Mobilité et gestion efficace des fréquences dans un réseau ad hoc à forte efficacité

Stephane Pomportes

UNIVERSITÉ PARIS SUD

Comprendre le monde, construire l'avenir®

UNIVERSITE PARIS-SUD

ÉCOLE DOCTORALE : « *Sciences et Technologies de l'Information des Télécommunications et des Systèmes* »

Laboratoire : *Laboratoire des Signaux et Systèmes*

DISCIPLINE : PHYSIQUE

THÈSE DE DOCTORAT

soutenue le 12/12/2011

par

Stéphane POMPORTES

Mobilité et gestion efficace des fréquences dans un réseau ad hoc à forte efficacité

Directeur de thèse :	Véronique VEQUE	Professeur (Laboratoire des Signaux et Systèmes)
Co-directeur de thèse :	Joanna TOMASIK	Professeur Adjoint (SUPÉLEC)
	Anthony BUSSON	Maître de Conférences (L2S)

Composition du jury :

Président du jury :	Bertrand DUCOURTHIAL	Professeur (Université Technologique de Compiègne)
Rapporteurs :	Vincent VILLAIN	Professeur (Laboratoire MIS)
	Fabrice VALOIS	Professeur (INRIA SWING / CITI, INSA-Lyon)
Examinateurs :	Steven MARTIN	Maître de Conférences (Université Paris-Sud)
	Geraud ALLARD	Ingénieur (Sagem Défense Sécurité)

Table des matières

1 Introduction **5**

1.1 Communications lors d'intervention des forces de sécurité civile 5

1.2 Organisation du mémoire 9

I Modèles de mobilité adaptés aux équipes de la sécurité civile **11**

2 Des modèles de mobilité... **13**

2.1 Introduction 13

2.2 Modèles synthétiques 14

2.2.1 Modèles purement aléatoires 15

2.2.2 Modèles avec dépendance temporelle 19

2.2.3 Modèles avec dépendance spatiale 22

2.2.4 Modèles avec contraintes géographiques 23

2.2.5 Modèles de déplacement humain 27

2.3 Les modèles composites 29

2.4 Conclusion 31

3 Composite Mobility Model for Disaster Area (C2MDA) **33**

3.1 Introduction 33

3.2 Le modèle C2MDA 33

3.2.1 Gestion des déplacements 34

3.3 Gestion des obstacles dans C2MDA 34

3.4 Évaluation du C2MDA 37

3.4.1 Comparaison de la distribution spatiale des nœuds du RWP et du Lévy-walk . . . 37

3.4.2 Impact du C2MDA sur la taille du voisinage des mobiles pour des équipes de taille moyenne 38

3.4.3 Influence de la présence d'obstacles sur le degré des nœuds et les temps de contact dans le C2MDA 40

3.5 Conclusion 42

II Allocation distribuée de ressources radio **43**

4 État de l'art **45**

4.1 Introduction . . . 45

4.1.1 Accès par contention . . . 45

4.1.2 Accès par réservation . . . 46

4.1.3 Conclusions . . . 49

4.2 Modélisation du réseau . . . 49

4.2.1 Modélisation par graphe . . . 51

4.2.2 Modélisation issue de la théorie de l'information . . . 52

4.2.3 Importance de la modélisation des interférences . . . 53

4.3 Allocation de ressources et modélisation par graphe . . . 53

4.3.1 Coloration des nœuds . . . 53

4.3.2 Coloration des liens . . . 53

4.3.3 Unification des deux approches . . . 54

4.3.4 Allocation orientée nœud dans un contexte distribué . . . 55

4.3.5 Allocation sur les liens dans un contexte distribué . . . 56

4.4 Conclusion . . . 59

5 Détection et correction de conflits **61**

5.1 Introduction . . . 61

5.2 Modélisation . . . 62

5.2.1 Modélisation du réseau . . . 62

5.2.2 Modélisation des interférences . . . 62

5.2.3 Modèle à états . . . 62

5.2.4 Auto-stabilisation . . . 63

5.3 Algorithme de correction de conflits et preuves . . . 63

5.3.1 Correction des conflits existants . . . 64

5.3.2 Détection des ressources utilisables . . . 66

5.3.3 Spécifications . . . 69

5.3.4 Preuve des spécifications . . . 71

5.4 Algorithme 2 : Prise en compte de la priorité des arcs . . . 79

5.5 Conclusion . . . 85

6 Stratégies de répartition des ressources **87**

6.1 Introduction . . . 87

6.2 Allocation garantissant la connectivité . . . 87

6.2.1 Principes de l'algorithme DSATUR . . . 88

6.2.2 Présentation de notre algorithme de connectivité . . . 91

6.2.3 Validation de l'algorithme de connectivité 95
6.2.4 Conclusions 100
6.3 Allocation garantissant la QoS pour un trafic donné 100
6.3.1 Pondération des arcs en fonction des demandes 100
6.3.2 Présentation de l'algorithme 101
6.3.3 Validation de l'algorithme d'allocation avec QoS 102
6.3.4 Amélioration du temps de convergence 104
6.4 Conclusion 105

7 Conclusion et perspectives **107**

A Rappels de théorie des graphes **111**

Chapitre 1

Introduction

Pompiers, policiers, services médicaux d'urgences et services de secours représentent autant de corps de métiers mis à contribution au quotidien dans le cadre de la protection civile. Les circonstances pouvant les amener à intervenir peuvent être diverses. Dans chacun de ces scénarios d'intervention, la communication entre les différents corps de la protection civile ainsi qu'au sein de chaque équipe est primordiale. Nous avons donc travaillé sur de nouveaux algorithmes d'allocation de ressources ainsi que sur des modèles de mobilités qui permettent de valider nos solutions.

1.1 Communications lors d'intervention des forces de sécurité civile

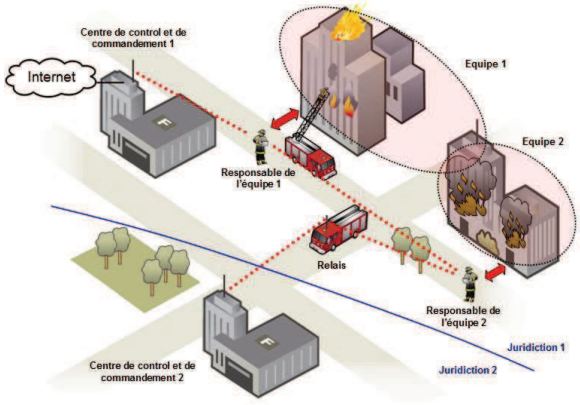


FIGURE 1.1: Intervention en cas d'incendie [RAF08]

Un premier scénario d'intervention est illustré par la figure 1.1. Il s'agit d'un incendie dans un immeuble d'habitation où interviennent une ou plusieurs équipes de pompiers accompagnées des services médicaux d'urgences. Dans ce scénario, la sécurité tant des civils présents sur les lieux que celles des intervenants doit être assurée. Ces derniers ont alors besoin de communiquer entre eux afin de s'échanger des informations sur l'état du bâtiment, l'avancée de l'incendie ou le signalement de blessés. Généralement,

la coordination de ces différentes équipes est sous la responsabilité d'un chef d'opération qui communique donc avec les responsables des différents groupes qui commandent à leur tour les actions de leurs équipiers.

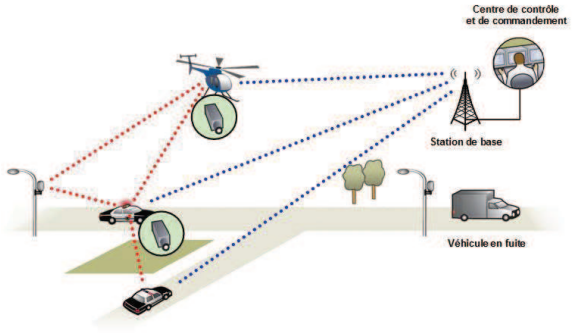


FIGURE 1.2: Poursuite d'un véhicule en fuite [RAF08]

Sur la figure 1.2, les forces de police interviennent dans le cadre de la poursuite d'un véhicule en fuite. Tout en poursuivant le fugitif, les policiers communiquent pour coordonner les véhicules suiveurs, demander à un centre de renseignement des informations sur le chauffeur grâce au numéro de plaque d'immatriculation relevé sur le véhicule. En fonction de la situation, ils peuvent également être amenés à demander des renforts par hélicoptère qui filment toute la scène en mouvement pour permettre au PC (Poste de Commandement) de coordonner l'action des poursuivants.

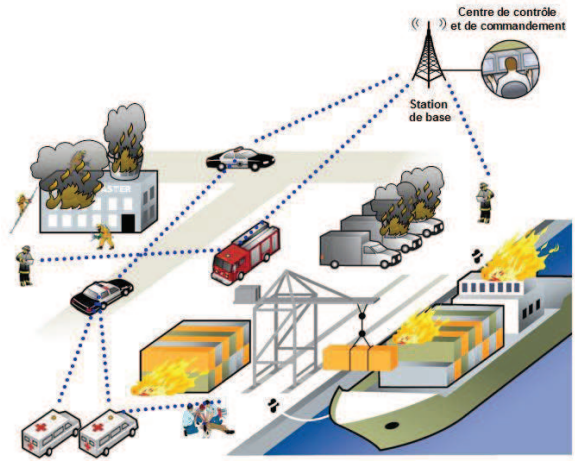


FIGURE 1.3: Explosion dans un port

Le scénario de catastrophe majeure est plus contraignant du point de vue des communications car l'infrastructure existante peut être endommagée ou détruite. La figure 1.3 représente une explosion dans un port où de nombreux acteurs de la protection civile interviennent : pompiers, police et services médicaux d'urgence. De même que précédemment, la scène peut être filmée depuis un hélicoptère et retransmise vers le PC. En effet, la taille de la zone d'intervention, les conditions d'accès et de déplacement ainsi que les différents incendies qui peuvent avoir été déclenchés par l'explosion représentent autant de contraintes dont les forces d'intervention doivent être informées. Le PC va délivrer ces informations, en

fonction du rôle des différentes équipes dans les plans d'intervention et des capacités des terminaux qu'ils transportent. Ces informations peuvent être délivrées oralement, ou sous forme de photo ou de vidéo de points précis.

A travers ces quelques exemples d'intervention des forces de sécurité civile, nous avons isolé quelques caractéristiques importantes des communication sur les scènes d'intervention. Les communications s'effectuent entre différents corps d'intervention de manière hiérarchique : une chef d'opération communique avec les chefs de groupe qui communiquent avec des équipes. Ces communications se font principalement de manière orale au moyen de terminaux de type téléphone portable mais des vidéos et photos de la scènes ou des parties de scène peuvent être nécessaires. Toutes les informations n'ont pas la même importance ou le même niveau de criticité : une alerte à l'explosion imminente est plus critique que la position d'un blessé léger ou un commandement est plus important qu'une notification de position. Cela se traduit par des niveaux de priorité entre les communications qui dépendent directement de l'organisation des opérations et de la criticité des informations. Compte-tenu du caractère vital des interventions, il faut souligner que les communications doivent être faites avec une qualité de service garantie notamment en termes de fiabilité et de délai. Enfin, nous remarquons que tous les acteurs se déplacent énormément que ce soit à pied, avec un véhicule terrestre ou par hélicoptère.

L'identification des besoins de communication des forces de sécurité civile dans les réseaux de circonstance ont guidé quelques-uns des problèmes abordés dans cette thèse en particulier la modélisation de la mobilité des forces d'intervention. Ainsi, la première partie de cette thèse porte sur l'étude des modèles de mobilité humaine ou véhiculaire sur une scène de catastrophe. Cette étude a porté sur différents modèles de mobilité et établit lesquels illustraient le mieux les caractéristiques de déplacement que nous venons de décrire.

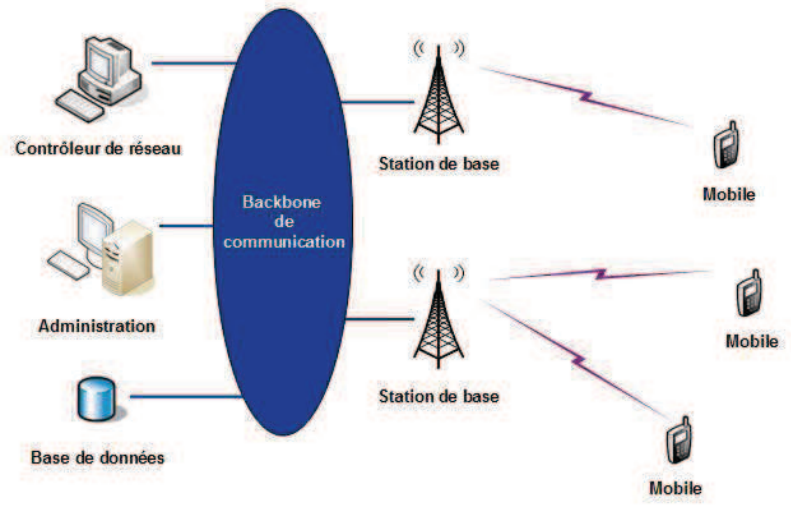


FIGURE 1.4: Architecture d'un système PMR [RAF08]

Depuis de nombreuses années, les forces de sécurité civile possèdent leur propre infrastructure de communication fondée sur la technologie PMR (*Professional/Private Mobile Radio*) [Ket04], et dont les

principaux éléments sont :

- une *bande de fréquence* radio autour de 400 MHz réservée à ce système,
- les *stations mobiles* sont des terminaux de communication renforcés pour supporter des conditions d'utilisation extrêmes et sont utilisées par les équipes d'intervention, piétons ou véhicules ;
- les *stations de base*, équipées de systèmes émetteurs/récepteurs radios, permettent de relier les stations mobiles au réseau d'infrastructure ;
- l'*infrastructure de communication* est un réseau filaire qui interconnecte l'ensemble des stations de base ainsi que les équipements de contrôle,
- divers *équipements de contrôle du réseau* sont utilisés principalement pour gérer les ressources radios ainsi que le routage des flux de données.

Les systèmes PMR utilisent donc un réseau d'infrastructure ce qui peut poser problème en cas de catastrophe majeure puisque celui-ci peut être détruit ou grandement endommagé. En cas de tremblement de terre ou d'explosion chimique, les stations de base peuvent être détruites, les connexions du réseau *backbone* peuvent être coupées tout comme l'alimentation électrique des différents équipements du réseau d'infrastructure. Pour pallier ces différents problèmes, certains organismes tels que le programme Américain SAFECOM ont préconisés [SAF06] que les réseaux de communications dédiés aux forces de la sécurité civile devaient être capables de former des réseaux ad hoc, encore appelés réseaux de circonstance, capables de fonctionner et de s'auto-organiser en l'absence de toute infrastructure.

Certaines normes largement utilisées telles que l'IEEE 802.11 permettent déjà aux terminaux de communiquer entre eux. Cependant, les débits, la portée ainsi que la qualité de service offerte ne sont pas suffisants pour satisfaire les différents besoins des forces de la sécurité civile. Ces dernières ont en effet besoin de :

- *services de données :* lors d'une intervention, les équipes ont besoin de consulter la carte du terrain ou le plan du bâtiment dans lesquels elles interviennent. Elles peuvent également souhaiter accéder à d'autres informations telles que les emplacements des conduites de gaz ou le registre du personnel présent ;
- *services de phonie :* la communication audio est la base de la coordination des différents intervenants. Le réseau utilisé doit donc fournir un système à haut niveau de qualité de service en termes de disponibilité, de latence et de fiabilité des liens ;
- *services vidéo :* le centre de commandement en charge de la coordination des équipes peut utiliser un service de vidéo en temps réel pour évaluer rapidement la situation du lieu de l'incident. Des photos ou vidéos peuvent également être transmises à d'autres équipes d'intervention afin d'adapter le matériel utilisé au type de sinistre rencontré. Le réseau doit ainsi pouvoir supporter le transport d'informations à haut débit et en temps réel.

La particularité de ces services est qu'ils doivent être de haut niveau avec des garanties en termes de fiabilité, de disponibilité ou de délai. Seules certaines technologies existantes telles que la norme IEEE 802.16 (WiMAX) [80204] ou le LTE (Long Term Evolution) [STB09] permettent de répondre à toutes ces

contraintes tout en offrant une portée de communication suffisante. Cependant, aucune de ces technologies ne prévoit de mode ad hoc. De nouvelles solutions doivent donc être trouvées pour répondre aux besoins spécifiques des forces de la protection civile. Au niveau de l'allocation de ressources par exemple, les solutions fondées sur des protocoles d'accès aléatoire ne permettent pas de garantir la qualité de service requise par ces applications. Il faut donc développer de nouveaux protocoles d'allocation de ressources décentralisés pour fonctionner dans un réseau ad hoc, et avec réservation de manière à satisfaire la QoS demandée. Ce problème fera l'objet de la seconde partie de cette thèse.

1.2 Organisation du mémoire

Les travaux de cette thèse s'intéressent à :

- la modélisation du déplacement des équipes de la protection civile,
- l'allocation de ressources dans les réseaux ad hoc.

La première partie porte sur les modèles de mobilité et est constituée de deux chapitres. Le premier dresse un état de l'art des modèles existants tandis que le second est dédié à la présentation du modèle de mobilité que nous avons développé.

La seconde partie de cette thèse traite de l'allocation de ressources dans les réseaux ad hoc. Son premier chapitre présentera les différentes solutions existantes. Dans nos travaux, nous avons décidé de découper le problème de l'allocation en deux étapes distinctes : la correction des conflits d'une part et la répartition des ressources d'autre part. Le chapitre suivant est consacré à la gestion des conflits. Nous y présenterons tout d'abord un premier algorithme auto-stabilisant ainsi que les preuves de son temps de convergence. Notre second algorithme est une amélioration du premier car il limite le nombre de ressources libérées et introduit un ordre de priorité entre les différentes allocations. Le dernier chapitre porte sur l'allocation de ressources et la spécification de deux algorithmes. Le premier garantit la connectivité du réseau en attribuant une ressource à chaque lien de communication du réseau. Le second garantit un certain niveau de qualité de service en adaptant la quantité de ressources attribuées à la demande de chaque lien.

Première partie

Modèles de mobilité adaptés aux équipes de la sécurité civile

Chapitre 2

Des modèles de mobilité...

2.1 Introduction

Dans les réseaux ad hoc mobiles, la plupart des travaux proposant une solution protocolaire évaluent ses performances à l'aide de simulations. Dans [KCC05], les auteurs ont mené une étude portant sur 150 articles scientifiques afin de mieux connaître cette évaluation de performances dans ce type de réseau. Les résultats ont montré que la validation par simulation était utilisée dans près de 75% des travaux.

Pour que les résultats de cette évaluation de performance soient pertinents, il est important que les simulations reproduisent le plus fidèlement possible les conditions réelles dans lesquelles le protocole doit fonctionner. L'environnement de simulation doit ainsi prendre en compte de nombreux paramètres tels que la portée de transmission, la limitation et le partage de la bande passante ou encore les pertes de paquets. Lorsque le réseau est constitué d'éléments mobiles, des connexions peuvent apparaître ou disparaître au gré des déplacements de ces unités. Le simulateur doit donc tenir compte de cet aspect à l'aide d'un modèle de mobilité adapté au type de déplacement rencontré dans le réseau.

De nombreux modèles de mobilité ont été proposés au cours de ces dernières années. Les différents travaux qui ont étudié ces modèles [HFB06, CBD02, MM09, BH04] les classent généralement en deux grandes familles :

– *Les modèles basés sur des traces :* Basés sur les données récoltées lors d'essais réels tels que la position GPS de plusieurs mobiles au cours du temps, ces modèles permettent de reproduire le déplacement enregistré lors des simulations. En reproduisant à l'identique les caractéristiques mesurées dans les essais, les modèles basés sur des traces permettent de se rapprocher d'une mobilité réelle. Cet avantage est également son principal inconvénient. En effet, les traces collectées par un mobile représentent le déplacement de celui-ci dans des conditions particulières. Pour pouvoir changer certains paramètres de simulation comme la position d'obstacles ou la taille de la zone sur laquelle les unités se déplacent, il faut refaire de nouveaux tests afin de capturer les informations correspondantes à ces paramètres. Ce manque de souplesse ainsi que la difficulté que peut représenter l'accès à des données issues de la mobilité réelle sont les raisons pour lesquelles ces

modèles sont de loin les moins nombreux [MM09].

– *Les modèles synthétiques :* Le déplacement des unités y est défini à partir de composantes simples tels que la vitesse ou la destination du mobile. Suivant les modèles, la valeur de ces paramètres peut changer à intervalle régulier, ou au contraire être modifiée uniquement lorsqu'un événement se produit comme, par exemple, le passage d'un mobile à un point particulier. Ainsi, contrairement aux modèles basés sur les traces qui reproduisent un mouvement préenregistré, les modèles synthétiques génèrent une trajectoire pour chaque mobile.

Certains travaux se sont intéressés aux traces, non pas pour les rejouer, mais pour identifier les caractéristiques du déplacement humain ou animal de manière plus générale. Certains modèles synthétiques y font donc référence, mais elles servent alors uniquement de base à la construction du modèle ou de trajectoire de référence pour comparer les résultats du modèle synthétique avec un déplacement réel.

Ce chapitre se concentre uniquement sur les modèles synthétiques qui sont présentés en détails dans le paragraphe suivant.

2.2 Modèles synthétiques

Dans un modèle synthétique, le mouvement d'un agent mobile est généralement constitué d'une succession d'étapes pendant lesquelles les mobiles se déplacent à vitesse et orientation constantes. La trajectoire de chaque mobile peut donc être définie par un vecteur vitesse et sa fréquence d'évolution. Dans certains modèles, le vecteur vitesse est défini explicitement. Lorsque ce n'est pas le cas, la trajectoire est généralement décrite à l'aide d'un algorithme indiquant la manière de choisir certains paramètres tels que la distance à parcourir, le temps de trajet, l'orientation du déplacement ou tout simplement la vitesse du mobile. Ces informations permettent de reconstituer le vecteur vitesse à travers ses différentes composantes : sa norme (la vitesse à proprement parler), son orientation et son sens. Quelle que soit la manière dont la trajectoire est définie, il est donc toujours possible de retrouver le vecteur vitesse des mobiles.

Pour éviter que les mobiles ne se déplacent en ligne droite et à vitesse constante pendant toute la durée de la simulation, il est important que ce vecteur vitesse puisse changer au cours du temps. Suivant le modèle de mobilité, l'évolution de ce vecteur vitesse peut se faire de manière :

– *périodique* : le temps de simulation est découpé en intervalles de durée identique. Au début de chaque période de temps, un nouveau vecteur vitesse est défini pour chaque mobile.

– *événementielle* : Les changements du vecteur vitesse sont déclenchés par des événements discrets tels que la rencontre des extrémités de la zone de simulation ou l'arrivée d'un mobile à un point défini comme sa destination.

– *continue* : Lorsque la trajectoire des mobiles prend en compte l'accélération, l'évolution du vecteur vitesse peut se faire de manière continue. Afin d'éviter les changements brusques de trajectoire et de vitesse, cette approche est notamment utilisée dans [Bet01].

Chaque trajectoire correspond à une combinaison particulière du vecteur vitesse et de sa fréquence de

changement de valeur. En fixant chacun de ces deux paramètres, les modèles de mobilité ont la capacité de donner des propriétés particulières au mouvement des mobiles. Les modèles existants sont classés en fonction de ces propriétés. Au cours de ce chapitre, cinq catégories de modèles seront présentées :

- *Les modèles purement aléatoires :* ces modèles se caractérisent par une définition du vecteur vitesse totalement indépendante de l'environnement et de l'historique. À chaque changement il s'agit généralement de choisir aléatoirement chacune des composantes du vecteur vitesse sur son ensemble de définition. La principale différence entre les modèles purement aléatoires réside dans la fréquence à laquelle les changements de ce vecteur ont lieu.
- *Modèles avec dépendance temporelle :* dans la réalité, les mobiles sont soumis à un certain nombre de forces physiques qui contraignent leur déplacement. Ainsi, il leur est impossible de changer instantanément de vitesse ou de direction. Pour tenir compte de cette continuité des différents aspects de la mobilité, les modèles avec dépendance temporelle intègrent l'historique du déplacement dans la définition du vecteur vitesse.
- *Modèle avec dépendance spatiale :* Suivant le type de déplacement étudié, certains mobiles peuvent se déplacer en groupe, des groupes peuvent interagir entre eux. Pour gérer ces mobiles qui ont des trajectoires similaires, certains travaux proposent des modèles avec dépendance spatiale.
- *Modèle avec restriction géographique :* la zone de simulation n'est pas toujours utilisable dans sa totalité. Il peut exister des plans d'eau, des rochers ou encore des bâtiments que les mobiles doivent contourner. Les modèles avec restriction géographique permettent d'adapter la trajectoire des mobiles afin de prendre en compte ces différents obstacles.
- *Modèles de déplacement humain :* Certaines études se sont intéressées au déplacement humain. Les résultats ont montré que certaines caractéristiques telles que la longueur des déplacements, le temps de pause ou encore la vitesse de déplacement suivent certaines lois probabilistes. Des modèles ont donc été conçus afin de reproduire le plus fidèlement possible ce type de déplacement.
- *Modèles composites :* lorsque le mouvement modélisé est complexe, comme par exemple le déplacement de militaires sur une zone de combat, il est nécessaire de combiner plusieurs des caractéristiques évoquées ci-dessus. Les modèles composites proposent donc de faire cohabiter plusieurs modèles existants afin de reproduire le déplacement d'unités mobiles dans des conditions particulières.

2.2.1 Modèles purement aléatoires

Les modèles purement aléatoires sont les premiers modèles à avoir été proposés. Leur plus grande force est leur simplicité qui permet de les utiliser, de les étudier et de les implémenter très facilement. Cependant, cette simplicité n'est pas sans conséquence et de nombreuses études ont montré que certaines caractéristiques de ces modèles n'étaient pas souhaitables lors de l'analyse de performances d'un protocole.

Random Walk Le *Random Walk* (RW) a été introduit pour la première fois par Einstein en 1926 [Woo28] en tant qu'approximation du mouvement aléatoire des particules. Dans ce modèle, le vecteur vitesse de chaque mobile est défini part sa norme et sa direction. Ce vecteur est modifié périodiquement et la nouvelle valeur de ses paramètres est choisie uniformément sur $[V_{min},V_{max}]$ pour la vitesse et $]0,2\pi]$ pour la direction. V_{min} et V_{max} sont deux constantes représentant respectivement la vitesse minimale et maximale à laquelle un mobile peut se déplacer.

Bien qu'à l'origine, ce modèle n'était pas prévu pour décrire le déplacement de mobiles dans un réseau ad hoc, de nombreux travaux l'utilisent [JHR01, SPR07, SPR08] et étudient son influence sur les protocoles de communication [AK11, CLYE08, La10]. Un exemple de trajectoire générée à partir de ce modèle est présenté sur la figure 2.1

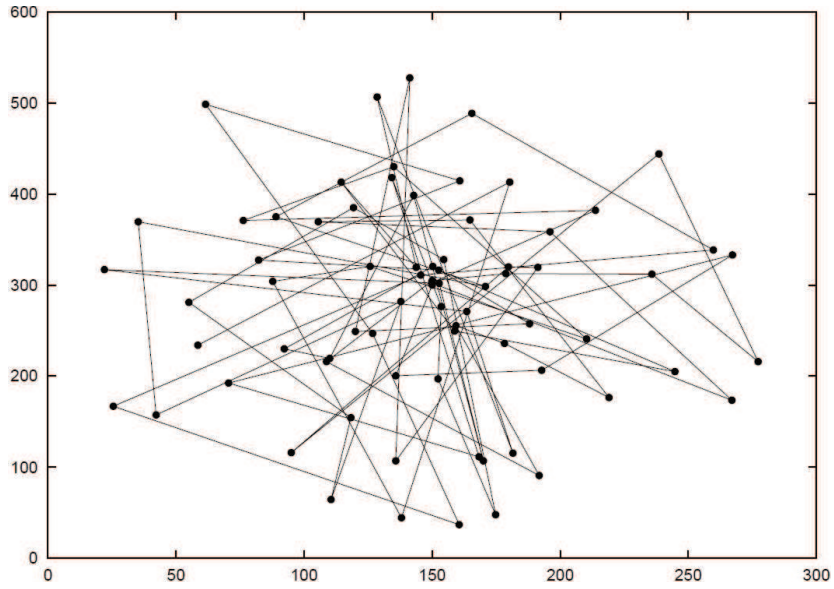


FIGURE 2.1: Trajectoire d'un mobile générée à partir du modèle Random Walk [CBD02].

Random WayPoint Le modèle *Random WayPoint* (RWP) a été présenté pour la première fois dans [JM96]. La trajectoire des mobiles y est définie par deux paramètres : leur destination et leur vitesse. Dans ce modèle, le changement de vecteur vitesse est événementiel et survient à chaque fois qu'un mobile atteint sa destination. Ce modèle prévoit également un temps de pause entre chaque déplacement. Le fonctionnement de ce modèle est résumé par l'algorithme 2.2.1. Un exemple de trajectoire générée par ce modèle est présenté sur la figure 2.2.

Algorithme 2.2.1: Random WayPoint appliqué pour chaque mobile p

Données :

$Dest_p \leftarrow$ coordonnées cartésiennes du point vers lequel se dirige le mobile p ;

$V_p \leftarrow$ vitesse du mobile p définie sur l'intervalle [V_{min},V_{max}];

$Pause_p \leftarrow$ temps de pause du mobile p défini sur l'intervalle [P_{min},P_{max}];

1. Choisir une nouvelle valeur pour $Dest_p$ uniformément dans chacune des directions;
2. Choisir une nouvelle valeur pour V_p uniformément sur l'intervalle [V_{min},V_{max}] ;
3. Déplacer ce mobile p vers $Dest_p$ à la vitesse V_p;
4. Choisir une nouvelle valeur pour $Pause_p$ uniformément sur l'intervalle [P_{min},P_{max}];
5. Effectue une pause d'une durée $Pause_p$;
6. Recommencer à l'étape 1 jusqu'à la fin de la simulation;

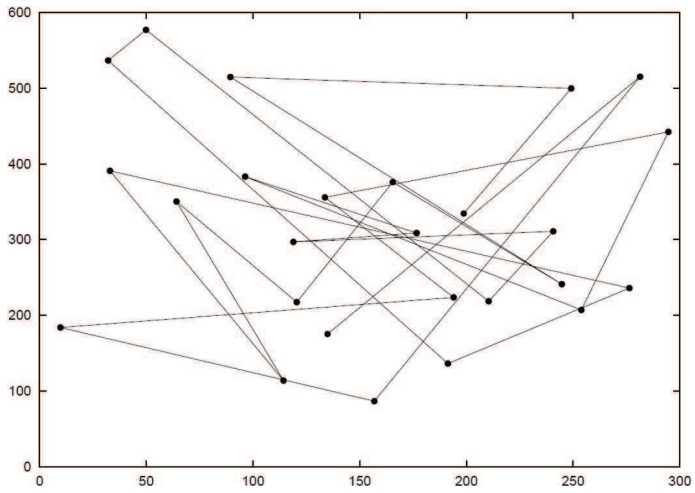


FIGURE 2.2: Trajectoire d'un mobile générée à partir du modèle Random WayPoint [CBD02].

La principale différence entre ce modèle et le RW repose sur la fréquence de changement des vecteurs vitesse, périodique dans le RW et évenementielle dans le RWP. La définition de ce vecteur est quant à elle sensiblement identique. En effet, dans ces deux modèles, la norme du vecteur vitesse est choisie uniformément sur l'ensemble [V_{min},V_{max}]. Concernant la direction, le RW la définit explicitement et elle est choisie uniformément sur l'intervalle]0, 2π]. Dans le RWP, l'orientation du vecteur vitesse d'un mobile est déterminée par sa destination. Puisque celle-ci est choisie sur l'ensemble de la zone de simulation et uniformément dans ses deux directions, l'orientation du vecteur vitesse dans le RWP est également choisie sur l'intervalle]0, 2π]. Cependant, contrairement au RW, elle n'est pas choisie uniformément sur l'ensemble de cet intervalle. En effet un mobile situé par exemple à proximité du bord droit de la zone de simulation aura plus de destinations possibles sur sa gauche que sur sa droite. La probabilité qu'il parte vers la gauche est donc plus élevée.

Le RWP est relativement simple à utiliser et à implémenter. C'est sans doute la raison pour laquelle ce modèle est le plus utilisé dans les travaux sur les réseaux ad hoc mobiles [KCC05]. Cependant, cette simplicité en fait également un modèle peu réaliste. De nombreuses études se sont intéressées à ses

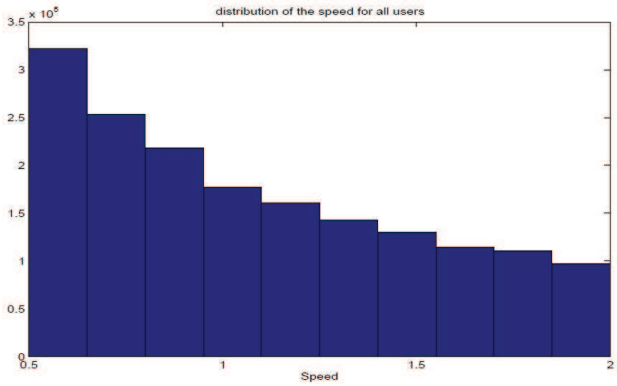


FIGURE 2.3: Distribution de la vitesse instantanée des mobiles pour le RWP après une journée de simulation. [LB07]

propriétés et ont mis en évidence un certain nombre de caractéristiques indésirables :

- *une convergence de la vitesse instantanée vers V_{min} :* Pour étudier la vitesse des mobiles, il est parfois nécessaire de faire la distinction entre la vitesse événementielle, mesurée lorsqu'un mobile change de vecteur vitesse, et la vitesse instantanée, mesurée à n'importe quel moment de la situation. Avec RWP, la vitesse événementielle est explicitement définie, elle a donc une distribution uniforme sur l'intervalle [V_{min},V_{max}]. Intuitivement on pourrait penser qu'il en va de même pour la vitesse instantanée. Cependant, les auteurs de [YLN03], qui ont étudié le modèle RWP, ont montré que cette vitesse instantanée avait tendance à diminuer au cours du temps en convergeant vers V_{min}. Cette convergence vers la vitesse minimale peut s'expliquer de la manière suivante. Lorsqu'un mobile choisit une nouvelle destination et une nouvelle vitesse, la durée du trajet est proportionnelle à la distance qu'il doit parcourir et inversement proportionnelle à la vitesse choisie. Si un mobile choisit une destination éloignée et une vitesse faible, la durée du trajet peut être très longue. Il se trouve alors « piégé » à cette vitesse faible. Au contraire, les mobiles qui choisissent une vitesse élevée et une destination proche vont effectuer leur trajet très rapidement. Ainsi, même si les nouvelles vitesses sont choisies uniformément, les mobiles les plus rapides changent plus souvent de vitesse que les mobiles les plus lents. La distribution de la vitesse instantanée qui initialement était uniforme, se déséquilibre progressivement pour adopter la distribution présentée à la figure 2.3 issue de l'étude [LB07] et représentant la distribution des vitesses instantanées dans un réseau après une journée de simulation.
- *une forte concentration des mobiles dans le centre de la zone de simulation :* De nombreux travaux tels que [BRS03, GSB+08] ont étudié l'évolution de la distribution des mobiles. Les résultats ont montré que le RWP avait tendance à concentrer les mobiles au milieu de la zone de simulation. Cette propriété peut avoir des conséquences non négligeables sur les résultats de simulations. Par exemple, dans le cas de réseaux cellulaires, les cellules centrales auront une charge plus importante ainsi que des taux d'entrée et sortie plus important que les cellules en bordure de zone de simulation.
- *une oscillation du nombre de voisins :* Dans [RMSM01], les auteurs ont observé que le degré de

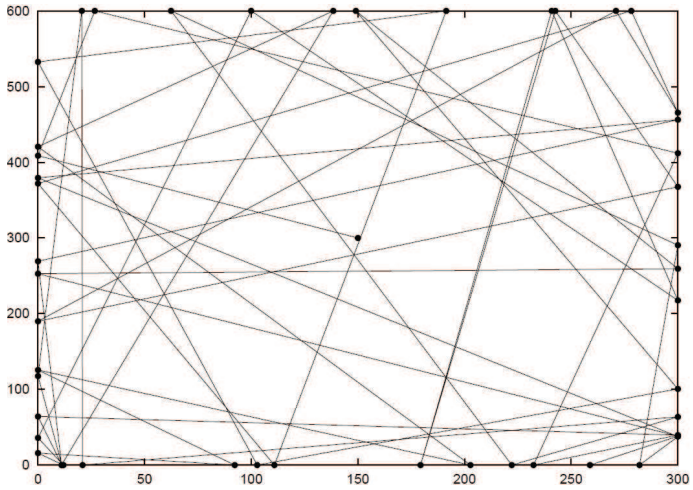


FIGURE 2.4: Trajectoire d'un mobile générée à partir du modèle Random Direction. [CBD02]

chaque mobile variait au cours du temps pendant toute la durée de la simulation. Cette oscillation est directement liée au point précédent. Lorsqu'un mobile se déplace, plus il se rapproche du centre et plus la densité est élevée, augmentant ainsi son degré. Lorsqu'il s'en éloigne, cette densité diminue et son degré également. Chaque mobile voit dont son degré fluctuer à chaque fois qu'il s'éloigne ou se rapproche du centre de la zone de simulation.

Random Direction Pour atténuer les défauts du RWP, les auteurs de [RMSM01] ont proposé un nouveau modèle : le *Random Direction* (RD) dont le fonctionnement est le suivant. Initialement, l'ensemble des mobiles est placé sur la zone de simulation en choisissant leur position uniformément dans les deux directions de l'espace. Chaque mobile tire une direction et une vitesse, toutes deux choisies uniformément sur [0, 2π] pour la direction et sur [V_{min}, V_{max}] pour la vitesse. Les mobiles vont ensuite se déplacer dans la direction choisie à la vitesse choisie jusqu'à ce qu'il rencontre le bord de la zone de simulation. Lorsqu'un mobile atteint cette bordure, il effectue une pause dont la durée est choisie uniformément sur [P_{min},P_{max}]. À la fin de cette pause, le mobile choisit une nouvelle direction et une nouvelle vitesse. Contrairement au premier tirage, l'ensemble des valeurs possibles pour la direction est restreinte pour empêcher le mobile de sortir de la zone de simulation. Par exemple, dans le cas d'un mobile atteignant le bord supérieur de la zone de simulation, l'intervalle de définition sera réduit de moitié afin d'obliger ce mobile à redescendre. Le tirage de la vitesse se fait toujours uniformément sur [V_{min}, V_{max}].

Contrairement au RWP, RD ne concentre pas les mobiles au milieu de la zone de simulation. Il limite ainsi le phénomène d'oscillation du degré. Un exemple de trajectoire générée par le modèle *Random Direction* est présenté sur la figure 2.4.

2.2.2 Modèles avec dépendance temporelle

Dans les modèles purement aléatoires que nous venons d'exposer, les changements de direction peuvent se faire instantanément, les modifications de la vitesse ne connaissent aucune transition et passent

brusquement d'une valeur à une autre. Pour rendre les déplacements plus réalistes, certains travaux proposent de tenir compte de l'historique des mouvements à chaque modification de la trajectoire ou de la vitesse. Ces modèles présentent donc une dépendance temporelle.

Modèle de Gauss-Markov Dans [LH03], les auteurs présentent le modèle de mobilité de Gauss-Markov. Dans ce modèle, le temps est discrétisé en intervalles de durée constante. A chaque instant t, un nouveau vecteur vitesse, noté $\vec{v_t}$, est choisi.

Le nom de ce modèle fait référence aux deux propriétés que possède le processus à l'origine du vecteur vitesse des mobiles. Ce processus est en effet markovien car son état futur ne dépend pas des états passés mais uniquement de son état présent. Il contient également une composante gaussienne car pour pouvoir introduire une modification entre deux valeurs consécutives, ce processus fait intervenir une distribution normale. L'expression du vecteur vitesse dans un espace à deux dimensions est définie par :

$$\vec{v}_{t+1} = \begin{cases} v_x^{t+1} = \alpha v_x^t + (1-\alpha)\mu_x + \sigma_x\sqrt{1-\alpha^2}w_x^t \\ v_y^{t+1} = \alpha v_y^t + (1-\alpha)\mu_y + \sigma_y\sqrt{1-\alpha^2}w_y^t \end{cases} \tag{2.1}$$

où $\vec{\mu} = [\mu_x, \mu_y]$ représente le vecteur vitesse moyen et $\vec{\sigma} = [\sigma_x, \sigma_y]$ son écart type. $\vec{w} = [w_x, w_y]$ représente le vecteur d'un processus gaussien normalisé $\mathcal{N}(0,1)$.

