ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE
CONCENTRATION RÉSEAUX DE TÉLÉCOMMUNICATIONS
M.Ing.

PAR
Stéphane BOYER

ALGORITHMES INCITATIFS POUR LE PARTAGE DE RESSOURCES DANS LES RÉSEAUX 802.11 EN PRÉSENCE DE NOEUDS ÉGOÏSTES

MONTRÉAL, LE 22 DÉCEMBRE 2011

PRÉSENTATION DU JURY

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Jean-Marc Robert, directeur de mémoire
Département de génie logiciel et des TI, École de technologie supérieure

M. Michel Kadoch, président du jury
Département de génie électrique, École de technologie supérieure

M. Abdelouahed Gherbi, membre du jury
Département de génie logiciel et des TI, École de technologie supérieure

IL A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 12 DÉCEMBRE 2011

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

REMERCIEMENTS

Je tiens à remercier mon directeur de recherche M. Jean-Marc Robert pour son aide financière ainsi que de ces encouragements qui m'ont permis de mener à terme ce projet. De plus, je tiens à remercier Mme. Chantal Gamache du service aux étudiants de l'ETS pour son aide à la rédaction de ce document.

ALGORITHMES INCITATIFS POUR LE PARTAGE DE RESSOURCES DANS LES RÉSEAUX 802.11 EN PRÉSENCE DE NOEUDS ÉGOÏSTES

Stéphane BOYER

RÉSUMÉ

La présence des noeuds égoïstes dans les réseaux 802.11 est un problème d'importance car leurs comportements peuvent être nuisible à tous. La détection des noeuds égoïstes permet d'identifier la source du problème. Ensuite, puisque la détection seule n'empêche rien, des actions doivent être entreprises contre les noeuds égoïstes. Ces actions ont pour but de faire cesser ces comportements égoïstes. Dans le cas d'un noeud égoïste qui est rationnel, ses punitions reçues contre ses comportements égoïstes devraient l'inciter à redevenir collaboratif.

Dans le cadre de cette étude, les noeuds dit égoïstes ont modifié leur implémentation de la norme 802.11. Des périodes de retrait plus courtes sont attendues par les noeuds égoïstes, ce qui leurs donnent des avantages en terme de débits de données transférés. Pour contrer ce problème, des algorithmes de détection et de réaction contre les noeuds égoïstes sont développés.

Mots-clés : IEEE 802.11, DCF, backoff, noeud égoïste

ALGORITHMES INCITATIFS POUR LE PARTAGE DE RESSOURCES DANS LES RÉSEAUX 802.11 EN PRÉSENCE DE NOEUDS ÉGOÏSTES

Stéphane BOYER

ABSTRACT

The presence of selfish nodes in 802.11 networks is a problem because their behaviour can be harmful for everyone. Therefore, it is important to detect selfish nodes. However, detection alone cannot prevent selfish behaviours. Actions must be initiated against selfish nodes. The main goal for these actions is to punish selfish behaviour. If a selfish node is rational, the potential punishments will incite the selfish node to behave cooperatively.

In this thesis, selfish nodes have modified their implementation of the 802.11 protocol. Selfish nodes have reduced their backoff values. This results with a higher bandwidth for these nodes. Reduced backoff values can be done by using a narrower contention window. Contention windows, as well as all communications between nodes, are specified by the 802.11 protocol. Thus, a selfish node would simply have to deviate from the protocol specifications.

To resolve this problem, detection methods and actions against selfish nodes are developed. If the proposed solutions are applied, networks containing selfish nodes may perform better.

Keywords: IEEE 802.11, DCF, backoff, selfish

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

ACK	Acquittement
ADC	Anomaly Detection Component
BEB	Binary Exponential Backoff
CSMA/CA	Carrier sense multiple access with collision avoidance
CTS	Clear To Send
CUSUM	Nonparametric Cumulative Sum
CW	Contention Window ou fenêtre de contention
DEC	Deviation Estimation Component
DIFS	Distributed Coordination Function Interframe Space
DMC	Decision Making Component
DOMINO	Detection Of greedy behavior in the MAC layer of IEEE 802.11 public NetwOrks
GNU GPL	GNU General Public License
IEEE	Institute of Electrical and Electronics Engineers
MAC	Media Access Control
NAV	Network Allocation Vector
OSI	Open Systems Interconnection
PRB	Predictable Binary Backoff
RTS	Request To Send
SIFS	Short Interframe Space
SPRT	Sequential Probability Ratio Test
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

INTRODUCTION

La bande passante limitée est un problème commun à tous les réseaux de télécommunication. Pour échanger des données entre deux équipements, des liaisons doivent être établies. Ces liaisons utilisent divers supports de transport tels que des fils de cuivre, des fibres optiques ou des ondes radio. Les données doivent être encodées puis modulées en signaux pouvant transiter dans ces supports. Ces signaux peuvent plus ou moins s'atténuer selon la distance entre les équipements, leur fréquence de modulation et leur puissance d'émission. De plus, l'éventail des fréquences disponibles pour transporter ces signaux n'est pas illimité. Tous ces facteurs limiteront la largeur de bande nécessaire pour transporter ces signaux. Puisque la largeur de bande est limitée, la quantité d'information qu'il sera possible de transférer sera également limitée (Gokhale, 2004).

Lorsqu'une ressource est limitée, une compétition peut s'installer entre les différents consommateurs de cette ressource dans le but de l'obtenir en totalité ou en partie. Les protocoles de communication, tels que la norme 802.11 de l'IEEE (IEEE, 1999), essaient, entre autres de favoriser un partage équitable de la bande passante. La bande passante devrait être partagée équitablement entre les différents noeuds du réseau si tous sont régis par la norme 802.11. Dans un milieu où une ressource est limitée et où aucune autorité ne peut régir son utilisation, la prise de contrôle de la ressource peut se faire de façon anarchique si tous décident de se l'accaparer en même temps, sans contrôle. Dans ces circonstances, si chacun essaie de s'accaparer de la bande passante en ne respectant pas la norme 802.11, il y aura une augmentation du nombre de paquets perdus et une baisse dans les débits de données transmises (Liu *et al.*, 2009). Tous risquent d'y perdre.

Le problème avec la norme 802.11 est que chacun des noeuds du réseau a la responsabilité d'appliquer la norme. Rien n'empêche un noeud de faire fi de ce que décrit la norme dans le but d'avoir une partie plus grande de la bande passante par rapport à ce qu'il aurait obtenu en

respectant la norme. Le fait qu'un noeud s'octroie une part plus grande de la bande passante se fait au détriment des autres noeuds.