Le vecteur vitesse à un instant $t+1$, $\vec{v}_{t+1}$, dépend donc à la fois de sa vitesse à l'instant t, $\vec{v}_t$, mais également d'une composante totalement indépendante : $\vec{w_t}$.

La constante $\alpha \in [0,1]$ permet d'établir le niveau de dépendance temporelle du vecteur vitesse. Plus ce facteur est élevé, plus le vecteur vitesse à l'instant $t+1$ est proche de celui à l'instant t. Lorsque $\alpha = 1$, le modèle Gauss-Markov devient un modèle *fluid-flow* [FM94] caractérisé par un vecteur vitesse constant.

Au contraire plus ce facteur est faible et plus les fluctuations du vecteur vitesse sont importantes. Lorsque $\alpha = 0$, il n'existe plus aucune dépendance avec le mouvement précédent. La trajectoire des mobiles est alors la même que pour le modèle *Random-Walk* présenté au paragraphe 2.2.1.

Smooth Random Dans [Bet01], les auteurs présentent un nouveau modèle dont la particularité est de définir un changement totalement continu du vecteur vitesse. Celui-ci est défini à partir de ses deux composantes : la vitesse et la direction. Pour obtenir une variation continue de ce vecteur, les auteurs introduisent un vecteur accélération permettant ainsi aux mobiles de passer progressivement d'un vecteur vitesse à un autre.

Pour déterminer la vitesse instantanée d'un mobile v(t), ce modèle utilise deux variables :

– la vitesse à atteindre *v*(t)*.

– l'accélération instantanée *a(t)*.

Dans les modèles présentés jusqu'à maintenant, lorsqu'un changement de vitesse intervenait, il était directement appliqué à la vitesse instantanée, provoquant un brusque changement de celle-ci. Dans le modèle *Smooth Random* lorsqu'une nouvelle vitesse est définie, elle n'est pas appliquée à v(t) mais à

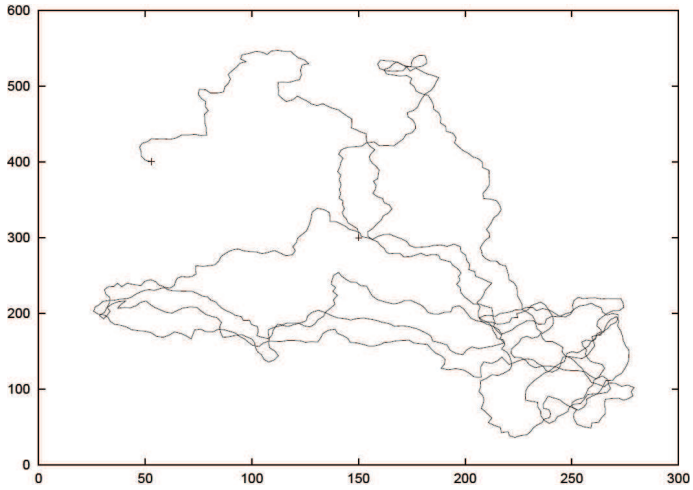


FIGURE 2.5: Trajectoire d'un mobile générée à partir de modèle Gauss-Markov. [CBD02]

v*(t). La vitesse instantanée est ensuite modifiée progressivement à l'aide de l'accélération a(t) afin d'atteindre la vitesse souhaitée v*(t).

Les vitesses v(t) et v*(t) sont bornées par une vitesse minimale de 0 m/s et une vitesse maximale V_{max}. Les auteurs font l'hypothèse que dans la réalité, la vitesse de déplacement n'est pas distribuée uniformément mais qu'il existe des vitesses auxquelles ces mobiles préfèrent se déplacer. Le modèle *Smooth Random* prévoit donc que les mobiles possèdent un ensemble de vitesse privilégiées qui auront une plus forte probabilité d'être choisies. Dans ce modèle, les auteurs proposent de définir la fréquence de changement de la vitesse v*(t) par un processus de Poisson.

Lorsqu'une nouvelle vitesse est choisie, le modèle détermine la valeur de l'accélération a(t) permettant de passer de la vitesse actuelle v(t) à la vitesse à atteindre v*(t). Si *v*(t) < v(t)*, une nouvelle valeur de *a(t)* est choisie uniformément dans l'ensemble [d_{max},0[, où d_{max} représente la décélération maximale. Si *v*(t) > v(t)* la nouvelle valeur de l'accélération est prise dans l'ensemble]0,a_{max}], où a_{max} représente l'accélération maximale.

Pour déterminer l'orientation instantanée du vecteur vitesse, notée $\varphi(t)$, le modèle utilise également deux variables :

- $\Delta\varphi(t)$, le changement de direction à l'instant t,
- $\varphi^*(t)$ la direction à atteindre.

Les variables $\varphi(t)$ et $\varphi^*(t)$ sont définies dans l'intervalle $[0, 2\pi]$. Tout comme pour la vitesse à atteindre v*(t), les les instances des modifications de $\varphi^*(t)$ suivent une loi de Poisson. La valeur de $\Delta\varphi$(t) représente l'angle entre l'orientation actuelle du mobile et son orientation souhaitée :

$$\Delta\varphi(t) = |\varphi^*(t) - \varphi(t)| \tag{2.2}$$

Pour éviter un brusque changement de direction, le modèle définit une vitesse de rotation. Pour cela, il introduit la variable Δt_c qui représente le temps alloué pour effectuer la rotation $\Delta\varphi$(t). Δt_c est choisie

uniformément sur [t_{min}, t_{max}] à chaque changement de $\varphi^*(t)$. La vitesse de rotation est alors définie par $\frac{\Delta\varphi(t)}{\Delta t_c}$.

Une propriété intéressante de ce modèle, en plus d'avoir des changements progressifs de vecteur vitesse, est d'avoir un rayon de courbure proportionnel à la vitesse du mobile. En effet puisque Δt_c est indépendant de la vitesse instantanée du mobile, plus la vitesse est élevée et plus la distance parcourue pendant Δt_c est grande. Le rayon de courbure est donc plus important pour des vitesses élevées ce qui correspond au comportement d'un mobile dans des conditions réelles.

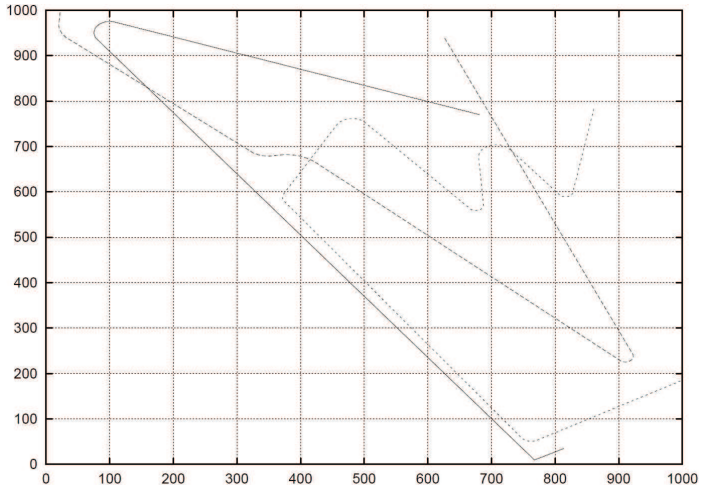


FIGURE 2.6: Trajectoire d'un mobile générée à partir du modèle Smooth Random [Bet01].

2.2.3 Modèles avec dépendance spatiale

Dans les modèles présentés jusqu'à maintenant, la trajectoire de chaque élément du réseau est indépendante de celle des autres mobiles. Cependant, lorsque l'on souhaite représenter le mouvement de mobiles se déplaçant en groupe ou convergeant vers une destination commune, il est important de pouvoir lier le déplacement de plusieurs mobiles.

Cette dépendance spatiale dans le déplacement des unités peut avoir un impact sur certaines caractéristiques de la communication comme la durée de vie des liens ou l'évolution du degré des mobiles.

En effet, les membres d'un même groupe suivent globalement la même trajectoire et sont proches les uns des autres. La durée de vie des connexions à l'intérieur d'un groupe est donc plus élevée qu'entre des mobiles appartenant à des groupes différents.

Dans le cas des modèles précédents, les unités étaient réparties de manière plus ou moins homogène sur la zone de simulation. Certaines zones pouvaient être plus denses mais cette augmentation de la densité était progressive, les fluctuations du degré se faisaient donc aussi de manière progressive. Dans le cas d'un déplacement en groupe les rencontres ne vont pas se faire mobile par mobile mais groupe par groupe. L'évolution du degré du mobile va donc se faire par palier dont la hauteur dépend de la taille des groupes. Toutes ces caractéristiques peuvent avoir un impact sur les performances d'un protocole. C'est pourquoi, certains travaux proposent des modèles qui prennent en compte cette dépendance spatiale.

Reference Point Group Model Les auteurs de [HGPC99] proposent un modèle de déplacement en groupe avec point de référence nommé RPGM (*Reference Point Group Mobility Model*) possédant les propriétés suivantes :

- chaque groupe dispose d'un chef qui sert de point de référence pour déterminer le déplacement global du groupe. Ce chef peut être un mobile comme un point virtuel,
- les membres d'un groupe suivent globalement le déplacement de leur chef tout en conservant une certaine liberté de mouvement,
- la trajectoire du chef de groupe est définie par le modèle RWP présenté au paragraphe 2.2.1.

Le vecteur vitesse d'un membre de groupe, noté $\vec{V_i^t}$ est défini par :

$$\overrightarrow{V_i^t} = \overrightarrow{V_{groupe}^t} + \overrightarrow{RM_i^t} \tag{2.3}$$

où $\overrightarrow{V_{groupe}^t}$ représente le vecteur vitesse du chef de groupe. Le vecteur $\overrightarrow{RM_i^t}$ définit la liberté de mouvement du mobile i par rapport au déplacement de son chef de groupe. Les coordonnées de ce vecteur sont

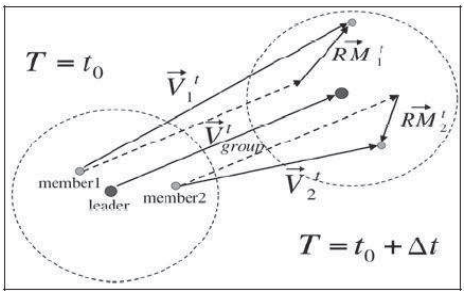


FIGURE 2.7: Exemple de mouvement généré par le modèle RPGM. [BH04]

calculées en fonction de deux variables aléatoires indépendantes :

- la distance r entre un mobile i et le chef du groupe est distribuée uniformément dans l'intervalle [0,r$_{max}$], où r_{max} représente l'écart maximum autorisé entre un membre et son chef de groupe,
- l'écart, noté b, entre la direction d'un mobile i et celle de son chef de groupe est distribuée uniformément sur l'intervalle [0,2π].

Par conséquent, si le chef de groupe se déplace entre les points [x$_0$,y$_0$] et [x$_1$,y$_1$] et avec b$\in$[0,2π] et r$\in$[0,r$_{max}$], l'expression du vecteur est définie par :

$$\overrightarrow{RM_i^t} = \begin{cases} RM_{i,x}^t = r \times cos(a+b) \\ RM_{i,y}^t = r \times sin(a+b) \end{cases} \tag{2.4}$$

où $a = arctan((y_1 - y_0)/(x_1 - x_0))$.

2.2.4 Modèles avec contraintes géographiques

Dans la plupart des modèles, la zone de simulation se résume à un rectangle dans lequel les mobiles sont libres de se déplacer. Dans la réalité, il existe de nombreuses contraintes géographiques telles que

des arbres, des plans d'eau ou des bâtiments qui doivent être évités. Pour être réaliste, un modèle doit donc tenir compte de ces obstacles et modifier la trajectoire des mobiles en conséquence.

2.2.4.1 Modélisation basée sur un graphe

Dans [THB+02] les auteurs proposent de remplacer la zone de simulation par un graphe $G(V, E)$ permettant de modéliser une carte routière. L'ensemble E permet de définir les différentes routes qu'un mobile peut emprunter tandis que les éléments de V modélisent les carrefours entre ces différentes routes. Dans ce modèle, les nœuds de l'ensemble V éseésentent ainsi les positions où un mobile peut changer de vecteur vitesse.

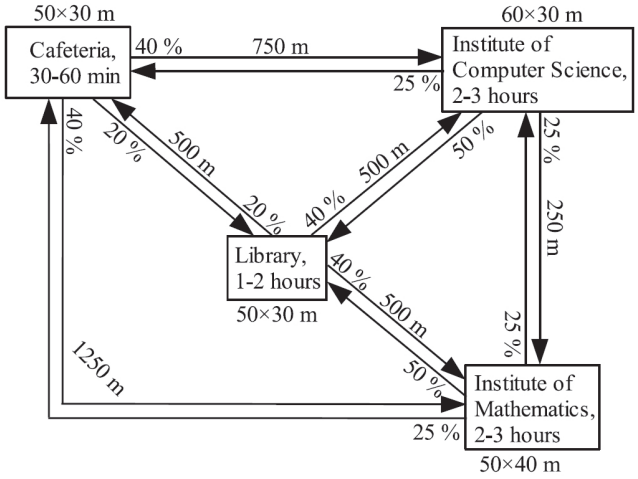


FIGURE 2.8: Modélisation de la zone de simulation par un graphe. [BRS05]

En 2005, [BRS05] réutilise ce concept de graphe mais avec une approche différente. Dans ce modèle, la zone de simulation est toujours définie à l'aide d'un graphe, mais cette fois les nœuds de l'ensemble V représentent des zones de tailles diverses dans lesquelles les mobiles peuvent se déplacer. Lorsqu'un mobile peut passer d'une zone à une autre, les nœuds qui représentent ces zones sont reliés par un arc. Ce graphe est orienté et pondéré. Le poids d'un arc e reliant les nœuds v_1 à v_2 détermine la probabilité de passer de la zone modélisée par v_1 à celle modélisée par v_2. Un exemple d'une telle zone de simulation est donné par la figure 2.8. Le déplacement des mobiles à l'intérieur de ces zones est défini par le modèle RWP (cf. paragraphe 2.2.1).

La même année, [HMS+05] propose le modèle Weight Waypoint Mobility (WWM) afin d'adapter cette idée au cas particulier des campus universitaires. La nouveauté de cet article est de faire varier le poids des arcs suivant l'heure. Par exemple, le poids des arcs entrant vers la cafétéria est augmenté pendant l'heure du déjeuner. Pour fixer ces poids ainsi que leurs fluctuations, les auteurs ont mené une étude sur les habitudes d'un échantillon de la population des étudiants.

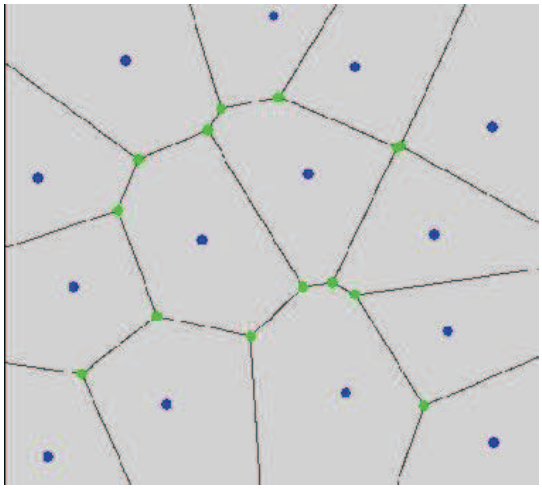


FIGURE 2.9: Décomposition de l'espace en diagramme de Voronoï

2.2.4.2 Contournement d'obstacles et diagrammes de Voronoï

Les diagrammes de Voronoï permettent de partitionner un espace. Ce partitionnement est déterminé par un ensemble fini de points appelés *germes*. Un élément e de cette partition, nommé *cellule*, associé avec un germe g est constitué des points de l'espace plus proches de g que de n'importe quel autre germe.

Définition : Soit S un ensemble de n germes de l'espace euclidien en dimension d. Pour chaque site p de S, la cellule de Voronoï $V(p)$ de p est l'ensemble des points de l'espace qui sont plus proches de p que de tous les autres sites de S. Le diagramme de Voronoï de $V(S)$ est la décomposition de l'espace formé par les cellules de Voronoï des sites.

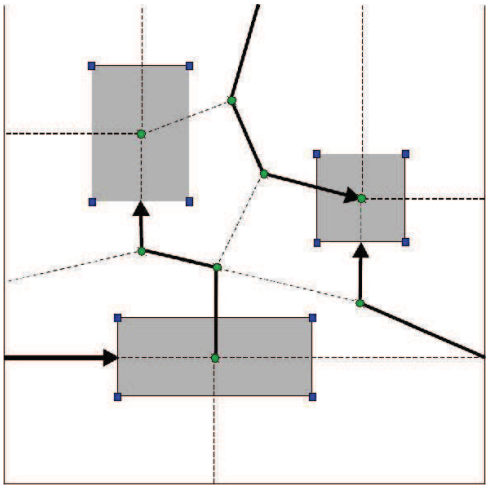


FIGURE 2.10: Utilisation des diagrammes de Voronoï pour le contournement d'obstacles. Les rectangles bleus représentent les germes du diagramme de Voronoï. Les cercles verts correspondent aux points de jonction entre plusieurs cellules [JBRAS03].

À partir de ces diagrammes, il est possible de construire un graphe $G_C(V_C,E_C)$ basé sur les frontières des cellules de Voronoï. Chaque sommet de l'ensemble V_C correspond alors à l'intersection entre plusieurs frontières de cellules. Les arêtes de l'ensemble E_C représentent quant à elles les frontières elles-mêmes.

Sur la figure 2.9, les sommets du graphe sont représentés par les points verts.

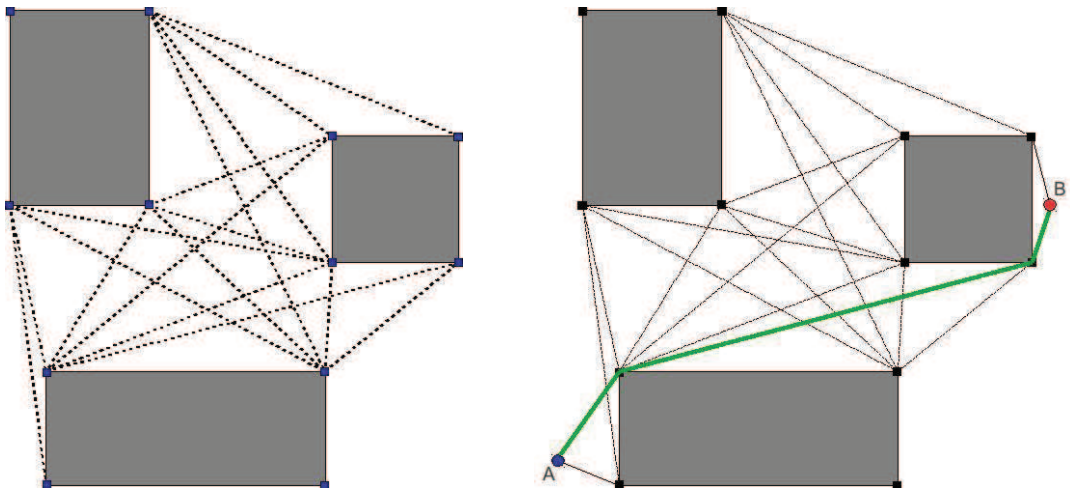
Plusieurs modèles [JBRAS03, JBRAS05] ont utilisé ce diagramme pour permettre aux mobiles de contourner les obstacles présents dans la zone de simulation. Pour cela, ils ont considéré les coins des obstacles comme germes du diagramme de Voronoï. Le graphe $G_C(V_C,E_C)$ construit à partir des cellules de ce diagramme leur permet ensuite de définir les chemins que les mobiles peuvent emprunter pour contourner les obstacles. La figure 2.10 représente un exemple de diagramme de Voronoï construit de cette manière. Notons que ce modèle n'exclut pas les mouvements à l'intérieur des bâtiments puisqu'il permet de définir des points d'entrée pour chaque obstacle qui doivent ensuite être ignorés ou non suivant la nature de l'obstacle.

2.2.4.3 Contournement d'obstacles et graphe de visibilité

Dans [LPW79] Lozano-Pérez et Wesley proposent un outil pour contourner les obstacles : le graphe de visibilité. Ce graphe, noté $G_{\text{vis}}(V_{\text{vis}},E_{\text{vis}})$ est construit de la manière suivante :

- chaque sommet $v \in V_{\text{vis}}$ modélise le coin d'un obstacle,
- Chaque arête $e \in E_{\text{vis}}$ relie deux sommets v_1 et v_2 du graphe si et seulement si les coins correspondant à v_1 et v_2 sont mutuellement visibles, deux sommets étant mutuellement visibles s'il n'existe aucun obstacle entre eux.

La figure 2.11a représente un exemple de graphe de visibilité construit à partir d'une zone de simulation dans laquelle les obstacles ont été disposés de la même manière que sur la figure 2.10.



(a) Construction du graphe de visibilité d'une zone de simulation

(b) Chemin permettant à un mobile d'atteindre sa destination en contournant les obstacles

FIGURE 2.11: Contournement d'obstacles à l'aide d'un graphe de visibilité

Pour trouver le chemin permettant à un mobile d'atteindre sa destination, deux nouveaux sommets sont ajoutés à ce graphe afin de modéliser la position actuelle du mobile ainsi que sa destination. L'ensemble E_{vis} est ensuite mis à jour afin d'intégrer ces nouveaux sommets au reste du graphe. Chaque arête se voit ensuite attribuer un poids correspondant à la distance euclidienne séparant les deux coins modélisés

par ses extrémités. En appliquant l'algorithme de Dijkstra [Dij59] il est alors possible de trouver le plus court chemin reliant la position actuelle du mobile à sa destination. La figure 2.11b représente le chemin trouvé à l'aide de cette méthode pour relier deux points, A et B, appartenant à la zone de simulation.

2.2.5 Modèles de déplacement humain

Pour obtenir un modèle de mobilité fidèle aux mouvements humains, de nombreux travaux ont analysé nos modes de déplacement. Ces analyses ont été réalisées soit :

- *à partir de données récupérées auprès des opérateurs de téléphonie mobile :* chaque téléphone mobile se trouve dans la zone de couverture d'une ou plusieurs stations de base. La localisation géographique de ces dernières est connue ainsi que leur portée approximative. La position d'un mobile peut donc être déterminée par triangulation. Ces traces permettent d'avoir une vision de la mobilité à l'échelle d'une ville, d'un pays voire à l'échelle mondiale. De plus, les individus tracés ne sont pas influencés par l'expérience puisqu'ils n'ont pas connaissance de la collecte de données. Celles-ci sont cependant parcellaires puisque la collecte de données ne se fait que lorsque le mobile est en communication,
- *à partir de traces collectées sur un ensemble de volontaires équipés de capteurs GPS :* le principe de fonctionnement d'un GPS est sensiblement le même que pour les mobiles. Ces capteurs sont capables d'identifier les satellites dont ils reçoivent le signal. En connaissant la position de ces satellites, il leur est alors possible de déterminer leur propre position géographique. Ces expériences se font généralement sur une échelle plus faible, de la taille d'un campus universitaire ou d'une ville. Le nombre d'individus tracés y est également plus faible mais les traces sont prises de manière beaucoup plus fréquente et la précision géographique y est plus importante que dans le cas de traces obtenues par des téléphones portables.

La plupart des travaux s'accordent à dire que de nombreuses caractéristiques du déplacement humain suivent une loi de puissance tels que :

- la vitesse [KKK06],
- le temps de pause [KKK06, RSH+08],
- la longueur des déplacements entre deux changements de direction [GHB08, BHG06, JYZ09, RSH+08],
- le temps entre deux rencontres successives de la même paire d'individus. [CHC+07, HCS+05, RSH+08].

Dans [RSH+08] les auteurs proposent un modèle de mobilité dont certaines caractéristiques du déplacement suivent une loi de puissance particulière : la loi de Lévy. Dans ce modèle, les mobiles se déplacent par étapes. Chaque étape est caractérisée par quatre paramètres :

- *la longueur de vol, l,* qui définit la distance que doit parcourir un mobile,
- *la direction, θ*,
- *le temps de vol, Δt_v*,

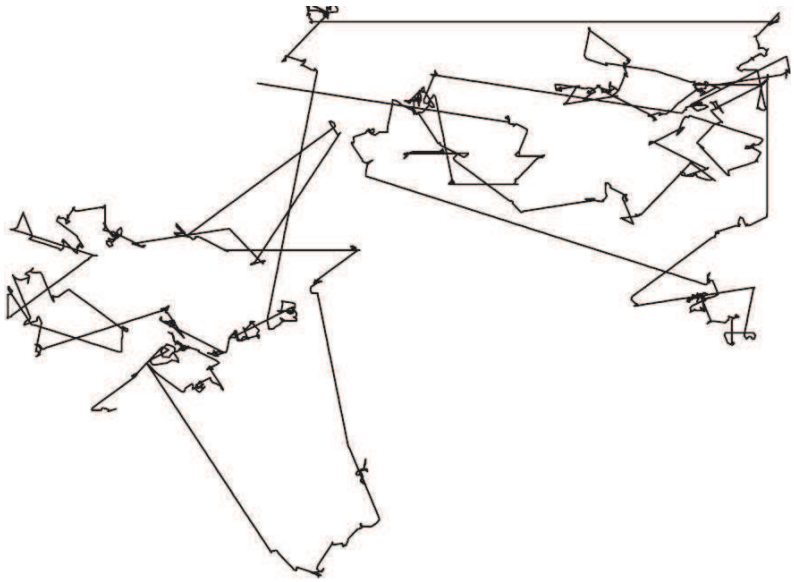


FIGURE 2.12: Exemple de trajectoire obtenu par le modèle de Lévy-walk [RSH+08].

– *le temps de pause,* Δt_p.

Au cours d'une étape, chaque mobile détermine la valeur de ces variables puis se déplace d'une longueur l dans la direction θ. Sa vitesse est fixée par le temps de déplacement Δt_v. Arrivé à destination, le mobile effectue une pause d'une durée Δt_p. La valeur de ces paramètres est choisie aléatoirement au début de chaque étape. La direction θ est choisie uniformément sur l'intervalle [0,2π[. La longueur de vol et le temps de pause suivent une approximation de la loi de Lévy et sont définis par :

$$l = \frac{1}{|x|^{1+\alpha}} \tag{2.5}$$

$$\Delta t_p = \frac{1}{|x|^{1+\beta}} \tag{2.6}$$

α et β sont deux constantes prises dans l'intervalle]0,2[. Le temps de vol est choisi aléatoirement suivant la formule :

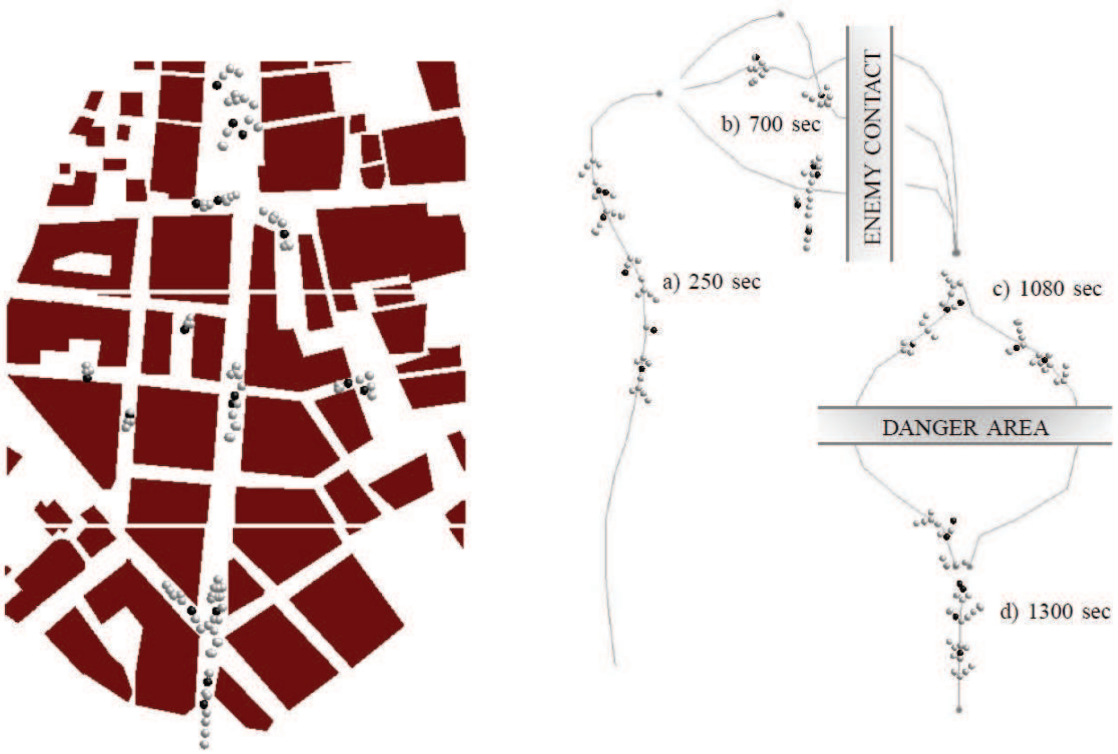
$$\Delta t_v = k l^{1-\rho} \tag{2.7}$$

où k et ρ sont deux constantes de simulation. La constante k représente le facteur d'échelle entre la longueur et le temps de vol. La constante ρ, avec $0 \leq \rho \leq 1$, permet de définir le niveau de dépendance du temps de vol par rapport à la longueur de vol. Lorsque la valeur de ρ est nulle, l et Δt_v sont proportionnels, la vitesse étant constante. Lorsque ρ vaut 1, le temps de déplacement est constant et la vitesse est alors proportionnelle à la distance à parcourir. Dans ce même article, [RSH+08], les auteurs ont comparé les traces issues de situations réelles et les traces obtenues grâce à leur modèle de mobilité. Les résultats ont montré qu'en fixant les constantes à $k = 18.72$ et $r = 0.79$ pour des déplacement inférieurs

à 500 m et $k = 1.37$, $r = 0.79$ dans les autres cas, le modèle est fidèle aux traces du déplacement humain.

2.3 Les modèles composites

La dépendance temporelle ou spatiale, la gestion des contraintes géographiques ou encore les spécificités du déplacement humain représentent autant de caractéristiques du déplacement. Dans la plupart des modèles proposés, un seul de ces aspects est généralement pris en compte. Or, pour simuler une situation concrète telle que le déplacement de véhicules sur autoroute ou l'intervention d'un groupe de secouristes, il est nécessaire de gérer plusieurs aspects de la mobilité en même temps. Pour cette raison, certains travaux proposent des solutions hybrides faisant souvent interagir plusieurs modèles de mobilité existants.



(a) Intervention de militaire en zone urbaine [RW07].

(b) Intervention de militaire en zone dégagé [RW07].

FIGURE 2.13: Exemples de zones d'interventions utilisés par le modèle présenté dans [RW07].

Déplacement de militaires : dans [RW07] les auteurs proposent un protocole de communication pour des troupes militaires. Dans les simulations réalisées, le déplacement des unités suit un modèle composite qui tient compte :

- *de la dépendance spatiale :* les unités mobiles sont divisées en plusieurs groupes de tailles différentes capables de se scinder et de se regrouper au cours de la simulation.

– *des contraintes géographiques :* deux scénarios sont étudiés, l'intervention en zone urbaine et en zone dégagée. Dans le premier scénario, illustré par la figure 2.13a, les bâtiments sont pris en compte à la fois pour le déplacement des unités mais également pour la qualité de la transmission radio. En effet, lorsqu'un signal traverse un obstacle, une partie de son énergie est absorbée. Tenir compte des obstacles permet donc d'avoir une meilleur approximation de la portée radio d'un signal. Dans le second scénario, aucun obstacle n'est présent dans la zone de simulation cependant celle-ci est découpée en plusieurs zones (zone de danger, de contact avec l'ennemi) qu'il faut éviter ou au contraire aborder de manière spécifique. La figure 2.13b représente l'espace de simulation pour le scénario de l'intervention en zone rurale.

Intervention sur les lieux de catastrophes les auteurs de [AGPG+07] proposent un modèle adapté à ce type de mobilité basé sur leurs observations du déplacement des forces de la sécurité civile allemande lors d'exercices réels en vue de la préparation des Journées Mondiales de la Jeunesse 2005 et de la coupe du monde de football 2006. Dans ce modèle, la zone de simulation est divisée en sous-zones, chacune d'entre elles ayant un rôle particulier : centre d'opération, parking de départ des ambulances, zones d'intervention et de traitement des blessés. La figure 2.14 représente cette division de la zone d'intervention.

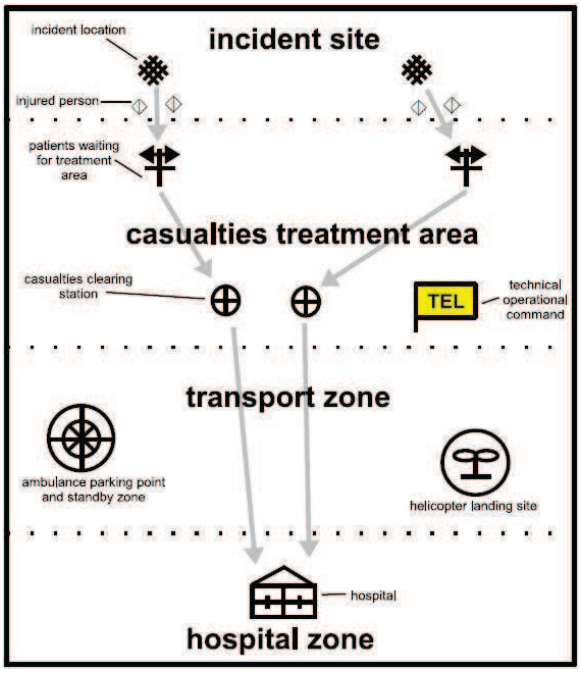


FIGURE 2.14: Division de la zone d'intervention par les forces de la sécurité civile [AGPG+07]

Les agents mobiles sont également répartis en différentes catégories (équipes de sauvetage, brigades de pompiers, etc.) afin de gérer les spécifications de chacune d'entre elles (vitesse, zone d'intervention, etc.). L'évitement d'obstacle est réalisé à l'aide d'un graphe de visibilité.

Un autre modèle a été proposé dans [PBDK09] pour l'intervention sur une scène de désastre : Le *Mission Critical Mobility* (MCM). Contrairement au modèle précédent, le MCM prend en compte la

dépendance spatiale en déplaçant les mobiles par groupe.

Dans le modèle MCM, les mobiles se déplacent donc en groupe selon un modèle équivalent au RPGM avec un déplacement global du groupe basé sur RWP. Les groupes sont ensuite répartis en deux catégories :

- *les groupes d'intervention d'urgence :* chargés de modéliser le déplacement de pompiers, de policiers ou de militaires, les groupes appartenant à cette catégorie se déplacent d'un site d'intervention à un autre. Lorsqu'un groupe arrive sur un site, il effectue une pause, d'une durée P choisie uniformément entre [P_{min},P_{max}]. Cette pause permet de représenter le temps d'intervention. Lorsqu'elle est terminée, le groupe choisit un nouveau site d'intervention et une nouvelle vitesse de déplacement. Chaque nouveau site est choisi uniformément parmi l'ensemble des points de la zone de simulation où aucun n'obstacle n'est présent. La vitesse est quant à elle choisie uniformément sur [0,V_{max}]. Pour que tous les membres d'un groupe ne se concentrent pas sur un seul point, chacun d'entre eux détermine sa propre destination. Celle-ci ne doit cependant pas être située à une distance supérieure à r_{max} du site d'intervention choisi.
- *les équipes médicales :* L'un des rôles de ces équipes est de rapatrier les blessés depuis les sites d'intervention jusqu'à un centre où ils seront soignés. Pour cela, les mobiles appartenant à cette catégorie doivent revenir à un point particulier de la zone de simulation entre chaque intervention. Le choix des sites et de la vitesse de déplacement suit le même processus que pour les groupes d'intervention d'urgence.

Tout comme pour le modèle [AGPG+07], le MCM utilise un graphe de visibilité pour permettre aux mobiles de contourner les obstacles situés sur la zone de simulation.

À chaque fois qu'un nouveau site d'intervention est choisi, le modèle lui attribue un niveau de priorité qui peut prendre trois états :

- *Normale :* aucune action spécifique n'est nécessaire, le groupe se rend simplement sur le site d'intervention,
- *Sérieux :* lorsqu'un groupe doit intervenir sur un tel site, il fait appel à une équipe de renfort de même catégorie que lui.
- *Complexe :* le groupe fait appel à une équipe de chaque catégorie pour venir l'assister.

2.4 Conclusion

De nombreux modèles de mobilité ont été proposés afin de représenter le déplacement de mobiles. Dans les premières solutions, la trajectoire des mobiles est purement aléatoire. Cette approche relativement simple présente cependant un certain nombre de propriétés indésirables mises en évidence par plusieurs études. Certains travaux ont cherché à représenter des mouvements plus proches de la réalité. Ils ont pour cela défini des modèles capables de prendre en compte des caractéristiques particulières telles que les dépendances spatiale et temporelle ou encore la gestion des contraintes géographiques. Quelques études ont abordé le problème sous un autre angle en examinant des traces de mobilité humaine. Leurs travaux

ont permis de démontrer que certaines propriétés de ce déplacement pouvaient être modélisées par une loi de puissance.

Enfin, pour représenter des mouvements plus complexes, certains chercheurs ont proposé des modèles composites qui définissent la trajectoire des mobiles à partir de plusieurs modèles existants.

Chapitre 3

Composite Mobility Model for Disaster Area (C2MDA)

3.1 Introduction

L'état de l'art sur les modèles de mobilité réalisé au chapitre précédent montre que les modèles proposés jusqu'à présent permettent de représenter un grand nombre de caractéristiques du déplacement tels que les changements progressifs de vitesse et de direction, le déplacement en groupe ou la gestion des obstacles. Pour décrire les déplacements plus complexes, comme par exemple celui des militaires ou des équipes de secouristes, plusieurs travaux ont proposé des modèles composites construits à partir de plusieurs modèles existants.

A notre connaissance, tous les modèles composites font intervenir le RWP pour définir la trajectoire des mobiles. Dans le chapitre précédent, nous avons vu que de nombreux travaux ont étudié ce modèle et ont mis en évidence certaines caractéristiques non souhaitables de ce modèle telles que la concentration des mobiles au milieu de la zone de simulation ou l'oscillation du nombre de voisins.

Pour éviter de reproduire ces caractéristiques indésirables, nous avons décidé de concevoir un nouveau modèle de mobilité adapté au déplacement des forces de la sécurité civile et qui ne fait à aucun moment intervenir le RWP ou tout autre modèle purement aléatoire.

3.2 Le modèle C2MDA

Ce chapitre présente notre modèle composite : le *Composite Mobility Model For Disaster Area* (C2MDA) [PTV10, PTV11]. Ce modèle permet de générer un déplacement des unités proche de celui des forces de la sécurité civile. Le C2MDA intègre pour cela trois composantes caractéristiques du mouvement de ce type d'agents mobiles : un déplacement de groupe, une gestion des obstacles présents sur la zone d'intervention et un mouvement reproduisant les principales propriétés du déplacement humain.

3.2.1 Gestion des déplacements

Pour que le mouvement des unités soit le plus réaliste possible, le modèle C2MDA permet de regrouper les agents en équipes de taille variable. Le déplacement de ces équipes se fait à l'aide du modèle RPGM décrit dans le chapitre précédent. Dans ce modèle, chaque groupe est composé d'un chef de groupe et de plusieurs membres. Le déplacement du chef définit le déplacement général du groupe. Les membres doivent quant à eux suivre le mouvement de leur chef mais avec un certain degré de liberté. Dans l'article d'origine, les auteurs proposent d'utiliser le RWP pour le déplacement du leader. Dans le C2MDA, nous préférons utiliser un modèle plus proche du déplacement humain : le modèle de Lévy-walk.

Ce modèle présenté également au chapitre précédent s'appuie sur de nombreuses études qui ont montré que certaines caractéristiques de la mobilité humaine suivent une loi de puissance. Dans le cas particulier de la longueur des déplacements, respecter une loi de puissance signifie que les mobiles effectuent de nombreux déplacements de faible amplitude et de longs trajets considérablement plus rares. Les études [GHB08, BHG06, JYZ09, RSH+08, CHC+07, HCS+05, KKK06] n'ont pas été menées dans le cadre spécifique de l'intervention d'équipe de la protection civile. Cependant, ce modèle semble également approprié pour ce type de mobilité. En effet, le rôle de ces équipes consiste à se rendre sur un site, effectuer leur intervention puis se rendre sur le site suivant. Le mouvement de ces unités est donc constitué de nombreux déplacements de faible amplitude réalisés lorsqu'ils interviennent sur un site mais également de trajets beaucoup plus longs lorsque ces équipes passent d'un site à un autre. L'utilisation d'une loi de puissance pour modéliser le déplacement de ces agents est donc appropriée.

La modélisation de la vitesse dans Lévy-Walk correspond également aux déplacements réels des unités d'intervention. Nous venons de voir que les déplacements de faible amplitude pouvaient modéliser l'intervention des agents sur un site. Dans ce cas, ces mobiles se déplacent généralement à pied. Au contraire, lorsque ces agents effectuent de longs trajets afin de changer de site, ils ont tendance à utiliser un véhicule pour leur déplacement. Le mouvement de ces unités est alors beaucoup plus rapide. Une corrélation entre vitesse et longueur de déplacement tel que le propose le Lévy-Walk est donc également adapté au type de déplacement étudié.

Grâce à ces deux modèles, RPGM et Lévy-walk, le C2MDA prend en compte un déplacement en groupe, adapté aux spécificités du mouvement des équipes de secouristes. Ces modèles ne tiennent cependant pas compte de la présence d'obstacles dans la zone de simulation. C'est pour cette raison que le C2MDA intègre une autre composante afin de tenir compte de ces contraintes géographiques.

3.3 Gestion des obstacles dans C2MDA

En cas de catastrophe, le lieu d'intervention des équipes de sauvetage peut contenir de nombreux obstacles tels que des débris ou des bâtiments détruits que les mobiles doivent éviter. La plupart des modèles composites utilisent un graphe de visibilité, présenté dans le chapitre précédent, pour établir les routes permettant de les contourner.

Les trajectoires obtenues par ce graphe ont la particularité de passer au plus proche des obstacles ce qui permet d'obtenir un trajet qui minimise la distance parcourue. Cependant, dans le cas d'une scène de catastrophe, les obstacles peuvent être en feu, instables, ou présentant un risque élevé d'explosion. Lors d'interventions dans un tel environnement, les équipes d'intervention cherchent à conserver une distance de sécurité lorsqu'ils les contournent même si cela rallonge leur trajet. Le modèle C2MDA n'utilise donc pas de graphe de visibilité mais un diagramme de Voronoï pour tracer le chemin des secouristes en évitant les obstacles. Contrairement aux graphes de visibilité, les routes fournies par un diagramme de Voronoï permettent en effet de passer à mi-chemin entre les obstacles, augmentant ainsi la distance entre les agents mobiles et ces objets potentiellement dangereux.

Traditionnellement les coins des obstacles servent de germes aux cellules de Voronoï. Cependant, il peut arriver que cette approche ne fournisse pas suffisamment de routes pour pouvoir contourner les obstacles. Ce problème survient notamment lorsque des obstacles de forme allongée se trouvent à proximité les uns des autres. La figure 3.1 présente un exemple de construction classique du diagramme de Voronoï où ce problème survient. Les frontières non utilisables ont été représentées en rouge tandis que celles pouvant servir de route ont été marquées en noir. Il est ainsi possible d'observer que sur cet exemple, les arcs noirs sont insuffisants pour contourner les obstacles.

Ce problème est dû à un nombre insuffisant de points de référence permettant de définir la position des bâtiments. Pour résoudre ce problème, nous avons apporté les modifications suivantes à cette construction du diagramme de Voronoï.

Tout d'abord, de nouveaux germes ont été ajoutés sur les obstacles. Ces derniers correspondent à la projection orthogonale des coins de chaque obstacle sur les arêtes de tous les autres. Cette première modification crée de nouveaux chemins permettant aux agents de circuler entre les obstacles. D'autres germes ont également été ajoutés sur la bordure de la zone de simulation. Chacun d'entre eux est la projection orthogonale des coins des obstacles sur ces bordures. Les nouvelles routes créées grâce à ces nouveaux germes permettent aux agents de passer entre les obstacles et la bordure de la zone de simulation.

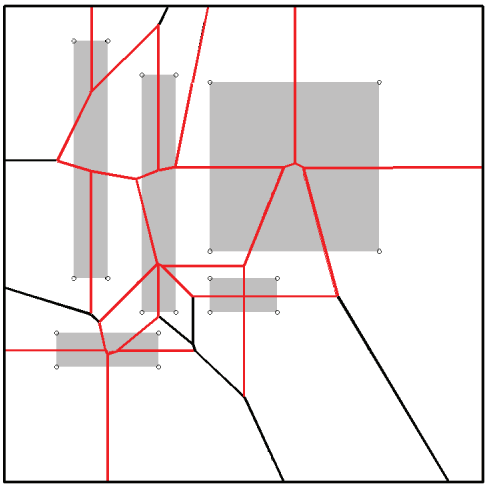


FIGURE 3.1: Construction classique du diagramme de Voronoï

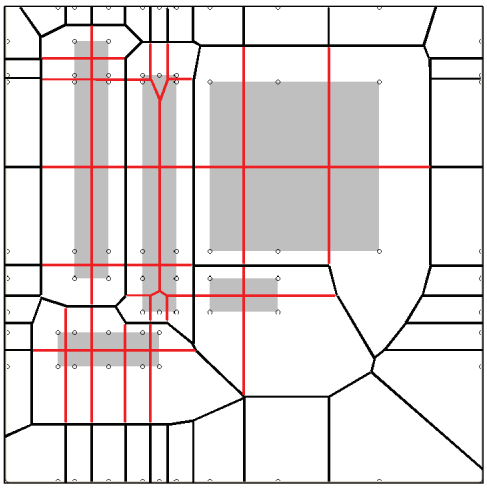


FIGURE 3.2: Diagramme de Voronoï avec points de référence supplémentaires

La figure 3.2 montre le diagramme de Voronoï obtenu grâce à notre approche. Le nombre de routes utilisables est cette fois suffisant pour contourner chaque obstacle de la zone de simulation. Par rapport à la carte générée par le graphe de visibilité, les routes proposées permettent aux agents de conserver une certaine distance avec les obstacles. En fait, chaque point de la route se trouve au centre de l'espace libre entre les différents bâtiments, augmentant ainsi l'espace entre les agents et les obstacles. A chaque

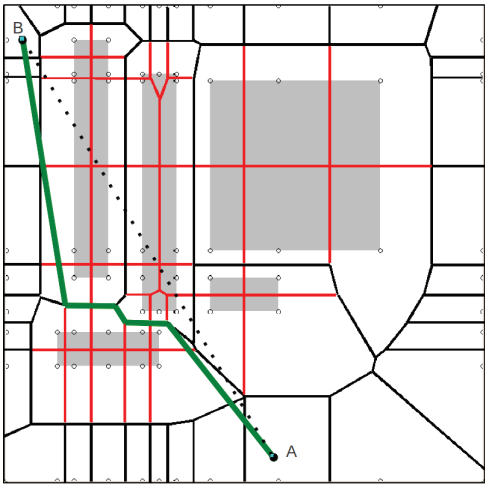


FIGURE 3.3: Chemin issu du diagramme de Voronoï, permettant à un mobile d'atteindre sa destination en contournant les obstacles

fois qu'un agent mobile doit atteindre une destination tout en contournant des obstacles, il va rechercher le plus court chemin entre sa position actuelle et sa destination en utilisant les routes offertes par le diagramme de Voronoï de la manière suivante. Un graphe pondéré $G(V,E)$ est défini de telle sorte que :

- chaque sommet $v \in V$ représente un point de croisement entre plusieurs cellules de Voronoï.
- lorsqu'il existe une bordure de cellule de Voronoï entre deux points de croisements p_1 et p_2, leurs sommets associés sont reliés par une arête $(v_1,v_2) \in$ E. le poids de cette arête correspond alors à la distance euclidienne entre les points p_1 et p_2.

Deux sommets sont ensuite ajoutés. Le premier correspond à la position actuelle de l'agent, le second à sa destination. De nouveaux arcs sont également créés entre chacun de ces nouveaux nœuds et tout autre sommet du graphe si et seulement si ces nouvelles arêtes ne croisent aucun obstacle. L'algorithme de Dijkstra [Dij59] est ensuite appliqué sur ce graphe afin de trouver le plus court chemin entre la position actuelle du mobile et sa destination. La figure 3.3 montre un exemple de chemin qui peut être choisi par un agent positionné au point A et qui souhaite se rendre au point B.

Ainsi, en définissant de nouveaux points de référence sur les bâtiments, nous arrivons à obtenir suffisamment de chemins pour pouvoir contourner les obstacles présents dans la zone de simulation quelle que soit leur position.

3.4 Évaluation du C2MDA

Nous avons évalué les performances de notre modèle afin de répondre à deux questions :

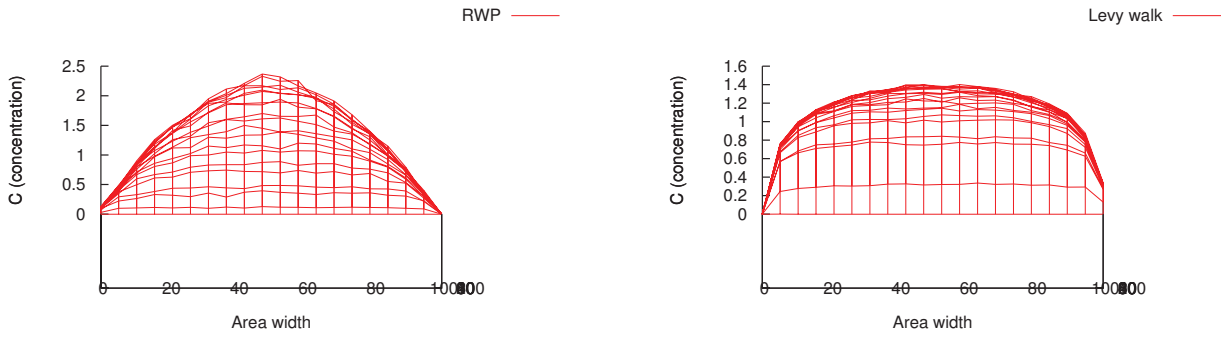
1. Le modèle Lévy-walk a-t-il tendance, comme le RWP, à concentrer les mobiles au centre de la zone de simulation ?
2. Le C2MDA étant un modèle composite, permet-il de conserver les caractéristiques de chacun des modèles qui le compose ?

3.4.1 Comparaison de la distribution spatiale des nœuds du RWP et du Lévy-walk

Les travaux qui se sont intéressés au RWP ont montré que ce modèle avait tendance à concentrer les mobiles dans le centre de la zone de simulation. Cette propriété engendre plusieurs problèmes tels qu'une oscillation du nombre de voisins. Dans le C2MDA, le déplacement des mobiles en l'absence d'obstacle est principalement défini par le Lévy-walk. Aucune étude ne s'étant penchée sur la distribution des mobiles dont la trajectoire suit une distribution de Lévy, nous avons décidé de l'évaluer et de la comparer au RWP dans les mêmes conditions de simulations.

Dans ces simulations, les paramètres du Lévy-walk ont été fixés à $\alpha = 1.5$ et $\beta = 1$ qui correspondent aux valeurs utilisées dans [RSH+08] pour reproduire les caractéristiques obtenues par les traces de déplacement. Dans ce même article, les auteurs proposent des valeurs de k et r qui permettent de modéliser au mieux le déplacement humain. Nous avons donc suivi ces recommandations et fixé ces constantes à $k = 18.72$ et $r = 0.79$ pour des déplacements inférieurs à 500 m et $k = 1.37$, $r = 0.79$ dans les autres cas.

L'ensemble des simulations a été réalisé sur une zone de 800m×800m avec 40 mobiles. Pour normaliser les résultats, nous avons introduit une mesure permettant de comparer la distribution de ces modèles par rapport à une distribution uniforme des mobiles sur la zone de simulation. La zone de simulation a ainsi été découpée en une multitude de cases élémentaires et le nombre de passages des mobiles dans chacune de ces cases a été comptabilisé et noté NA. Celui-ci est comparé au nombre de passages qu'aurait effectué



(a) Concentration des nœuds sur la zone de simulation pour le modèle RWP

(b) Concentration des nœuds sur la zone de simulation pour le modèle Lévy-walk

FIGURE 3.4: Répartition des nœuds sur la zone de simulation pour les modèles RWP et Lévy-walk

ces mobiles avec une distribution uniforme noté U. Nous obtenons ainsi une métrique sur la concentration appelée C et définie par :

$$C = \frac{NA}{U}. \tag{3.1}$$

Une valeur de C supérieure à 1 signifie que la concentration des mobiles est plus élevée que la moyenne.

La figure 3.4 présente les résultats de ces simulations. La courbe 3.4a, correspondant au RWP, montre, comme nous pouvions nous y attendre, une forte concentration au centre de la zone de simulation qui est au maximum 2,5 fois plus élevée qu'une distribution uniforme. Pour le Lévy-walk, la distribution obtenue montre une bien meilleure répartition. En effet, la concentration est supérieure à la normale sur le centre mais elle ne dépasse jamais les 40%. Cette différence est provoquée en grande partie par les effets de bord qui limitent le passage des nœuds aux extrémités de la zone de simulation. Par compensation, la concentration dans la zone centrale est donc plus élevée mais dans des proportions bien moindre que pour le RWP. Ainsi, contrairement au RWP, le modèle Lévy-Walk ne concentre pas les mobiles dans le centre de la zone de simulation évitant ainsi les problèmes d'oscillation du nombre de voisins liés à ce phénomène.

3.4.2 Impact du C2MDA sur la taille du voisinage des mobiles pour des équipes de taille moyenne

Pour vérifier que notre modèle conserve les caractéristiques des modèles qui le composent, nous nous sommes intéressés à l'impact des différents modèles (RPGM, Lévy-walk et C2MDA) sur le nombre moyen de voisins que possède chaque agent. Nous avons également étudié le cas du RWP puisque ce modèle est utilisé dans le RPGM classique. Nous avons ainsi une double lecture des résultats : premièrement, les modèles qui utilisent un déplacement de groupe (RPGM, C2MDA) sont comparés aux autres (RWP et

Lévy-walk) ; deuxièmement, les modèles basés sur un modèle purement aléatoire (RWP et RPGM) sont comparés aux modèles basés sur un déplacement de Lévy-walk (Lévy-walk et C2MDA).

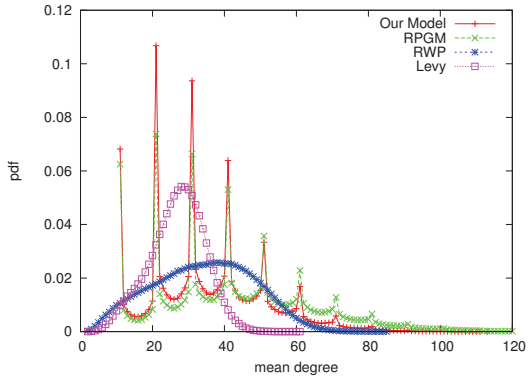


FIGURE 3.5: fonction de densité du degré des mobiles pour une taille de groupe de dix membres.

La figure 3.5 représente l'évolution de la densité de probabilité du nombre moyen de voisins pour les modèles RWP, Lévy-walk, RPGM et C2MDA. Les courbes du RWP et du RPGM montrent que lorsque les mobiles se déplacent en groupe, le degré des mobiles a tendance à prendre des valeurs spécifiques. Ces valeurs correspondent aux multiples de la taille des groupes ce qui peut s'expliquer de la manière suivante. Dans les modèles utilisant la gestion de groupe, lorsqu'un nœud rencontre un voisin appartenant à un autre groupe, il y a de fortes chances pour que ce nœud rencontre l'ensemble des membres du groupe. Les rencontres ne se font donc pas nœud par nœud mais groupe par groupe. Ceci explique ces fortes densités observées pour le modèle RPGM. La courbe de notre modèle présente les mêmes pics à intervalles réguliers que le RPGM. Le C2MDA a donc su conserver l'impact du déplacement de groupe sur la densité moyenne de voisins.

Cette même figure permet également d'observer que la courbe tracée selon le Lévy-walk est plus resserrée autour de la moyenne que celle du RWP. Cette propriété peut s'expliquer de la manière suivante. Les mobiles étant répartis de manière plus homogène avec le Lévy-Walk, les mobiles n'alternent pas entre des zones à forte densité (au centre de la zone de simulation) et des zones dont la densité d est beaucoup plus faible. L'amplitude de la variation du nombre de voisins est donc plus réduite dans le Lévy-Walk que dans le RWP. La distribution plus uniforme du Lévy-walk a ainsi tendance à limiter les fluctuations du nombre moyen de voisins. En ce qui concerne les courbes correspondants au RPGM et au C2MDA, les mêmes constatations peuvent être faites. Le C2MDA, basé sur le Lévy-walk, possède en effet une courbe plus recentrée sur sa moyenne que le RPGM qui est lui basé sur le RWP. Le C2MDA a ainsi su conserver la caractéristique du Lévy-walk en ce qui concerne la densité moyenne des nœuds.

Le modèle C2MDA a donc préservé la signature des deux principaux modèles qui le composent, le RPGM et le Lévy-walk, en ce qui concerne la distribution du degré des nœuds.

3.4.3 Influence de la présence d'obstacles sur le degré des nœuds et les temps de contact dans le C2MDA

Lorsqu'un signal se propage, l'affaiblissement que celui-ci subit dépend du milieu traversé. L'air provoque ainsi une atténuation beaucoup plus faible que certains matériaux tel que le béton. Ainsi, la présence d'obstacles dans une zone de simulation n'influence pas uniquement la trajectoire des mobiles mais perturbe également les télécommunications entre les différents éléments du réseau. Nous avons donc étudié l'influence de ces obstacles à travers trois éléments liés à la communication des mobiles :

- *le degré des nœuds*
- *la durée des connexions* correspondant à l'intervalle de temps pendant lequel deux nœuds sont continuellement en capacité de communiquer.
- *la durée d'inter-contact* : pour un couple de nœuds la durée d'inter-contact correspond à l'intervalle de temps séparant deux connexions successives.

Pour identifier les mobiles capables de communiquer, nous avons posé l'hypothèse suivante. Chaque mobile possède une portée radio correspondant à la distance r à laquelle son signal peut être reçu par un autre élément du réseau. Cette portée est identique pour tous les mobiles et constante tout au long de la simulation. Pour tenir compte des obstacles, nous considérons également que la trajectoire entre deux mobiles doit être totalement dégagée pour que ces derniers puissent communiquer. Ainsi, deux mobiles peuvent communiquer si la distance qui les sépare est inférieure à la portée radio r et s'il n'existe pas d'obstacle entre ces deux mobiles.

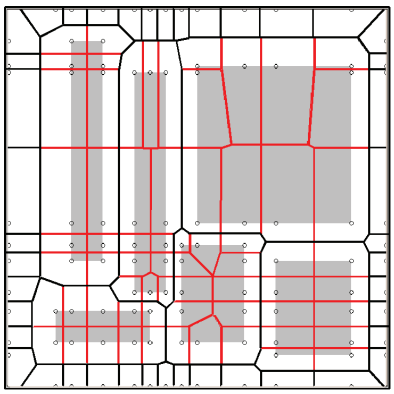


FIGURE 3.6: Position des obstacles lors des simulations

Les simulations ont été réalisées dans une zone de 600m×600m avec une portée radio de 175m et des groupes de dix membres. La position des obstacles sur la zone de simulation est présentée à la figure 3.6.

La figure 3.9 montre l'influence des obstacles sur la distribution du degré moyen des nœuds du réseau. Dans le cas d'une zone de simulation totalement dégagée, cette distribution s'étale de 0 à 200 nœuds. En présence d'obstacles, la probabilité pour qu'un nœud possède plus de 100 voisins est extrêmement faible. Les obstacles entraînent donc une forte diminution du degré moyen des nœuds.

L'étude des temps de connexion et d'inter-contact a été réalisée avec les mêmes paramètres de simulation. Les résultats obtenus sont représentés sur la figure 3.7 pour les temps de connexion et 3.8 pour

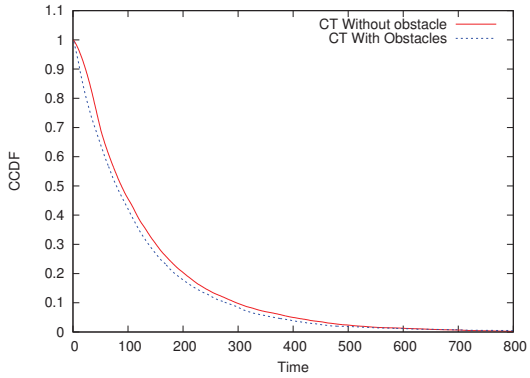


FIGURE 3.7: fonction de répartition complémentaire (CCDF) pour les durées de connexion

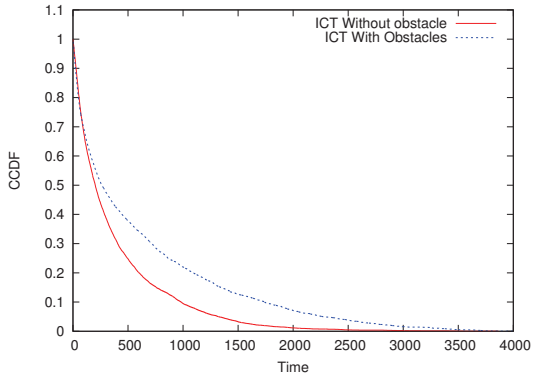


FIGURE 3.8: fonction de répartition complémentaire (CCDF) pour les durées d'inter-contact

les temps d'inter-contact. Ces figures montrent la fonction de répartition complémentaire de ces variables aléatoires (CCDF = 1-CDF). Cette fonction permet de représenter la probabilité que la durée de connexion ou d'interconnexion soit supérieure à une certaine valeur.

La figure 3.7 indique que les obstacles n'ont que peu d'influence sur les temps de connexion. Ainsi, l'absence d'obstacle ne permet d'augmenter que de 2% le nombre de mobiles qui peut maintenir une connexion supérieure à 200 secondes.

Dans le cas des temps d'inter-contact, cette différence est cette fois beaucoup plus marquée. La figure 3.8 permet en effet de constater qu'en l'absence d'obstacle, seuls 10% des mobiles doivent attendre plus de 1000 secondes entre deux rencontres. Lorsque des obstacles sont présents ce taux passe à 23% soit plus du double. Le temps d'interconnexion est donc beaucoup plus long en présence d'obstacles.

Cette figure 3.8 montre également que les durées d'inter-contact dans le modèle C2MDA suivent une loi de puissance ce qui est conforme à ce que plusieurs études portant sur le déplacement humain, tels que [CHC+07, HCS+05, RSH+08], ont pu observer.

La gestion des obstacles a donc une influence non négligeable sur les temps d'inter-contact et le nombre de voisins des agents mobiles. Notre modèle, en gérant la présence de ces contraintes géographiques, permet ainsi d'obtenir un déplacement assez réaliste. De plus, la forme en loi de puissance des temps d'inter-contact montre bien que le C2MDA conserve les caractéristiques du déplacement humain.

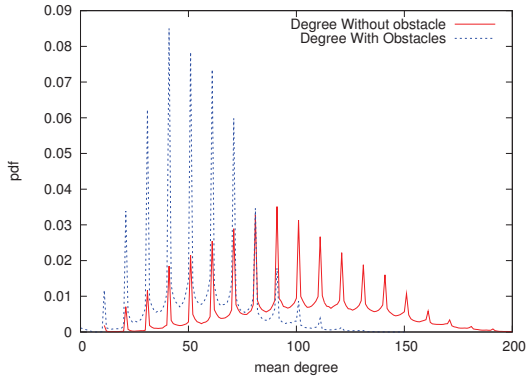


FIGURE 3.9: Influence des obstacles sur le degré moyen des nœuds

3.5 Conclusion

Les différents modèles de mobilité composites proposés jusqu'à présent utilisent le RWP ou une de ses variantes pour représenter le déplacement d'agents mobiles. Les limites de ce modèle ayant été souligné dans divers travaux, nous proposons un nouveau modèle qui s'affranchit totalement du RWP : le C2MDA qui fait l'objet de ce chapitre.

Le C2MDA est construit à partir de plusieurs modèles existants afin de reproduire certaines caractéristiques du déplacement des équipes de la sécurité civile lorsqu'ils interviennent sur une scène de catastrophe. Le RPGM est ainsi utilisé pour modéliser le déplacement de groupe, Le Lévy-walk pour ses caractéristiques proches de la mobilité humaine et les diagrammes de Voronoï permettent de contourner les obstacles présents dans la zone de simulation.

Les modèles RPGM et Lévy-walk ont été adaptés afin de pouvoir être intégrés au C2MDA. Pour les diagrammes de Voronoï, nous avons montré que les approches existantes ne permettaient pas toujours de contourner les obstacles. Nous avons donc proposé d'améliorer ces solutions en ajoutant de nouveaux points de référence, permettant ainsi de circuler entre les obstacles mais également de les contourner.

L'évaluation de notre modèle montre que les caractéristiques du déplacement des agents conservent les propriétés réalistes du modèle Lévy-walk et du déplacement de groupe du RPGM tout en intégrant l'influence des obstacles gérés par les diagrammes de Voronoï. Les résultats ont également montré que le modèle de Lévy-walk permettait une meilleure distribution spatiale des mobiles sur la zone de simulation.

Deuxième partie

Allocation distribuée de ressources radio

Chapitre 4

État de l'art

4.1 Introduction

Dans les réseaux de communications, pour qu'un émetteur puisse envoyer des informations à un récepteur, un support de transmission, appelé medium, doit relier ces deux équipements. Dans les réseaux filaires, ce support est généralement constitué de câbles de cuivre ou de fibre optique. La nature du support permet alors une propagation unidirectionnelle du signal, depuis l'émetteur jusqu'au récepteur. Tout équipement situé à proximité mais n'étant pas directement relié à ce support ne pourra donc pas recevoir le signal. Dans les réseaux sans fil, lorsqu'un équipement souhaite transmettre un signal, il utilise le support hertzien. Cette fois-ci, le signal se propage dans toutes les directions de l'espace permettant ainsi à chaque équipement situé à proximité de l'émetteur de recevoir son signal.

Pour que la transmission puisse s'effectuer correctement, il est important que l'émetteur ait un accès exclusif au support de transmission. Si deux signaux sont émis simultanément, ces derniers risquent d'interférer entre eux, empêchant tout récepteur de pouvoir les interpréter correctement. L'accès au support doit donc être contrôlé par un mécanisme capable d'empêcher les transmissions simultanées tout en permettant aux équipements qui souhaitent communiquer de pouvoir accéder au support [SWB09].

4.1.1 Accès par contention

La majorité des protocoles de contrôle d'accès utilisent une approche probabiliste à travers des mécanismes par *contention* généralement basés sur le CSMA (*Carrier Sense Multiple Access*) dont le principe est le suivant [80207]. Chaque équipement qui souhaite transmettre écoute le support de transmission. Si celui-ci n'est pas utilisé, cet équipement transmet son message. Si ce n'est pas le cas, il continue à écouter le canal jusqu'à ce qu'il soit libre. À ce moment, il choisit aléatoirement un intervalle de temps pendant lequel il va continuer d'attendre. Si, à l'issue de ce délai, le canal est toujours libre, il transmettra son message. Ce mécanisme sert à éviter que deux stations qui attendaient la libération du canal transmettent au même moment. Cependant, il est toujours possible qu'une telle situation se produise.

Les trames des deux stations entrent alors en collision et doivent être retransmis. Pour éviter qu'une nouvelle collision n'apparaisse lors de la retransmission, chaque émetteur attend également pendant une durée choisie aléatoirement avant de réémettre son message.

Lorsqu'un support est très sollicité, de nombreux équipements peuvent attendre l'accès. Dans ce cas, c'est le temps d'attente, choisi aléatoirement, qui détermine le prochain équipement qui pourra transmettre. Cet ordonnancement se faisant sur un critère aléatoire, chaque équipement possède la même probabilité d'accéder au canal. Ce type de mécanisme est censé permettre une répartition équitable de la bande passante entre les différents équipements d'une même zone [VGL08]. Cependant, de nombreux travaux ont montré que dans certains cas le partage de la bande passante pouvait être très inégal [DGL02, TC05].

4.1.2 Accès par réservation

Le contrôle d'accès par réservation consiste à diviser la bande passante offerte par le support en plusieurs canaux de communication puis à répartir ces canaux entre les différents équipements du réseau.

4.1.2.1 Division de la bande passante

L'accès au médium peut alors se faire en parallèle sur les différents canaux disponibles. Pour obtenir ces canaux, la bande passante peut être divisée :

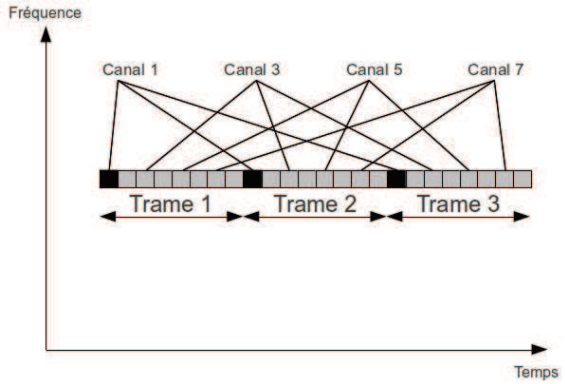


FIGURE 4.1: Canaux de communications issus d'un découpage temporel.

- *en temps* : les ressources radio sont alors constituées d'un ensemble d'intervalles de temps. Ces intervalles sont regroupés sous forme de trames périodiques de taille constante. Chaque canal de communication correspond à la répétition d'un même intervalle de temps dans les différentes trames qui se succèdent. Grâce à ce découpage, il est possible d'obtenir N canaux différents où N représente le nombre d'intervalles de temps contenus dans chaque trame. La figure 4.2 présente un exemple de découpe temporelle.
- *en fréquence* : Le principe est de diviser la bande de fréquence disponible en un ensemble de bandes de largeur spectrale plus faible. Chacune de ces bandes constitue alors un canal de communication. Pour limiter les interférences entre canaux adjacents, chaque canal est séparé du voisin par une

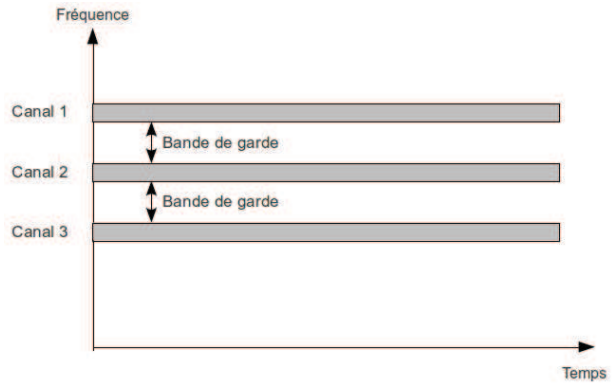


FIGURE 4.2: Canaux de communications issus d'un découpage fréquentielle.

bande de garde qui est inutilisée pour la transmission.

Certaines solutions hybrides emploient ces deux techniques simultanément. Les ressources issues de ce double découpage en temps et en fréquence sont alors appelées *Resource Blocks* [DS10].

4.1.2.2 Répartition des ressources

La répartition des canaux aux utilisateurs est ensuite réalisée à l'aide d'un algorithme d'*allocation de ressources*. Elle peut être *statique* auquel cas chaque utilisateur se voit attribuer un ou plusieurs canaux de manière permanente [RGC04, MDS10], ou *dynamique* lorsque l'attribution des canaux est fréquemment modifiée pour adapter l'allocation à certaines contraintes.

Allocation statique L'allocation statique est la plus simple et nécessite peu de temps de calcul. Dans sa forme la plus basique, cette allocation consiste à allouer une ressource à chaque équipement du réseau permettant ainsi d'obtenir une répartition uniforme des ressources. Cependant, la demande des différents équipements d'un réseau est rarement uniforme. Pour tenir compte de cet aspect, certains travaux ont proposé une allocation basée sur une estimation du trafic. Les auteurs de [RGC04] proposent ainsi d'allouer une quantité de ressources différente pour chaque équipement en fonction de la demande de ces derniers. Cependant, en raison de l'approche statique, cette solution ne permet pas de prendre en compte les fluctuations des demandes au cours du temps.

Allocation dynamique Certains travaux tels que [RC04, RC005] ont donc proposé une *allocation dynamique*. La répartition des canaux est alors recalculée régulièrement afin d'adapter la quantité de ressources allouée à la variation des demandes que ce soit en terme de débit ou de temps d'acheminement. Ces allocations nécessitent cependant plus de temps de calcul puisque l'allocation est réactualisée fréquemment.

allocation hybride Une stratégie hybride existe dans laquelle une partie des canaux est attribuée de manière permanente, tandis que l'autre est allouée à la demande. Cette approche, utilisée dans [RBAB06, KV05, KV06], permet de réaliser une allocation minimale pour par exemple garantir la connectivité du

réseau tout en conservant des ressources pour répondre aux fluctuations des demandes des différents utilisateurs.

Allocation dans les réseaux cellulaires Le problème de l'allocation de ressources dépend également de la nature du réseau considéré. Dans les réseaux cellulaires, l'espace est partitionné en zones élémentaires appelées *cellules*. Au centre de chaque cellule se trouve un équipement radio, appelé station de base servant d'interface entre un réseau d'infrastructure, et les mobiles situés dans sa zone de couverture.

Dans ce type de réseau, les communications ont lieu uniquement entre mobiles et station de base. L'allocation de ressources consiste alors à répartir les canaux entre chaque mobile et la station de base à laquelle il est rattaché. Cette répartition se fait généralement en deux temps [Wes02]. Tout d'abord, les canaux fréquentiels disponibles pour l'ensemble du réseau cellulaire sont répartis entre les différentes stations de bases. Le nombre de fréquences disponibles est cependant limité et ne permet pas aux stations de base d'attribuer un canal à chaque mobile qui souhaite communiquer. Pour pallier ce problème, ces canaux vont à nouveau être divisés mais cette fois en temps. Ces sous-canaux, issus d'un double découpage en temps et en fréquence sont ensuite distribués aux mobiles qui souhaitent communiquer avec la station de base.

Ces stations de base représentent une interface entre des équipements sans fil et des infrastructures tels que le réseau d'un opérateur téléphonique. Dans ce cas, les équipements sans fil, qui sont des téléphones mobiles, ont ainsi la possibilité de communiquer avec n'importe quel poste fixe ou mobile accessible depuis le réseau de l'opérateur. Cependant, les équipements sans fil d'un réseau cellulaire n'ont pas la possibilité de communiquer directement entre eux mais doivent obligatoirement passer par l'intermédiaire d'une station de base même lorsqu'ils sont à porté radio.

Allocation dans les réseaux ad hoc Pour s'affranchir de cette contrainte, de nouvelles solutions ont été proposées pour permettre à ces équipements radio de communiquer directement entre eux sans passer par un réseau d'infrastructure, formant ainsi un réseau dit *ad hoc*. Ces réseaux étant constitués uniquement de terminaux, leur déploiement est plus simple et plus rapide que celui d'un réseau cellulaire [PG10]. Cependant, l'absence totale d'infrastructure présente aussi certains inconvénients. Dans les réseaux cellulaires, certaines tâches telles que l'allocation de ressources ou le routage sont gérés de manière centralisée, soit par les stations de base, soit par un autre équipement du réseau d'infrastructure [PG10]. Dans les réseaux ad hoc, il n'existe aucun équipement de ce type, ce sont donc les terminaux qui doivent s'auto-organiser afin de répartir les ressources ou de trouver les routes permettant d'acheminer les messages [Haa01].

Dans ce contexte distribué, les algorithmes d'allocation doivent tout d'abord identifier les équipements susceptibles de communiquer et veiller à ce que les ressources attribuées ne puissent pas créer d'interférences. Une méthode simple pour y arriver consiste à attribuer chaque canal à un seul équipement du réseau. Cependant cette méthode n'utilise pas efficacement les ressources disponibles. En effet, puisque la puissance des signaux diminue avec la distance, il existe une distance au delà de laquelle deux

équipements peuvent utiliser le même canal sans interférer. Chaque canal peut donc être utilisé par plusieurs équipements à partir du moment où ces derniers sont suffisamment éloignés les uns des autres [GKJ07]. En gérant efficacement cette *réutilisation spatiale* des ressources, l'algorithme d'allocation peut minimiser le nombre total de ressources nécessaires pour réaliser sa tâche. Cette notion de réutilisation spatiale est représentée par la figure 4.3. Dans celle-ci chaque émetteur possède une zone d'interférence à l'intérieur de laquelle ses signaux sont suffisamment puissants pour générer des interférences. Cette zone est modélisée par un disque centré sur chaque émetteur. Dans cette figure, les zones d'interférence des émetteurs A, B et C se recoupent, ils doivent donc utiliser des canaux différents afin d'éviter tout risque d'interférence. Par contre les équipements A et de D possèdent des zones d'interférences totalement disjointes, ils peuvent utiliser la même ressource sans créer d'interférences.

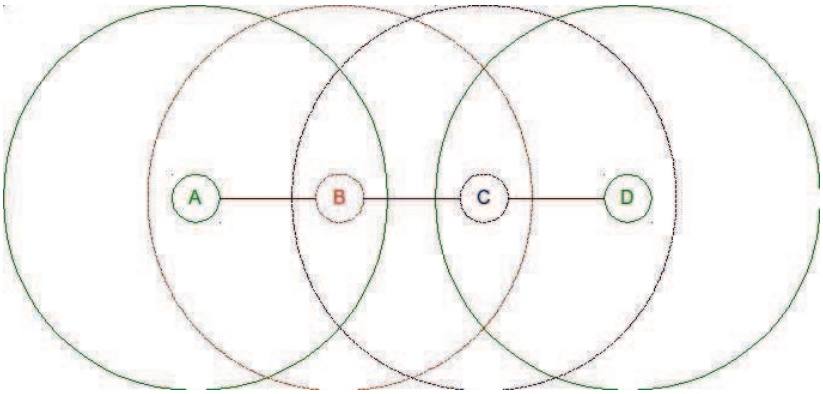


FIGURE 4.3: Réutisation spatiale des canaux de communication.

4.1.3 Conclusion

Un algorithme d'allocation de ressources doit donc :

- *identifier les équipements suffisamment proches géographiquement pour pouvoir communiquer,* afin que les ressources soient attribuées uniquement aux équipements qui peuvent les utiliser.
- *éviter de créer des interférences* en allouant des ressources différentes aux équipements suffisamment proches pour que leurs signaux interfèrent,
- *maximiser la réutilisation spatiale* en attribuant la même ressource aux équipements suffisamment éloignés pour que leurs signaux n'interfèrent pas.

Pour cela le réseau doit être modélisé afin d'identifier les équipements suffisamment proches pour communiquer ainsi que les éléments suffisamment éloignés pour ne pas créer d'interférences.

4.2 Modélisation du réseau

Pour qu'un récepteur puisse recevoir un signal de manière correcte, il faut que le rapport entre la puissance de ce signal et le niveau de bruit soit suffisamment élevé. Plus un récepteur est éloigné de l'émetteur, plus la puissance de réception est faible. A partir d'une certaine distance, le rapport signal

sur bruit (SNR, *Signal-to-Noise Ratio*) passe en dessous du seuil de détection et le signal reçu ne peut plus être correctement interprété. Il existe donc une zone autour des émetteurs, que nous appellerons *zone de transmission*, dans laquelle un nœud peut recevoir correctement un signal.