Rien n'empêche un noeud d'être égoïste et d'agir pour son seul profit. Il devient donc nécessaire que les autres noeuds collaboratifs respectant la norme essaient de contrer un noeud égoïste et ses conséquences. Ainsi, un noeud collaboratif peut récupérer une partie de la bande passante volée par le noeud égoïste en devenant à son tour égoïste. Par contre, si chacun des noeuds décide de devenir égoïste, alors il s'en suivra une perte de performance pour l'ensemble des noeuds. Pour forcer le respect de la norme 802.11 par tous et le plus souvent possible, un noeud ne doit pas devenir égoïste automatiquement. Un noeud collaboratif doit appliquer un processus de détection rigoureux des noeuds égoïstes. Cette détection peut être suivie ou non d'une action réfléchie dont le but est de contrer l'action du noeud égoïste. Si le noeud est rationnel, cette action devrait le ramener à la raison en l'incitant à respecter la norme. Sinon, il devrait remarquer qu'être égoïste ne lui apporte que des désavantages.

Pour bien comprendre la façon dont le comportement égoïste est produit, le chapitre 1 récapitule les aspects pertinents à la problématique du fonctionnement de la norme 802.11. Ensuite, le chapitre 2 décrit les différentes techniques déjà connues relativement à la problématique présentée dans ce document. Les algorithmes tentant de détecter, et par la suite, de contrecarrer les noeuds égoïstes sont décrits dans le chapitre 3. Pour assurer la validité des algorithmes développés, le chapitre 4 expose les différents résultats de l'implémentation des algorithmes selon différents cas d'utilisation. La conclusion fait un retour sur les résultats obtenus pour les algorithmes déployés. En fait, ce travail apporte les détails de l'implémentation des algorithmes du problème cité dans l'article (Boyer *et al.*, 2012).

CHAPITRE 1

PROBLÉMATIQUE

Dans ce chapitre, un retour sur certains éléments de la norme 802.11 est fait. Entre autres, les modes de fonctionnement de la norme et les fenêtres de contention y sont présentés. La définition de ces éléments permettent une meilleure compréhension de la problématique et des solutions proposées. Par la suite, les caractéristiques des noeuds égoïstes sont définies. La détection et les réactions possibles contre les noeuds égoïstes sont décrites brièvement. Finalement, les objectifs de ce document sont décrits.

1.1 Réseaux basés sur la norme 802.11

1.1.1 Norme 802.11

La norme 802.11 définit un format de données et un processus d'échange des données au niveau de la couche physique et de la couche liaison dans les réseaux sans fils. Cette norme spécifie, notamment, les ondes hertziennes comme support de transport de données. Plusieurs amendements ont été proposés à la norme 802.11 originale. Un de ces amendements est l'amendement IEEE 802.11b-1999 (IEEE, 2000) ou, plus simplement, 802.11b, qui définit l'utilisation des ondes hertziennes de fréquence de 2.4 GHz pour la transmission des données. La capacité théorique de transmission de ces réseaux est de 11 Mbps, mais en réalité, cette capacité est moindre à cause des différents processus de synchronisation d'envoi des paquets de données. Les paquets de contrôle sont transférés à un débit beaucoup plus bas, soit de 1 Mbps.

1.1.2 Modes de fonctionnement

Deux types de modes de fonctionnement sont définis par la norme 802.11. Le mode ad hoc définit un fonctionnement où chaque membre communique directement avec les autres sans passer

par une autorité centrale. Le deuxième mode de fonctionnement est le mode infrastructure. Un point d'accès, généralement unique, est défini et les membres du réseau doivent s'associer à ce dernier pour communiquer avec les autres membres. Ce point d'accès agit comme l'autorité par laquelle toutes les communications doivent passer. Il n'y a donc pas de communication directe entre les membres du réseau même si tous partagent le même support. En pratique, ce point d'accès est ensuite lié au monde extérieur (par exemple un réseau local ou Internet).

Dans les deux modes de fonctionnement, les communications entre les éléments du réseau doivent être synchronisés, car le support est partagé par tous. L'amendement 802.11b fournit des mécanismes qui essaient de diminuer la probabilité de collision lorsque plusieurs paquets de données doivent être transmis simultanément par plusieurs participants. Il n'est donc pas possible pour plus d'un membre de communiquer avec succès exactement au même moment. Si deux signaux radio sont reçus simultanément avec la même puissance par un récepteur, ce dernier aura beaucoup de difficulté à distinguer les deux signaux. Les données seront corrompues. Elles seront alors rejetées et devront être retransmisses.

1.1.3 Processus d'échange des données

Pour éviter une corruption des données lorsque deux noeuds veulent communiquer simultanément, deux mécanismes distincts ont été mis en place. L'un ou l'autre est activé selon la taille des paquets de données à transmettre. Un de ces mécanismes est utilisé pour les paquets de petite taille tandis que l'autre est utilisé pour les paquets de grande taille. La classification est établie, au niveau du réseau, sur la base d'un seuil appelé *RTSThreshold*. Deux classes de paquets sont définies, soient les petits paquets dont la taille est plus petite que ce seuil et les grands paquets.

Soit un scénario avec trois noeuds, tel qu'illustré à la figure 1.1. La topologie est définie avec un point d'accès P au centre. Deux noeuds A et B placés de part et d'autre de ce dernier. La

puissance d'émission de A n'est pas assez élevée pour joindre B et vice-versa. Par contre, P peut communiquer simultanément avec A et B. Puisque des ondes hertziennes ici sont utilisées, une puissance d'émission insuffisante limite la portée de la transmission. Cependant, il n'est pas toujours possible d'augmenter la puissance des ondes radio pour atteindre une portée plus grande.

La norme 802.11 utilise un mécanisme de type *Carrier Sense Multiple Access with Collision Avoidance* (*CSMA/CA*). Avant l'envoi d'un paquet quelconque sur le support, un noeud écoute d'abord le support pour vérifier s'il est libre. À la figure 1.1, si A et B veulent transmettre un paquet, alors ils écouteront d'abord le support. N'entendant pas ce que l'autre transmet, ils détecteront à tort que le support est libre et transmettront leur paquet simultanément. A transmet le paquet P_A en même temps que B transmet P_B. Cette transmission simultanée produit une corruption des données reçues au niveau de P. Or, un point d'accès, tel que P, dans un réseau en mode infrastructure a toujours la possibilité d'entendre des paquets provenant de tous les membres du réseau et d'en transmettre à tous. Ainsi, un processus de coordination des transmissions impliquant le point d'accès permet de régler en partie le problème des noeuds cachés.