Cette zone de transmission permet donc d'identifier les équipements à portée radio. Lorsque cette zone est surévaluée, l'algorithme d'allocation attribue des canaux à des équipements qui sont trop éloignés pour communiquer, ces ressources sont alors perdues. À l'opposé, une sous-évaluation de la taille de cette zone empêche l'algorithme d'attribuer des ressources à des équipements qui peuvent en réalité communiquer. Il y aura donc moins de chemins possibles pour acheminer les messages entre deux équipements du réseau. Si le nombre de chemins était déjà limité, cette sous-évaluation peut entraîner la séparation du réseau en plusieurs composantes qui n'auront aucun moyen de communiquer entre-elles.

FIGURE 4.4: Interférence entre deux couples émetteur-récepteur

Lorsque deux couples émetteur-récepteur se trouvent trop proches l'un de l'autre, le signal de l'un des émetteurs peut interférer avec le signal transmis par l'autre émetteur. Prenons le cas de deux couples (A_T,A_R) et (B_T,B_R), avec A_T et B_T deux émetteurs, et A_R et B_R deux récepteurs tel que le représente la figure 4.4. Le signal de A_T s'adresse uniquement au récepteur A_R avec lequel il communique. Le récepteur B_R considère donc ce signal comme une perturbation qui vient s'ajouter au bruit ambiant. Plus A_T et B_R sont proches et plus le niveau de bruit détecté par ce dernier augmente, diminuant ainsi les capacités de B_R à recevoir correctement le signal de B_T. Il faut donc maintenir une distance minimale entre le récepteur d'un couple et les autres émetteurs afin que les signaux de deux émetteurs n'interfèrent pas. Il existe donc une zone autour de chaque émetteur dans laquelle les récepteurs des autres couples ne doivent pas se trouver. Nous appelons cette zone : *zone d'interférence*.

La connaissance de ces zones d'interférence permet à l'algorithme d'allocation de déterminer les équipements qui sont suffisamment éloignés pour utiliser la même ressource sans que leurs signaux n'interfèrent. Si elle est surévaluée, la réutilisation spatiale n'est pas optimale, les ressources sont donc sous-exploitées. Si elle est sous-évaluée, les équipements utilisant la même ressource sont trop proches et des interférences apparaissent. Certaines ressources allouées deviennent alors inutilisables.

Une mauvaise estimation de ces deux zones peut donc avoir des conséquences sur les performances des protocoles de communications. Les travaux existants proposent deux grandes approches pour modéliser ces zones[GH01]. La première modélise le réseau à l'aide de graphes [MDS10, SS00] en s'appuyant généralement sur les distances séparant les équipements. La seconde repose sur une détection physique du niveau de bruit et de la puissance d'émission des émetteurs [SCS99, NTN10].

4.2.1 Modélisation par graphe

Dans l'approche graphe, les nœuds à portée radio sont représentés à l'aide d'un *graphe de connectivité* qui peut être orienté si les travaux considèrent des liens unidirectionnels ou non-orientés dans le cas contraire. Ce graphe, que nous supposerons e, est noté $G_C(V_C,E_C)$ où V_C représente l'ensemble des nœuds du graphe et E_C l'ensemble de ses arêtes. Ces ensembles sont construits de la manière suivante :

- Chaque équipement du réseau est modélisé par un nœud v $\in V_C$.
- Lorsque deux équipements sont à portée radio, les nœuds correspondant à ces équipements sont reliés par une arête e $\in E_C$.

Pour déterminer l'ensemble E_C, les articles [MDS10, SGJ09, AGK08] supposent que chaque nœud a une zone de transmission régulière, en forme de disque dont le rayon r est identique pour l'ensemble des nœuds. L'ensemble E_C est alors défini par :

$$\forall (i,j) \in V_a^2, (i,j) \in E_C \text{ si et seulement si } distance(i,j) \leq r \tag{4.1}$$

Il suffit alors de calculer la distance euclidienne entre chaque paire de nœuds pour savoir s'ils sont à portée radio. Dans la pratique, la portée radio n'est pas la même dans tout le réseau notamment à cause du niveau de bruit qui peut fluctuer d'un endroit à l'autre. La présence d'obstacles peut également diminuer la portée radio dans certaines directions, ou au contraire l'augmenter en jouant le rôle de guide d'ondes. Une méthode plus réaliste consiste donc à utiliser le rapport signal sur bruit pour définir l'ensemble E_C [RL93, SS00]. Pour tout couple de nœuds $(i,j) \in V_C$, ce rapport signal-sur-bruit, noté $\Gamma_{i,j}$, est défini par :

$$\Gamma_{i,j} = \frac{\frac{P_i}{L_b(i,j)}}{N_R}, \tag{4.2}$$

où P_i représente la puissance d'émission des signaux de i, $L_b(i,j)$ représente le facteur d'atténuation entre les deux nœuds i et j et N_R représente la puissance du bruit au niveau du récepteur.

Lorsque $\Gamma_{i,j}$ est supérieur au seuil de détection, noté γ, alors un lien peut exister entre les nœuds i et j. L'ensemble E est donc défini par :

$$(i,j) \in E_C \text{ si et seulement si } \Gamma_{i,j} > \gamma \tag{4.3}$$

Pour modéliser les interférences, la plupart des travaux admettent généralement qu'elles peuvent être de deux types :

- *les interférences primaires :* elle se produisent lorsqu'un équipement doit réaliser plusieurs opérations d'émission/réception simultanément. Par exemple recevoir les signaux de deux émetteurs différents.
- *les interférences secondaires :* un récepteur se trouve dans la zone d'interférence d'un émetteur qui n'est pas celui avec lequel il communique.

Pour éviter ces interférences, l'allocation doit s'assurer que pour tout couple de liens $(i,j),(k,l)$ utilisant la même ressource :

- les nœuds i, j, k et l sont mutuellement distincts (interférences primaires)
- les liens (i,l) et (k,j) n'appartiennent pas à l'ensemble E_C (interférences secondaires)

4.2.2 Modélisation issue de la théorie de l'information

Dans cette approche, le rapport SNR est systématiquement utilisé pour déterminer les couples de nœuds pouvant communiquer. Ce rapport permet de définir l'ensemble C des nœuds à portée radio par :

$$C = \{(i,j) | \Gamma_{i,j} > \gamma_{SNR}\}. \tag{4.4}$$

La principale différence entre les deux approches, graphe et théorie de l'information, réside dans la gestion des interférences. L'approche graphe ne tient généralement compte que des émetteurs situés dans le voisinage d'un récepteur pour identifier les interférences. Cette approche ne prend donc en compte qu'une partie des émetteurs du réseau. L'approche issue de la théorie de l'information s'intéresse à l'ensemble des émetteurs du réseau pour déterminer si la réception d'un signal est interférée. Ainsi dans le cas d'un couple de nœuds (i,j) où i est l'émetteur et j le récepteur, les interférences subies par ce couple sont définies par :

$$I_K(i,j) = \sum_{(k_e,k_r) \in C \setminus (i,j)} \frac{P_{k_e}}{L_b(k_e,j)}, \tag{4.5}$$

où k_e et k_r sont respectivement l'émetteur et le récepteur du couple (k_e,k_r). $I_K(i,j)$ permet ainsi de calculer le niveau d'interférence du récepteur j pour le couple (i,j).

Pour tenir compte de ces interférences, l'approche de la théorie de l'information s'appuie sur le rapport signal-sur-interférence-plus-bruit (SINR) défini par :

$$\Gamma'_{i,j} = \frac{\frac{P_i}{L_b(i,j)}}{I_K(i,j) + N_R} \tag{4.6}$$

Le lien (i,j) est alors considéré interféré si $\Gamma'_{i,j} < \gamma_{SINR}$.

Ces deux approches ont donc des visions radicalement différentes. D'un coté une approche graphe ou la modélisation des interférences est simplifiée et limitée au voisinage. De l'autre, une approche théorie de l'information, qui est beaucoup plus réaliste, en prenant en compte l'ensemble des émetteurs du réseau pour calculer le niveau d'interférence de chaque nœud. Certains travaux [BR03, GH01], se sont intéressés à la comparaison de ces deux approches afin d'évaluer les forces et les faiblesses de chacun.

4.2.3 Importance de la modélisation des interférences

Dans [GH01], les auteurs proposent une comparaison entre les deux approches. Les résultats de cette étude indiquent que l'approche graphe est plus rapide et réactive mais indiquent également que de nombreuses interférences ne sont pas détectées. Les auteurs ont montré que la plupart des interférences restantes ne sont pas dues à un défaut de l'approche graphe, mais à l'hypothèse généralement admise selon laquelle les zones de transmission et d'interférence sont de tailles identiques. Ils font ainsi remarquer qu'un signal peut être trop faible pour être correctement reçu, mais encore suffisamment puissant pour interférer la réception d'un autre signal. Par conséquent, la zone d'interférence doit être considérée comme plus étendue que la zone de transmission.

À notre connaissance, seuls certains articles [FGR03, AEvdH09, BM10] sur le coloriage de graphe tiennent compte d'une zone d'interférence supérieure à la zone de transmission dans leurs travaux. Ce problème porte le nom de k-distance où k est la distance minimale en dessous de laquelle une couleur ne peut pas être réutilisée.

En tenant compte des remarques de cette étude, nous avons donc décidé de nous intéresser à l'approche graphe mais en faisant l'hypothèse d'une zone d'interférence deux fois plus étendue que la zone de transmission et ce, pour l'ensemble des travaux présentés dans cette thèse.

4.3 Allocation de ressources et modélisation par graphe

Le problème de l'allocation de ressources peut être traité comme un problème de coloration de graphe [Kub04]. La coloration de graphe consiste à attribuer une couleur à certains éléments du graphe. Une couleur représente une ressource : canal fréquentiel, intervalle de temps ou Ressource block. La répartition des couleurs peut se faire soit sur les nœuds soit sur les arêtes d'un graphe.

4.3.1 Coloration des nœuds

La coloration des nœuds consiste à attribuer une couleur à chaque sommet du graphe de manière à ce que deux sommets adjacents ne possèdent pas la même couleur. Ce type de coloriage modélise bien les problèmes d'allocation *broadcast* où chaque nœud communique avec l'ensemble de ses voisins. En effet, dans ce type d'allocation lorsqu'un nœud communique, l'ensemble de ses voisins doit être prêt à recevoir son signal, ils ne doivent donc pas utiliser le même canal que lui.

4.3.2 Coloration des liens

Il peut arriver que les équipements ne souhaitent communiquer qu'avec un seul de leurs voisins. Dans ce cas, les ressources doivent être attribuées au lien existant entre cet émetteur et son unique récepteur. La coloration des arêtes est alors la modélisation la plus adaptée ; elle consiste, par exemple, à attribuer une ressource à chaque arête du graphe de telle sorte que deux arêtes adjacentes ne possèdent pas la même couleur.

4.3.3 Unification des deux approches

Des tentatives d'unification de ces deux méthodes d'allocations ont été proposées, tels que [Ram97] présenté plus loin, l'objectif étant de définir un modèle capable de fournir des algorithmes pouvant être utilisés aussi bien pour l'allocation lien que pour l'allocation sur les nœuds.

Dans l'article [Ram97] Ramanathan propose un cadre très étendu au problème de l'allocation quelle que soit la nature des ressources (fréquence ou intervalle de temps) et capable de gérer une allocation sur les nœuds comme sur les liens. Pour cela, il définit un ensemble de onze contraintes élémentaires sur les allocations. Chacune de ces contraintes impose des restrictions notamment sur la distance minimale entre deux éléments utilisant la même ressource. En combinant ces contraintes, Ramanathan a montré qu'il était possible de modéliser des problèmes aussi variés que :

- l'allocation de fréquence dans les réseaux cellulaires
- l'allocation de ressources temporelles et/ou fréquentielles sur les nœuds
- l'allocation de ressources temporelles et/ou fréquentielles sur les liens unidirectionnels
- l'allocation de ressources temporelles et/ou fréquentielles sur les liens bidirectionnels
- l'allocation de ressources temporelles et/ou fréquentielles sur les liens unidirectionnels avec de multiples antennes directionnelles.

Dans ce même article, l'auteur propose trois algorithmes issus de ce cadre générique. Ils sont centralisés et fonctionnent en deux phases :

- *un étiquetage* pendant lequel chaque nœud reçoit une étiquette unique comprise entre 1 et N où N représente le nombre de nœuds du graphe. Cette étape permet de classer les nœuds avant de leur attribuer une couleur,
- *un coloriage* pendant lequel chaque nœud, classé par ordre décroissant d'étiquette, reçoit une couleur. La couleur choisie est celle qui respecte l'ensemble des contraintes élémentaires fixées. Lorsque plusieurs couleurs peuvent être attribuées, la couleur retenue sera celle qui possède l'identifiant le plus petit.

La phase d'étiquetage est la plus complexe, car c'est elle qui introduit l'ordre dans lequel les ressources vont être distribuées aux nœuds. Ramanathan a donc proposé trois approches différentes pour cette phase avant de comparer leurs performances par simulations pour un réseau sans-fil. Ces stratégies qui ont donné leur nom aux algorithmes qui les implémentent sont :

- *RAND* : comme son nom l'indique cet algorithme répartit uniformément les labels sur les différents nœuds.
- *MNF* : Minimum Neighbor First. Dans cet algorithme les nœuds sont étiquetés par ordre de degré croissant. Les nœuds possédant le moins de voisins se voient donc attribuer les étiquettes les plus petites.
- *PMNF* : Progressive Minimum Neighbors First. Comme pour l'algorithme précédent, les nœuds sont étiquetés par ordre de degré croissant. Cependant, dans le PMNF, lorsqu'un nœud A reçoit une étiquette, il est ensuite supprimé du réseau pour le reste de l'étape d'étiquetage. Le degré des

voisins de A se voit alors diminuer. La classification des nœuds par degré décroissant change donc après chaque étiquetage, ce qui conduit à une solution différente du MNF.

L'approche aléatoire avait déjà été proposée précédemment [Hu93] tout comme l'ordonnancement des nœuds en fonction de leur degré [WP67] . Cependant, grâce à son cadre de travail unifié, Ramanhatan à pu comparer leurs performances. Néanmoins, tous ces algorithmes utilisent une approche centralisée, qui se prête mal aux réseaux ad-hoc. Nous allons donc voir les différentes solutions proposés pour ce type de réseau.

A notre connaissance, aucune adaptation du cadre unifié de Ramanathan n'a été proposée pour l'allocation de ressources dans un contexte distribué. Les travaux existants portent donc soit sur une allocation des nœuds soit une allocation des liens, mais n'offrent pas de solution commune à ces deux approches. Nous allons donc voir dans un premier temps les solutions spécifiques aux allocations sur les nœuds.

4.3.4 Allocation orientée nœud dans un contexte distribué

Dans [RWMX09], les auteurs proposent l'algorithme DRAND qui est une adaptation de l'algorithme RAND [Ram97] aux environnements distribués. Les nœuds du réseau ne sont pas synchronisés, cependant, ils exécutent l'algorithme à intervalle régulier. Il est ainsi possible de définir un intervalle de temps, appelé *ronde*, pendant lequel chaque nœud exécute son algorithme.

La solution proposé par DRAND est basé sur une loterie. Au cours de chaque ronde, les nœuds réalisent une épreuve de Bernouilli de probabilité $p = 0.5$. S'il gagne, il participe à une loterie dans laquelle ses chances de gagner sont de $1/k$ où k est le nombre de voisins à un et deux sauts qui n'ont pas encore de ressources attribuées. Les nœuds qui perdent cette loterie attendent la prochaine ronde. Lorsqu'un nœud A gagne, il envoie une requête à ses voisins afin de les prévenir qu'il est prêt à choisir une ressource. S'il existe au moins un voisin B de A qui a déjà reçu une telle demande de la part d'un autre de ses voisins, il rejette la demande de A. Dans le cas contraire, B informe le nœud A qu'il est prêt et lui transmet la liste des ressources utilisables. Si A reçoit un seul rejet, il annule la phase d'allocation. Si par contre, tous ses voisins sont libres, alors il choisit une des ressources dont la disponibilité est commune à chacun de ses voisins. Les auteurs de cet article ont prouvé que la complexité de cet algorithme était de $O(\delta)$ en temps, où δ représente la taille maximum du 2-voisinage dans l'ensemble du réseau.

Dans [HT04], les auteurs proposent un algorithme distribué et auto-stabilisant adapté au problème de l'allocation de ressources temporelles sur les nœuds. L'auto-stabilisation est la capacité d'un algorithme à pouvoir converger en un temps fini vers un état stable. Elle a été introduite pour la première fois par Dijkstra dans [Dij59]. Contrairement à la plupart des articles, les auteurs de [HT04] s'intéressent aux réseaux dont les équipements ne possèdent pas d'identifiant unique. Pour parvenir à attribuer une ressource à chaque nœud du réseau, l'algorithme proposé fonctionne en quatre phases :

– *Identification* : chaque nœud reçoit un identifiant unique dans le voisinage. Plusieurs nœuds du réseau peuvent donc posséder le même identifiant, mais ces nœuds ne peuvent pas être à moins de

trois sauts les uns des autres.

- *Élection de leader* : Plusieurs nœuds du réseau sont désignés comme *leaders*. Ces nœuds sont choisis de manière à ce que chaque nœud du graphe puisse être soit un *leader* soit le voisin d'un *leader*. Cette élection s'appuie sur l'identifiant attribué lors de la phase précédente. Plus un nœud possède un identifiant petit, plus sa priorité pour être *leader* sera grande. Lorsqu'un nœud possède plusieurs *leaders* dans son voisinage, il se rattache à celui d'identifiant le plus faible.
- *Attribution d'une couleur :* Chaque *leader* élu au cours de l'étape précédente est chargé d'attribuer une couleur à chacun de ses voisins. Afin de le faire, chaque nœud transmet la liste des couleurs utilisées dans son voisinage. Chaque *leader* a ainsi la connaissance de l'ensemble des couleurs utilisées dans son 2-voisinage et peut attribuer une couleur disponible à chacun de ses voisins dont il est le *leader*.
- *Attribution des intervalles de temps :* A partir du coloriage précédemment réalisé, chaque nœud va déterminer un ensemble d'intervalles pendant lesquelles ce dernier pourra émettre. Le temps total d'émission pour chaque nœud est calculé en fonction du nombre de couleurs utilisées dans le 2-voisinage.

Les auteurs de cet article ont prouvé de manière formelle la convergence de chacune de ces phases.

4.3.5 Allocation sur les liens dans un contexte distribué

Dans l'allocation sur les nœuds, chaque équipement avait la possibilité d'utiliser les canaux qui lui étaient attribués pour communiquer avec l'ensemble de ses voisins. Lorsque les communications ont lieu entre un émetteur et un unique récepteur les ressources seront beaucoup plus efficacement utilisées si elles sont attribuées uniquement à ce couple émetteur-récepteur. L'allocation se fait donc sur les liens du réseau. Chaque ressource peut alors servir :

- pour les deux sens de communications : le lien est donc bidirectionnel. La répartition de la bande passante entre les deux sens de communication est alors laissée à la charge des nœuds.
- pour un seul sens de communication : le lien est unidirectionnel et chacune de ses extrémités a un rôle bien particulier, émetteur pour l'un, récepteur pour l'autre.

Les liens bidirectionnels permettent une certaine souplesse dans le réarrangement des ressources puisque les nœuds peuvent décider localement de la répartition des ressources entre les deux sens de communication. Cependant cette souplesse se fait au détriment de la réutilisation spatiale. Dans l'allocation unidirectionnelle, chaque extrémité d'un lien a un rôle bien précis : soit émetteur, soit récepteur. Grâce à cette différentiation, nous allons voir que la distance minimale entre deux liens qui utilisent la même ressource est plus faible que dans le cas de lien bidirectionnel.

(a) Liens unidirectionnels (b) Liens bidirectionnels

FIGURE 4.5: Distance minimale de réutilisation spatiale

La figure 4.5 montre la différence entre la distance minimale de réutilisation du lien unidirectionnel et bidirectionnel. Nous supposons dans cet exemple que les interférences surviennent entre un émetteur et un récepteur séparé d'un saut. Dans le cas des liens orientés, la distinction entre les rôles d'émetteur et de récepteur permet de réutiliser la ressource allouée au lien AB pour le lien DC. Dans le cas de lien bidirectionnel, cette distance est trop courte puisque B pourrait être émetteur et C récepteur au même moment. Ces deux nœuds étant voisins, il y a risque d'interférence. C'est pourquoi la ressource allouée au lien AB ne peut être réutilisée que pour le lien DE. Les distances minimales à respecter entre deux allocations utilisant la même ressource sont donc plus grandes pour les liens bidirectionnels. La réutilisation spatiale optimale est, par conséquent, moins bonne que pour les liens unidirectionnels. Dans un réseau où la charge des liens change fréquemment de sens, la souplesse de l'approche bidirectionnelle peut être appréciable. Par contre, si les liens sont fortement asymétriques ou que l'optimisation de l'utilisation des ressources est un point critique, alors l'approche unidirectionnelle sera plus adaptée.

4.3.5.1 Liens bidirectionnels

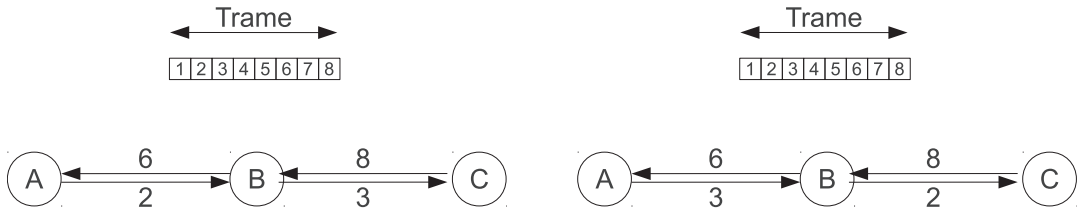
Dans [ST05], les auteurs présentent un algorithme permettant de répartir les ressources disponibles entre différentes sessions. Une session étant définie comme une demande de bande passante entre un nœud source et un nœud destination. La solution proposée par les auteurs ne cherche pas seulement à satisfaire la demande mais à répartir la totalité de la bande passante entre les différentes sessions en tenant compte de la demande de chacun. La stratégie de répartition de l'algorithme est basée sur une équité max-min [ZM11], ainsi, lorsque cet algorithme a terminé sa répartition, la part de ressources reçue par chaque session est proportionnelle à sa demande initiale.

L'article [GZH06] s'intéresse à la minimisation du temps de distribution des messages mais cette fois dans le cas particulier des réseaux de capteurs. Ces réseaux sont organisés autour d'un nœud puits chargé de récupérer l'ensemble des informations collectées par les capteurs. Les communications partent donc de sources multiples vers une seule destination. Les auteurs de cet article ont étudié et proposé des solutions pour le cas particulier des réseaux linéaires, multilignes et des arbres. Ils ont également proposé une généralisation aux topologies quelconques en appliquant un algorithme permettant d'obtenir un arbre couvrant minimum.

4.3.5.2 Liens unidirectionnels

Dans les travaux portant sur des liens unidirectionnels, l'article [DV07b], avec une version distribué dans [DV07a], utilise une stratégie min-max pour minimiser les délais de transmission aller-retour dans les réseaux où les ressources sont des intervalles de temps. Dans le cas d'une chaîne composée de trois nœuds A, B et C, chaque message transmis par A à destination de C doit être envoyé de A à B puis de B à C. Si l'intervalle de temps utilisé par le lien BC précède celui utilisé par le lien AB, la retransmission devra attendre le changement de trame. Si par contre l'intervalle de temps utilisé par BC succède à celui utilisé par AB, la retransmission se fera aussitôt. La figure 4.6 donne deux exemples d'allocation

dans cette topologie. Dans la figure 4.6a lorsqu'un message est émis par A à destination de C, celui-ci est d'abord envoyé à B au cours du deuxième intervalle de temps de la trame. Le nœud B peut ensuite réémettre ce message vers C au cours du troisième intervalle de temps. La durée de transmission de A à C est donc de celle de deux intervalles de temps. Par contre, avec l'allocation de la figure 4.6b, la transmission de A vers B se fait au cours du troisième intervalle de temps. La transmission de B vers C devra attendre le deuxième intervalle sur la trame suivante. La durée totale de transmission est alors quasiment égale à la durée d'une trame.



(a) Allocation des canaux permettant un délai d'acheminement minimum entre A et C

(b) Allocation des canaux offrant un délai d'acheminement maximum entre A et C

FIGURE 4.6: Impact du choix des canaux sur le délai de transmission de bout en bout

Il est donc possible de minimiser le délai d'acheminement d'un message entre deux nœuds en choisissant efficacement l'ordre dans lequel les intervalles de temps sont attribués aux liens. Il est également possible qu'un chemin plus long en nombre de sauts se révèle plus court en durée parce qu'il minimise le temps de retransmission à chaque saut. Pour cette raison, la solution présentée dans cet article s'intéresse à la fois à l'ordonnancement des intervalles et au routage pour minimiser les délais.

L'algorithme d'ordonnancement tente de répartir au mieux les ressources pour minimiser les délais de retransmission tandis que le routage consiste à exécuter l'algorithme de Bellman-Ford [BTW09] qui permet de trouver le chemin le plus court dans un graphe valué dont les poids peuvent être négatifs. Ici les auteurs utilisent cet algorithme dans le graphe de conflits. Habituellement, celui-ci est non orienté et composé d'un ensemble de nœuds V_C et un ensemble d'arêtes E_C. Chaque élément de V_C représente un arc du graphe de communication. Chaque élément de E_C permet alors de modéliser un conflit existant entre deux arcs du graphe de communication. Dans cet article, les auteurs vont plus loin et utilisent un graphe de conflits orienté. Ainsi lorsque deux nœuds A et B de V_C correspondent à deux arcs en conflit dans le graphe de communication, il existe deux arcs dans E_C une de A vers B et l'autre de B vers A. Grâce à ce dédoublement, il est possible d'associer un poids à chaque sens du lien. Le poids de chaque arc est fixé en fonction du temps d'acheminement nécessaire. Ce temps est calculé grâce à la position dans une trame des intervalles de temps utilisés par les extrémités de cet arc.

4.3.5.3 Comparaison des approches bidirectionnelles et unidirectionnelles

Les auteurs de [NEL07] ont étudié la différence de performances entre les solutions considérant des liens unidirectionnels et celles considérant des liens bidirectionnels. Ils ont considéré le cas où les paquets de données sont transmis de manière unidirectionnelle mais avec la présence d'un mécanisme d'acquitte-

ment des messages. Pour chaque couple de nœuds (A,B), il existe donc deux flux : un flux de données de A vers B et un flux d'acquittements de B vers A. Ils ont calculé théoriquement la capacité du réseau sur deux topologies particulières et ont mesuré par simulation la capacité pour des topologies aléatoires. Sur une topologie triangulaire (trois paires de nœuds réparties en triangle) la capacité du réseau est doublée en considérant des liens unidirectionnels. Pour le cas de grille, le gain est de 33%. Les simulations ont montré un gain moyen de 70% pour des réseaux contenant entre 40 et 90 paires de nœuds. Ces résultats peuvent s'expliquer par une meilleure différentiation, dans l'approche unidirectionnelle, du rôle de chaque extrémité d'un lien : soit émetteur, soit récepteur. La réutilisation spatiale est alors plus efficace, comme nous l'avons vu sur la figure 4.3, permettant ainsi d'augmenter le nombre d'allocations réalisables avec la même quantité de ressources.

4.4 Conclusion

Les travaux qui ont étudié les différentes approches, graphe et théorie de l'information, s'accordent à dire que l'approche graphe permet d'obtenir des solutions plus rapides et plus réactives que l'approche basée sur la théorie de l'information. Ils font cependant remarquer que l'hypothèse généralement admise dans l'approche graphe selon laquelle les zones de transmission et d'interférence sont de même taille n'est pas réaliste. Les algorithmes d'allocation ne prennent donc pas en compte toutes les interférences ce qui nuit aux performances de la solution. De plus, pour utiliser efficacement les ressources, certains travaux tentent de maximiser la réutilisation spatiale des canaux. Pour cela ils cherchent à minimiser la distance entre deux éléments utilisant le même canal tout en veillant à ce que cette distance soit suffisamment grande pour ne pas générer d'interférences. Or, si la zone d'interférence est sous-évaluée, plus un algorithme cherche à maximiser la réutilisation spatiale et plus il risque d'augmenter le nombre d'interférences dans le réseau, rendant la tentative de réutilisation spatiale contre-productive.

Considérer une zone d'interférence supérieure à la zone de transmission présente deux difficultés majeures. Tout d'abord cela signifie qu'à chaque allocation, il faut prendre en compte plus de nœuds ou de liens afin de vérifier qu'aucune interférence ne soit générée. Dans un contexte distribué, cela signifie que les nœuds interférés ne se situent plus uniquement dans le voisinage mais que chaque nœud doit tenir compte d'éléments du réseau situés à deux ou trois sauts. Une zone d'interférence supérieure à la zone de transmission signifie également que l'ensemble des récepteurs interférés par un nœud n'est pas forcément à portée radio de celui-ci. Ainsi la détection et la correction de conflits dans un environnement distribué ne peut plus se faire directement entre l'émetteur à l'origine de l'interférence et le récepteur interféré. Il faut donc mettre en place de nouveaux protocoles capables de détecter ces conflits d'allocation et de les corriger en s'appuyant sur des voisins communs à ces deux nœuds.

Parmi les approches graphe, il est possible d'utiliser une stratégie orientée nœud ou orientée lien. La première est adaptée aux réseaux avec des transmissions de type *broadcast* tandis que la seconde est plus efficace dans un réseau où les demandes de débit sont importantes mais où les connexions sont plutôt point-à-point. Certaines études se sont intéressées à la différence de performances entre les solutions

qui considèrent des liens bidirectionnels et celles qui tiennent compte du sens de communication dans l'allocation des ressources. Elles ont montré que l'allocation sur des liens unidirectionnels permettait, dans la plupart des cas, d'obtenir de meilleurs résultats. Il existe cependant très peu de travaux proposant des solutions considérant une allocation de canaux sur des liens unidirectionnels.

Ces travaux ont été réalisés en tenant compte de ces différents points. Les travaux présentés dans les chapitres suivants portent donc sur l'allocation de ressources sur des liens unidirectionnels pour une approche graphe et en considérant une zone d'interférence deux fois supérieure à la zone de transmission.

Chapitre 5

Détection et correction de conflits

5.1 Introduction

Le chapitre précédent à montré que le principal problème de l'approche graphe pour l'allocation de ressources était l'hypothèse selon laquelle la zone de transmission et la zone d'interférence sont de taille identique. Les études [GH01, BR03] ont montré que cette hypothèse, faite par la plupart des travaux, était à l'origine de nombreuses interférences non détectées et d'une modélisation erronée.

Nous avons donc décidé de proposer de nouvelles solutions à l'allocation de ressources fondées sur un modèle plus réaliste où la zone d'interférence est deux fois plus étendue que la zone de transmission. La principale conséquence de ce changement d'hypothèse porte sur la détection et la correction des conflits. Lorsque l'on considère que la zone d'interférence d'un émetteur est confondue avec sa zone de transmission, cela signifie que cet émetteur est capable de communiquer avec l'ensemble des récepteurs qu'il interfère. Il lui suffit alors de connaître les canaux utilisés en réception par ses voisins pour détecter la présence d'éventuels conflits. Il ne lui reste plus qu'à libérer ou demander au récepteur de libérer le canal pour résoudre le conflit.

Par contre, lorsque la zone d'interférence est deux fois plus étendue que la zone de transmission, un émetteur ne peut plus communiquer directement avec l'ensemble des récepteurs qu'il interfère. En effet, un récepteur peut se trouver au-delà de la zone de transmission tout en étant toujours dans sa zone d'interférence. Dans ce cas, la détection du conflit passe obligatoirement par d'autres nœuds du réseau qui doivent alors informer l'émetteur de la présence de ce récepteur. La détection et la correction de conflits deviennent alors des problèmes à part entière. C'est pourquoi nous avons décidé de séparer le problème de l'allocation de ressources en deux parties :

- *la détection et la correction de conflit*, chargée de résoudre les conflits existants mais également d'identifier les ressources utilisables pour de nouvelles allocations.
- *la répartition de ressources*, chargée d'attribuer les canaux en fonction des demandes et de veiller à maximiser la réutilisation spatiale pour gérer efficacement les ressources disponibles.

Ce chapitre est consacré à la détection et la correction de conflits à travers la présentation de deux

algorithmes. Le premier est auto-stabilisant et écrit dans le modèle à états que nous présentons ci-dessous. Nous avons prouvé que cet algorithme permettait de corriger l'ensemble des conflits. Nous avons borné le temps de convergence et prouvé que lorsque cet algorithme a atteint l'état stable, il est possible d'identifier les ressources pouvant être allouées sans créer de conflits avec les allocations existantes.

Le modèle à état ne permet pas d'utiliser cet algorithme dans des conditions de communications réelles. C'est pourquoi, le second algorithme est à la fois une adaptation du premier dans un modèle plus réaliste mais également une amélioration de celui-ci afin de gérer plus efficacement la libération des canaux en cas de conflit.

5.2 Modélisation

5.2.1 Modélisation du réseau

Le réseau est modélisé par un graphe orienté symétrique G(V,E) où V représente l'ensemble des nœuds et E, l'ensemble des arcs. Ce graphe est appelé *graphe de connectivité*. Chaque nœud modélise une unité de communication, et chaque arc (e,r) représente un lien unidirectionnel entre un émetteur $e \in V$ et son récepteur associé, $r \in V$. Nous considérons uniquement le cas de graphes connexes. Nous supposons l'existence d'un ensemble H de ressources discrètes indexées de 0 à M qui seront associées à chaque arc de l'ensemble E.

5.2.2 Modélisation des interférences

Pour modéliser les interférences, nous définissons un graphe non orienté $G_C = (V_C, E_C)$ constitué d'un ensemble de sommets V_C et d'arêtes E_C. L'ensemble V_C a une relation un-pour-un avec l'ensemble E du graphe de connectivité ; c'est-à-dire que pour chaque lien $e \in E$ il existe un sommet $v_c \in V_C$. Nous appelons ce graphe, *graphe des conflits*. Une arête (e_1,e_2) de E_C, existe seulement si l'émetteur de e_1 (resp. e_2) et le récepteur de e_2 (resp. e_1) sont à au plus deux sauts l'un de l'autre. Partant de l'hypothèse qu'il n'existe pas d'interférence entre deux ressources distinctes, chaque ressource peut être considérée comme indépendante.

5.2.3 Modèle à états

Le premier algorithme présenté dans cette partie est écrit dans le modèle à état. Contrairement au modèle classique de communication, les nœuds ne s'échangent pas de message mais sont capables de lire les variables partagées de leurs voisins. Dans ce modèle, chaque noeud exécute le même algorithme constitué de variables et d'actions. Le premier algorithme considéré dans cette partie utilise une variable par ressource pour chaque noeud. Chacune de ces variables représente l'état de la ressource qui lui est associée. Chaque noeud est capable de lire ses propres variables et les variables de ses voisins. Cependant, il ne peut modifier que ses propres variables. Une action gardée a la forme $< etiquette >::< garde >\rightarrow< declaration >$. L'étiquette permet d'identifier l'action. La garde est un prédicat pouvant porter aussi bien

sur les variables locales que sur les variables des voisins. La déclaration est un ensemble d'instructions modifiant les variables locales. Lorsque la garde est vraie, l'action correspondante est dite activable. Par extension, un noeud est dit activable lorsqu'il possède au moins une action activable. L'activation d'un noeud consiste à exécuter l'ensemble des instructions de toutes ses actions activables. L'état d'un noeud est défini par l'état de l'ensemble de ses variables. L'ensemble des états de tous les noeuds du système est appelé configuration. L'exécution d'un système est constitué de l'ensemble des configurations c_1, c_2,..., c_n où c_{i+1} est obtenu à partir de la configuration c_i après activation d'au moins un noeud. La complexité en temps est ici exprimée en *ronde*, la ronde étant définie comme le sous-ensemble minimal de configurations dans lequel chaque noeud est activé au moins une fois.

5.2.4 Auto-stabilisation

Comme tout système, un réseau peut être sujet à des fautes telles que la perte de messages ou la disparition d'un nœud. Lorsqu'un tel événement survient, l'algorithme doit permettre au système de continuer à fonctionner correctement. Cette capacité est la tolérance aux fautes. Pour les systèmes distribués, Dijkstra a introduit dans [Dij74] la notion d'auto-stabilisation comme la capacité d'un système à converger, en un temps fini, vers un état légitime lorsqu'une faute survient.

5.3 Algorithme de correction de conflits et preuves

Notre premier algorithme de correction de conflits a deux objectifs :

– corriger les conflits existant dans le réseau,

– prévenir les conflits futurs en identifiant les ressources utilisables pour de nouvelles allocations.

Nous avons fait l'hypothèse que les canaux de communications étaient indépendants. La correction des conflits peut donc également se faire de manière indépendante. C'est pourquoi, par souci de clarté, l'étude de l'algorithme se fera en considérant le cas d'un réseau possédant un unique canal. Dans le cas plus général d'un réseau à M canaux, il suffit d'appliquer cet algorithme à chacune de ces ressources.

Nous appelons *allocation* le fait d'attribuer une ressource à un arc du graphe de connectivité. Les extrémités de cet arc, c'est-à-dire l'émetteur et le récepteur, sont dites *associées* pour la ressource allouée. Pour une même ressource, lorsqu'un émetteur est voisin d'un récepteur et que ces deux nœuds ne sont pas associés, il y a conflit d'allocation.

Pour qu'un émetteur puisse détecter un conflit, il lui faut savoir s'il existe un récepteur dans sa zone d'interférence, c'est-à-dire dans son 2-voisinage. De manière symétrique, un récepteur a besoin de détecter la présence d'émetteurs dans son 2-voisinage pour savoir si l'allocation à laquelle il participe est en conflit avec une autre allocation dans le réseau. Pour effectuer cette détection, chaque nœud possède une variable par ressource appelée *channelState*. Chaque nœud possède donc M variables *channelState*. La valeur de cette variable permet aux nœuds de détecter la présence d'émetteur ou de récepteur dans leur 2-voisinage. Nous allons maintenant voir quels sont les états que peut prendre cette variable.

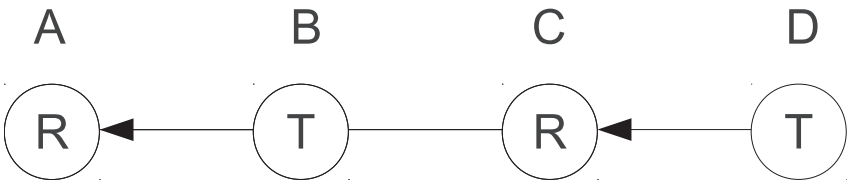


FIGURE 5.1: Exemple de conflit à un saut

5.3.1 Correction des conflits existants

Pour corriger les conflits existants, l'algorithme doit détecter les conflits à un saut et à deux sauts. Les premiers ayant lieu entre un émetteur et un récepteur voisins mais non associés, il suffit que ces nœuds informent leur voisinage de leur état pour qu'un conflit puisse être détecté. Pour identifier la position des émetteurs et des récepteurs, les deux premiers états que peut prendre *channelState* sont donc :

– l'état R pour indiquer que la ressource est utilisée en réception

– l'état T lorsqu'elle est utilisée en transmission.

La figure 5.1 représente un exemple de conflit à un saut. Les cercles modélisent les nœuds. La valeur à l'intérieur de ces cercles indique l'état de la variable *channelState* de ces nœuds. Les flèches représentent les arcs alloués. Elles relient donc chaque émetteur avec son récepteur associé. Dans cette figure, il existe deux allocations. La première entre l'émetteur B et le récepteur A. La seconde entre l'émetteur D et le récepteur C. Il y a donc conflit à un saut entre l'émetteur B et le récepteur C. Grâce aux états R et T que peut prendre *channelstate*, le nœud B détecte la présence d'un récepteur non associé dans son voisinage tout comme le nœud C détecte la présence d'un émetteur non associé dans son voisinage. Le conflit est ainsi détecté.

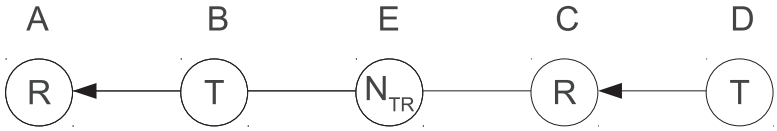
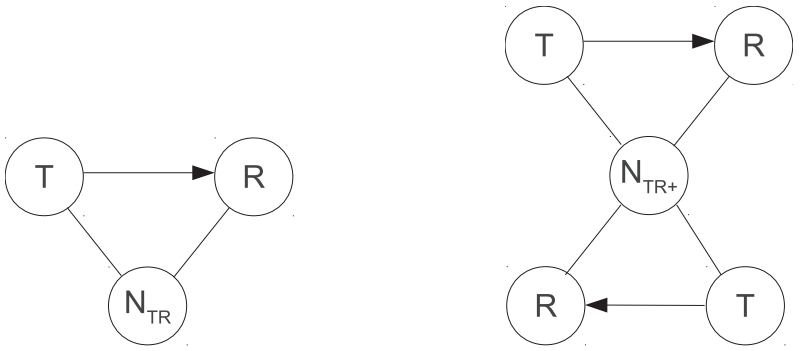


FIGURE 5.2: Exemple de conflit à deux sauts

Cependant, pour détecter la présence d'émetteur ou récepteur à deux sauts, ces deux états ne sont pas suffisants. La figure 5.2 donne un exemple de conflit à deux sauts. Dans cette configuration, le conflit qui a lieu entre B et C ne peut pas être détecté directement puisque ces deux nœuds ne sont pas voisins. Cependant, B et C ont un voisin commun : E. En indiquant qu'il est voisin à la fois d'un émetteur et d'un récepteur, ce nœud va informer B de la présence d'un récepteur dans son 2-voisinage et C de la présence d'un émetteur dans son 2-voisinage. Pour cela, deux nouveaux états sont ajoutés à l'ensemble de définition de *channelState* :

– N_{TR}, pour indiquer la présence d'*au moins* un émetteur et d'*exactement* un récepteur dans le voisinage,

– N_{TR+} pour indiquer la présence d'*au moins* un émetteur et d'*au moins* deux récepteurs dans le voisinage.



(a) État N_{TR} n'indiquant pas la présence d'un conflit

(b) État N_{TR} indiquant la présence d'un conflit

FIGURE 5.3: Utilisation de l'état N_{TR+}

Ces deux états permettent de faire la distinction entre les deux cas présentés par les figures 5.3a et 5.3b. Dans le premier cas, Figure 5.3a, un nœud est voisin d'un émetteur et d'un récepteur associés ; il n'y a donc aucun conflit. Dans le cas illustré sur la figure 5.3b, le même nœud est cette fois-ci voisin de deux couples (émetteur, récepteur). Si un seul état existait, il ne serait pas possible de faire la distinction entre ces deux cas de figure. Par contre, en introduisant l'état N_{TR+}, les émetteurs dans la figure 5.3b sont certains qu'il existe au moins deux récepteurs dans leur 2-voisinage. L'un de ces récepteurs peut être leur récepteur associé, mais le second appartient obligatoirement à une autre allocation. Ces quatre états (T, R, N_{TR} e N_{TR+}) permettent ainsi de détecter les conflits d'allocation. Il reste cependant des configurations où ces états ne sont pas suffisants.

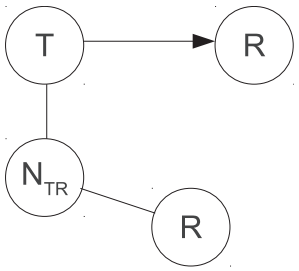


FIGURE 5.4: État N_{TR} indiquant la présence d'un conflit

Ainsi, la figure 5.4 donne un exemple de configuration qui présente un autre type de conflit d'allocation. Pour différencier les situations des figures 5.4 et 5.3a, l'algorithme s'appuie sur le fait qu'un nœud dans l'état N_{TR} ne peut être voisin que d'un seul récepteur. L'émetteur sait qu'il existe un seul récepteur dans son 2-voisinage. Il lui faut alors déterminer s'il s'agit de son récepteur associé ou d'un récepteur appartenant à une autre allocation. Pour cela, l'émetteur a besoin de connaître la liste des voisins de son voisin dans l'état N_{TR}. C'est pourquoi, en plus de la variable *channelState*, chaque nœud peut accéder à la liste des voisins de ses voisins. Ainsi lorsqu'un émetteur détecte un voisin dans l'état N_{TR} lui-même voisin de son récepteur associé, il est sûr qu'il se trouve dans le cas de la figure 5.3a et qu'il n'existe donc aucun conflit. Dans le cas contraire, le récepteur situé dans son 2-voisinage appartient à une autre

allocation, il y a donc conflit.

Grâce à ces informations, les nœuds participant à une allocation sont capables de détecter la présence de conflits dans leur zone d'interférence. Il leur suffit alors de libérer la ressource en conflit pour le corriger.

5.3.2 Détection des ressources utilisables

Le second rôle de l'algorithme consiste à prévenir les conflits en permettant à l'algorithme d'allocation de détecter les ressources utilisables pour une nouvelle allocation. Pour qu'une nouvelle allocation ne crée pas de conflit, il faut qu'il n'y ait aucun récepteur dans le 2-voisinage du nouvel émetteur et aucun émetteur dans le 2-voisinage du nouveau récepteur. Puisque l'algorithme est distribué, chaque nouvelle allocation est créée localement par une des extrémités de l'allocation : soit l'émetteur, soit le récepteur. Ces deux nœuds ont un rôle identique, aucun critère ne permet donc de privilégier l'un par rapport à l'autre. Dans nos travaux, nous faisons l'hypothèse que les allocations sont gérées par les émetteurs.

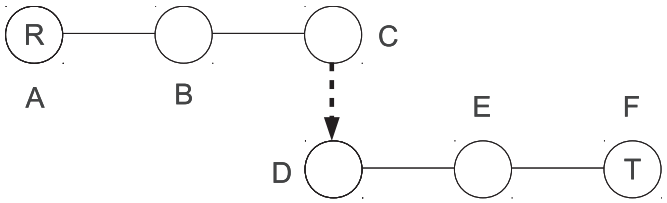


FIGURE 5.5: Création d'un conflit en cas d'allocation d'une ressource entre les nœuds C et D.

Ainsi, l'algorithme de correction de conflits doit informer chaque émetteur potentiel de la présence de récepteurs dans son 2-voisinage mais également de la présence d'émetteurs dans le 2-voisinage de tous ses voisins. Dans la Figure 5.5, le noeud C souhaite allouer une ressource vers le nœud D. Cette allocation ne peut avoir lieu pour deux raisons :

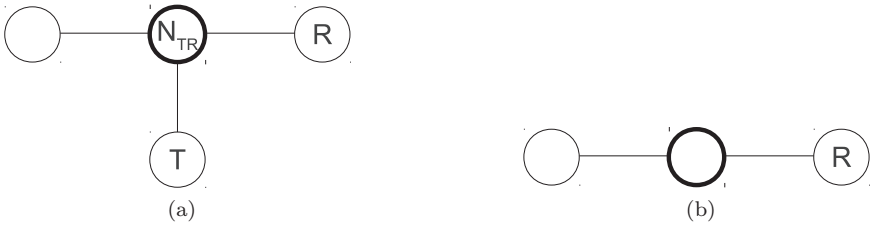
- C est dans le 2-voisinage du récepteur A
- D appartient au 2-voisinage de l'émetteur F

Le nœud C étant chargé de l'allocation, c'est lui qui doit détecter ces deux cas. De manière générale, pour qu'une allocation ne génère pas de conflit, tout nœud qui souhaite devenir émetteur doit pouvoir détecter :

- la présence de récepteur dans son 2-voisinage.
- la présence d'émetteur dans le 2-voisinage de ses voisins

5.3.2.1 Détection des récepteurs dans le 2-voisinage d'un nœud

La correction des conflits existants utilise les différents états que peut prendre la variable *channelState*. Certains de ces états informent déjà les nœuds de la présence de récepteurs dans leur zone d'interférence. Grâce à l'état R, chaque nœud est capable de détecter la présence de récepteurs dans son voisinage strict. La présence de voisins dont la variable est dans l'état N_{TR} ou N_{TR+} permet à un nœud de détecter la présence de certains récepteurs dans le 2-voisinage. Cependant, la fig 5.6 montre que lorsque le voisinage

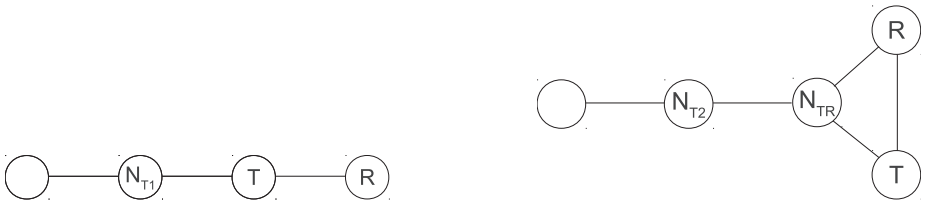


(a) (b)

FIGURE 5.6: Rôle de l'état N_R pour la détection de récepteur dans le 2-voisinage

d'un nœud ne contient que des récepteurs, aucun état ne lui permet d'informer l'ensemble de son voisinage de la présence de ces récepteurs. C'est pourquoi, un nouvel état a été introduit : N_R. Cet état permet à un nœud d'informer son voisinage qu'il est lui-même voisin d'un récepteur. Grâce à cet état chaque nœud est capable de détecter la présence de tout récepteur dans son 2-voisinage.

5.3.2.2 Détection des émetteurs dans le 2-voisinage des voisins d'un nœud



(a) Rôle de l'état N_{T1} (b) Rôle de l'état N_{T2}

FIGURE 5.7: Détection d'émetteurs dans le 3-voisinage

Un nœud A_T qui souhaite créer une allocation avec un voisin B_R doit également vérifier qu'il n'existe aucun émetteur dans le 2-voisinage de B_R. La variable *channelState* doit donc indiquer la présence d'émetteurs à deux sauts. Certains états présentés précédemment le permettent déjà puisque :

- N_{TR}, N_{TR+} indiquent par définition la présence d'un émetteur dans le voisinage.
- N_R indique la présence d'un récepteur dans le voisinage. Ce récepteur étant associé avec un émetteur, cet état indique également la présence d'un émetteur dans le 2-voisinage.

La figure 5.7 montre que dans certains cas, ces états sont insuffisants pour indiquer la présence d'émetteurs dans le 2-voisinage. C'est pourquoi nous introduisons deux nouveaux états :

- N_{T1}, lorsqu'il existe un émetteur dans le voisinage
- N_{T2} lorsqu'il existe un émetteur à deux sauts. Pour pouvoir détecter cette situation un noeud N s'appuie sur ses voisins. Si la variable *channelState* de l'un d'entre eux est dans l'état N_{T1}, N_{TR} ou N_{TR+}, alors N est forcément à deux sauts d'un émetteur.

Grâce à ces nouveaux états, chaque nœud peut indiquer la présence d'émetteurs à un saut (N_{T1}, N_{TR} et N_{TR+}) ou à deux sauts (N_{T2}). Pour reprendre l'exemple du nœud A_T qui souhaite créer une allocation avec son voisin B_R : A_T est donc capable de détecter s'il existe un émetteur dans le 2-voisinage de B_R.

Les états de cette variable sont résumés dans le tableau 5.1. La figure 5.8 illustre les règles utilisées par l'algorithme pour choisir son état. Dans cette figure, les traits pleins indiquent les liaisons qui doivent

TABLE 5.1: Description de la variable *channelState*

État	**Portée**	**Signification**
T	local	ressource utilisée localement pour transmettre
R	local	ressource utilisée localement pour recevoir
N_R	un saut	ressource utilisée en réception par au moins un voisin
N_{T1}	un saut	ressource utilisée en émission par au moins un voisin
N_{TR}	un saut	ressource utilisée en émission par au moins un voisin et en réception par exactement un voisin
N_{TR+}	un saut	ressource utilisée en émission par au moins un voisin et en réception par au moins deux voisins
N_{T2}	deux sauts	ressource utilisée en transmission par au moins un voisin à deux sauts
F	-	Aucun émetteur à deux sauts et aucun récepteur dans le voisinage

exister. Les traits barrés indiquent des liaisons qui ne doivent pas exister. Les traits pointillés indiquent les liaisons qui ne sont pas nécessaires. Cependant s'il existe une liaison pour un trait pointillé, une liaison doit également exister pour chacun des autres traits pointillés. Les arcs indiquent les couples émetteur-récepteur associés.

L'algorithme 5.3.1 [PTBV10] repose donc principalement sur l'échange de deux informations :

– la liste des voisins

– l'état de la variable *channelState* pour chaque ressource.

Par exemple, considérons le cas d'un émetteur (Fig. 5.8, premier cadre en haut à gauche). Pour qu'un émetteur soit dans un état correct, il faut qu'il soit associé à un récepteur et qu'il n'y ait aucun récepteur dans son 2-voisinage. Le récepteur se charge de détecter la présence d'émetteur dans son 2-voisinage. Dans le schéma étudié, un arc relie l'émetteur à un récepteur, représentant ainsi la présence obligatoire du récepteur associé. Le lien barré entre l'émetteur et un autre récepteur indique qu'aucun autre récepteur ne doit se trouver dans le voisinage de l'émetteur étudié. Pour indiquer qu'il ne doit pas non plus exister de récepteurs dans le 2-voisinage, un autre lien barré relie l'émetteur avec des nœuds dans l'état T (indique un conflit entre l'émetteur étudié et le récepteur associé au nœud dans l'état T), N_R et N_{TR+} (indiquent la présence de récepteur dans le 2-voisinage de l'émetteur étudié). Le cas de l'état N_{TR} est particulier. Comme nous l'avons vu dans le paragraphe précédent, cet état indique la présence d'un unique récepteur dans le voisinage. Cet unique récepteur peut être le récepteur associé à l'émetteur étudié auquel cas, il n'y a pas de conflit. Cette situation est représentée par les liens en trait discontinue qui indique une situation correcte tant qu'un lien existe pour chaque trait discontinu. Mais, cet unique récepteur peut ne pas être associé avec l'émetteur étudié. Dans ce cas, le nœud N_{TR} traduit un conflit, ce qui est schématisé par un trait barré entre l'émetteur étudié et un voisin dans l'état N_{TR}.

L'algorithme possède six actions pour mettre à jour l'état d'une variable *channelState* :

– $N_R - Action_p$ qui permet de faire passer la variable dans l'état N_R

– $N_{T1} - Action_p$ qui permet de faire passer la variable dans l'état N_{T1}

– $N_{T2} - Action_p$ qui permet de faire passer la variable dans l'état N_{T2}

– $N_{TR} - Action_p$ qui permet de faire passer la variable dans l'état N_{TR}

– $N_{TR+} - Action_p$ qui permet de faire passer la variable dans l'état N_{TR+}

– $N_F - Action_p$ qui permet de faire passer la variable dans l'état N_F.

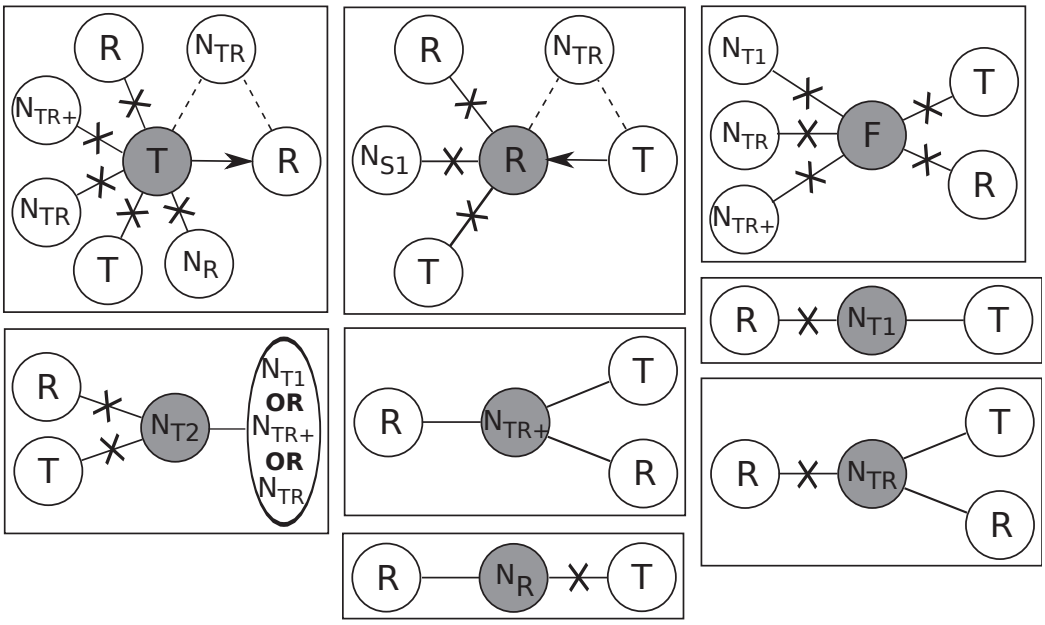


FIGURE 5.8: Critères de sélection de l'état pour la variable *channelState*

Aucune action ne permet à l'algorithme 5.3.1 d'attribuer les états T et R à *channelState* puisque la création d'allocation est réalisée par un autre algorithme. Les algorithmes d'allocation feront l'objet du chapitre suivant. Pour qu'une de ces actions soit activable, il faut que certains des prédicats de l'algorithme 5.3.1 soient vérifiés. Ces prédicats ont été conçus afin de lever toute ambiguïté sur le choix de l'état d'une variable. En effet, en se basant uniquement sur les définitions données par le tableau 5.1, il serait possible d'attribuer simultanément plusieurs valeurs à *channelState*. Par exemple, un nœud peut être voisin d'un récepteur et situé à deux sauts d'un émetteur, il a alors le choix entre les états N_R et N_{T2}. Pour lever cette ambiguïté, les prédicats permettent d'introduire un ordre de priorité entre les différents états. Cet ordre suit deux règles. L'information sur le voisinage ($\{N_R, N_{TR}, N_{TR+}, N_{T1}\}$) est prioritaire sur le 2-voisinage (N_{T2}) et sur l'état par défaut (F). Lorsque ce critère ne suffit pas, la priorité se fait sur la quantité d'information véhiculée par l'état de la variable ($N_{TR+} > N_{TR} > N_{T1} \geq N_R$ et $N_{T2} > F$).

5.3.3 Spécifications

Nous nous intéressons dans ce paragraphe aux spécifications de notre algorithme et à leurs preuves formelles. Nous considérons ici qu'un système est dans un état stable s'il n'existe plus de nœuds activables dans le système. De manière formelle, notre algorithme doit vérifier les propriétés suivantes :

Sûreté :

1. Un système stable ne présente aucun conflit d'allocation.

Algorithme 5.3.1: Détection et correction de conflit

Variables :

$StatesSet = \{T, R, N_R, N_{TR}, N_{TR+}, N_{T1}, N_{T2}, F\}$

$S_p \in StatesSet$: *channelState* of node p.

$Neigh_p$: ensemble des voisins de p.

$OtherEnd(p)$: nœud associé à p.

Predicates :

$$\begin{aligned}
\text{R-Isolated}_p &\equiv S_p = R \wedge (\nexists q \in Neigh_p :: (S_q = T) \wedge (q = OtherEnd(p)) \\
\text{T-Isolated}_p &\equiv S_p = T \wedge (\nexists q \in Neigh_p :: (S_q = R) \wedge (q = OtherEnd(p)) \\
\text{T-1HopInterf}_p &\equiv S_p = T \wedge (\exists q \in Neigh_p :: S_q = R \wedge q \neq OtherEnd(p))) \\
\text{R-1HopInterf}_p &\equiv S_p = R \wedge (\exists q \in Neigh_p :: S_q = T \wedge q \neq OtherEnd(p))) \\
\text{T-2HopInterf}_p &\equiv S_p = T \wedge (\exists q \in Neigh_p :: (S_q \in \{T, N_{TR+}, N_R\}) \\
&\qquad \vee (S_q = N_{TR} \wedge OtherEnd(p) \notin Neigh_q)) \\
\text{R-2HopInterf}_p &\equiv S_p = R \wedge \exists q \in Neigh_p :: (S_q = R) \\
\text{IncorrectT}_p &\equiv S_p = T \wedge (T\text{-}Isolated_p \vee T\text{-}1HopInterf_p \vee T\text{-}2HopInterf_p) \\
\text{IncorrectR}_p &\equiv (R\text{-}Isolated_p \vee R\text{-}1HopInterf_p \vee R\text{-}2HopInterf_p)
\end{aligned}$$

$$\begin{aligned}
\text{Pre-NeighRcv}_p &\equiv ((\exists q \in Neigh_p :: S_q = R) \wedge (\nexists q' \in Neigh_p :: S_{q'} = T)) \\
\text{Pre-1HopNeighTrans}_p &\equiv ((\exists q \in Neigh_p :: S_q = T) \wedge (\nexists q' \in Neigh_p :: S_{q'} = R)) \\
\text{Pre-2HopNeighTrans}_p &\equiv ((\exists q \in Neigh_p :: S_q \in \{N_{T1}, N_{TR}, N_{TR+}) \\
&\qquad \wedge (\nexists q' \in Neigh_p :: S_{q'} \in \{T, R\})) \\
\text{Pre-NeighTR}_p &\equiv ((\exists! q_R \in Neigh_p :: S_{q_R} = R) \wedge (\exists q_T \in Neigh_p :: S_{q_T} = T)) \\
\text{Pre-NeighTR+}_p &\equiv (\exists (q_T, q_{R1}, q_{R2}) \in Neigh_p^3 :: (S_{q_{R1}} = S_{q_{R2}} = R) \wedge (S_{q_T} = T)) \\
\text{Pre-Free}_p &\equiv (\nexists q \in Neigh_p :: S_q \in States \backslash \{N_R, N_{T2}, F\})
\end{aligned}$$

$$\begin{aligned}
\text{NeighRcv}_p &\equiv Pre\text{-}NeighRcv_p \wedge (IncorrectT_p \vee IncorrectR_p \\
&\qquad \vee S_p \in States \backslash \{T, R, N_R\}) \\
\text{1HopNeighTrans}_p &\equiv Pre\text{-}1HopNeighTrans_p \wedge (IncorrectT \vee IncorrectR_p \\
&\qquad \vee S_p \in States \backslash \{T, R, N_{T1}\}) \\
\text{2HopNeighTrans}_p &\equiv Pre\text{-}2HopNeighTrans_p \wedge (IncorrectT \vee IncorrectR_p \\
&\qquad \vee S_p \in States \backslash \{T, R, N_{T2}\}) \\
\text{NeighTR}_p &\equiv Pre\text{-}NeighTR_p \wedge (IncorrectT \vee IncorrectR_p \\
&\qquad \vee S_p \in States \backslash \{T, R, N_{TR}\}) \\
\text{NeighTR+}_p &\equiv Pre\text{-}NeighTR+_p \wedge (IncorrectT \vee IncorrectR_p \\
&\qquad \vee S_p \in States \backslash \{T, R, N_{TR+}\}) \\
\text{Free}_p &\equiv Pre\text{-}Free_p \wedge (IncorrectT \vee IncorrectR_p \\
&\qquad \vee S_p \in States \backslash \{T, R, F\})
\end{aligned}$$

Actions :

$$\begin{aligned}
&N_R\text{-}Action_p :: NeighRcv_p \longrightarrow S_p := N_R \\
&N_{T1}\text{-}Action_p :: 1HopNeighTrans_p \longrightarrow S_p := N_{T1} \\
&N_{T2}\text{-}Action_p :: 2HopNeighTrans_p \longrightarrow S_p := N_{T2} \\
&N_{TR}\text{-}Action_p :: NeighTR_p \longrightarrow S_p := N_{TR} \\
&N_{TR+}\text{-}Action_p :: NeighTR+_p \longrightarrow S_p := N_{TR+} \\
&F\text{-}Action_p :: Free \longrightarrow_p S_p := F
\end{aligned}$$

2. Dans un système stable, tout nœud p de V dont la variable *channelState* est dans l'état F ou N_{T2} et qui ne possède pas de voisin dans l'état N_R peut créer une allocation vers un voisin dans l'état F sans créer de conflit.

Vivacité :

1. Tout système atteint un état stable en au plus cinq rondes quel que soit le graphe de connectivité, la valeur de *channelState* de chaque nœud et la taille du réseau.

5.3.4 Preuve des spécifications

La preuve de ces spécifications est présentée dans ce paragraphe. Nous travaillons sous l'hypothèse qu'il n'y a pas d'allocation ni de changement topologique durant la phase de convergence.

Lemme 5.3.1. *Au moins un des prédicats* − {Pre-NeighRcv$_p$, Pre-1HopNeighTrans$_p$, Pre-2HopNeighTrans$_p$, Pre-NeighTR$_p$, Pre-NeighTR+$_p$, Pre-Free$_p$ }− est toujours vrai

Démonstration. Supposons par l'absurde qu'il existe un nœud A pour lequel aucun de ces prédicats ne soit vrai. Puisque le graphe de connectivité $G = (V, E)$ est connexe, A possède au moins un voisin B.

- Si B est un récepteur et que A n'a pas de voisin dont la variable *channelState* est dans l'état T, alors le prédicat *Pre-NeighRcv$_p$* est vérifié.
- Si B est le seul récepteur de A et que A possède des émetteurs dans son voisinage, alors le prédicat *Pre-NeighTR$_p$* est vérifié.
- Si B n'est pas le seul récepteur de A et que A possède des émetteurs dans son voisinage, alors le prédicat *Pre-NeighTR+$_p$* est vérifié.
- Si B est un émetteur et que A n'a pas de voisin dont la variable *channelState* est dans l'état R, alors le prédicat *Pre-1HopNeighTrans$_p$* est vérifié.
- Si B est un émetteur et que A possède un unique récepteur dans son voisinage, alors le prédicat *Pre-NeighTR$_p$* est vérifié.
- Si B est un émetteur et que A possède au moins deux récepteurs dans son voisinage, alors le prédicat *Pre-NeighTR+$_p$* est vérifié.
- Si l'état de *channelState* de B appartient à l'ensemble $\{N_{T1}, N_{TR}, N_{TR+}\}$ et que A ne possède aucun émetteur ou récepteur dans son voisinage, alors le prédicat *Pre-2HopNeighTrans$_p$* est vérifié.
- Si la variable de *channelState* des voisins de A est dans l'état F, N_R ou N_{T2}, le prédicat *Pre-Free$_p$* est vérifié.

Ainsi, quel que soit le voisinage de A, il existe toujours un des prédicats − *Pre-NeighRcv$_p$, Pre-1HopNeighTrans$_p$, Pre-2HopNeighTrans$_p$, Pre-NeighTR$_p$, Pre-NeighTR+$_p$, Pre-Free$_p$* − qui est vrai. □

Grâce au lemme précédent, nous avons montré que quelle que soit la topologie, au moins un des prédicats était vérifié pour chaque noeud du réseau. L'algorithme peut donc toujours attribuer un état

à chaque variable *channelState* quelle que soit la topologie. Nous allons maintenant montrer que deux actions ne peuvent pas être simultanément activables et que donc l'algorithme est déterministe puisqu'il n'y a jamais deux états possibles pour une même variable.

Lemme 5.3.2. *Il y a au plus une action activable pour chaque nœud.*

Démonstration. Deux actions sont simultanément activables si leur garde est vraie simultanément. Dans notre algorithme, deux gardes sont vraies simultanément si au moins deux prédicats de l'ensemble $P = \{Pre\text{-}NeighRcv_p, Pre\text{-}1HopNeighTrans_p, Pre\text{-}2HopNeighTrans_p, Pre\text{-}NeighTR_p, Pre\text{-}NeighTR+_p, Pre\text{-}Free_p\}$ sont vrai simultanément. Supposons par l'absurde qu'il existe au moins deux prédicats de l'ensemble P simultanément vrai.

- Supposons que le prédicat *Pre-NeighRcv* soit l'un de ces deux prédicats. Pour qu'il soit vérifié, le voisinage doit contenir au moins un récepteur et aucun émetteur. Or :
  - la présence d'un récepteur dans le voisinage ne permet pas aux prédicats *Pre-1HopNeighTrans*, *Pre-2HopNeighTrans*, et *Pre-Free* d'être vérifiés.
  - Un émetteur doit être présent dans le voisinage pour que les prédicats *Pre-NeighTR* and *Pre-NeighTR+* soient vérifiés.

  Ainsi, le prédicat *Pre-NeighRcv* ne peut pas être vrai si un des autres prédicats de P est vrai.
- Supposons que le prédicat *Pre-1HopNeighTrans* soit l'un de ces deux prédicats. Pour qu'il soit vérifié, il faut que le voisinage contienne au moins un émetteur et aucun récepteur. Or :
  - la présence d'un voisin dans l'état T ne permet pas aux prédicats *Pre-NeighRcv*, *Pre-2HopNeighTrans* et *Pre-Free* d'être vrais.
  - la présence d'un récepteur dans le voisinage est indispensable dans les prédicats *Pre-NeighTR* et *Pre-NeighTR+*.

  Ainsi, le prédicat *Pre-NeighRcv* ne peut pas être vrai si un des autres prédicats de P est vrai.
- Supposons que le prédicat *Pre-2HopNeighTrans* soit l'un de ces deux prédicats. Pour qu'il soit vérifié, au moins un voisin doit être dans un état N_{T1},N_{TR} ou N_{TR+}, mais aucun émetteur ou récepteur ne doit être présent dans le voisinage. Or :
  - la présence d'un récepteur dans le voisinage est indispensable pour vérifier les prédicats *Pre-NeighRcv*, *Pre-NeighTR* et *Pre-NeighTR+*.
  - Un émetteur doit être présent dans le voisinage pour que le prédicat *Pre-1HopNeighTrans* soit vérifiés.
  - la présence de voisin dont la variable *channelState* est dans l'un des états N_{T1},N_{TR} ou N_{TR+} ne permet pas de vérifier le prédicat *Pre-Free*.

  Ainsi, le prédicat *Pre-2HopNeighTrans* ne peut pas être vrai si un des autres prédicats de P est vrai.
- Supposons que le prédicat *Pre-NeighTR* soit l'un de ces deux prédicats. Pour qu'il soit vérifié, un unique récepteur doit se trouver dans le voisinage, ainsi qu'au moins un émetteur. Or :
  - la présence d'un voisin dans l'état T ne permet pas aux prédicats *Pre-NeighRcv*, *Pre-2HopNeighTrans*

et *Pre-Free* d'être vrais.
  - la présence d'un récepteur dans le voisinage ne permet pas au prédicat *Pre-1HopNeighTrans* d'être vrai.
  - Au moins deux récepteurs doivent être présents pour que le prédicat *Pre-NeighTR+* soit vérifié.

  Ainsi, le prédicat *Pre-NeighTR* ne peut pas être vrai si un des autres prédicats de P est vrai.
- Supposons que le prédicat *Pre-NeighTR+* soit l'un de ces deux prédicats. Pour qu'il soit vérifié, au moins deux récepteurs doivent se trouver dans le voisinage, ainsi qu'au moins un émetteur. Or :
  - la présence d'un voisin dans l'état T ne permet pas aux prédicats *Pre-NeighRcv*, *Pre-2HopNeighTrans* et *Pre-Free* d'être vrais.
  - la présence d'un récepteur dans le voisinage ne permet pas au prédicat *Pre-1HopNeighTrans* d'être vrai.
  - un *unique* récepteur doit être présent pour que le prédicat *Pre-NeighTR* soit vérifié.

  Ainsi, le prédicat *Pre-NeighTR+* ne peut pas être vrai si un des autres prédicats de P est vrai.
- Supposons que le prédicat *Pre-Free* soit l'un de ces deux prédicats. Pour qu'il soit vérifié, les seuls valeurs de *channelState* autorisés dans le voisinage sont les états N_R,N_{T2} ou N_F. Or :
  - la présence d'un voisin dans l'état T est nécessaire pour vérifier les prédicats *Pre-1HopNeighTrans*, *Pre-NeighTR*, *Pre-2HopNeighTrans* et *Pre-NeighTR+*.
  - la présence d'un voisin dans l'état R est nécessaire pour vérifier le prédicat *Pre-NeighRcv*.

  Ainsi, le prédicat *Pre-Free* ne peut pas être vrai si un des autres prédicats de P est vrai.

Deux prédicats de l'ensemble P ne peuvent donc pas être simultanément vrai.

□

Proposition 5.3.3. *Tous les conflits à un saut sont corrigés à la fin de la première ronde.*

Démonstration. Supposons par l'absurde qu'à la fin de la première ronde un conflit existe entre un émetteur A_T et un de ses voisins, un récepteur B_R. Cette situation est représentée sur la figure 5.9 où A_T est associé avec le récepteur A'_R et B_R avec l'émetteur B'_R.

Dans l'algorithme, il n'existe aucune action qui permette à une variable *channelState* de passer dans l'état T ou R. Ces deux nœuds étaient donc dans le même état au début de la première ronde. Puisque les nœuds A_T et B_R ne sont pas associés, les prédicats *T-1HopInterf*$_{A_T}$ et *R-1HopInterf*$_{B_R}$ sont vrais. Puisque ces prédicats sont vrais, alors les prédicats *IncorrectT*$_{A_T}$ et *IncorrectR*$_{B_R}$ sont également valides.

D'après les lemmes 1 et 2, tout nœud a une et une seule action activable au départ de la première ronde. Puisque durant une ronde, chaque nœud activable est activé, la variable *channelState* ne peut pas être dans l'état T pour A_T et dans l'état R pour B_T à la fin de la première ronde. □

Dans cet algorithme, le principal objectif est la correction des conflits. La première proposition a démontré que l'algorithme était capable de corriger l'ensemble des conflits à un saut à la fin de la première ronde. Dans le cas des conflits à deux sauts, les nœuds interférant ne sont pas voisins, mais possèdent au moins un voisin commun comme le montre la figure 5.10 où les nœuds A_T et B_R sont

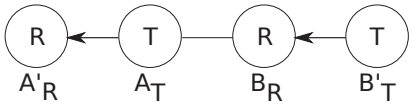


FIGURE 5.9: Conflit à un saut

en conflit. C'est sur ce voisin que l'algorithme va s'appuyer pour détecter le conflit. La correction des conflits à deux sauts se fait donc en deux temps. Tout d'abord, le voisin commun doit mettre à jour sa variable *channelState* pour indiquer qu'il y a des émetteur(s) et récepteur(s) dans son voisinage. Une fois que cette mise à jour est réalisée, A_T et B_R sont capables de détecter le conflit et de le corriger.

La seconde proposition indique que la correction de ces conflits à deux sauts est réalisée en au plus deux rondes.

Proposition 5.3.4. *À la fin de la seconde ronde, tous les conflits à deux sauts sont corrigés.*

Démonstration. Supposons par l'absurde qu'il existe un conflit à deux sauts à la fin de la seconde ronde entre un émetteur A_T et un récepteur B_R. Puisque c'est un conflit à deux sauts, il existe au moins un nœud C_i voisin de A_T et de B_R. Cette configuration est représentée sur la figure 5.10. Pour prouver cette proposition, nous allons montrer que la présence de C_i, quelle que soit la valeur de sa variable *channelState* permet aux nœuds en conflit de détecter et de corriger le conflit existant entre A_T et B_R avant la fin de la seconde ronde.

- Supposons que C_i soit un émetteur ou un récepteur :
  - Si C_i n'est pas associé à A_T ou B_R, il y a conflit à un saut entre C_i et ces deux nœuds. Ce type de conflit est résolu en une ronde d'après la proposition 1.
  - Si C_i est le récepteur associé à A_T, alors B_R est voisin d'un récepteur. Nous savons que dans un état correct, chaque récepteur est associé avec un émetteur. Par conséquent, lorsque deux récepteurs sont voisins, l'un des récepteurs est en conflit avec l'émetteur associé de l'autre récepteur. Lorsqu'un tel cas apparaît, le prédicat *R-2HopInterf* est vérifié ce qui rend le prédicat *IncorrectR* vrai pour chacun des récepteurs. Puisque *IncorrectR* est valide et qu'un des prédicats du lemme 1 est valide également, une des actions est activable au début de la première ronde pour ces deux récepteurs. Par conséquent, la variable *channelState* de ces nœuds n'est plus dans l'état R à la fin de la première ronde. Il n'y a donc plus d'émetteur à deux sauts d'un récepteur, le conflit est corrigé.

  - Si C_i est l'émetteur associé à B_R alors A_T est voisin d'un émetteur. Pour les mêmes raisons que celles évoquées ci-dessus, lorsque deux émetteurs sont voisins, l'un des émetteurs est en conflit avec le récepteur associé à l'autre émetteur. Cette situation où deux émetteurs sont voisins est détectée par la validation du prédicat *T-2HopInterf*. Par le même mécanisme, ces deux émetteurs possèdent une action activable au début de la première ronde. Par conséquent, la variable *channelState* de ces deux nœuds ne peut plus être dans l'état à la fin de la première

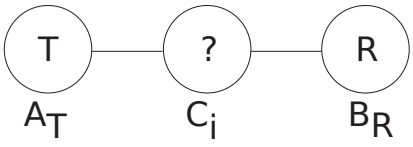


FIGURE 5.10: Conflits à deux sauts

ronde.

- Supposons maintenant les cas où C_i n'est ni émetteur ni récepteur :
  - Si C_i n'est voisin d'aucun autre récepteur que B_R alors C_i est voisin d'un unique récepteur et d'au moins un émetteur. Le prédicat *Pre-NeighTR*$_{C_i}$ est donc vrai. Si C_i n'est pas encore dans l'état N_{TR}, la validation de ce prédicat rend l'action N_{TR}-*Action*$_{C_i}$ activable au début de la première ronde. Ainsi quelle que soit la situation, C_i est dans l'état N_{TR} à la fin de la première ronde. Au début de la seconde ronde, le nœud A_T est donc voisin d'un nœud dans l'état N_{TR} et qui n'est pas voisin de son récepteur associé. Le prédicat *T-2HopInterf*$_{A_T}$ est donc vrai ce qui permet de valider le prédicat *IncorrectT*$_{A_T}$. le nœud A_T est donc activable et ne sera plus dans l'état T à la fin de la seconde ronde.
  - Si C_i possède plusieurs récepteurs dans son voisinage, le prédicat *Pre-NeighTR+* est vrai puisqu'il possède également un émetteur (A_T). Par conséquent, la variable *channelState* de C_i est dans l'état N_TR+ à la fin de la première ronde. Le nœud A_T possède donc un voisin dans l'état N_{TR+} au début de la première ronde ce qui valide le prédicat *T-2HopInterf*$_{A_T}$ et donc le prédicat*IncorrectT*$_{A_T}$. Le nœud A_T est donc activable et sa variable *channelState* quitte l'état T avant la fin de la seconde ronde.

Ainsi, quelle que soit la valeur de la variable *channelState* du nœud C_i, le conflit entre A_T et B_R est détecté et corrigé avant la fin de la seconde ronde. □

Grâce aux deux propositions précédentes, nous avons montré que l'algorithme permettait de corriger l'ensemble des conflits existants. À la fin de la seconde ronde, l'ensemble des couples émetteurs-récepteurs associés sont stables puisqu'il n'existe plus de conflits à corriger. L'algorithme doit maintenant stabiliser l'état des autres nœuds du réseau. Cette convergence va se faire en plusieurs étapes. Chaque étape permettant de stabiliser un groupe de nœuds particulier. Les premiers nœuds à atteindre l'état stable sont donc ceux dont la variable *channelState* est dans l'état T ou R. Cette stabilisation va ensuite se propager une ronde plus tard à leurs voisins ; les nœuds dans l'état N_{T1}, N_R, N_{TR} et N_{TR+} forment donc le second groupe à se stabiliser. Enfin, cette vague de stabilisation atteindra les nœuds dans l'état N_{T2} et F. Lorsque cette dernière phase de stabilisation sera terminée, le système sera alors totalement stable. La prochaine proposition s'intéresse à la première phase : la stabilisation des nœuds dans l'état T et R. Les deux premières rondes ayant permis de corriger tous les conflits, l'ensemble des émetteurs-récepteurs des allocations restantes sont déjà dans un état stable au début de la seconde ronde. Il reste cependant des nœuds dans l'état T ou R qui n'appartiennent plus à une allocation valide. En effet, les corrections de conflits des deux premières rondes ont pu faire apparaître des nœuds isolés, c'est-à-dire des émetteurs

qui n'ont plus de récepteurs associés ou des récepteurs qui n'ont plus d'émetteurs associés. les prédicats *T-Isolate* et *R-Isolate* de l'algorithme sont là pour détecter ces situations et forcer ces nœuds à quitter leur état T ou R. La correction de l'état de ces nœuds nécessite une ronde. La proposition suivante prouve qu'à la fin de la troisième ronde, l'ensemble de ces nœuds isolés a quitté l'état T ou R.