Si deux noeuds transmettent des paquets de petite taille simultanément, alors la probabilité de collision est plus petite qu'avec des paquets de grande taille. Ainsi, pour réduire les risques de collision de paquets de grande taille transmis simultanément, un mécanisme de synchronisation plus lourd que celui prévu pour la transmission éventuelle simultanée de paquets de petite taille est mis en place.

1.1.3.1 Transmission des paquets de petite taille

À un débit de transmission égal, un paquet de petite taille transmis accaparera moins longtemps le support qu'un paquet de grande taille. Ainsi, les risques de collisions associés à la transmis-

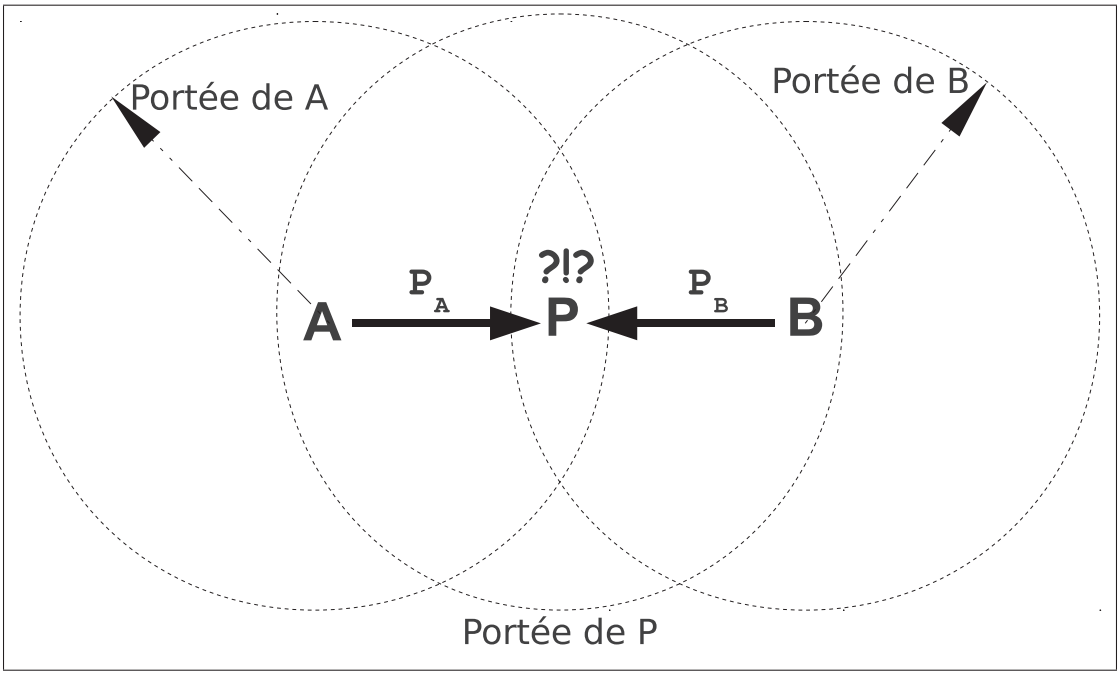

Figure 1.1 Problèmes des noeuds cachés

sion d'un paquet de petite taille sont moins grands que ceux associés à la transmission d'un paquet de grande taille. De plus, le mécanisme de synchronisation pour la transmission des paquets de grande taille est coûteux en temps de préparation. La norme 802.11 définit l'envoi des paquets de petite taille en utilisant un mécanisme de synchronisation réduit. Après un certain temps d'attente aléatoire, si l'émetteur ne détecte pas d'activité sur le support, alors il transmettra directement sans synchronisation un paquet de petite taille. L'émetteur attendra ensuite un paquet d'acquittement des données appelé *ACK* provenant du destinataire. Si cet acquittement ne parvient pas à l'émetteur après un certain temps d'attente, alors le paquet de données est retransmis jusqu'à ce qu'il soit transmis avec succès. Le nombre de retransmissions possibles est défini par le seuil *ShortRetryLimit*. Le paquet est détruit si le nombre de retransmissions dépasse ce seuil.

1.1.3.2 Transmission des paquets de grande taille

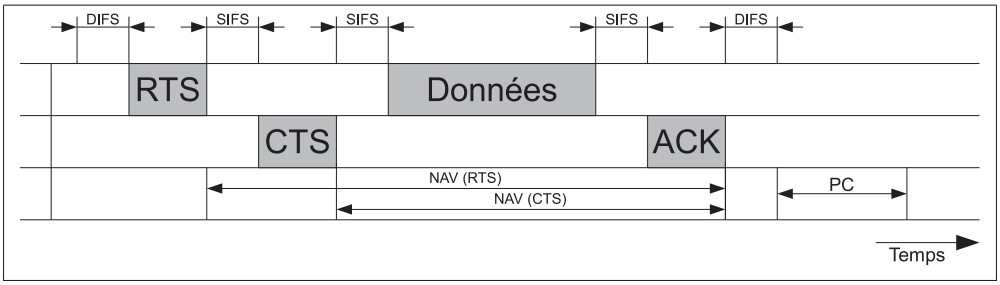

Figure 1.2 Séquence des paquets de contrôle pour les grands paquets de données
Adaptée de IEEE (1999)

Lorsqu'il y a transmission d'un paquet de grande taille, les risques de collisions sont plus grands. La norme 802.11 définit, alors, un mécanisme de synchronisation empêchant l'envoi simultané de données par au moins deux noeuds d'un même réseau. Ce mécanisme est mis en place pour régler, entre autres choses, le problème des noeuds cachés. Avant l'envoi du paquet de données, des paquets de contrôle sont d'abord envoyés pour réserver le support de façon exclusive à la transmission du paquet de données. Lorsqu'un noeud veut envoyer un paquet de grande taille, il attend d'abord une période de temps dite période de retrait, puis écoute le support. Si le support est libre, il envoie un paquet de contrôle de type *Request To Send* (*RTS*) au point d'accès. Grâce à un paquet de contrôle *Clear To Send* (*CTS*), le point d'accès fera savoir au noeud émetteur s'il peut transmettre. Le paquet *RTS* inclut un champ contenant la durée de transmission requise appelé *Duration*. Cette durée correspond au temps de transmission de la séquence complète des paquets de contrôle et de données jusqu'à l'acquittement final (ACK). Lorsqu'un paquet de type *RTS* est reçu par un noeud collaboratif, respectant la norme, autre que le destinataire, ce noeud met son interface radio en veille. Il cesse d'écouter et ne transmet plus sur le support pour la durée prescrite dans le champ *Duration*. Il met à jour son *Network Allocation Vector* (*NAV*) avec cette valeur et attendra cette durée avant de réutiliser son interface radio. Le paquet *CTS* contient également un champ *Duration*. Sa valeur est égale