Proposition 5.3.5. *l'ensemble des nœuds dont la variable* channelState *est dans l'état R ou T atteint l'état stable avant la fin de la troisième ronde*

Démonstration. Nous considérons qu'un ensemble est stable si et seulement si aucun élément ne peut quitter ou rejoindre cet ensemble. De manière générale, nous appellerons E_X l'ensemble des nœuds dont la variable *channelState* est dans l'état X.
Nous rappelons qu'un noeud, qu'il soit émetteur ou récepteur, est dit isolé s'il n'a pas de nœud associé. Par exemple dans le cas d'un émetteur A associé à un récepteur B, si B change d'état à cause de la détection d'un conflit, A sera dit isolé car il aura perdu son récepteur associé. La proposition 1 prouve que les conflits à un saut sont corrigés à la fin de la première ronde. Pour corriger ce conflit, l'algorithme change l'état de la variable *channelState* pour au moins un des nœuds en conflit. Ces nœuds chez qui cette variable est modifiée peuvent être associés avec d'autre nœuds qui se retrouvent isolés. Une telle situation est détectée par les prédicats *R-Isolate* dans le cas d'un récepteur isolé et *T-Isolate* dans le cas d'un émetteur isolé. Ces deux prédicats lorsqu'ils sont vrais entraîne la validation des prédicats *IncorrectR* ou *IncorrectS*. Ces derniers forcent ces nœuds isolés à quitter leur état T ou R en rendant une des actions activable. Puisque cette situation est détectable dès le début de la seconde ronde, ces nœuds sont activables à ce moment. Ils ne peuvent donc plus être dans l'état T ou R à la fin de la seconde ronde.

La proposition 2 prouve quant à elle que les conflits à deux sauts sont corrigés à la fin de la seconde ronde. La correction de ces conflits peut également faire apparaître des nœuds isolés. Cette situation est détectable dès le début de la troisième ronde. Pour les mêmes raisons que précédemment, ces nœuds sont alors activables. Les nœuds isolés suite à la correction des conflits à deux sauts ne peuvent donc plus être dans l'état T ou R à la fin de la troisième ronde.
Puisque tous les conflits sont résolus, et que les nœuds isolés suite à la correction de ces conflits sont corrigés, plus aucun nœud dans l'état T ou R ne peut changer d'état. Aucun nœud ne peut donc plus quitter l'ensemble $\mathrm{E}_{T,R}$. De plus, l'algorithme ne possède pas d'actions permettant à un nœud de passer dans l'un de ces deux états. Aucun nœud ne peut rejoindre cet ensemble $\mathrm{E}_{T,R}$. L'ensemble $\mathrm{E}_{T,R}$ est donc stable avant la fin de la troisième ronde. □

Cette stabilisation va donc pouvoir se propager aux nœuds dont l'état dépend de la présence d'émetteur ou de récepteur dans le voisinage. Ainsi la quatrième ronde va permettre de stabiliser l'ensemble $\mathrm{E}_{N_R,N_{TR},N_{TR+},N_{T1}}$ des nœuds dans l'état N_R, N_{TR}, N_{TR+} ou N_{T1}.

Proposition 5.3.6. *Tout nœud dans l'état N_R, N_{TR}, N_{TR+} ou N_{T1} est stable à la fin de la quatrième ronde.*

Démonstration. Nous considérons d'abord l'ensemble E_{N_R} des nœuds dont *channelState* est dans l'état N_R. Pour qu'un nœud puisse rejoindre cet ensemble, il faut que l'action N_R-*Action* soit activée. Pour cela, le prédicat *Pre-1HopNeighRcv* doit être vérifié. Ce prédicat dépend uniquement de nœuds appartenant aux ensembles E_T et E_R. D'après la proposition 3, ces ensembles sont stables à la fin de la troisième ronde. Par conséquent, si le prédicat *Pre-1HopNeighRcv* est vrai pour un nœud, il l'est dès la fin de la troisième ronde. Puisque tout nœud activable au cours d'une ronde est activé, les derniers nœuds à rejoindre l'ensemble E_{N_R} le font avant la fin de la quatrième ronde.
Il en va de même pour les autres ensembles $E_{N_{TR}}, E_{N_{TR+}}$ et $E_{N_{T1}}$ dont les prédicats dépendent également des nœuds appartenant aux ensembles E_T et E_R. Ainsi, aucun nœud ne peut rejoindre les ensembles $E_{N_R}, E_{N_{TR}}, E_{N_{TR+}}$ et $E_{N_{T1}}$ après la fin de la quatrième ronde.
Supposons maintenant qu'un nœud A quitte l'ensemble E_{N_R} durant la cinquième ronde. Nous avons vu qu'aucun nœud ne peut rejoindre les ensembles $E_{N_{TR}}, E_{N_{TR+}}$ et $E_{N_{T1}}$ et qu'aucune règle ne permet à un nœud de passer dans un état T ou R. A quitte donc l'état N_R pour l'état N_{T2} ou F. Ce changement d'état au cours de la cinquième ronde signifierait que A a perdu les récepteurs se trouvant dans son voisinage et responsable de son état N_R. Or l'ensemble E_R est stable depuis la fin de la troisième ronde. A ne peut donc pas quitter l'état N_R durant la cinquième ronde. L'ensemble E_{N_R} est donc stable à la fin de la quatrième ronde.
De même, un nœud B appartenant à un des ensembles $E_{N_{TR}}, E_{N_{TR+}}$ ou $E_{N_{T1}}$ et qui changerait d'état au cours de la cinquième ronde ne pourrait rejoindre que l'ensemble E_F ou $E_{N_{T2}}$. Or, quelle que soit l'ensemble auquel appartient B, ce changement ne pourrait avoir lieu qu'en cas de disparition d'émetteur et/ou de récepteurs dans son voisinage. Ce qui est rigoureusement impossible après la fin de la troisième ronde d'après la proposition 3. Aucun nœud ne peut donc quitter ou rejoindre les ensembles $E_{N_{TR}}, E_{N_{TR+}}$ et $E_{N_{T1}}$ après la fin de la quatrième ronde. □

Proposition 5.3.7. *Tout nœud dans l'état N_{T2} ou F est stable avant la fin de la cinquième ronde.*

Démonstration. Supposons par l'absurde qu'au cours de la sixième ronde, un nœud A rejoint ou quitte l'ensemble E_F ou $E_{N_{T2}}$. D'après les propositions 3 et 4, les ensembles de nœuds E_T, E_R, E_{N_R}, $E_{N_{TR}}$, $E_{N_{TR+}}$ et $E_{N_{T1}}$ sont stables à la fin de la quatrième ronde. Par conséquent, le nœud A ne peut quitter ou joindre ces états durant la sixième ronde. Les échanges ne peuvent donc avoir lieu, à partir de la fin de la quatrième ronde, qu'entre les ensembles E_F et $E_{N_{T2}}$.

Supposons que A quitte l'ensemble E_F pour rejoindre l'ensemble $E_{N_{T2}}$ durant la sixième ronde. Dans ce cas, le prédicat *Pre-2HopNeighTransp*$_A$ doit être vrai. Puisque ce prédicat dépend uniquement de nœuds appartenant aux ensembles déjà stabilisés à la fin de la quatrième ronde, *Pre-2HopNeighTransp*$_A$ est vrai depuis le début de la cinquième ronde. Par conséquent, si A doit rejoindre l'ensemble $E_{N_{T2}}$, il le fait au plus tard durant la cinquième ronde.

Supposons maintenant que A quitte l'ensemble $E_{N_{T2}}$ pour rejoindre l'ensemble E_F,. Dans ce cas, c'est le prédicat *Pre-Free*$_A$ qui doit être vrai. Puisque celui-ci dépend également des états appartenant

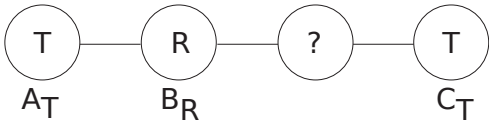


FIGURE 5.11: A_T est responsable du conflit

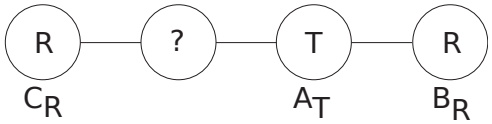


FIGURE 5.12: B_R subit le conflit

aux ensembles stables depuis la fin de la quatrième ronde, s'il doit changer d'état, il le fait au plus tard au cours de la cinquième ronde.

Ainsi, aucun nœud ne peut quitter ou rejoindre les ensembles E_F et $E_{N_{T2}}$ au cours de la sixième ronde. Ces ensembles sont donc stables à la fin de la cinquième ronde. □

Preuve de vivacité : Tout système atteint l'état stable en au plus cinq rondes. Quel que soit le graphe de connectivité, la valeur des variables *channelState* et la taille du réseau.

D'après les propositions 3, 4 et 5, tous les ensembles de nœuds sont stables à la fin de la cinquième ronde. Par conséquent, l'ensemble du réseau est stable à la fin de la cinquième ronde.

Preuve de sûreté 1 : Un système stable ne contient aucun conflit d'allocation.

Si un réseau présente des conflits alors qu'il a atteint l'état stable, cela signifie que des conflits n'ont pas été détectés. D'après notre définition des conflits, ces derniers ne peuvent survenir qu'entre un émetteur et un récepteur situé à au plus deux sauts. Les propositions 1 et 2 ont montré que ce type de conflit pouvait être détecté et corrigé. Il ne peut donc pas rester de conflit dans un système stable.

Preuve de sûreté 2 : Dans un système stable, tout nœud p de V dont la variable *channelState* est dans l'état F ou N_{T2} et qui ne possède pas de voisin dans l'état N_R peut créer une allocation vers un voisin dans l'état F sans créer de conflit.

Soit A_T un nœud appartenant à l'ensemble E_F ou $E_{N_{T2}}$ et qui ne possède pas de voisin dont *channelState* est dans l'état N_R. Soit B_R un nœud appartenant à l'ensemble E_F. Supposons par l'absurde qu'un conflit survienne lorsqu'une ressource est allouée entre l'émetteur A_T et le récepteur B_R.

- Supposons le cas d'un conflit à un saut. Dans un tel cas, B_R est un voisin d'un émetteur ou A_T est voisin d'un récepteur. Puisque le système est stable, la variable *channelState* de ces deux nœuds ne pourraient pas être dans l'état F ou N_{T2}. Un conflit à un saut ne peut donc pas survenir.

- Supposons le cas d'un conflit à deux sauts :
  - Si A_T est responsable du conflit, alors il existe un récepteur C_R à deux sauts de A_T. Dans ce

cas, représenté par la figure 5.11, le voisin commun de ces deux nœuds doit être dans l'état N_R puisque le système est stable, ce qui est impossible d'après nos hypothèses. Il ne peut donc pas exister de récepteur à deux sauts de A_T.

– Si B_R subit le conflit, alors il existe un nœud C_T situé à deux sauts de B_R. Dans ce cas, représenté par la figure 5.11, le voisin commun de ces deux nœuds doit être dans l'état N_{T1} puisque le système est stable. Ce qui est impossible puisque dans ce cas, B_R ne pourrait pas être dans l'état F mais N_{T2}. Il ne peut donc pas exister d'émetteur à deux sauts de B_R. La création de cette allocation ne peut donc pas créer de conflit à un ou deux sauts. Dans un système stable, tout nœud p de V dont la variable *channelState* est dans l'état F ou N_{T2} et qui ne possède pas de voisin dans l'état N_R peut donc créer une allocation vers un voisin dans l'état F sans créer de conflit.

5.4 Algorithme 2 : Prise en compte de la priorité des arcs

Dans l'algorithme présenté au paragraphe précédent, la correction des conflits passe par la suppression de certaines allocations. Or, ces ressources étaient utilisées pour la communication dans le réseau. Un algorithme de répartition de ressources doit donc se charger de trouver de nouveaux canaux disponibles pour les remplacer. Cependant, ces nouvelles allocations vont représenter un travail supplémentaire pour cet algorithme de répartition. De plus, tant que ces ressources n'ont pas été réattribuées, le débit entre certains nœuds peut être bridé, ce qui peut être particulièrement gênant dans le cas de communications audio ou vidéo. Dans le pire des cas, la connexion peut même être rompue lorsque cette libération affecte le dernier canal disponible pour un lien de communication. Il est donc important que l'algorithme de correction de conflits limite la quantité d'allocations supprimées.

Certaines allocations peuvent également être considérées comme plus importantes que d'autres, par exemple la libération d'un canal nécessaire au maintient de la connexité du réseau doit être évitée. Choisir intelligemment les allocations à conserver peut donc avoir son utilité. Pour cette raison, nous proposons une nouvelle approche de la correction de conflits qui prend en compte la priorité des allocations. Afin de pouvoir utiliser cet algorithme dans des conditions réelles, celui-ci doit être écrit dans le modèle à passage de message. Pour cela, chaque nœud émet de manière périodique un message à destination de ses voisins. Ces messages permettent de transmettre la liste des voisins, l'état des variables *channelState* ainsi que la valeur des poids, présentée dans la suite de ce paragraphe, pour chaque canal. Le fonctionnement de la correction de conflits est donné par l'algorithme 5.4.1 dont nous allons maintenant détailler le principe de fonctionnement.

L'algorithme présenté dans le paragraphe précédent permettait de garantir la correction de conflits avec un nombre de variables partagées. Cependant cet algorithme ne cherchait ni à limiter le nombre de libérations de ressources ni à privilégier certaines allocations lors de la phase de correction. La figure 5.13 montre un cas où l'algorithme précédent libère plus de ressources que nécessaire. Dans cet exemple, la topologie du réseau est constituée de quatre nœuds sur lesquels sont effectués deux allocations du

même canal. Une première entre les nœuds A_T et A_R, une seconde entre les nœuds B_T et B_R. Ces deux

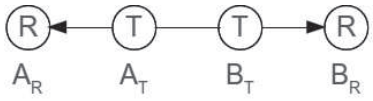


FIGURE 5.13: Exemple de symétrie

allocations sont en conflits puisque l'émetteur d'un couple se trouve à deux sauts du récepteur de l'autre couple. La solution la plus efficace pour corriger ce conflit consiste à libérer une des deux allocations et à laisser l'autre intacte. L'algorithme d'allocation va permettre aux deux émetteurs de détecter ce conflit grâce au prédicat $T-2HopInterf_p$. Ce prédicat forcera ces deux nœuds à quitter leur statut d'émetteur et donc à libérer le canal, corrigeant ainsi le conflit. Cependant, cette correction entraîne la libération des deux allocations, alors qu'une seule aurait suffi.

Cette double libération est due à la présence d'une symétrie dans la topologie. En effet, le point de vue des émetteurs A_T et B_T est strictement identique. L'algorithme étant déterministe, deux conditions identiques entraînent deux réactions identiques. Ces deux émetteurs vont donc appliquer les mêmes actions, qui conduiront à la libération des deux allocations.

Pour éviter ce genre de situations il faut pouvoir casser cette symétrie. La solution consiste généralement à attribuer un identifiant unique à chaque élément. Cet identifiant permettant de choisir, parmi ces deux émetteurs, celui qui doit effectuer la libération. La question est alors de savoir comment choisir cet identifiant. Affecter des identifiants arbitraires mais uniques permet de résoudre le problème, cependant le choix entre deux allocations sera lui aussi arbitraire et ne permettra pas de prendre en compte l'importance relative des allocations. Une autre solution consiste à affecter un poids à chaque allocation, représentative de son importance. Le choix entre plusieurs allocations est alors plus efficace. Cependant, il est difficile de garantir que ce poids soit unique pour l'ensemble des allocations du réseau. Il faut ainsi choisir entre un identifiant unique mais non représentatif ou un poids représentatif mais qui ne garantit pas que le problème de symétrie sera résolu.

Dans notre nouvel algorithme de correction de conflits, nous avons décidé d'utiliser une solution mixte dans laquelle l'ordre de priorité des allocations est déterminé à partir de deux critères exprimés par les poids :

- P_1 permettant de fixer un poids représentatif de l'importance de l'allocation,
- P_2 correspondant à un identifiant unique afin de garantir la suppression de toute symétrie y compris lorsque le poids P_1 de plusieurs allocations est identique.

Lorsque l'algorithme doit choisir entre la libération de plusieurs allocations, il choisira de conserver celle possédant le poids P_1 le plus élevé. En cas d'égalité, le choix portera alors sur le second critère P_2.

Le choix de P_1 dépend fortement de la stratégie de répartition adoptée. C'est la raison pour laquelle nous supposerons que la valeur de ce critère est attribuée par l'algorithme de répartition de ressources. Par exemple, P_1 peut prendre la valeur du poids P ou du poids modifié P' qui sont définis dans les algorithmes 5.3.1 et 5.4.1.

Par contre, le poids P_2 ne dépend pas de la stratégie de répartition. Nous proposons donc une solution permettant d'attribuer une valeur unique pour chaque allocation utilisant le même canal. Chaque allocation permet d'attribuer un canal à un couple émetteur-récepteur particulier. Avec la réutilisation spatiale, il est possible d'attribuer un même canal à plusieurs couples émetteur-récepteur. Chaque canal c est donc utilisé par un ensemble de couples émetteur-récepteur, noté E^c_{ER}. Puisqu'un nœud ne peut pas utiliser la même ressource pour deux communications différentes, chaque émetteur de l'ensemble E^c_{ER} est unique. En utilisant l'identifiant de l'émetteur comme identifiant de l'allocation et en supposant que chaque nœud possède un identifiant unique, il est possible d'attribuer un identifiant unique à chaque allocation du réseau. Pour cette raison, dans notre algorithme, la valeur de P_2 pour chaque allocation est déterminé par l'identifiant de l'émetteur. La figure 5.14 reprend l'exemple de la figure 5.13 mais avec

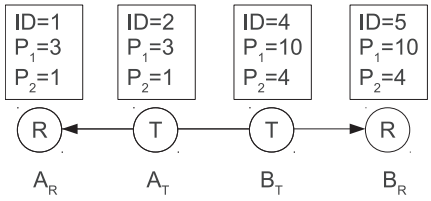


FIGURE 5.14: Utilisation des critères d'ordonnancement

cette fois les critères P_1 et P_2. Il est possible de voir que pour chacune des allocations, la valeur de P_2 correspond à l'identifiant de l'émetteur. Dans cet exemple, le couple B_T et B_R possède une valeur de P_1 plus élevée. L'algorithme d'allocation va donc privilégié cette allocation et supprimer l'allocation entre A_T et A_R pour corriger le conflit.

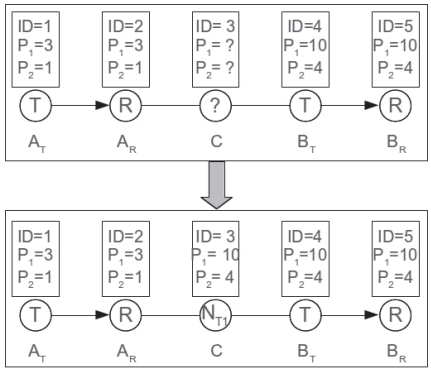


FIGURE 5.15: Détection d'un conflit à deux sauts

Dans le cas de la figure 5.15, les deux allocations sont cette fois-ci séparées par un nœud C. Avec l'algorithme 5.3.1, la variable *channelState* de ce nœud passe dans l'état N_{TR}, indiquant la présence d'un émetteur et d'un récepteur dans le voisinage. Le prédicat $T - 2HopInterf_p$ du nœud B_T se serait alors activé entraînant la libération de l'allocation entre les nœuds B_T et B_R. L'algorithme aurait donc corrigé le conflit en supprimant un minimum de ressources. Cependant le choix de l'allocation à supprimer aurait été réalisé de manière totalement arbitraire.

Avec l'utilisation des nouveaux critères, le nœud C à la capacité d'identifier quelle allocation est

prioritaire, ici l'allocation B_T et B_R puisque c'est elle qui possède le poids P_1 le plus élevé. Le nœud C doit ensuite choisir son état et la valeur de ses propres variables pour informer le nœud A_R qu'il doit libérer le canal. Dans le nouvel algorithme, la valeur de *channelState* n'est plus déterminée en fonction de l'état de tous les voisins mais uniquement de ceux qui ont la priorité la plus élevé. Dans l'exemple de la figure 5.15, le nœud C va donc fixer l'état de *channelState* à N_{T1} et la valeur de ses variables à $P_1 = 10$ et $P_2 = 4$. Il indique ainsi qu'il est voisin d'un émetteur dont la priorité est fixée par le couple ($P_1 = 10$, $P_2 = 4$). Ainsi, le nœud A_R va détecter la présence d'un émetteur dont la priorité est plus élevé que la sienne et va donc libérer le canal.

Avec ce système l'état N_{TR+} n'est plus nécessaire. En effet cet état signifie qu'un nœud est voisin d'un émetteur et d'au moins deux récepteurs. Dans notre nouvel algorithme, cela signifierait que ces différents nœuds possède une priorité identique. Or, le couple (P_1,P_2) est unique pour chaque allocation. Ainsi, seuls l'émetteur et le récepteur d'une allocation peuvent posséder la même valeur de (P_1,P_2). Il est donc impossible d'avoir un émetteur et deux récepteurs possédant le même couple (P_1, P_2). C'est pourquoi nous avons décidé de supprimer cet état. En contre-partie nous en ajoutons un nouveau, l'état N_{R2} indiquant la présence d'un récepteur à deux sauts. Cet état n'apporte aucun changement au niveau de la correction de conflits, mais il va permettre à l'algorithme de répartition des ressources d'être plus efficace dans la réutilisation spatiale des ressources.

Dans le paragraphe 5.3.2, nous avons vu que dans l'algorithme 5.3.1, l'information sur le voisinage ($\{N_R, N_{TR}, N_{T1}\}$) est prioritaire sur le 2-voisinage (N_{T2}) et sur l'état par défaut (F), et que lorsque ce critère ne suffit pas, la priorité se fait sur la quantité d'information véhiculée par l'état de la variable ($N_{TR} > N_{T1} \geq N_R$ et $N_{T2} > N_{R2} > F$).

Pour conserver cette hiérarchisation notre nouvel algorithme étudie par ordre de priorité :

1. l'ensemble des voisins dans l'état T ou R, appelé SET_1. Si cet ensemble est vide il passe à l'étape suivante,
2. l'ensemble des voisins dans l'état $\{N_{T1}, N_{TR}\}$, appelé SET_2. si cet ensemble est vide, il passe à l'étape suivante,
3. l'ensemble des voisins dans l'état $\{N_{R1}\}$, appelé SET_3.

L'algorithme 5.4.1 [PTBV11] résume le fonctionnement de la correction de conflits. Il fonctionne donc de la manière suivante pour chaque nœud p et chaque canal c :

– Si le canal c est utilisé pour émettre (lignes 1 à 6) :
  - la ligne 2, qui correspond au prédicat $T - Isolated_p$ de l'algorithme 5.3.1, permet de vérifier si l'émetteur est isolé. Si c'est le cas, l'erreur est corrigée par la ligne 3 qui force la variable *channelState* à prendre l'état F. L'état correct sera ensuite trouvé à l'aide du bloc de lignes 13 à 30.
  - Les lignes 4 et 5 permettent de détecter la présence de conflits et correspondent aux règles $T-1HopInterf_p$ et $T-2HopInterf_p$. La correction se fait également en positionnant la variable *channelState* dans l'état F.

– Si le canal c est utilisé pour recevoir (lignes 7 à 12) :
  - la ligne 2, qui correspond au prédicat $R-Isolated_p$, permet de vérifier si ce récepteur est isolé. Si c'est le cas, l'erreur est corrigée par la ligne 3 qui force la variable *channelState* à prendre l'état F. L'état correct sera ensuite trouvé à l'aide du bloc de lignes 13 à 30.
  - les lignes 10 et 11 correspondent aux nouveaux prédicats $R-1HopInterf_p$ et $R-2HopInterf_p$ permettant de détecter la présence d'un conflit lorsque p est récepteur. La ligne 12 permet alors de corriger le conflit.
– Si le canal c n'est utilisé ni pour recevoir ni pour émettre (lignes 13 à 30) :
  - l'algorithme isole l'ensemble SET_1 des voisins de p dont la valeur de la variable *channelState* pour le canal c est dans l'état T ou R. Si cet ensemble n'est pas vide, l'algorithme entre alors dans le bloc de ligne 14 à 21. Dans celui-ci les conditions des lignes 16 et 17 permettent de reproduire le prédicat $Pre-1HopNeighTrans_p$. Si ces conditions sont vérifiées, la variable *channelState* passe alors dans l'état N_{T1} (ligne 18). Les conditions 16 et 19 correspondent au prédicat $Pre-1HopNeighTR_p$. Leur vérification entraîne le passage de la variable *channelState* dans l'état N_{TR} (ligne 20). La condition de la ligne 21 correspond au prédicat $Pre-1HopNeighRcv_p$. Si elle est vérifiée, la variable *channelState* passe alors dans l'état N_{R1} (ligne 22). Les valeurs des poids $P_{1,c}$ et $P_{2,c}$ correspondent à celles de l'élément de SET_1 possédant la priorité maximale.
  - Si aucun voisin n'est dans l'état T ou R, l'algorithme isole l'ensemble SET_2 des voisins de p dont la valeur de la variable *channelState* pour le canal c est dans l'état N_{T1} ou N_{TR}. Si cet ensemble n'est pas vide, l'algorithme fixe la variable *channelState* dans l'état N_{T2} (ligne 24). Les valeurs des poids $P_{1,c}$ et $P_{2,c}$ correspondent à celles de l'élément de SET_2 possédant la priorité maximale.
  - Si les ensembles SET_1 et SET_2 sont vides, l'algorithme isole l'ensemble SET_3 des voisins de p dont la valeur de la variable *channelState* pour le canal c est dans l'état N_{R1}. Si cet ensemble n'est pas vide, l'algorithme fixe la variable *channelState* dans l'état N_{R2} (ligne 28). Les valeurs des poids $P_{1,c}$ et $P_{2,c}$ correspondent alors à celles de l'élément de SET_3 possédant la priorité maximale.
  - Si les ensemble SET_1, SET_2 et SET_3 sont tous vides, c'est-à-dire si l'ensemble des voisins se trouve dans l'état N_{R2}, N_{T2} ou F, l'algorithme fixe la variable *channelState* dans l'état N_{R2} (ligne 28). La valeur des poids $P_{1,c}$ et $P_{2,c}$ est alors fixée à une valeur arbitraire de -1. Cependant, le poids d'un canal dans l'état F n'était jamais vérifié, la valeur fixée est sans importante.

Cet algorithme 5.4.1 permet donc de corriger les conflits en tenant compte de la priorité des différentes allocations. Cette priorité est obtenue à l'aide de deux poids P_1 et P_2. Le premier critère permet d'affecter un poids à chaque allocation en fonction de la stratégie d'allocation retenue. Le second critère permet quant à lui de départager deux allocations qui partagent la même valeur de P_1. L'association de ces deux critères permet donc d'obtenir une relation d'ordre stricte sur les différentes allocations du réseau. Cet algorithme est également conçu pour fonctionner dans le modèle à passage de message afin d'être

Algorithme 5.4.1: Algorithme de correction de conflits avec gestion de poids

function : $updateStates_{p,i}(newRBState, newP_{1,i}(p), newP_{2,i}(p))$: update RBstate, $P_{1,i}(p)$ and $P_{2,i}(p)$ of i for p

1 **if** $RBState_i(p) = T$ **then**
2 | **if** $RBState_i(PA_i(p)) \notin \{R, F, N_{R2}\}$ ***OU*** *(*$RBState_i(PA_i(p)) \in \{R, N_{R2}\}$ ***ET*** $PA_i(PA_i(p)) \neq p$*)* **then**
3 | | $updateStates_{p,i}(F, -1, -1)$;
4 | **else if** $\exists v \in V(p) \setminus \{PA_i(p)\}$, $P_{1,i}(v) > P_{1,i}(p)$ ***OU*** *(*$P_{1,i}(v) = P_{1,i}(p)$ ***ET*** $P_{2,i}(v) > P_{2,i}(p)$*)* **then**
5 | | **if** $RBState_i(v) \in \{T, R, N_{R1}, N_{TR}\}$ **then**
6 | | | $updateStates_{p,i}(F, -1, -1)$;
7 **if** $RBState_i(p) = R$ **then**
8 | **if** $RBState_i(PA_i(p)) \neq T$ ***OU*** $PA_i(PA_i(p)) \neq p$ **then**
9 | | $updateStates_{p,i}(F, -1, -1)$;
10 | **else if** $\exists v \in V(p) \setminus \{PA_i(p)\}$, $P_{1,i}(v) > P_{1,i}(p)$ ***OU*** *(*$P_{1,i}(v) = P_{1,i}(p)$ ***ET*** $P_{2,i}(v) > P_{2,i}(p)$*)* **then**
11 | | **if** $RBState_i(v) \in \{T, R, N_{T1}, N_{TR}\}$ **then**
12 | | | $updateStates_{p,i}(F, -1, -1)$;
13 **if** $RBState_i(p) \notin \{T, R\}$ **then**
14 | **if** $\exists\ v \in Voisins(p),\ RBState_i(v) \in \{T, R\}\}$ **then**
15 | | $SET_1 \leftarrow \{v \in Voisins(p) | RBState_i(v) \in \{T, R\}\}$;
| | $V_{P_{MAX}} \leftarrow \{v \in Voisins_{P_{1,i}^{MAX}(SET_1)} | P_{2,i}(v) = P_{2,i}^{MAX}(Voisins_{P_{1,i}^{MAX}(SET_1)})\}$;
16 | | **if** $\exists\ v_T \in V_{P_{MAX}},\ RBState_i(v_T) = T$ **then**
17 | | | **if** $\nexists\ v_R \in V_{P_{MAX}}, RBState_i(v_R) = R$ **then**
18 | | | | $updateStates_{p,i}(N_{T1}, P_{1,i}^{MAX}(SET_1), P_{2,i}^{MAX}(SET_1))$;
19 | | | **else**
20 | | | | $updateStates_{p,i}(N_{TR}, P_{1,i}^{MAX}(SET_1), P_{2,i}^{MAX}(SET_1))$;
21 | | **else**
22 | | | $updateStates_{p,i}(N_{R1}, P_{1,i}^{MAX}(SET_1), P_{2,i}^{MAX}(SET_1))$;
23 | **sinon si** $\exists\ v \in V_{P_{MAX}},\ RBState_i(v_T) \in \{N_{T1}, N_{TR}\}$ **alors**
24 | | $SET_2 \leftarrow \{v \in Voisins(p) | RBState_i(v) \in \{N_{T1}, N_{TR}\}\}$;
| | $V_{P_{MAX}} \leftarrow \{v \in Voisins_{P_{1,i}^{MAX}(SET_2)} | P_{2,i}(v) = P_{2,i}^{MAX}(Voisins_{P_{1,i}^{MAX}(SET_2)})\}$;
| | $updateStates_{p,i}(N_{T2}, P_{1,i}^{MAX}(SET_2), P_{2,i}^{MAX}(SET_2))$;
25 | **else**
26 | | $SET_3 \leftarrow \{v \in Voisins(p) | RBState_i(v) \in \{N_{R1}\}\}$;
| | $V_{P_{MAX}} \leftarrow \{v \in Voisins_{P_{1,i}^{MAX}(SET_3)} | P_{2,i}(v) = P_{2,i}^{MAX}(Voisins_{P_{1,i}^{MAX}(SET_3)})\}$;
27 | | **if** $\exists\ v \in V_{P_{MAX}},\ RBState_i(v) \in \{N_{R1}\}$ **then**
28 | | | $updateStates_{p,i}(N_{R2}, P_{1,i}^{MAX}(SET_3), P_{2,i}^{MAX}(SET_3))$;
29 | | **else**
30 | | | $updateStates_{p,i}(F, -1, -1)$;

utilisable dans un environnement réel.

5.5 Conclusion

Au cours de ce chapitre, nous avons proposé deux algorithmes de correction de conflits. Le premier, écrit dans le modèle à état permet notamment de corriger l'ensemble des conflits avec un nombre réduit de variables partagées. Cette correction ainsi que le temps de convergence de cet algorithme ont été prouvés.

Un second algorithme a été proposé afin d'améliorer la libération des ressources. Cette amélioration permet à la fois de diminuer le nombre de libérations nécessaire à la correction des conflits mais également d'éviter de libérer les allocations les plus importantes. Pour cela, il introduit une relation d'ordre stricte entre les différentes allocations existantes. Cette relation est obtenue à l'aide de deux critères. Le premier permet d'attribuer un poids en fonction de la stratégie retenue par l'algorithme de répartition de ressources. Le second permet d'attribuer un identifiant unique à chaque allocation.

Bien que le second algorithme soit plus efficace dans la libération, il introduit deux nouveaux paramètres dont la valeur doit être transmise par chaque nœud à son voisinage. Cet algorithme doit donc partager plus d'informations avec ses voisins que l'algorithme 5.3.1.

Chapitre 6

Stratégies de répartition des ressources

6.1 Introduction

Dans notre approche de l'allocation de ressources, nous avons décidé de traiter séparément la détection et la correction des conflits d'une part et la répartition des ressources d'autre part. Le chapitre précédent a permis de présenter deux algorithmes capables à la fois de corriger les conflits existants, mais également d'identifier les canaux qui peuvent être utilisés sans créer de conflits avec les allocations existantes.

Le problème de la répartition des ressources est fortement lié à la stratégie d'allocation retenue. Il est ainsi possible d'effectuer une allocation équitable en attribuant la même quantité de ressources à chaque arc du réseau. Il est également possible de tenir compte du trafic en allouant plus de canaux aux arcs les plus sollicités ou encore d'optimiser la répartition afin de minimiser le temps de transport des messages. Chacun de ces problèmes nécessite un algorithme de répartition de ressources spécifique.

Dans ce chapitre, nous présentons deux nouveaux algorithmes de répartition de ressources. Le premier, que nous appellerons algorithme de connectivité, garantit une connectivité maximale en allouant une quantité minimale de ressources à chaque arc du graphe de connectivité définit dans le chapitre précédent. Le second, que nous appellerons algorithme d'allocation à la demande, permet de tenir compte du trafic et de l'évolution de celui-ci au cours du temps en allouant les ressources en fonction de la demande de chaque arc de ce même graphe.

6.2 Allocation garantissant la connectivité

L'algorithme présenté ici consiste à attribuer un canal à chaque arc du graphe de connectivité en minimisant le nombre total de canaux utilisés. Ce problème est très proche de celui posé par la coloration de graphe : attribuer une couleur à chaque élément (nœud ou arc) d'un graphe en utilisant le moins de

couleur possible. Les couleurs utilisées jouent alors le même rôle que les canaux dans l'allocation de ressources. La NP-complétude de ces problèmes a été démontré par Karp dans [Kar72].

Le problème de coloration de graphe correspondant à notre problème consisterait à attribuer, de manière distribuée, une couleur à chaque arc du graphe en veillant à ce que la couleur utilisée par un arc a ne soit pas utilisée par d'autres arcs situés dans le 2-voisinage de a. À notre connaissance aucun travail portant sur la théorie des graphes ne traite de ce problème spécifique.

Nous avons donc décidé de nous inspirer de l'algorithme DSATUR présenté dans [Bre79] en l'adaptant à nos hypothèses c'est-à-dire une allocation distribuée sur des liens unidirectionnels en considérant une zone d'interférence deux fois plus étendue que la zone de transmission. Nous commencerons donc cette partie par une brève présentation de DSATUR puis nous verrons les différentes transformations que nous proposons d'appliquer à cet algorithme pour obtenir une coloration des arcs conforme à nos hypothèses. Nous aborderons ensuite les difficultés posées par l'adaptation de cet algorithme à un environnement distribué. Enfin, nous présenterons notre algorithme distribué ainsi que l'évaluation de ses performances sur différentes topologies.

6.2.1 Principes de l'algorithme DSATUR

DSATUR [Bre79] est une heuristique de coloriage de graphe avec une approche glouton. Son fonctionnement repose sur une allocation séquentielle des couleurs sur les nœuds du graphe de connectivité. Les couleurs sont numérotées par ordre croissant de première utilisation. La couleur 1 représente ainsi la première couleur à avoir été utilisée. L'ordre dans lequel les allocations sont réalisées est fondé sur un critère appelé *degré de saturation*. Ce degré est défini pour chaque nœud n comme étant le nombre de couleurs différentes utilisées dans le voisinage de n. Plus le degré de saturation d'un nœud est élevé et plus celui-ci aura de difficultés à trouver une ressource disponible parmi les couleurs déjà utilisées. Ainsi, pour minimiser le nombre de couleurs nécessaires, il est important de donner la priorité aux nœuds possédant le degré de saturation le plus élevé. C'est pourquoi, l'algorithme calcule ce degré de saturation pour chaque nœud du graphe, choisit le nœud de degré le plus élevé et lui alloue la première couleur disponible. Lorsque plusieurs nœuds possèdent le même degré de saturation, l'algorithme s'appuie sur leur degré, c'est-à-dire leur nombre de voisins, pour les départager. L'algorithme ne précise pas comment gérer les cas où des nœuds possèdent le même degré de saturation et le nombre de voisins.

Algorithme 6.2.1: Algorithme DSATUR

Initialisation :
1. 1. Ordonner les nœuds par degré décroissant;
2. 2. Affecter la première couleur au nœud de degré maximum;
3. 3. Choisir le nœud présentant le degré de saturation maximum. En cas d'égalité choisir le nœud de degré maximum dans le sous-graphe non colorié;
4. 4. Colorer le nœud choisi avec la couleur disponible possédant l'identifiant le plus petit ;
5. 5. Si tous les nœuds sont colorés, arrêter. Sinon reprendre à 3;

Afin d'adapter cet algorithme à nos hypothèses, nous allons tout d'abord le modifier pour qu'il effectue

une coloration des arcs en tenant compte d'une zone d'interférence deux fois plus étendue que la zone de transmission. Nous verrons ensuite comment adapter ce modèle à un environnement distribué.

6.2.1.1 Adaptation de DSATUR à l'allocation des arcs avec une zone d'interférence quelconque

Pour généraliser cet algorithme à une zone d'interférence quelconque, il est nécessaire de redéfinir les deux critères d'ordonnancement de l'algorithme : le degré de saturation et le nombre de voisins.

Dans l'algorithme original, le degré de saturation permet de connaître le nombre de couleurs différentes utilisées dans le voisinage d'un nœud p. Puisqu'un nœud ne peut choisir une couleur qui est déjà utilisée dans sa zone d'interférence, ce degré représente également le nombre de couleurs qui ne peuvent être choisies pour p. Lorsque la zone d'interférence est deux fois plus étendue que la zone de transmission, ce degré doit alors tenir compte de l'ensemble des couleurs utilisés dans son 2-voisinage. Il est ainsi possible de généraliser à une zone d'interférence N fois supérieure à la zone de transmission. Le degré de saturation d'un nœud représentera alors la quantité de couleurs différentes utilisées dans le N-voisinage de celui-ci.

Puisque nous souhaitons obtenir une coloration des arcs du graphe, il faut donc considérer le degré de saturation de ces derniers. Par analogie avec la définition de l'algorithme d'origine, le degré de saturation d'un arc doit correspondre au nombre de couleurs utilisées par l'ensemble des arcs appartenant à sa zone d'interférence. La nouvelle définition du degré de saturation devient :

Définition 6.1. *Le degré de saturation d'un arc p dans le cas d'une zone d'interférence N fois plus étendue que la zone de transmission est noté $D_S^N(p)$. Il représente la quantité de couleurs différentes utilisées par les arcs situés dans le N-voisinage de p.*

Le second critère d'ordonnancement doit également être redéfini. Dans l'algorithme original, celui-ci correspond au degré du nœud p étudié, c'est-à-dire au nombre de voisin de celui-ci. Ce critère permet de connaître le nombre de nœuds qui ne peuvent utiliser la même couleur que p. Pour pouvoir généraliser à une zone d'interférence N fois plus étendue que la zone de transmission, il faut comptabiliser l'ensemble des nœuds situés dans le N-voisinage de p. Le terme de "degré" n'est donc plus approprié et nous lui préférerons celui de *degré d'interférence*. Tout comme pour le critère précèdent, il est important de considérer le degré d'interférence des arcs plutôt que celui des nœuds. la nouvelle définition de ce deuxième critère devient donc :

Définition 6.2. *Le degré d'interférence d'un arc p dans le cas d'une zone d'interférence N fois plus étendue que la zone de transmission est noté $D_I^N(p)$. Il représente le nombre total d'arcs présents dans le N-voisinage de p.*

L'algorithme 6.2.2 correspond à l'adaptation de l'algorithme 6.2.1 dans le cadre d'une allocation sur les arcs avec une zone d'interférence deux fois plus étendue que la zone de transmission.

Algorithme 6.2.2: Algorithme DSATUR adapté à la coloration d'arc avec une zone d'interférence deux fois plus étendue que la zone de transmission

Données :

$D_I^2(p) \leftarrow$ nombre d'arcs appartenant à la zone d'interférence de l'arc p;

$D_S^2(p) \leftarrow$ nombre de couleurs différentes utilisées dans la zone d'interférence de l'arc p ;

Initialisation :

1. 1. Ordonner les arcs par D_I^2 décroissant;
2. 2. Affecter la première couleur à l'arc de D_I^2 maximum;
3. 3. Choisir l'arc présentant le D_S^2 maximum. En cas d'égalité choisir l'arc de D_I^2 dans le sous-graphe non colorié;
4. 4. Colorer l'arc choisi avec la couleur disponible possédant l'identifiant le plus petit;
5. 5. Si tous les arcs sont colorés, arrêter. Sinon reprendre à 3;

Cet algorithme est cependant adapté à une coloration centralisée. Or, nous nous intéressons à une allocation distribuée des ressources et certains aspects de l'algorithme 6.2.2 sont difficiles à adapter à un environnement réparti.

Tout d'abord, l'allocation est réalisée de manière séquentielle ce qui nécessite une coordination des nœuds à l'échelle du réseau entre chaque allocation. Un tel niveau de coordination n'est pas du tout adapté à un environnement distribué où la prise de décision est locale. Ensuite, pour calculer le degré de saturation il est nécessaire de connaître le nombre de ressources différentes utilisées dans le 2-voisinage de chaque arc. Chaque nœud doit donc collecter des informations dans une zone importante ce qui induit une utilisation élevée de la bande passante pour relayer ces informations. De plus, le temps nécessaire à l'acheminement de ces informations augmente le temps de convergence de l'algorithme. Enfin, le degré de saturation est une information fréquemment modifiée puisque celle-ci peut changer après chaque allocation. Il est donc nécessaire de la mettre à jour entre chaque allocation et d'informer le 2-voisinage de l'arc ayant réalisé la dernière allocation de cette mise à jour.

L'adaptation de l'algorithme DSATUR à un environnement distribué présente de nombreuses difficultés. La suite de cette partie est consacrée à l'étude de chacun de ces problèmes et aux adaptations possibles pour obtenir une allocation des ressources efficace tout en conservant un temps de convergence et une quantité d'information échangées raisonnables dans un environnement distribué.

6.2.1.2 Adaptation de DSATUR à un environnement distribué

Le principe de l'algorithme DSATUR consiste à colorer les liens par ordre décroissant suivant deux critères :

1. le degré de saturation
2. le degré d'interférence

degré de saturation en environnement distribué : Le premier critère est difficilement utilisable dans un environnement distribué car sa valeur doit être mise à jour après chaque nouvelle allocation pour l'ensemble des arcs appartenant à la zone d'interférence de l'arc ayant reçu l'allocation. Cette variable va donc régulièrement changer de valeur et sa mise à jour se fera au rythme de la propagation des messages. Le choix d'un critère aussi versatile risque d'avoir des conséquences sur la pertinence de l'ordonnancement qui est la base de l'algorithme. Ce critère n'est donc pas adapté à une utilisation dans un environnement distribué.

Degré d'interférence en environnement distribué : Le second critère, le nombre de liens appartenant à la zone d'interférence, est lui beaucoup moins versatile puisqu'il ne varie qu'en cas de changement topologique. Contrairement au degré de saturation, qui donne une information sur la saturation instantanée d'un arc, ce critère permet de connaître le niveau de saturation d'un lien à la fin de l'allocation. Par conséquent, utiliser ce critère fait perdre la connaissance instantanée de la saturation mais permet d'avoir une solution utilisable dans un environnement distribué et qui reste basée sur le degré de saturation des arcs.

Ce critère n'en reste pas moins difficile à utiliser car il nécessite la collecte d'informations dans le 2-voisinage de chaque arc. Or, dans un réseau, seuls les nœuds sont des objets réels, capables de collecter des informations et d'allouer des ressources. Cette collecte ne peut donc se faire que sur le prédécesseur ou le successeur de chaque arc. Pour cela, l'un de ces deux nœuds doit récupérer des informations dans son 3-voisinage pour calculer le poids de chacun de ses arcs. Ce mécanisme nécessite de relayer un volume d'information trop important pour ce genre de réseau qui augmenterait l'utilisation de la bande passante ainsi que le temps de convergence de l'algorithme. Dans la solution présentée ici, nous avons décidé de nous limiter au nombre de liens qui peuvent être détectés par l'émetteur ou le récepteur ; c'est-à-dire les arcs qui lui sont attachés ainsi que les arcs de leurs voisins. Puisque l'algorithme de correction de conflits utilise déjà cette liste de voisin, cette information est déjà connue.

Certes, cette mesure ne donne pas une valeur exacte du nombre d'arcs interférés, mais il permet d'avoir une idée de la densité de liens présents autour des extrémités du lien et donc d'avoir une estimation du niveau de contrainte du lien étudié. Ces observations sont à la base de l'algorithme d'allocation que nous allons présenter maintenant.

6.2.2 Présentation de notre algorithme de connectivité

Le premier algorithme de répartition de ressources que nous présentons [PTBV11] est distribué et permet d'attribuer un canal à chaque arc du graphe de connectivité en cherchant à minimiser le nombre total de canaux utilisés. Il fonctionne en parallèle avec un des algorithmes de correction de conflits présenté au chapitre précédent et repose sur les principes suivants :

– Chaque nouvelle allocation est réalisée par le prédécesseur de l'arc concerné, c'est-à-dire par

l'émetteur.

- Un ordonnancement local permet d'éviter que deux voisins puissent effectuer leur allocation en même temps.
- L'ordonnancement est réalisé grâce au poids, attribué à chaque nœud, qui identifie le niveau de contrainte de celui-ci.

Puisque notre algorithme est distribué, la décision d'allouer une ressource doit être prise localement, soit par le prédécesseur (l'émetteur), soit par le successeur (le récepteur) de cet arc. Aucun critère ne nous permettant de privilégier l'un par rapport à l'autre, nous avons décidé que l'attribution de nouvelles ressources serait gérée par le prédécesseur de chaque arc, comme nous en faisions l'hypothèse dans la présentation des algorithmes de correction de conflits.

6.2.2.1 Critères d'ordonnancement

Dans notre algorithme, l'ordonnancement des allocations privilégie les arcs les plus contraints. Contrairement à l'algorithme DSATUR qui utilise un ordonnancement global, notre solution se limite à un ordonnancement local plus approprié à un système distribué. Cet ordonnancement est basé sur trois critères :

- un poids P_n lié au nombre de voisins situés dans le 2-voisinage.
- un poids modifié P'_n qui prend en compte les allocations déjà réalisées sur le nœud n
- un identifiant ID_n unique pour chaque nœud qui garantit une relation d'ordre strict entre les allocations.

Premier critère, le poids P_n : pour les raisons présentées plus haut, nous avons choisi d'utiliser une estimation du nombre d'arcs situés dans le 2-voisinage d'un nœud comme critère d'ordonnancement. Chaque nœud n du réseau possède ainsi un poids P_n représentant cette estimation et défini par $P_n = A^s_n + \sum_{v \in \text{Voisins(n)}} A^s_v$, où A^s_n représente le nombre d'arc sortant de n, c'est-à-dire les arcs pour lesquels n est émetteur. Pour connaître le nombre d'arcs sortant d'un voisin, les nœuds n'ont pas besoin d'échanger de nouvelles données. Puisqu'un nœud possède autant d'arcs sortant que de voisins, il suffit de connaître le degré d'un nœud pour connaître la valeur de A^s_v. Dans nos algorithmes de correction de conflit, chaque nœud v transmet déjà la liste de ses voisins. Chaque voisin de v peut donc réutiliser cette information pour calculer la valeur de A^s_v.

Grâce à ce poids, chaque nœud est capable d'obtenir une estimation du nombre d'arcs situés dans son 2-voisinage, cependant ce critère n'est pas suffisant. Prenons l'exemple d'un réseau comprenant cinq nœuds répartis selon une topologie en chaîne et représenté sur la figure 6.1. Nous supposons ici que le réseau possède 16 canaux de communications représentés en dessous des nœuds. La valeur affectée à chacun de ces canaux représente l'état de la variable *channelState* de l'algorithme de détection et de correction de conflits. Ici, l'ensemble de ces canaux sont libres (état F) à l'exception du premier canal qui est utilisé par l'arc B→D. Les trois critères d'ordonnancement utilisés par l'algorithme de répartition de ressources sont représentés au dessus de chaque nœud. Dans le cas du nœud A, le calcul de son poids

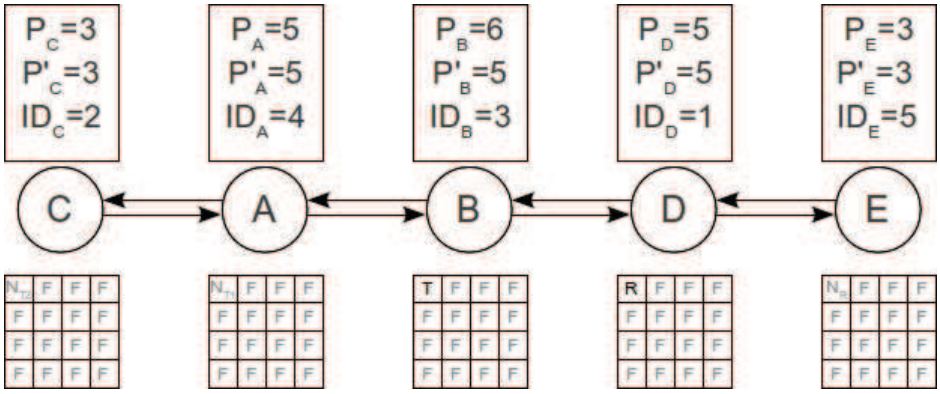


FIGURE 6.1: Calcul des critères d'ordonnancement

P_A est réalisé de la manière suivante. Celui-ci est voisin de deux nœuds, C et B et possède donc deux arcs sortants. Ces voisins possèdent un degré de 1 pour C et de 2 pour B. Le poids de A, qui correspond à la somme de ces trois valeurs, est donc bien de cinq. En se basant uniquement sur ce critère et en supposant un ordonnancement local :

- les nœuds C et E possèdent un poids plus faible que leur unique voisin. Ils ne sont donc pas prioritaire pour l'allocation,
- les nœuds A et D possèdent un poids plus faible que leur voisin B. Ils ne sont donc pas prioritaire pour l'allocation.
- le nœud B possède un poids plus élevé que ses voisins. Il est donc prioritaire pour l'allocation.

Si nous basons l'ordonnancement uniquement sur ce poids, le nœud B va donc pouvoir effectuer ses allocations en premier. Cependant, les nœuds A et D ne peuvent pas détecter la fin des allocations de B, les empêchant ainsi de réaliser leurs propres allocations. De plus, même si un tel mécanisme était présent, les nœuds A et D devraient attendre que B ait effectué l'ensemble de ses allocations. Ce poids ne tient donc pas compte des allocations déjà effectuées par B dans le calcul de son poids.

Deuxième critère, le poids modifié P'_n : pour éviter ces problèmes, l'algorithme utilise un second critère d'ordonnancement, le poids modifié, noté P'_n et tel que $P'_n = P_n$ - *Nombre de liens sortants de n ayant déjà reçu une allocation* si tous les arcs n'ont pas encore reçu d'allocation, et 0 sinon. Ce nouveau poids est toujours basé sur le nombre d'arc situé dans le 2-voisinage de l'émetteur, mais contrairement au poids classique P_n, il tient compte du nombre de ressources déjà alloué et permet de "retirer" les nœuds ayant terminé leurs allocations. En effet, le graphe étant connexe, tout nœud à au moins un voisin et donc au moins un arc sortant. Le poids minimum d'un noeud n'ayant pas terminé son allocation est donc de 1. En fixant le poids modifié des nœuds ayant terminé leur allocation à zéro, cela revient à les exclure de l'ordonnancement permettant à l'algorithme de se concentrer uniquement sur les nœuds qui ont encore des allocations à réaliser.

Pour utiliser ce poids modifié, un nœud n a besoin de connaître le poids modifié P' de chacun de ses voisins pour ensuite le comparer au sien. Pour éviter d'augmenter la taille des messages utilisés par

l'algorithme de répartition de ressources nous avons essayer de trouver une solution permettant à chaque nœud d'identifier le poids modifié de chacun de ses voisins à partir des informations disponible. Reprenons le cas de la figure 6.1 et étudions le calcul du poids modifié de B par le nœud A. En supposant que chaque nœud transmet la valeur de son poids, A peut accéder à la valeur de P_B. Pour trouver la valeur de P'_B, il lui faut encore connaître le nombre d'arcs sortant de B qui ont déjà reçu une allocation. En d'autres termes, A doit trouver le nombre de canaux utilisés par B en tant qu'émetteur. Dans l'algorithme de correction de conflit, chaque nœud possède une variable *channelState* pour chaque ressource du réseau. Lorsqu'un nœud utilise un canal pour émettre, il fixe la valeur de la variable *channelState* à l'état T. Puisque chaque nœud partage la valeur de ses variables *channelState* avec l'ensemble de ses voisins, A est capable de connaître la quantité de ressources utilisées par B en émission. dans l'exemple ici, A peut donc savoir que le nœud B utilise le second canal en émission. Le nœud A possède donc toutes les informations pour calculer la valeur de P'_B.

6.2.2.2 Mécanisme d'ordonnancement

Pour trouver un ordonnancement, l'algorithme d'un nœud n compare ces deux critères ainsi que l'identifiant unique pour l'ensemble des nœuds appartenant au voisinage de n. L'ordre dans lequel ces critères sont étudiés est le suivant :

1. poids modifié
2. poids
3. identifiant

Ainsi, pour savoir s'il est prioritaire, chaque nœud n du réseau compare son poids modifié P' avec celui de l'ensemble de ses voisins. S'il existe un voisin possédant un poids modifié plus élevé, le nœud attend. En cas d'égalité, la comparaison se fait sur le poids afin de donner la priorité au nœud ayant le plus d'arcs dans son 2-voisinage. Si ce poids est également identique, alors la comparaison se fait sur l'identifiant qui étant unique pour chaque nœud permet de casser les symétries et donc d'éviter que deux voisins effectuent leur allocation simultanément. Quand il devient prioritaire pour l'allocation, le nœud alloue une ressource sur le lien qu'il partage avec son voisin de poids modifié le plus élevé. Ce nœud doit alors identifier les ressources qui peuvent être utilisées sans créer de conflit. Nous avons vu au chapitre précédent qu'un nœud A_T peut allouer une ressource sur un arc dirigé vers B_R sans créer de conflits si sa variable *channelState* est dans l'état F ou N_{T2}, qu'il ne possède pas de voisin dans l'état N_R et que la variable de B_R est dans l'état F.

Ces critères lui permettent d'identifier l'ensemble des canaux utilisables pour l'allocation. Pour essayer d'utiliser au mieux les ressources disponibles, l'algorithme va rechercher les canaux qui permettent de maximiser la réutilisation spatiale. La figure 6.2 présente un exemple dans lequel la réutilisation spatiale est maximale. Pour cela, A_T doit se trouver à trois sauts des récepteurs. Or, si la variable *channelState* de ce nœud est dans l'état N_{T2} cela signifie qu'il y a un émetteur à deux sauts, et donc que le récepteur associé à cet émetteur est à trois sauts. Choisir un canal dans l'état N_{T2} pour le nœud A_T permet donc

de maximiser la réutilisation des ressources. Ainsi, lorsqu'il a le choix entre des ressources dans l'état F et des ressources dans l'état N_{T2}, A_T choisira une des ressources dans l'état N_{T2}.

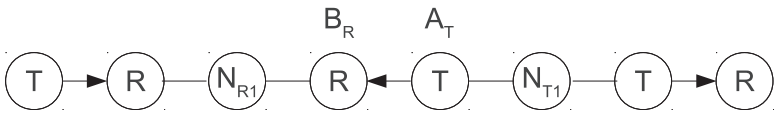


FIGURE 6.2: Maximisation de la réutilisation spatiale

Il est également possible d'améliorer la réutilisation spatiale en permettant à A_T d'identifier les canaux où le nœud B_R se trouve à 3 sauts d'un émetteur. le premier algorithme de correction de conflits (algorithme 5.3.1) ne permettait pas de détecter ces situations. C'est pour cette raison que nous avons introduit l'état N_{R2} dans l'algorithme 5.4.1. En effet Lorsque B_R possède un canal dans l'état N_{R2}, cela signifie qu'il possède un récepteur à deux saut, et donc un émetteur à trois sauts. Le nœud A_T, responsable de l'attribution des ressources, est donc capable d'identifier ces situations et ainsi d'améliorer la réutilisation spatiale.

Algorithme 6.2.3: Algorithme de garantie de connectivité

Données :

$P_n \leftarrow$ Poids du nœud n;

$P'_n \leftarrow$ Poids modifié du nœud n;

$ID_n \leftarrow$ Identifiant du nœud n;

P'max$_{V_n} \leftarrow$ Poids modifié le plus élevé dans le voisinage de n;

Pmax$_{V_n} \leftarrow$ Poids le plus élevé dans le voisinage de n;

IDmax$_{V_n} \leftarrow$ Identifiant le plus élevé dans le voisinage de n;

1 **tant que** $P'_n \geq P'max_{V_n}$ ***OU*** $P_n \geq Pmax_{V_n}$ ***OU*** $ID_n > IDmax_{V_n}$ **faire**
2 candidats ←Voisins$_p \setminus$ {voisins dont le lien a déjà reçu une allocation};
3 candidat ← nœud ∈ candidats de poids modifié maximum;
4 **si** ∃ *un canal c tq c est utilisable et maximise la réutilisation spatiale* **alors**
5 Allocation du canal c sur le lien sortant vers candidat;
6 **sinon si** ∃ *un canal c tq c est utilisable* **alors**
7 Allocation du canal c sur le lien sortant vers candidat;

6.2.3 Validation de l'algorithme de connectivité

Pour valider cet algorithme, nous avions besoin d'un environnement de simulation simple capable de générer un réseau constitué d'un ensemble d'équipements pouvant :

- fonctionner indépendamment les uns des autres,
- exécuter les algorithmes de correction de conflits et de répartition de ressources,

– communiquer avec les équipements situés à une distance inférieure à une valeur seuil appelée *portée radio*.

Nous avons donc décidé de développer notre propre simulateur, écrit en Java et adapté à nos besoins.

6.2.3.1 Topologies étudiées

Trois types de topologies ont été étudiées lors des simulations :

– *chemin :* les nœuds sont répartis sur une ligne et peuvent communiquer avec au plus deux nœuds. La figure 6.3 illustre un exemple de topologie chemin.

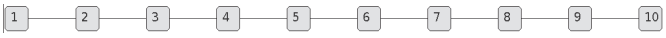


FIGURE 6.3: Exemple d'un chaîne où les identifiants des nœuds sont ordonnées linéairement.

– *grille :* les nœuds sont répartis sur deux dimensions de manière déterministe, chaque équipement pouvant communiquer avec au plus deux voisins sur chacune de ces dimensions. La figure 6.4 présente un exemple de topologie grille.

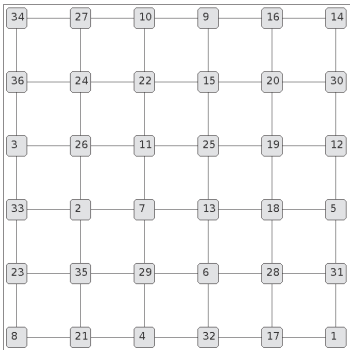


FIGURE 6.4: Exemple d'une grille où les identifiants des nœuds sont distribués uniformément sur deux dimensions.

– *géometrique aléatoire :* les équipements sont placés dans la zone de simulation. Les nœuds sont distribués uniformément dans les deux dimensions. Lorsque la distance entre deux nœuds est inférieure à la portée radio, ces deux nœuds peuvent communiquer.

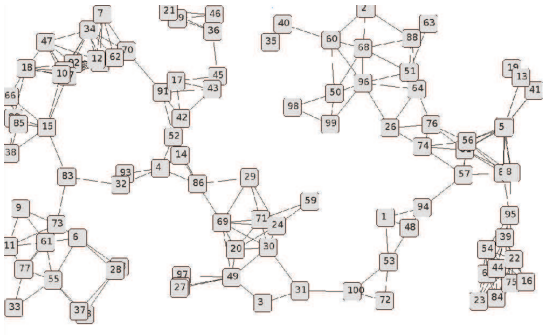


FIGURE 6.5: Exemple d'une topologie géometrique aléatoire

6.2.3.2 Déroulement des simulations

Une fois les nœuds positionnées, le simulateur va les activer. Ces équipements ayant un fonctionnement indépendant, il n'y a aucune raison pour qu'ils s'activent simultanément. Le moment de l'activation de chaque nœud est donc choisi uniformément entre 0 et 500 ms, l'instant 0 correspondant au lancement de la simulation.

Lorsqu'un nœud est activé par le simulateur pour la première fois, il exécute l'algorithme de correction de conflits puis celui de répartition de ressources. Chacun de ces algorithmes envoie ensuite périodiquement un message à destination des voisins du nœud qui les exécutent. Ce message contient les informations nécessaires au fonctionnement de ces deux algorithmes.

Dans les simulations, ces algorithmes envoient un message toutes les 500 ms. Puisque ce temps est configurable, nous avons décidé d'étudier le temps de convergence en nombre de rondes, une ronde étant l'intervalle de temps nécessaire pour que tous les nœuds émettent un message.

6.2.3.3 Résultats des simulations réalisées

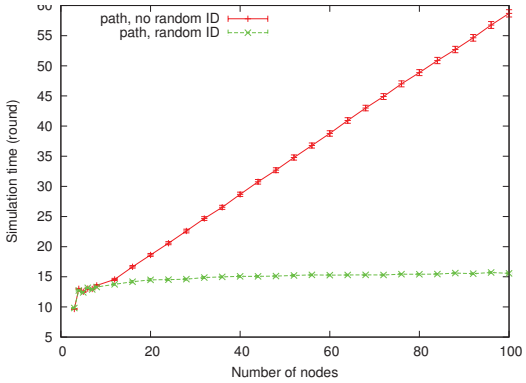
Les résultats obtenus sont présentés dans les figures 6.6 et 6.7. Chaque point de ces graphiques a été calculé avec un intervalle de confiance de 95%. Dans les figures 6.6a, 6.6b 6.7a 6.7b chaque point est la moyenne de 100 simulations pour les topologies de type chaîne et grille. Pour les topologies aléatoires des figure 6.6c et 6.7c, chaque point est la moyenne de 1000 simulations.

Les simulations portent sur le nombre de ressources utilisées en fonction de la taille du réseau. Théoriquement, cette quantité dépend de la taille d'une clique maximale du graphe de conflit. La taille d'une clique maximale correspond au nombre maximum d'arcs mutuellement interférant. Pour vérifier si notre solution supporte le passage à l'échelle, la taille du réseau doit augmenter sans que la taille de cette clique maximale soit modifiée. Pour les chemins et les grilles, la taille de cette clique reste constante à partir d'une certaine taille de réseau. Ceci n'est pas forcément le cas pour les topologies aléatoires. De plus, il est irréaliste d'extraire les topologies aléatoires présentant la même taille de clique maximale puisque le calcul de cette taille est un des 21 problèmes NP-complets de Karp [Kar72]. Néanmoins, on peut supposer que la taille de la clique maximale augmentera en même temps que le degré moyen des nœuds. C'est pourquoi dans les simulations réalisées, nous avons fait varier la portée radio afin de conserver un degré moyen constant à l'aide de la formule :

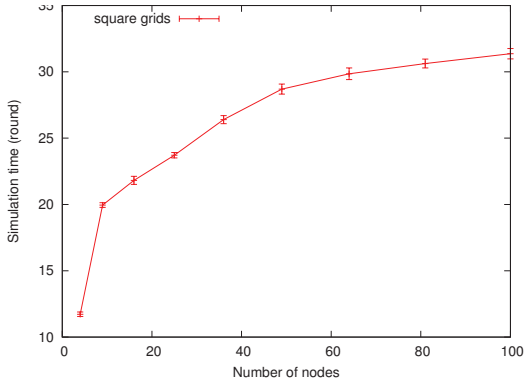
$$R = \sqrt{\frac{D.A_W.A_H}{\pi(N-1)}} \tag{6.1}$$

, où R représente la portée radio, D le degré moyen souhaité, A_W la longueur et A_H la largeur de la zone de simulation et N le nombre de nœuds du réseau.

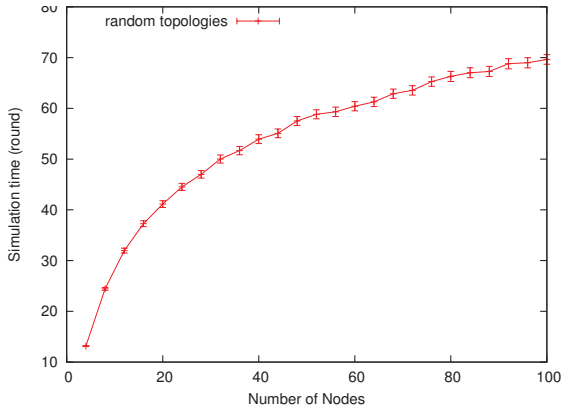
Temps de convergence : Les figures 6.6a, 6.6b et 6.6c montrent l'évolution du temps de convergence en fonction du nombre de nœuds pour les différentes topologies. Lorsque les nœuds forment une chaîne, l'ordre dans lequel les identifiants sont répartis peut avoir une influence sur le temps de convergence. Nous avons donc représenté le cas ou les identifiants sont répartis uniformément sur la chaîne et le cas où ces identifiants sont ordonnés linéairement comme sur la figure 6.3.



(a) Résultats pour les topologies de type chaîne



(b) Résultats pour les topologies de type grille



(c) Résultats pour les topologies aléatoire avec un degré moyen des noeuds constant

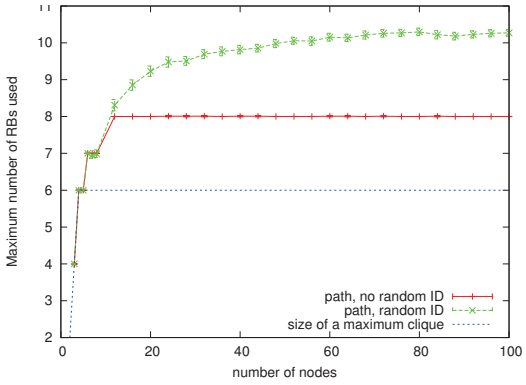
FIGURE 6.6: Temps de convergence (en ronde) de l'algorithme de connectivité

La figure 6.6a montre que le temps de convergence augmente linéairement avec le nombre de nœuds lorsque les identifiants sont ordonnés alors qu'il converge rapidement lorsque ces identifiants sont aléatoirement répartis.

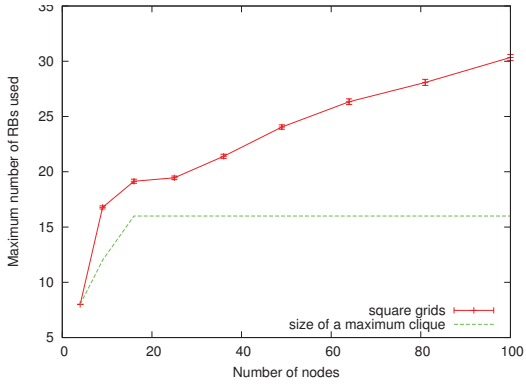
Ce phénomène est lié au critère d'ordonnancement. Dans la présentation de l'algorithme 6.2.3 nous avons vu que la priorité d'un nœud dépendait d'abord de son poids modifié. Lorsque celui-ci est identique, la distinction se fait sur le poids et en dernier lieu sur l'identifiant. Dans une chaîne, tous les nœuds possèdent le même poids et poids modifié à l'exception des deux premiers et des deux derniers. La comparaison porte alors sur les identifiants. Lorsque ceux-ci sont répartis aléatoirement, plusieurs nœuds peuvent avoir la même priorité dans leur voisinage. Dans ce cas, plusieurs allocations peuvent avoir lieu simultanément. De plus, ce nombre d'allocations simultanées augmente avec la taille du réseau. L'augmentation du nombre de nœuds a ainsi deux effets antagonistes sur le temps de convergence.

D'un coté l'augmentation du nombre de nœuds qui rend le temps de convergence plus long à cause de l'accroissement du nombre d'allocations à réaliser. De l'autre l'augmentation du nombre d'allocations simultanées qui tend à faire diminuer ce temps de convergence. Par contre, lorsque les identifiants sont ordonnés comme sur la figure 6.3, il ne peut pas y avoir plusieurs allocations simultanées, le temps de convergence croît alors proportionnellement avec le nombre de nœuds.

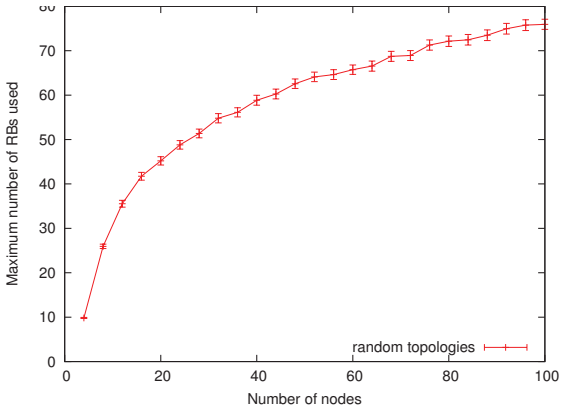
Les figures 6.6b et 6.6c s'intéressent également à l'évolution du temps de convergence en fonction du nombre de nœuds mais pour des topologies de type grille et aléatoire dans lesquelles les nœuds ont des identifiants attribués uniformément. Ces figures indiquent aussi que le temps de convergence de l'algorithme n'augmente pas linéairement. Tout comme pour le cas de la chaîne avec identifiants aléatoires, c'est l'augmentation du nombre d'allocations simultanées qui compense l'augmentation du nombre d'allocations à réaliser.



(a) Résultats pour les topologies de type chaîne



(b) Résultats pour les topologies de type grille



(c) Résultats pour les topologies aléatoire avec un degré moyen des noeuds constant

FIGURE 6.7: Évolution du nombre de ressources utilisées par l'algorithme de connectivité

Évolution du nombre de ressources utilisées : Les figures 6.7a, 6.7b et 6.7c portent sur l'évolution du nombre de ressources utilisées en fonction du nombre de nœuds du réseau. La première constatation est que l'algorithme nécessite plus de ressources pour les topologies aléatoires que pour les topologies grille qui demande elle-même plus de ressources que les topologies en chaîne. La principale raison est que

le degré moyen des nœuds suit la même tendance que le nombre de ressources demandées. En effet, une chaîne possède un degré moyen de deux, une grille un degré moyen de quatre et les topologies aléatoires un degré moyen de cinq. En ce qui concerne le passage à l'échelle, on observe une convergence de la quantité de ressources nécessaires avec l'augmentation de la taille du réseau. Ce phénomène est lié à la réutilisation spatiale qui permet d'allouer une même ressource à plusieurs arcs du réseau. De la même manière que le nombre d'allocations simultanées compense l'augmentation du temps de convergence, la réutilisation spatiale permet de compenser l'augmentation de la quantité de ressources nécessaires.

6.2.4 Conclusion

L'algorithme de connectivité dont les performances sont évaluées ici, permet d'attribuer une ressource à chaque arc du graphe en minimisant le nombre total de ressources utilisés. Il fonctionne en parallèle de l'algorithme de correction de conflits et se sert des valeurs de ses paramètres afin de minimiser la quantité de données échangées mais également pour maximiser la réutilisation spatiale des ressources. Les résultats des simulations montrent que dans la plupart des cas, cet algorithme supporte bien l'augmentation de la taille du réseau. Cependant, il ne permet pas d'attribuer une quantité de ressources différente pour chaque arc du graphe de connectivité. C'est pourquoi, afin de prendre en compte le trafic du réseau et ses variations au cours du temps, nous avons décidé de concevoir un nouvel algorithme dédié à ce problème.

6.3 Allocation garantissant la QoS pour un trafic donné

Dans la plupart des réseaux, les besoins en ressources ne sont pas identiques pour chaque arc du graphe de connectivité. Ils dépendent de certains facteurs tels que la topologie ou le trafic et peuvent fluctuer dans le temps. Par exemple, lorsque deux nœuds souhaitent passer d'une communication audio à une transmission vidéo, la demande en ressource entre ces deux équipements va augmenter. L'algorithme de répartition de ressources doit alors attribuer de nouveaux canaux de communications à chaque arc du chemin reliant ces deux équipements afin de satisfaire la demande. Inversement, ces canaux devront être libérés lorsque la transmission vidéo sera terminée afin de pouvoir utiliser ces ressources pour satisfaire d'autres demandes. L'algorithme étudié dans cette partie s'intéresse à cette problématique et adapte l'allocation aux demandes du réseau.

6.3.1 Pondération des arcs en fonction des demandes

Afin de tenir compte de ces demandes dans notre modèle, le graphe de connectivité est désormais pondéré, le poids de chaque arc représentant la quantité de canaux nécessaire pour satisfaire la demande de celui-ci. Nous nous intéressons dans ce document au problème de l'allocation de ressources. Nous supposerons donc que le prédécesseur de chaque arc est informé de la demande en ressource de celui-ci.

L'algorithme présenté dans cette partie est basé sur la même stratégie que l'algorithme de connectivité : ordonner les allocations en donnant la priorité aux arc qui auront le plus de difficultés à trouver

des ressources disponibles. Cependant, il ne suffit plus d'estimer le nombre d'arcs dans la zone d'interférence, il faut également pouvoir estimer le nombre de ressources requises pour chacun de ces arcs. Pour cette raison la solution présentée ici introduit une nouvelle pondération qui tient compte du nombre de ressources requises dans le voisinage d'un arc.

6.3.2 Présentation de l'algorithme

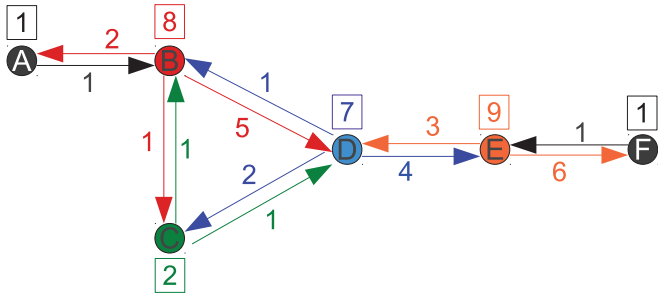


FIGURE 6.8: Calcul du poids

Nous supposons qu'un nœud est à même de connaître la quantité de ressources demandées pour l'ensemble des liens auxquels il est attaché. Cette information n'est cependant pas suffisante pour estimer la quantité de ressources requise dans la zone d'interférence d'un arc puisqu'elle ne donne qu'une vision locale de la demande en ressources.

Pour améliorer cette estimation, chaque nœud transmet une information supplémentaire : la quantité de ressources requise par l'ensemble de ses liens sortants. En additionnant la quantité de ressources pour l'ensemble de ses liens sortants avec celle de ses voisins, un nœud obtient une estimation de la quantité de ressources requises dans sa zone d'interférence. La figure 6.8 représente un exemple d'allocation ; la valeur indiquée au dessus de chaque nœud correspond à la somme du poids de chacun de ses arcs sortants et est transmise à l'ensemble des voisins. La limitation aux liens sortants permet d'éviter de compter plusieurs fois le même lien. Cette estimation est alors utilisée comme critère d'ordonnancement et remplace le poids du premier algorithme. À partir de ce poids, un nouveau poids modifié P'_n est calculé. Pour tenir compte du taux de satisfaction des demandes, c'est-à-dire du ratio entre quantité de ressources allouées et quantité de ressources demandée, la définition du poids modifié change et devient : $P'_n = P_n$ - *Quantité de ressources alloués sur les liens sortants de* n si toutes les demandes de ces liens sortants n'ont pas encore été pleinement satisfaites, et 0 sinon. Le critère d'ordonnancement ne change pas, il porte d'abord sur le poids modifié puis sur le poids et enfin sur l'identifiant.

Lorsqu'un nœud A est prioritaire pour une allocation, il identifie le nœud B qui possède le poids modifié le plus élevé dans son voisinage et attribue un nouveau canal sur l'arc $A \rightarrow B$. Le nœud A continue à allouer des ressources sur ses liens sortants selon les mêmes critères jusqu'à ce qu'il perde la priorité. Le choix de la ressource à allouer se fait de la même manière que pour l'algorithme précédent. L'algorithme cherche parmi l'ensemble des ressources libres celle qui lui permet de maximiser la réutilisation spatiale. L'identification des ressources maximisant la réutilisation spatiale se fait à l'aide des états utilisés par

l'algorithme de correction de conflits. Si une telle ressource n'existe pas l'algorithme prend une ressource libre disponible.

Algorithme 6.3.1: Algorithme d'allocation avec gestion des demandes

Données :

$P_n \leftarrow$ Poids du nœud n;

$P'_n \leftarrow$ Poids modifié du nœud n;

$ID_n \leftarrow$ Identifiant du nœud n;

P'max$_{V_n} \leftarrow$ Poids modifié le plus élevé dans le voisinage de n;

Pmax$_{V_n} \leftarrow$ Poids le plus élevé dans le voisinage de n;

IDmax$_{V_n} \leftarrow$ Identifiant le plus élevé dans le voisinage de n;

```
1 tant que P'_n ≥ P'max_{V_n} OU P_n ≥ Pmax_{V_n} OU ID_n > IDmax_{V_n} faire
2     candidats ←Voisins_p \ {voisins dont le lien est totalement satisfait};
3     candidat ← nœud ∈ candidats de poids modifié maximum;
4     si ∃ un canal c tq c est utilisable et maximise la réutilisation spatiale alors
5         Allocation de c sur le lien sortant vers candidat;
6     sinon si ∃ un canal c tq c est utilisable alors
7         Allocation de c sur le lien sortant vers candidat;
```

6.3.3 Validation de l'algorithme d'allocation avec QoS

6.3.3.1 Génération des demandes d'allocation

Pour valider l'algorithme 6.3.1, une fonction chargée de générer les demandes d'allocation a été ajoutée au simulateur java utilisé pour tester l'algorithme précédent. Pour créer une nouvelle demande, cette méthode choisit uniformément deux nœuds du réseau. Ces nœuds vont être considérés l'un comme la source et l'autre comme la destination de la demande d'allocation. L'algorithme de Dijkstra [Dij59] est ensuite utilisé pour trouver le plus court chemin reliant ces deux nœuds. La demande de ressources de chacun de ces arcs va ensuite être augmentée d'une valeur choisie uniformément sur l'intervalle [D_{min}, D_{max}] où D_{min} et D_{max} sont des paramètres de simulations. Cette fonction attribue également une durée de vie à ces allocations. Cette durée est choisie uniformément dans l'intervalle [TTL_{min}, TTL_{max}]. Tout comme pour D_{min} et D_{max}, TTL_{min} et TTL_{max} sont des paramètres de simulations.

Par exemple en prenant la même topologie que celle de la figure 6.8, dans un premier temps, le générateur d'allocation choisit le nœud A comme source et le nœud E comme destination. L'algorithme de Dijkstra identifie le plus court chemin : $A \rightarrow B \rightarrow D \rightarrow E$. L'algorithme détermine aléatoirement la quantité de ressources (4 unités) et le temps de vie de cette demande (2 minutes). La demande en ressource des arcs $A \rightarrow B$, $B \rightarrow D$ et $D \rightarrow E$ est alors augmentée de quatre unités. Au bout de deux minutes, ces quatre unités sont retranchées à la demande de ces arcs.