à la valeur du champ *RTS* précédemment envoyé moins la durée de transmission d'un *CTS* et de la durée d'un *Short Interframe Space* (*SIFS*). Dans le cas d'un noeud caché n'ayant pas reçu le paquet *RTS*, il recevra, durant la période indiquée dans ce paquet, le paquet *CTS* et mettra à jour à son tour son *NAV*. À la fin, si la séquence d'envoi des paquets, *RTS*, *CTS*, Données, *ACK*, prévu par la norme et illustrée à la figure 1.2 ne se termine pas effectivement par un *ACK*, alors toute la séquence est reprise du début. En cas d'erreur, le nombre de reprises avant d'abandonner l'envoi du paquet correspond au seuil défini par *LongRetryLimit*. Le paquet de données est détruit si ce seuil est dépassé.

1.1.4 Période de retrait et fenêtre de contention

Lorsqu'un paquet de données est prêt à être transmis, un noeud attend, au préalable, une période de temps aléatoire. Ce temps d'attente correspond à une période de retrait (*backoff*) où le noeud ne peut transmettre aucune donnée. Cette période d'attente cherche à réduire les risques de collision avec un paquet transmis en même temps par un autre noeud. Pour calculer cette période, un choix aléatoire est fait dans un intervalle discret entre 0 et la taille de la fenêtre de contention (*contention window* ou *CW*). La valeur de *CW* est une valeur discrète, situé entre une valeur minimale (*CWMin*) et une valeur maximale (*CWMax*). La valeur de *CW* est le nombre d'intervalles qu'attend le noeud pendant sa période de contention. Le temps d'attente réel, pour la période de contention, correspond au choix de *CW* multiplié par la constante *SlotTime*. Dans l'amendement 802.11b, la valeur de cette constante est égale à $20\mu s$.

La période de retrait est initialisée aussitôt que le noeud a un paquet de données à transmettre. Cette période d'attente se comporte comme un chronomètre. Ce chronomètre est arrêté pendant les périodes où le support est occupé puis redémarré lorsque le support redevient libre. Lorsque le temps de la période de retrait est écoulée, que le chronomètre atteint zéro, le noeud attend un temps fixe correspondant au *Distributed coordination function InterFrame Space* (*DIFS*). À la fin de cette période, le paquet de données de petite taille ou un paquet *RTS* est transmis

si un paquet de grande taille doit être transmis. S'il y a une collision, une nouvelle période de retrait sera attendue. La valeur de *CW* sera doublée jusqu'à concurrence de *CWMax* avant de répéter le processus jusqu'à l'envoi fructueux du paquet de données. À la fin, la valeur de *CW* sera remise à sa valeur initiale, soit *CWMin*, dans le but de recommencer le processus d'envoi d'un nouveau paquet de données.

Les collisions peuvent être détectées de différentes façons. Si les données contenues dans un paquet sont corrompues, il est possible que cela ait été causé par une collision. Indirectement, une collision peut être détectée lorsque, à la suite de l'envoi d'un paquet *RTS*, il n'y a pas la réception d'un paquet *CTS*. Un noeud suppose qu'il ne recevra pas un paquet *CTS* à la suite d'un temps d'expiration démarré à la suite de l'envoi de son paquet *RTS*. De la même façon, une collision a pu survenir si un paquet *ACK* n'est pas reçu à la suite de l'envoi d'un paquet de données.

1.1.5 Contexte actuel

À moins d'avis contraire, toutes les simulations incorporent les paramètres suivants. Tous les éléments dans la simulation du réseau implémentent en partie, ou en totalité, la norme 802.11 de l'IEEE. Plus particulièrement, les paramètres définis pour l'amendement 802.11b sont utilisés pour simuler la norme 802.11.

Le mode infrastructure est simulé. Il y a donc un point d'accès unique et plusieurs noeuds clients ainsi associés à ce point d'accès. Les noeuds sont tous à égale distance d du point d'accès. Ils sont tous distribués uniformément sur le cercle de rayon d. La figure 1.3 illustre un exemple de topologie avec 4 noeuds (N1, N2, N3 et N4) associés à un point d'accès (PA). La distance d est fixée dans tous les cas à 50 mètres.

Les communications entre les noeuds sont unidirectionnelles : les noeuds génèrent des paquets de données qui sont transmis au point d'accès. Les paquets de données sont générés avec un

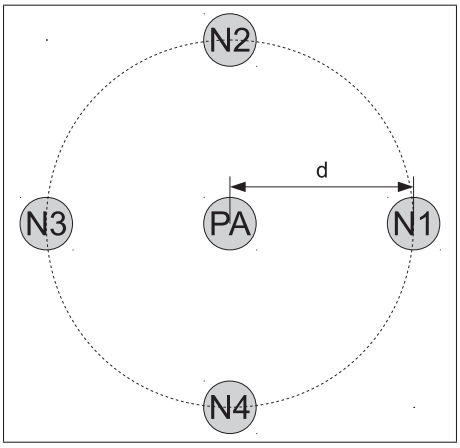

Figure 1.3 Topologie avec 4 noeuds et un point d'accès

débit constant et sont de type *User Datagram Protocol* (*UDP*). À la suite de l'envoi d'un paquet *UDP*, il n'y a donc pas d'acquittement de niveau 3 attendu après chaque paquet de données transmis, contrairement à des paquets de type *Transmission Control Protocol* (*TCP*). À noter que l'envoi d'un paquet *UDP* provoquera par la suite un acquittement de niveau 2. Ce paquet de type *ACK* provient de l'implantation de la norme 802.11.

Les paquets générés ne sont pas fragmentés lorsqu'ils seront transmis. La taille des files d'attente des noeuds a été réduite au minimum fonctionnel, soit un seul paquet.

Les algorithmes de routage, pour les réseaux sans fil, ont été désactivés, car ces algorithmes génèrent sporadiquement des paquets de données.

Dans ce contexte, si chacun de ces noeuds respectent intégralement la norme 802.11 et si chacun génère des données avec les mêmes paramètres, le point d'accès devrait mesurer des débits de données reçus très semblables pour chacun des noeuds. Ceci indique que chacun obtient une part égale de la bande passante totale.