6.3.3.2 Déroulement des simulations

L'algorithme 6.3.1 est conçu pour satisfaire les demandes de ressources et pour adapter l'allocation aux modifications de ces demandes. Plus le temps de convergence de l'algorithme est court, plus ce dernier est capable de répondre à un rythme élevé de modification des demandes d'allocations. Pour pouvoir étudier ce temps de convergence, plusieurs simulations ont été réalisées sur des topologies aléatoires. L'objectif de ces simulations est d'étudier l'impact de la longueur du chemin ainsi que celui de la quantité de ressources allouées sur le temps de convergence.

Pour chacune de ces simulations, la demande initiale des arcs est fixée à une unité et tant que ces demandes ne sont pas satisfaites, aucune demande n'est générée. Dans la première partie des simulations, l'algorithme 6.3.1 a donc le même comportement que l'algorithme 6.2.3.

Lorsque cette initialisation est terminée, le générateur d'allocation crée une première demande exprimant la quantité de ressources à allouer. Les nœuds concernés par cette demande attribuent de nouvelles ressources grâce à l'algorithme d'allocation 6.3.1 afin de satisfaire les arcs concernés. Si des conflits apparaissent suite à ces nouvelles allocations, ils seront alors corrigés par l'algorithme de correction de conflits qui fonctionne en parallèle avec l'algorithme d'allocation. Lorsque cette première demande est satisfaite et que l'algorithme de correction de conflits a terminé sa phase de convergence, le simulateur supprime la demande. L'algorithme d'allocation détecte ce nouveau changement et libère les ressources qui étaient liées à cette demande. Lorsque cette phase est terminée, une nouvelle demande est générée. Pour obtenir les résultats présentés ici, ce cycle a été effectué cent fois pour chaque simulation.

6.3.3.3 Temps de convergence

Nous nous sommes intéressés à l'impact de la longueur du chemin et du nombre de ressources demandées sur le temps de convergence de l'algorithme d'allocation. Les résultats, représentés par la figure 6.9, montrent que l'augmentation de ce temps de convergence dépend de ces deux paramètres.

Influence de la longueur des chemins En ce qui concerne l'influence de la taille du chemin, plus un chemin est long, plus il y a d'arcs sur lesquels il faut allouer de nouvelles ressources. L'algorithme 6.3.1 a donc plus d'allocations à réaliser, le temps de convergence s'en trouve donc augmenté. Nous pouvons cependant observer que l'augmentation du temps de convergence n'est pas proportionnelle à l'augmentation de la longueur du chemin. Ce phénomène est dû à l'accroissement du nombre d'allocations simultanées dans le réseau. En effet, l'ordonnancement s'effectue uniquement entre des nœuds voisins. Deux nœuds qui ne sont pas voisins peuvent donc allouer leurs ressources simultanément. Ainsi, plus la longueur du chemin augmente et plus le nombre d'allocations simultanées s'accroît. L'augmentation du nombre d'allocations à réaliser est donc compensé par l'augmentation du nombre d'allocations simultanées le long du chemin. Pour cette raison le temps de convergence n'est pas proportionnel à la longueur du chemin.

Influence du nombre de ressources demandées La courbe indique également que le temps de convergence augmente avec le nombre de ressources demandées. Ce résultat montre bien que l'augmentation du nombre d'allocations dues à l'accroissement de la demande induit une augmentation du temps de convergence. On peut cependant observer que ce temps de convergence n'augmente pas proportionnellement avec le nombre de demandes mais tend à se stabiliser.

Ce phénomène est dû à la première vague d'allocations qui suit la création d'une nouvelle demande. Au moment où cette demande est créée, le système est stable. Toutes les demandes sont donc satisfaites, le poids modifié de chaque nœud est nul. Suite à la création de la nouvelle demande, les nœuds concernés mettent à jour la valeur de leur poids et de leur poids modifié. Puisque la demande de certains de leurs liens n'est pas satisfaite, ils tentent de réaliser de nouvelles allocations. Pour cela, ils comparent d'abord leur poids modifié avec celui de leurs voisins.

Prenons l'exemple d'un nœud A dont la demande sur un de ses liens sortant est augmentée. Puisque la demande de ce lien n'est pas totalement satisfaite, Le poids modifié du nœud A est strictement supérieur à zéro. Le poids modifié de ses voisins qui sont concernés par la nouvelle allocation doit également changer. Cependant A n'a peut être pas encore reçu le message de ses voisins indiquant qu'il ont également de nouvelles allocations à réaliser. Du point de vue de A, ses voisins ont donc tous un poids modifié de zéro. Il va donc tenter d'allouer les ressources manquantes pour satisfaire la demande. Ce qui est vrai pour A l'est également pour ses voisins ainsi que pour tous les nœuds appartenant au chemin. L'ensemble des nœuds concernés par la nouvelle allocation aura donc le même comportement. Une importante vague d'allocation va donc avoir lieu dans le réseau. Ces allocations quasiment simultanées ont de fortes chances de créer des conflits. Cependant l'algorithme de correction de conflits pourra libérer certaines allocations afin de corriger les conflits. Pendant ce temps, les poids modifiés des nœuds auront le temps de converger. Ainsi, au moment où l'algorithme d'allocation tentera d'allouer de nouvelles ressources pour remplacer celles qui ont été libérées par la correction des conflits, l'ordonnancement sera de nouveau fondé sur une valeur correcte du poids modifié, ce qui permettra de n'avoir qu'une seule allocation dans le voisinage et donc de limiter les risque de conflits d'allocation.

Le fait que le temps de convergence n'augmente pas proportionnellement avec le nombre de demandes est du à la première vague d'allocation. En effet, plus la quantité de ressources demandées est élevée et plus le nombre d'allocations tentées pendant cette vague sera élevé. Certaines généreront des conflits, mais toutes les allocations qui passeront la phase de correction seront autant d'allocations que l'algorithme 6.3.1 ne devra pas retenter.

6.3.4 Amélioration du temps de convergence

Dans toutes les solutions que nous avons présentées, chaque nœud transmet périodiquement un message à l'ensemble de ses voisins. Ce message contient l'état des variables dont ces nœuds ont besoin pour prendre leurs décisions. Ce système permet à un nœud d'informer son voisinage des mises à jour effectuées sur ses variables de manière périodique. Lorsqu'un message est perdu, les nœuds destinataires de ce mes-

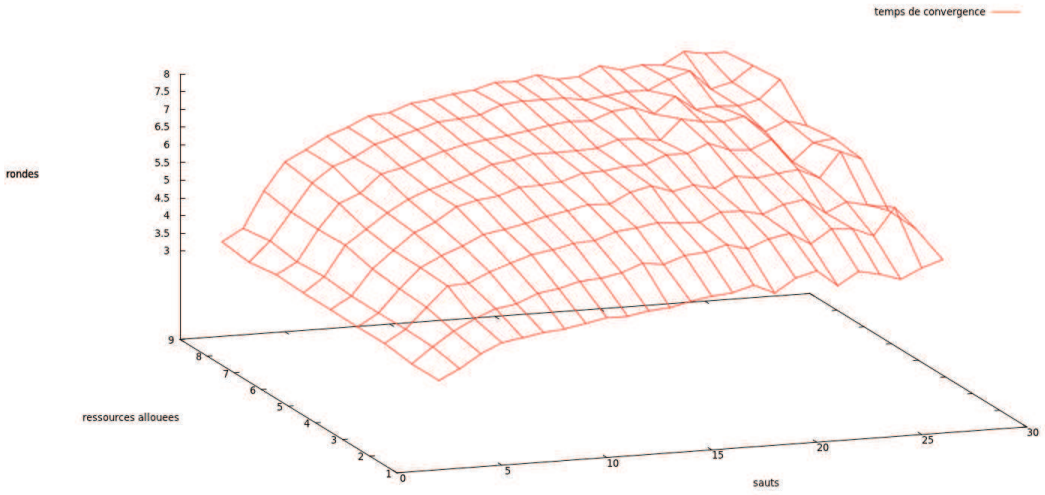


FIGURE 6.9: Influence de la longueur du chemin et de la quantité de ressources alloués sur le temps de convergence

sage n'ont pas accès aux dernières mises à jour que ce message pouvait contenir. La perte d'un message freine ainsi la propagation des informations et ralentira la convergence de l'algorithme. Le taux de pertes de messages influence donc le temps de convergence de l'algorithme. L'inconvénient de cette méthode est qu'elle nécessite une bande passante constante pour transmettre les messages de contrôle, que l'algorithme soit en convergence ou qu'il ait déjà réalisé l'ensemble des allocations demandées. Pour améliorer les performances de notre algorithme, nous avons introduit deux fréquences d'envoi des messages. Une première, très élevée, est utilisée par un nœud en phase de convergence. Une seconde, beaucoup plus faible, adaptée aux nœuds n'ayant plus de mise à jour à effectuer. Dans le second cas, il est toujours nécessaire de maintenir une fréquence d'envoi minimale à cause des pertes de messages ou tout simplement pour permettre à un nœud de détecter l'apparition ou la disparition d'un voisin lorsque les nœuds sont mobiles. La principale difficulté est de détecter l'état stable de la phase de convergence.

6.4 Conclusion

Au cours de ce chapitre, nous avons proposé deux algorithmes de répartition de ressources. Chacun d'entre eux est basé sur un ordonnancement des allocations en fonction de leur priorité. Bien que les ressources soient allouées sur les arcs, les nœuds sont les seuls éléments capables de réaliser ces allocations. Dans nos travaux nous supposons que le prédécesseur d'un arc (c'est-à-dire le nœud en position d'émetteur) est responsable des allocations sur celui-ci. Pour ordonner les allocations il faut donc ordonner les prédécesseurs de chaque arc. Cette classification est réalisée grâce à trois paramètres dont la définition est propre à chaque algorithme.

Le premier algorithme est conçu pour garantir la connectivité du réseau. Son rôle consiste à allouer un canal à chaque arc du graphe de connectivité tout en cherchant à minimiser le nombre total de canaux

utilisés. La priorité des allocations est calculée à partir d'une approximation du nombre d'arcs situés dans le 2-voisinage de chaque nœud. En cas d'égalité, l'ordonnancement se fait sur les identifiants des nœuds afin d'obtenir une relation d'ordre stricte qui évite que deux voisins ne décident d'allouer simultanément une même ressource.

Le second algorithme est prévu pour garantir un certain niveau de qualité de service. Son rôle consiste à attribuer des canaux en fonction de la demande de chaque arc du graphe de connectivité. La priorité des allocations est cette fois calculée à partir d'une approximation du nombre de ressources demandées par les arcs situés dans le 2-voisinage de chaque nœud. Tout comme pour l'algorithme précédent l'ordonnancement se fait en dernier recours sur les identifiants afin d'éviter des allocations simultanées dans le voisinage.

Chapitre 7

Conclusion et perspectives

Au cours de cette thèse, nous avons proposé à la fois de nouveaux algorithmes pour l'allocation de ressources dans les réseaux ad hoc ainsi qu'un modèle de mobilité, capable de reproduire le déplacement des forces de la protection civile. Cette thématique a été définie dans le projet RAF (Réseaux Ad hoc à forte efficacité) du pôle de compétitivité Systematic.

Dans un premier temps, nous nous sommes intéressés au problème de la modélisation du déplacement des équipes de la protection civile afin de valider nos solutions algorithmiques par simulation. Dans presque tous les travaux que nous avons étudiés, les modèles de mobilité ne tiennent compte que d'un seul de ces aspects. Quelques travaux proposent bien des modèles composites, construits à partir de plusieurs modèles existants afin de reproduire plusieurs caractéristiques du déplacement. Cependant, ils intègrent tous le Random Way Point (paragraphe 2.2.1), un modèle dont les faiblesses ont été mises en évidence par de nombreuses études.

Le modèle C2MDA de déplacement des équipes de la protection civile que nous avons développé est un modèle réaliste, capable de reproduire les principaux aspects du déplacement des secouristes, à savoir : une mobilité de groupe, un déplacement individuel qui reproduit les caractéristiques du déplacement humain et un mécanisme de contournement d'obstacles. Ce modèle ne repose pas sur RWP pour représenter le déplacement individuel des secouristes. Au contraire, pour générer la trajectoire des agents mobiles, nous nous sommes inspirés du Lévy-Walk, spécialement conçu pour reproduire les principales caractéristiques du déplacement humain(paragraphe 2.2.5). Le déplacement de groupe est quant à lui géré par le modèle RPGM qui permet d'affecter des trajectoires similaires à plusieurs mobiles (paragraphe 2.2.3). Pour contourner les obstacles présents dans la zone de simulation, certains auteurs ont proposé une solution fondée sur la construction d'un diagrammes de Voronoï (paragraphe 2.2.4.2). Cependant, nous avons mis en évidence que cette approche ne permettait pas toujours aux agents mobiles de contourner réellement les obstacles. En proposant de rajouter des points de référence supplémentaire pour la construction des chemins autour des obstacles, nous avons pu améliorer cette méthode et permettre aux agents mobiles de

contourner correctement les obstacles. L'évaluation de notre modèle C2MDA a montré que les trajectoires des agents mobiles reproduisaient les caractéristiques du mouvement humain tout comme celles d'un déplacement en groupe. Nous avons également montré que la présence d'obstacles avait une influence non négligeable sur le degré des nœuds et sur les temps d'inter-contact. Comme ces deux paramètres peuvent avoir un impact sur les protocoles testés, il est donc important de prendre en compte les obstacles lors des simulations.

Toutefois, notre modèle de mobilité ne gère pas les déplacements à l'intérieur des bâtiments. Or, les équipes d'interventions peuvent être amenées à y entrer. Il serait donc intéressant, dans la continuité de nos travaux, d'intégrer cet aspect dans notre modèle. D'autre part, nous avons considéré que tous les obstacles sont dangereux et ce avec le même niveau de gravité. Or certains obstacles peuvent ne présenter aucun danger, d'autres seulement un risque modéré tandis que certains obstacles doivent obligatoirement être contourner. Gérer ces différents niveaux de danger permettrait également d'améliorer le réalisme de notre modèle.

La seconde partie de cette thèse consistait à développer de nouveaux algorithmes d'allocation de ressources. Au cours d'une intervention, les forces de la protection civile peuvent avoir besoin d'établir des communications audio ou vidéo, que ce soit avec leurs collègues ou avec le coordinateur de l'opération. Ces communications sont donc au cœur de la réussite des opérations et sont d'un intérêt stratégique. Le réseau de communication doit donc offrir une qualité de services adaptée à chacune de ces communications. Actuellement, la gestion de cette qualité de service passe par des approches centralisées. Or, ces équipes peuvent avoir besoin de communiquer en l'absence de toute infrastructure. Nous avons donc cherché de nouvelles solutions à la problématique de l'allocation de ressources dans les réseaux ad hoc avec une gestion de la qualité de service. Dans un premier temps, le projet RAF a choisi une technologie de communication qui permette une vraie garantie de ressources : WiMAX en mode ad hoc. Nos algorithmes d'allocation ont donc porté sur l'allocation de time-slots par lien de manière à d'abord, éviter d'allouer la même ressource à deux liens dans une zone d'interférence donnée puis à prendre en compte le trafic dans la politique d'allocation.

Dans la plupart des travaux existants, les auteurs supposent que les zones d'interférence et de transmission sont de taille identique. Certaines études ont rappelé que dans la réalité, un signal dont la puissance est trop faible pour être correctement reçu pouvait être encore suffisamment forte pour générer des interférences. Ces études ont donc montré que cette hypothèse, généralement admise, ne permettait pas de modéliser correctement les interférences et que, par conséquent, de nombreux conflits d'allocation n'étaient pas détectés. Nos travaux sur l'allocation de ressources ont donc été réalisés en prenant en compte une zone d'interférence deux fois plus étendue que la zone de transmission. Dans ces conditions, la détection et la correction de conflits devient un problème à part entière. Afin de simplifier les solutions proposées, nous avons donc décidé de séparer le problème de la répartition des ressources de celui de la détection et de la correction de conflits. Cette séparation permet également de développer des stratégies

différentes (allocation équitable, gestion du trafic, etc.) pour chacune de ces parties afin d'adapter la solution obtenue au type de réseau considéré.

Pour détecter et corriger les conflits, nous avons développé un premier algorithme distribué. Nous avons prouvé le caractère auto-stabilisant de celui-ci ainsi que son temps de convergence, borné et indépendant de la taille du réseau. Afin d'améliorer la libération des ressources, nous avons proposé un second algorithme qui permet à la fois de diminuer le nombre d'allocations supprimées mais également de tenir compte de la priorité des allocations lors de ces suppressions. Chacun de ces deux algorithmes permet de régler les problèmes liés aux conflits d'allocations.

Nous avons également proposé deux solutions au problème de la répartition des ressources disponibles. La première permet d'attribuer un canal à chaque lien unidirectionnel du réseau tout en minimisant le nombre total de canaux nécessaires. Les simulations effectuées pour différentes topologies ont montré que l'augmentation de la taille du réseau n'avait qu'un impact limité sur les performances de l'algorithme en termes de temps de convergence et de quantité de ressources nécessaires. Nous avons ensuite proposé une seconde solution qui permet de prendre en compte la demande de chaque lien unidirectionnel du réseau. Cette solution permet ainsi de garantir une qualité de service en attribuant et en adaptant les ressources en fonction du trafic et de ses fluctuations au cours du temps. Là encore, les simulations réalisées ont montré que l'augmentation des demandes dans le réseau avaient un impact limité sur son temps de convergence.

Toutefois, ces algorithmes sont conçus pour une allocation de ressources sur les liens. Nous avons vu dans l'état de l'art qu'il était également possible d'attribuer les ressources aux nœuds du réseau. Les performances de ces deux approches dépendent principalement du type de communication considéré. Dans le cas de communications point-à-point, l'allocation sur les liens est plus efficace. Par contre, lorsque les communications sont de type *broadcast*, l'allocation sur les nœuds présente de meilleurs résultats. Dans la réalité, les équipements d'un réseau peuvent avoir besoin de ces deux types de communication. Trouver une solution à l'allocation sur les nœuds et surtout une solution hybride, nœuds et liens, permettrait d'adapter le type d'allocation en fonction de la nature des communications présentes dans le réseau. D'autre part, les algorithmes présentés dans ce mémoire constituent une première étape dans laquelle nous nous sommes focalisés sur la gestion d'une zone d'interférence supérieure à la zone de transmission. Dans la continuité des travaux de cette thèse, nous pourrions étudier l'adaptation de ces algorithmes à un réseau mobile et évaluer leurs performances à l'aide de notre modèle C2MDA. Nous pensons que ces algorithmes devraient fonctionner dans ce type de réseau à condition que les changements topologiques liés à la mobilité des nœuds ne soient pas trop importants par rapport au temps de convergence de notre solution.

Annexe A

Rappels de théorie des graphes

Quelques définitions présentées dans cette partie sont issues de [Ber66].

Définition A.1. Graphe orienté symétrique *Un graphe orienté G(V,E) est symétrique si pour tout arc* $[u \to v] \in E$*, il existe un arc* $[v \to u] \in E$*.*

Définition A.2. Adjacence d'un sommet *Soient un graphe non-orienté G = (V, E) et* $v \in V$ *, un sommet de G. L'ensemble Adj(v) est l'ensemble des sommets u de G tel qu'il existe une arête (u,v) dans le graphe G. On dit que u et v sont* ***voisins****.*

Définition A.3. Adjacence d'une arête *Soient un graphe non-orienté G = (V, E) et* $(u_1, v_1) \in E$ *, une arête de G. L'ensemble Adj(*(u_1, v_1)*) est l'ensemble des arêtes (u, v) de G tel qu'il existe un sommet commun entre arêtes* (u_1, v_1) *et (u, v).*

Définition A.4. Adjacence d'un arc *Soient un graphe orienté G = (V, E) et* $(u_1 \to v_1) \in E$ *, un arc de G. L'ensemble Adj(*$(u_1 \to v_1)$*) est l'ensemble des arcs* $(u \to v)$ *de G tel qu'il existe un sommet commun entre arcs* $(u_1 \to v_1)$ *et* $(u \to v)$*.*

Définition A.5. Chaîne dans un graphe *Soient G = (V, E) un graphe non orienté et u et* $v \in V$ *deux sommets de G. Une chaîne de longueur k-1 joignant les sommets u et v est une séquence de sommets* $< u = u_1, u_2, ..., u_k = v >$ *telle que pour tout i ∈ [1,k-1], il existe une arête* $[u_i\ ,\ u_{i+1}\ ] \in E$*.*

Définition A.6. Degré d'un sommet *Soit G = (V, E) un graphe et* $v \in V$ *un sommet de G. Le degré de v est la quantité* $|\ \{u : [u, v] \in E\}\ |$*.*

Définition A.7. Distance entre deux sommets *La taille de la chaîne de longueur minimale joignant les sommets u et v (respectivement du chemin allant de u à v) est appelé la distance entre les sommets u et v (respectivement du sommet u au sommet v).*

Définition A.8. Distance entre deux arêtes *Soient G(V,E) un graphe non orienté,* $[u, v]$ *et* $[u', v']$*, deux arêtes de ce graphe. Soit S, la séquence d'arêtes* $< [u, v] = [u_1, v_1], [u_2, v_2], ..., [u_k, v_k] = [u', v'] >$ *, telle que pour tout i ∈ [1,k-1], les arêtes* $[u_i, v_i]$ *et* $[u_{i+1}, v_{i+1}]$ *sont adjacentes. La distance entre les*

arêtes [u, v] et [u′,v′] est de k-1 où k représente le nombre d'éléments contenus dans la séquence S de taille minimale.

Définition A.9. Distance entre deux arcs *Soient G(V,E), un graphe orienté,* $[u \to v]$ *et* $[u' \to v']$*, deux arcs de ce graphe. Soit S, la séquence d'arc* $< [u \to v] = [u_1 \to v_1], [u_2 \to v_2], ..., [u_k \to v_k] = [u' \to v'] >$ *, telle que pour tout i ∈ [1,k-1], les arcs* $[u_i \to v_i]$ *et* $[u_{i+1} \to v_{i+1}]$ *sont adjacentes. La distance entre les arcs* $[u \to v]$ *et* $[u' \to v']$ *est de k-1 où k représente le nombre d'éléments contenus dans la séquence S de taille minimale.*

Définition A.10. N-voisinage d'un sommet *Soient G(V,E) un graphe et* $v \in V$ *un sommet de G. Le N-voisinage de v est l'ensemble des sommets u de G tel que la distance entre u et v est inférieur ou égale à N.*

Définition A.11. N-voisinage d'une arête *Soient G(V,E) un graphe et [u,v] ∈ E une arête de G. Le N-voisinage de* $[u, v]$ *est l'ensemble des arêtes* $[u', v']$ *de G tel que la distance entre* $[u, v]$ *et* $[u', v']$ *est inférieur ou égale à N.*

Définition A.12. N-voisinage d'un arc *Soient G(V,E) un graphe et* $[u \leftarrow v] \in E$ *un arc de G. Le N-voisinage de* $[u \leftarrow v]$ *est l'ensemble des arcs* $[u' \leftarrow v']$ *de G tel que la distance entre* $[u \leftarrow v]$ *et* $[u' \leftarrow v']$ *est inférieur ou égale à N.*

Définition A.13. Graphe complet *Soit G(V,E) un graphe. Le graphe G est* complet *si, pour tout couple de sommets (u,v) ∈ V , avec* $u \neq v$*, il existe une arête* $[u, v] \in E$*.*

Définition A.14. Clique *Soit G(V,E) un graphe. Une clique de ce graphe est un sous-ensemble de sommets formant un graphe complet.*

Bibliographie

[80204] IEEE standard for local and metropolitan area networks part 16 : Air interface for fixed broadband wireless access systems. Technical report, 2004.

[80207] IEEE Standard - Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pages C1 –1184, 12 2007.

[AEvdH09] O. Amini, L. Esperet, and J. van den Heuvel. A unified approach to distance-two colouring of planar graphs. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 273–282. Society for Industrial and Applied Mathematics, 2009.

[AGK08] Ehsan Aryafar, Omer Gurewitz, and Edward W. Knightly. Distance-1 Constrained Channel Assignment in Single Radio Wireless Mesh Networks. In *INFOCOM*, pages 762–770, 2008.

[AGPG+07] 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, 2007.

[AK11] Muhammad Zaheer Aslam and Abdur Rashid Khan. Comparison of Random Waypoint and Random Walk Mobility Model under DSR, AODV and DSDV MANET Routing Protocols. *CoRR*, abs/1104.2368, 2011.

[Ber66] C. Berge. *La théorie des graphes et ses applications.* Dunod, 1966.

[Bet01] Christian Bettstetter. Smooth is better than sharp : a random mobility model for simulation of wireless networks. In *Proceedings of the 4th ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, MSWIM '01, pages 19–27. ACM, 2001.

[BH04] F. Bai and A. Helmy. *Wireless Ad Hoc and Sensor Networks*, chapter A Survey of Mobility Modeling and Analysis in Wireless Adhoc Networks. Kluwer Academic Publishers, 2004.

[BHG06] D. BROCKMANN, L. HUFNAGEL, and T. GEISEL. The scaling laws of human travel. *Nature*, 439(7075) :462–465, 2006.

[BM10] J. Blair and F. Manne. An Efficient Self-stabilizing Distance-2 Coloring Algorithm. In *Structural Information and Communication Complexity : 16th International Colloquium,*

Sirocco 2009, Piran, Slovenia, May 25-27, 2009, Revised Selected Papers, volume 5869, page 237. Springer-Verlag New York Inc, 2010.

[BR03] A. Behzad and I. Rubin. On the performance of graph-based scheduling algorithms for packet radio networks. In *Global Telecommunications Conference, 2003. GLOBECOM'03. IEEE*, volume 6, pages 3432–3436. IEEE, 2003.

[Bre79] Daniel Brelaz. New methods to color the vertices of a graph. *Commun. ACM*, 22 :251–256, 1979.

[BRS03] Christian Bettstetter, Giovanni Resta, and Paolo Santi. The Node Distribution of the Random Waypoint Mobility Model for Wireless Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 2 :257–269, July 2003.

[BRS05] Sven Bittner, Wolf-Ulrich Raffel, and Manuel Scholz. The Area Graph-Based Mobility Model and Its Impact on Data Dissemination. In *Proceedings of the Third IEEE International Conference on Pervasive Computing and Communications Workshops*, PERCOMW '05, pages 268–272, 2005.

[BTW09] J. Baras, G. Theodorakopoulos, and J. Walrand. *Path Problems in Networks*. Morgan & Claypool Publishers, 2009.

[CBD02] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless communications and mobile computing*, 2(5) :483–502, 2002.

[CHC+07] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, pages 606–620, 2007.

[CLYE08] H. Cai, C.H. Lee, and D. Young-Eun. Invariance property of isotropic random walk mobility patterns in mobile ad-hoc networks. In *Communications, 2008. ICC'08. IEEE International Conference on*, pages 2141–2145. IEEE, 2008.

[DGL02] Dominique Dhoutaut and Isabelle Guérin Lassous. Impact of Heavy Traffic Beyond Communication Range in Multi-Hops Ad Hoc Networks. In *Proceedings of the Third International Network Conference (INC 2002)*, Plymouth, Royaume-Uni, July 2002.

[Dij59] E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1) :269–271, 1959.

[Dij74] E.W. Dijkstra. Self-Stabilizing Systems in Spite of Distributed Control. *Communications of the ACM*, 17(11) :643–644, 1974.

[DS10] K.L. Du and MNS Swamy. *Wireless communication systems : from RF subsystems to 4G enabling technologies*. Cambridge Univ Pr, 2010.

[DV07a] P. Djukic and S. Valaee. Distributed link scheduling for TDMA mesh networks. In *IEEE International Conference on Communications (ICC'07)*, pages 3823–3828. IEEE, 2007.

[DV07b] P. Djukic and S. Valaee. Link Scheduling for Minimum Delay in Spatial Re-Use TDMA . In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 3823–3828. IEEE, 2007.

[FGR03] G. Fertin, E. Godard, and A. Raspaud. Acyclic and k-distance coloring of the grid. *Information Processing Letters*, 87(1) :51–58, 2003.

[FM94] V.S. Frost and B. Melamed. Traffic modeling for telecommunications networks. *Communications Magazine, IEEE*, 32(3) :70–81, 1994.

[GH01] Jimmi Gronkvist and Anders Hansson. Comparison between graph-based and interference-based STDMA scheduling. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '01, pages 255–258. ACM, 2001.

[GHB08] M.C. Gonzalez, C.A. Hidalgo, and A.L. Barabási. Understanding individual human mobility patterns. *Nature*, 453(7196) :779–782, 2008.

[GKJ07] A.D. Gore, A. Karandikar, and S. Jagabathula. On high spatial reuse link scheduling in STDMA wireless ad hoc networks. In *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pages 736–741. IEEE, 2007.

[GSB+08] M. Gyarmati, U. Schilcher, G. Brandner, C. Bettstetter, Y. W. Chung, and Y. H. Kim. Impact of Random Mobility on the Inhomogeneity of Spatial Distributions. In *GLOBECOM*, pages 404–408, 2008.

[GZH06] Shashidhar Gandham, Ying Zhang, and Qingfeng Huang. Distributed Minimal Time Convergecast Scheduling in Wireless Sensor Networks. In *Proceedings of the 26th IEEE International Conference on Distributed Computing Systems*, ICDCS '06, pages 50–. IEEE Computer Society, 2006.

[Haa01] Z.J. Haas. Routing and mobility management protocols for ad-hoc networks, 2001. US Patent 6,304,556.

[HCS+05] Pan Hui, Augustin Chaintreau, James Scott, Richard Gass, Jon Crowcroft, and Christophe Diot. Pocket switched networks and human mobility in conference environments. In *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, WDTN '05, pages 244–251, 2005.

[HFB06] Jerome Haerri, Fethi Filali, and Christian Bonnet. Mobility Models for Vehicular ad hoc Networks : a Survey and Taxonomy. Technical Report EURECOM+1951, Institut Eurecom, France, 03 2006.

[HGPC99] 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*, MSWiM '99, pages 53–60, 1999.

[HMS+05] Wei-jen Hsu, Kashyap Merchant, Haw-wei Shu, Chih-hsin Hsu, and Ahmed Helmy. Weighted waypoint mobility model and its impact on ad hoc networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9 :59–63, January 2005.

[HT04] Ted Herman and Sébastien Tixeuil. A Distributed TDMA Slot Assignment Algorithm for Wireless Sensor Networks. In *Algorithmic Aspects of Wireless Sensor Networks*, pages 45–58. 2004.

[Hu93] L. Hu. Distributed code assignments for cdma packet radio network. *IEEE/ACM Transactions on Networking (TON)*, 1(6) :668–677, 1993.

[JBRAS03] 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, 2003.

[JBRAS05] A.P. Jardosh, E.M. Belding-Royer, K.C. Almeroth, and S. Suri. Real-world environment models for mobile network evaluation. *Selected Areas in Communications, IEEE Journal on*, 23(3) :622–632, 2005.

[JHR01] S. Jiang, D. He, and J. Rao. A prediction-based link availability estimation for mobile ad hoc networks. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1745–1752. IEEE, 2001.

[JM96] David B. Johnson and David A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In *Mobile Computing*, pages 153–181. Kluwer Academic Publishers, 1996.

[JYZ09] BIN JIANG, J. YIN, and S. ZHAO. Characterizing the human mobility pattern in a large street network. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 80(2), 2009.

[Kar72] R. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.

[KCC05] Stuart Kurkowski, Tracy Camp, and Michael Colagrosso. Manet simulation studies : The incredibles. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9 :50–61, 2005.

[Ket04] H.P.A. Ketterling. *Introduction to digital professional mobile radio*. Artech House Publishers, 2004.

[KKK06] M. Kim, D. Kotz, and S. Kim. Extracting a Mobility Model from Real User Traces. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–13. IEEE, 2006.

[Kub04] M. Kubale. *Graph colorings*, volume 352. Amer Mathematical Society, 2004.

[KV05] P. Kyasanur and N.H. Vaidya. Routing and interface assignment in multi-channel multi-interface wireless networks. In *Wireless Communications and Networking Conference*, volume 4, pages 2051–2056. IEEE, 2005.

[KV06] Pradeep Kyasanur and Nitin H. Vaidya. Routing and link-layer protocols for multi-channel multi-interface ad hoc wireless networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 10 :31–43, January 2006.

[La10] Richard La. Distributional Convergence of Intermeeting Times under the Generalized Hybrid Random Walk Mobility Model. *IEEE Transactions on Mobile Computing*, 9 :1201–1211, September 2010.

[LB07] Jean-Yves Le Boudec. Understanding the simulation of mobility models with Palm calculus. *Performance Evaluation*, 64 :126–147, February 2007.

[LH03] Ben Liang and Zygmunt J. Haas. Predictive distance-based mobility management for multidimensional PCS networks. *IEEE/ACM Trans. Netw.*, 11 :718–732, October 2003.

[LPW79] Tomás Lozano-Pérez and Michael A. Wesley. An Algorithm for Planning Collision-Free Paths Among Polyhedral Obstacles. *Commun. ACM*, 22 :560–570, October 1979.

[MDS10] Mahesh K. Marina, Samir R. Das, and Anand Prabhu Subramanian. A topology control approach for utilizing multiple channels in multi-radio wireless mesh networks. *Computer Networks : The International Journal of Computer and Telecommunications Networking*, 54 :241–256, February 2010.

[MM09] M. Musolesi and C. Mascolo. Mobility models for systems evaluation. A survey, 2009.

[NEL07] P.C. Ng, D.J. Edwards, and S.C. Liew. Colouring link-directional interference graphs in wireless ad hoc networks. In *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pages 859–863. IEEE, 2007.

[NTN10] D.T. Ngo, C. Tellambura, and H.H. Nguyen. Resource allocation for OFDMA-based cognitive radio multicast networks with primary user activity consideration. *Vehicular Technology, IEEE Transactions on*, 59(4) :1668–1679, 2010.

[PBDK09] Christos Papageorgiou, Konstantinos Birkos, Tasos Dagiuklas, and Stavros Kotsopoulos. Simulating mission critical mobile ad hoc networks. In *Proceedings of the 4th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks*, PM2HW2N '09, pages 143–150, 2009.

[PG10] Dr. Pradip Ghorpade Pravin Ghosekar, Girish Katkar. Mobile Ad Hoc Networking : Imperatives and Challenges. *IJCA Special Issue on MANETs*, (3) :153–158, 2010. Published by Foundation of Computer Science.

[PTBV10] S. Pomportes, J. Tomasik, A. Busson, and V. Vèque. Self-stabilizing algorithm of two-hop conflict resolution. *Stabilization, Safety, and Security of Distributed Systems*, pages 288–302, 2010.

[PTBV11] S. Pomportes, J. Tomasik, A. Busson, and V. Vèque. Resource Allocation in Ad Hoc Networks with Two-Hop Interference Resolution. *GLOBECOM*, pages 288–302, 2011.

[PTV10] S. Pomportes, J. Tomasik, and V. Veque. Ad hoc network in a disaster area : A composite mobility model and its evaluation. In *International Conference on Advanced Technologies for Communications (ATC)*, pages 17–22. IEEE, 2010.

[PTV11] S. Pomportes, J. Tomasik, and V. Veque. A composite mobility model for ad hoc networks in disaster areas. *REV Journal on Electronics and Communications*, 1(1), 2011.

[RAF08] RAF. *Recensement des Applications de Sécurité Civile*, 2008.

[Ram97] S. Ramanathan. A Unified Framework and Algorithm for (T/F/C)DMA Channel Assignment in Wireless Networks. In *Proceedings of the INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution*, INFOCOM '97, pages 900–, Washington, DC, USA, 1997. IEEE Computer Society.

[RBAB06] KN Ramachandran, EM Belding, KC Almeroth, and MM Buddhikot. Interference-Aware Channel Assignment in Multi-Radio Wireless Mesh Networks. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12. IEEE, 2006.

[RC005] *Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network*, volume 3, 2005.

[RC04] A. Raniwala and Tzi-cker Chiueh. Evaluation of a wireless enterprise backbone network architecture. In *Proceedings of the High Performance Interconnects, 2004. on Proceedings. 12th Annual IEEE Symposium*, HOTI '04, pages 98–104, 2004.

[RGC04] Ashish Raniwala, Kartik Gopalan, and Tzi-cker Chiueh. Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 8 :50–65, April 2004.

[RL93] Subramanian Ramanathan and Errol L. Lloyd. Scheduling algorithms for multihop radio networks. *IEEE/ACM Trans. Netw.*, 1 :166–177, April 1993.

[RMSM01] E.M. Royer, P.M. Melliar-Smith, and L.E. Moser. An analysis of the optimum node density for ad hoc mobile networks. In *Communications, 2001. ICC 2001. IEEE International Conference on*, volume 3, pages 857–861. IEEE, 2001.

[RSH+08] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong. On the Levy-Walk Nature of Human Mobility. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 924–932. IEEE, 2008.

[RW07] S. Reidt and S.D. Wolthusen. An evaluation of cluster head TA distribution mechanisms in tactical MANET environments. *Proc. of International Technical Alliance in Network and Informational Science*, pages 1–7, 2007.

[RWMX09] Injong Rhee, Ajit Warrier, Jeongki Min, and Lisong Xu. DRAND : Distributed Randomized TDMA Scheduling for Wireless Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 8 :1384–1396, October 2009.

[SAF06] the Statement of Requirements (SoR) for Public Safety Wireless Communications and Interoperability. The SAFECOM program, Department of Homeland Security, jan 2006.

[SCS99] S.M. Shin, C.H. Cho, and D.K. Sung. Interference-based channel assignment for DS-CDMA cellular systems. *Vehicular Technology, IEEE Transactions on*, 48(1) :233–239, 1999.

[SGJ09] Srikrishna Sridhar, Jun Guo, and Sanjay Jha. Channel assignment in multi-radio wireless mesh networks : a graph-theoretic approach. In *Proceedings of the First international conference on COMmunication Systems And NETworks*, COMSNETS'09, pages 180–189, Piscataway, NJ, USA, 2009. IEEE Press.

[SPR07] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and Focus : Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility. In *Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*, PERCOMW, pages 79–85, 2007.

[SPR08] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Efficient routing in intermittently connected mobile networks : the single-copy case. *IEEE/ACM Trans. Netw.*, 16 :63–76, February 2008.

[SS00] MM Sekhar and K.N. Sivarajan. Routing and scheduling in packet radio networks. In *Personal Wireless Communications, 2000 IEEE International Conference on*, pages 335–339. IEEE, 2000.

[ST05] Theodoros Salonidis and Leandros Tassiulas. Distributed dynamic scheduling for end-to-end rate guarantees in wireless ad hoc networks. In *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '05, pages 145–156. ACM, 2005.

[STB09] Stefania Sesia, Issam Toufik, and Matthew Baker. *LTE, The UMTS Long Term Evolution : From Theory to Practice*. Wiley Publishing, 2009.

[SWB09] S. Stanczak, M. Wiczanowski, and H. Boche. *Fundamentals of resource allocation in wireless networks : theory and algorithms*, volume 3. Springer Verlag, 2009.

[TC05] I. Tinnirello and S. Choi. Temporal fairness provisioning in multi-rate contention-based 802.11 e WLANs. In *World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. Sixth IEEE International Symposium on a*, pages 220–230. IEEE, 2005.

[THB+02] J. Tian, J. Haehner, C. Becker, I. Stepanov, and K. Rothermel. Graph-based mobility model for mobile ad hoc network simulation. In *isvlsi*, page 0337. Published by the IEEE Computer Society, 2002.

[VGL08] Rémi Vannier and Isabelle Guérin-Lassous. Partage équitable de la bande passante dans les réseaux ad hoc. In *Colloque Francophone sur l'Ingénierie des Protocoles (CFIP)*, Les Arcs, France, March 2008.

[Wes02] K. Wesoĺowski. *Mobile communication systems*. Wiley, 2002.

[Woo28] E.W. Woolard. Investigations on the Theory of the Brownian Movement, 1928.

[WP67] D.J.A. Welsh and MB Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1) :85, 1967.

[YLN03] 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. IEEE, 2003.

[ZM11] C. Zhou and NF Maxemchuk. Scalable max-min fairness in wireless ad hoc networks. *Ad Hoc Networks*, 9(2) :112–119, 2011.