1.2 Noeud égoïstes

Normalement, dans un réseau basé sur la norme 802.11, chacun des noeuds implémente et respecte les spécifications de la norme 802.11. Un tel noeud est dit collaboratif. Tous les noeuds collaboratifs ont un comportement semblable et les performances mesurées selon différents paramètres pour chacun de ces noeuds devraient être semblables.

Par contre, c'est la responsabilité de chaque noeud d'implanter correctement la norme. Même s'il s'agit d'une recommandation, un noeud pourrait décider de modifier ses paramètres par défaut dictés par la norme. Il pourrait également décider de ne pas respecter, en partie ou en totalité, l'implantation dictée par la description de la norme. Un noeud égoïste est un noeud qui modifie d'une quelconque façon l'implantation de la norme dans le but de tirer un avantage par rapport aux noeuds collaboratifs.

Le non-respect de la norme par un noeud égoïste pourrait lui permettre de transmettre plus de données que les autres noeuds collaboratifs. Si dans un réseau saturé tous les noeuds sont collaboratifs et que tous génèrent les données à transmettre au même débit, alors chaque noeud transmettra à des débits égaux. Ce n'est pas le cas lorsqu'il y a présence d'un noeud égoïste dans le réseau. Parce que la somme des débits de données transmis par les noeuds est limitée par la capacité du réseau, un noeud s'octroyant une part plus grande du débit le fera nécessairement au détriment des autres noeuds collaboratifs du réseau. Le noeud égoïste volera alors une partie de la capacité de transmission aux autres noeuds. Dans des cas extrêmes, ce vol peut même amener certains noeuds collaboratifs à ne plus être capables de transmettre dans le réseau.

1.2.1 Saturation

Un lien dans un réseau est dit saturé lorsque sa charge offerte atteint une valeur près ou égale à sa capacité (Jardosh *et al.*, 2005). Dans le cas présent, un réseau est saturé lorsqu'il n'est plus possible pour un de ses noeuds d'augmenter son débit de transmission de données sans que ceci

ne provoque la destruction de paquets supplémentaires. Si le nombre de retransmission dépasse le seuil permis, le paquet est détruit. Le temps d'inoccupation du support étant très court ou pratiquement nul dans un réseau saturé, l'ajout d'un nouveau paquet à transmettre ne peut que provoquer son éventuelle destruction. Un débit de données généré à la source plus petit que son débit de données effectivement transmis est un bon indice d'un noeud saturé. Également, la file d'attente du noeud rejette des paquets de données lorsque sa capacité est atteinte. Dans le cas actuellement étudié, ces deux indices sont vérifiés et sont suffisants.

Il peut y avoir différents niveaux de saturation. Le niveau de saturation pour un noeud peut être mesuré selon différentes méthodes. Il est possible de mesurer la différence entre le débit de données généré à l'intérieur d'un noeud et le débit effectivement transmis. Une manière plus simple de mesurer le niveau de saturation est de compter le nombre de paquets rejetés par sa file d'attente pendant une certaine période de temps. Soit deux réseaux avec des nombres différents de noeuds ayant la même configuration pour chacun des réseaux. Chacun des noeuds génèrent des paquets avec les mêmes paramètres de transmission. Si les deux réseaux ont le même nombre total de paquets perdus pour une période identique de temps, alors il est établi que ces réseaux sont saturés au même niveau.

Le niveau de saturation est un facteur important, car le comportement des algorithmes présentés peut varier selon le niveau de saturation des noeuds du réseau. Les tableaux 1.1 et 1.2 démontrent le niveau de saturation pour des réseaux de 5 et 20 noeuds. La taille des paquets est fixée à 1000 octets et la valeur *RTSThreshold* est fixée à 500 octets, tous les paquets transmis sont alors de grande taille. Une comparaison des deux tableaux montre des niveaux de saturation pratiquement identiques avec des débits de données générés et mesurés très différents.

Tableau 1.1 Comparaison des débits générés, mesurés et le nombre de paquets perdus avec 5 noeuds

Débit généré (bps)	**Débit mesuré (bps)**	**Paquets perdus (par 100 s.)**
840k	836k	49
852k	844k	96
868k	852k	196
896k	856k	502
938k	858k	992
1264k	860k	5000

Tableau 1.2 Comparaison des débits générés, mesurés et le nombre de paquets perdus avec 20 noeuds

Débit généré (bps)	**Débit mesuré (bps)**	**Paquets perdus (par 100 s.)**
207k	203k	55
212k	204k	101
221k	205k	198
246k	206k	500
287k	207k	997
611k	206k	5002

1.2.2 Équité de la norme 802.11

Soit un réseau en situation de saturation dont la topologie est illustré à la figure 1.3. Les noeuds 1 à 4 génèrent des paquets de données qui sont destinés au point d'accès et sont paramétrés identiquement. Les débits moyens de données, mesurés au niveau du point d'accès, sur des périodes de temps assez longues, devraient être très semblables pour les quatre noeuds. Ce fait n'est pas valable si les débits sont mesurées sur de courtes périodes de temps. Les réseaux basés sur la norme 802.11 engendrent une répartition inéquitable du trafic entre les noeuds sur des périodes brèves de temps (Koksal *et al.*, 2000). Ce fait est important à noter, car l'éventuelle

détection d'un noeud considéré égoïste, qui en réalité respecte parfaitement la norme, est à éviter. Par exemple, mesuré sur une courte période, le débit mesuré d'un noeud peut être le double par rapport aux autres noeuds d'un même réseau. Une précaution est prise pour éviter de biaiser les observations. C'est ainsi que les différents paramètres d'un noeud ou du réseau sont toujours mesurés sur des périodes de temps suffisamment longues pour amoindrir ce phénomène d'iniquité.

1.2.3 Génération d'un noeud égoïste

Pour qu'un noeud ait une plus grande part de la bande passante que les autres noeuds dans une situation de saturation, il doit modifier son implémentation de la norme 802.11 d'une quelconque façon. Il peut le faire en modifiant les paramètres déterminant sa fenêtre de contention, notamment en affectant des valeurs de *CWMin* et *CWMax* plus petites que celles définies par l'amendement 802.11b. Dans cette situation, le noeud égoïste attend en moyenne moins longtemps que les autres noeuds avant d'entamer le processus d'envoi de paquets de données. De ce fait, le noeud égoïste réussit à transmettre plus rapidement ses paquets que les autres noeuds et obtient ainsi une plus grande part de la bande passante. Il reste alors moins de bande passante pour l'ensemble des autres noeuds. De cette façon, le noeud égoïste réussit à voler de la bande passante aux noeuds restés collaboratifs.

1.2.4 Effets des noeuds égoïstes dans un réseau

La présence de noeuds égoïstes dans un réseau peut avoir des conséquences nuisibles pour les noeuds collaboratifs. Pour s'en convaincre, une simulation a été effectuée. La figure 1.4 illustre ce phénomène avec 4 noeuds devenant égoïstes à tour de rôle. Au début de la simulation, tous les noeuds sont collaboratifs et, à chaque période de 200 secondes, un noeud passe de l'état collaboratif à l'état égoïste. À la fin de la simulation, tous les noeuds sont égoïstes. Le débit de génération des données est de 2400kbps pour chacun des noeuds.

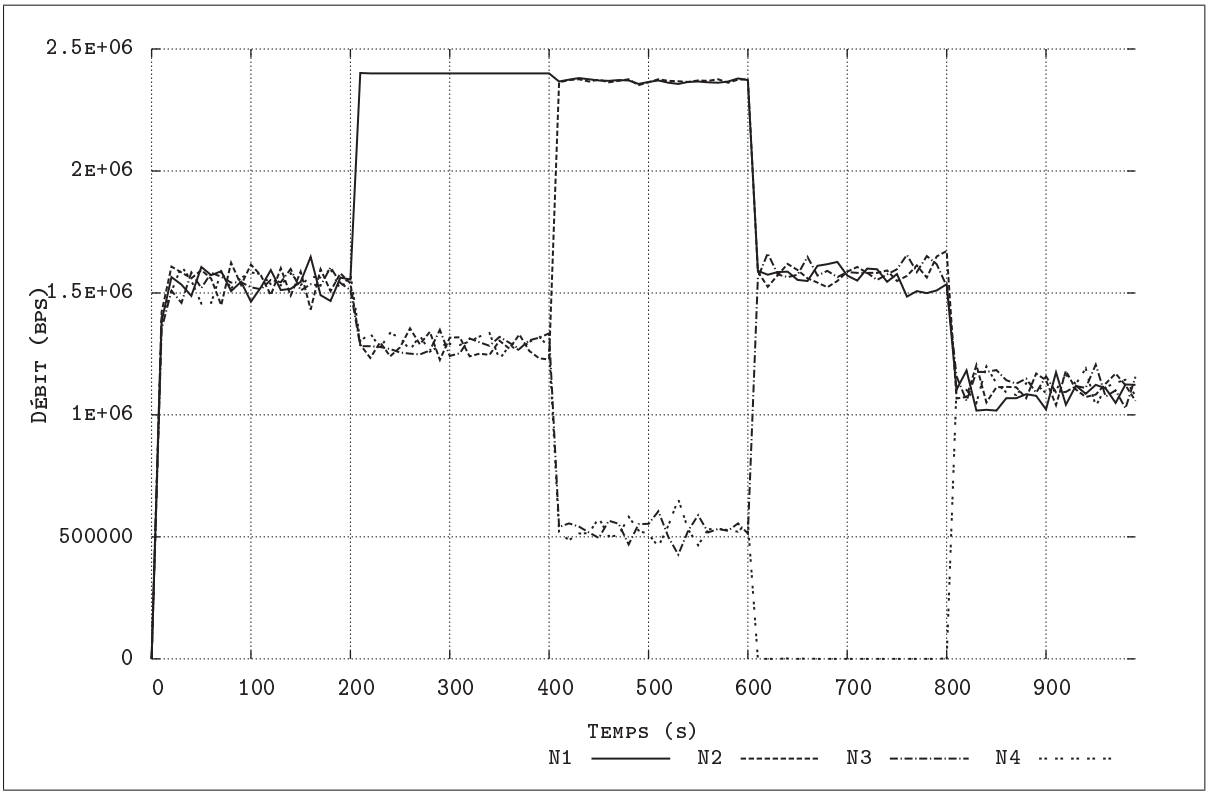

Figure 1.4 Impact des noeuds égoïstes sur les débits mesurés

Le tableau 1.3 décrit les débits de données moyens, mesurés pour chacun des noeuds, à différentes périodes de la simulation. Au début de la simulation, une saturation du réseau est notée, car les débits moyens mesurés lors de la première période entre 10s et 200s sont de beaucoup inférieurs aux débits de génération des données.

À 200s du début de la simulation, le noeud 1 (N1) passe de l'état collaboratif à l'état égoïste. Ses paramètres pour déterminer sa fenêtre de contention *CWMin* et *CWMax* passent des valeurs par défaut de 31 à 2 pour *CWMin* et de 1023 à 2 pour *CWMax*. Ces nouvelles valeurs sont les minima fonctionnels dans le cadre des simulations avec *ns-2*. Dans cette situation, le débit mesuré pour le noeud 1 passe de 1541kbps pour la période entre 10s et 200s à 2400kbps pour la période suivante où le noeud devient égoïste. Ce débit correspond exactement au débit de

génération de ses données, le noeud égoïste peut donc transmettre ses données sans aucune contrainte. Il y aura par contre une diminution des débits des trois autres noeuds pour la même période. L'augmentation du débit chez le noeud égoïste s'est faite au détriment des autres noeuds qui sont restés collaboratifs.

À 400s du début de la simulation, c'est au tour du noeud 2 (N2) de devenir égoïste. Le même phénomène est remarqué.

À 600s du début de la simulation, c'est au tour du troisième noeud (N3) de devenir égoïste. Dans cette situation, le noeud 4 (N4) est le perdant, car la présence de trois noeuds égoïstes dans le réseau ne lui permet plus de transmettre ses données.

La dernière période, où chaque noeud est égoïste, est caractérisée par le fait que chacun des noeuds a des débits de données mesuré semblable.

Les débits mesurés lors de la période située entre 10s et 200s où tous les noeuds sont collaboratifs sont beaucoup plus grands que les débits mesurés lors de la dernière période comprise entre 810s et 1000s, où tous les noeuds sont égoïstes. Ces faits conduisent à la conclusion suivante : il n'y a pas d'avantage pour l'ensemble du système à ce que tous les noeuds deviennent égoïstes en même temps. Ces faits conduisent à la conclusion suivante : un ou deux noeuds qui deviennent égoïstes en tire un gain énorme pour eux-même, au détriment des autres. En prenant la moyenne des débits mesurés pendant toute la durée de la simulation, soit entre 10s et 1000s, les noeuds qui sont restés égoïstes plus longtemps ont obtenu des débits mesurés plus grands que les noeuds qui sont restés collaboratifs plus longtemps.

1.2.5 Faux positifs et faux négatifs

Les algorithmes de détection des noeuds égoïstes n'étant pas parfaits, des erreurs de détection peuvent survenir de temps à autre. En fait, deux types d'erreurs peuvent survenir : des erreurs

Tableau 1.3 Débits moyens mesurés en bps par période avec des noeuds égoïstes

	Noeud 1	**Noeud 2**	**Noeud 3**	**Noeud 4**
Débit moyen 10-200s	1541k	1554k	1541k	1548k
Débit moyen 210-400s	2400k	1277k	1286k	1295k
Débit moyen 410-600s	2368k	2369k	532k	529k
Débit moyen 610-800s	1564k	1582k	1592k	0k
Débit moyen 810-1000s	1085k	1109k	1126k	1116k
Débit moyen 10-1000s	1802k	1583k	1213k	889k

de type faux positif ou des erreurs de type faux négatif. Un faux positif est un noeud qui est détecté à tort comme égoïste alors que, dans la réalité, il respecte le protocole. Un faux négatif est un noeud qui est détecté comme collaboratif, mais en réalité, il est un noeud égoïste. Les algorithmes présentés dans le cadre de ce mémoire ainsi que les différentes méthodes présentées dans le chapitre 2 tiennent généralement comptent de ces types d'erreurs.

1.3 Détection d'un noeud égoïste

Lorsqu'un noeud ne transmet pas ou n'est pas en veille, il lui est possible d'écouter les paquets de données ou de contrôle qui sont échangés sur le réseau. Dans ce cas, un noeud a la capacité d'écouter tout ce que le point d'accès transmet. À cause du problème des noeuds cachés, un noeud ne peut pas toujours entendre la totalité des communications transmise par les autres noeuds.

Dans le cas de la transmission d'un paquet de données de grande taille, un noeud qui est caché par rapport à un autre ne peut entendre que les paquets provenant du point d'accès. Les paquets de types *CTS* ou *ACK* sont entendus dans toutes les situations parce que transmis par le point d'accès. Par contre, il est possible que le paquet de type *RTS* et le paquet contenant les données ne soient pas toujours entendus.

Dans le cas de la transmission d'un paquet de données de petite taille, seul le paquet *ACK* est toujours entendu. Ainsi, la réception d'un paquet *ACK* permet de déduire qu'un paquet de données a été envoyé. L'en-tête d'un paquet *ACK*, comme normalement tous les types de paquets envoyés dans un réseau basé sur la norme 802.11, contient l'information sur l'émetteur du paquet ainsi que sur le destinataire. Un noeud voulant déterminer si un autre noeud, même caché, a envoyé un paquet de données n'a qu'à écouter les paquets *ACK* provenant du point d'accès. À la réception du paquet *ACK*, l'analyse de son en-tête permet à un noeud d'identifier l'émetteur du paquet de données envoyé précédemment et de récupérer l'information sur le destinataire.

Comme expliqué plus en détail dans le chapitre 3, pour vérifier, ensuite, si un noeud a envoyé plus souvent des paquets de données que les autres noeuds, chaque noeud maintient des compteurs de paquets *ACK* pour chacun des noeuds. En associant un compteur à chaque noeud connu du réseau et après une certaine période de temps ou un certain nombre de paquets totaux reçus, ces différents compteurs sont comparés les uns par rapport aux autres. Si un compteur se distingue des autres, alors, dans le contexte actuel, ceci peut indiquer que le noeud auquel le compteur est associé est égoïste.

1.4 Action contre les noeuds égoïstes

Lorsqu'il y a présence d'un noeud égoïste dans un réseau, son influence est généralement négative sur les autres noeuds. Le vol d'une partie de la bande passante, normalement attribuée aux noeuds collaboratifs, est généralement le motif, pour un noeud, de devenir égoïste. Dans le but de convaincre le noeud égoïste d'arrêter ces vols, des actions peuvent être entreprises contre le noeud égoïste. Ces actions visent à le forcer à arrêter le vol de la bande passante. Si le noeud égoïste n'est pas complètement refermé sur lui-même et est rationnel, prendre des actions contre lui peut le forcer à ne plus dévier de la norme. Si le noeud égoïste se rend compte

que le fait de modifier son implantation de la norme ne lui apporte aucun avantage marqué, ceci peut réussir à le convaincre à respecter la norme.

Les actions contre le noeud égoïste doivent être réfléchies. Elles ne doivent être accomplies que lorsque la présence d'un noeud égoïste est très probable. Elles ne doivent pas, non plus, être automatiques ou être posées dans le seul but de prévenir la présence des noeuds égoïstes. Une action posée contre un noeud égoïste doit entraîner le minimum de dommages collatéraux : elle ne doit pas trop nuire aux noeuds collaboratifs et doit avoir un impact négatif minimal sur le bon fonctionnement du réseau.

1.4.1 Brouillage de paquets

Une des façons de nuire au noeud égoïste est de systématiquement brouiller les paquets de données que le noeud égoïste envoie. Il faut faire attention, par contre, de ne brouiller que les paquets provenant du noeud égoïste et non pas les paquets provenant des autres noeuds. Autrement, l'action pourrait s'apparenter à une attaque de déni de service. À noter qu'il suffit de ne brouiller qu'une faible proportion des paquets afin qu'un réseau ne devienne pas complètement inutilisable (Zhou *et al.*, 2008), d'où le choix d'un brouillage très sélectif.

Lorsqu'un noeud collaboratif détecte un noeud égoïste, il brouille les paquets de données provenant du noeud égoïste en envoyant simultanément un paquet contenant de fausses données. Les signaux radio générés par les deux noeuds sont reçus simultanément au niveau du récepteur. La confusion au niveau du récepteur provoque la corruption des données reçues et dans le cas présent, c'est le résultat recherché. La conséquence est que le récepteur incorrectement reçoit les paquets de données provenant du noeud égoïste et, donc, ne les acquittent jamais. Le noeud égoïste, attendant un paquet *ACK* et ne le recevant jamais, va envoyer de nouveau son paquet de données et le processus recommence.

Dans le cas d'un paquet de données de grande taille, la valeur par défaut de la constante *LongRetryLimit*, déterminant le nombre de fois qu'un paquet de données est renvoyé, est fixée à 4. Le noeud égoïste renvoie alors quatre fois plus de paquets si tous ses paquets envoyés sont systématiquement bloqués. Le résultat final est que tous sont perdants. Le noeud égoïste transmet sans succès ses paquets alors que les autres noeuds verront leurs bandes passantes réduites de façon considérable.

1.4.2 Réduction des valeurs déterminant la fenêtre de contention

Sachant qu'un noeud est égoïste parce qu'il a réduit ses valeurs *CWMin* et *CWMax*, un noeud collaboratif peut imiter le noeud égoïste en réduisant à son tour ces deux valeurs à un niveau égal. Si chaque noeud collaboratif imite le noeud égoïste en utilisant des valeurs plus petites pour déterminer sa fenêtre de contention, alors chaque noeud transmet ses paquets de données avec des délais en moyenne semblable. Les différents noeuds se partageront alors une part égale de la bande passante totale. Ce processus revient à transformer temporairement les noeuds collaboratifs en noeuds égoïstes. Tel qu'expliqué à la section 1.2.4, si tous les noeuds devenaient égoïstes, alors tous seraient perdants. Cette solution doit donc être temporaire.

1.5 Hypothèses

Dans le cadre du présent travail, il est supposé que les noeuds égoïstes soient rationnels. Lorsqu'un noeud rationnel s'aperçoit que l'égoïsme de son comportement ne lui apporte aucun bénéfice, alors il sera tenté de redevenir collaboratif. En fait, les bénéfices que lui auraient apporté son comportement égoïste sont annihilés par la réaction des autres noeuds, ces autres noeuds implantant un algorithme de détection des comportements égoïstes. Le noeud égoïste étant rationnel, la seule façon de regagner la bande passante perdue à cause de son égoïsme est d'adopter le même comportement que les noeuds collaboratifs.

Également, les algorithmes présentés dans ce travail supposent la présence d'un seul noeud égoïste à la fois dans les réseaux. Si un réseau contient plus d'un noeud égoïste, alors le bon fonctionnement des algorithmes n'est pas garanti à moins d'une adaptation des algorithmes présentés.

1.6 Objectifs

En se basant sur la norme 802.11, sur les spécifications des noeuds égoïstes et sur le contexte actuel, un des objectifs de ce travail est de détecter les noeuds égoïstes parmi un ensemble de noeuds collaboratifs. Indépendamment des différents niveaux de saturation et des différentes configurations de réseau, ces détections doivent être faites de façon fiables. Notamment, la définition d'une méthode générique de détection des noeuds égoïstes permet d'appliquer une seule méthode de détection à toutes les situations.

Ensuite, trouver les meilleures statégies pour réagir contre la présence de noeuds égoïstes est l'objectif principal de ce travail. Deux stratégies principales sont proposées : le brouillage des paquets ou la transformation temporaire des noeuds collaboratifs en noeuds égoïstes. De ces deux stratégies, plusieurs variantes sont proposées. L'identification des variantes les plus performantes selon les différents cas d'utilisation est l'objectif à atteindre.

CHAPITRE 2

TRAVAUX ANTÉRIEURS

2.1 Solutions appliquées au point d'accès

Cette section regroupe certaines solutions qui peuvent être appliquées au niveau du point d'accès dans un réseau en mode infrastructure. À noter que dans certains cas, ces solutions peuvent être appliquées moyennant parfois certaines modifications, au niveau des noeuds d'un réseau ad hoc.

2.1.1 Système DOMINO

Le système DOMINO (Raya *et al.*, 2006) permet la détection de noeuds égoïstes à partir de l'observation des différentes modifications qu'un noeud égoïste peut apporter à la norme 802.11 originale. Selon les différents scénarios établis et en se basant sur diverses statistiques, il est possible de détecter les noeuds égoïstes.

DOMINO a été conçu pour être déployé au point d'accès. Par exemple, en supposant une communication unidirectionnelle entre un noeud et le point d'accès, le point d'accès peut compter le nombre de retransmissions d'un même paquet. Si un noeud retransmet un paquet reçu correctement, cela signifie qu'il n'a pas reçu le paquet *ACK*, suivant normalement la réception d'un paquet de données de petite taille, ou le paquet *CTS*, suivant la réception d'un paquet *RTS* lorsqu'un paquet de grande taille doit être envoyé. Il est possible que ces paquets aient été brouillés par un noeud égoïste, de façon volontaire, dans le but de faire augmenter la période de contention d'un noeud. Une augmentation s'effectue à chaque échec de retransmission. De ce fait, les noeuds dont les paquets ont été brouillés sont défavorisés lors de leurs transmissions.

Le système peut également vérifier si un noeud utilise la valeur de *DIFS* par défaut après chaque envoi de paquet *ACK*. Si un noeud transmet un paquet quelconque avant tous les autres noeuds, il se réservera le support avant tous les autres noeuds et gagnera un avantage.

Le système DOMINO peut comparer la valeur incluse dans le champ *Duration* et le temps réel d'occupation du support. Si un noeud inscrit dans ce champ une valeur beaucoup plus grande que le temps utilisé dans la réalité, alors il y aura du temps perdu qui ne sera pas disponible pour les autres noeuds. De la même façon que dans le cas d'un *DIFS* trop court, un noeud égoïste ayant des périodes de contention trop courtes aura un avantage sur les autres noeuds.

Après avoir recueilli différentes statistiques pour un noeud donné pendant une période d'observation, des tests sont effectués sur chaque statistique pour vérifier si les valeurs mesurées respectent ou non la norme. Le module effectuant les tests sur chaque statistique est divisé en deux parties : le *Deviation Estimation Component* (*DEC*) et le *Anomaly Detection Component* (*ADC*). Le *DEC* détermine l'écart entre la statistique observée et la valeur attendue. Le *ADC* détermine si le noeud respecte la norme en fonction de l'écart observé pour une statistique donnée.

Chaque couple de *DEC* et de *ADC* teste une seule statistique. Les résultats des différents tests sont recueillis par le *Decision Making Component* (*DMC*). Le *DMC*, détermine à la fin de la période d'observation, si le noeud est égoïste ou non. Si un noeud égoïste est détecté, le système DOMINO peut informer l'opérateur du réseau, car aucune manoeuvre pour contrecarrer un noeud égoïste n'est implantée.

2.1.2 Distribution statistique des valeurs de *CW* choisies

Serrano *et al.* (2010) évaluent statistiquement si un noeud choisit, de façon distribuée uniformément, ses valeurs de *CW* pour déterminer sa fenêtre de contention. En observant les délais entre deux paquets transmis consécutivement pour un noeud donné, il est possible de retrouver

les valeurs qu'un noeud a choisies pour déterminer sa fenêtre de contention. En particulier, un noeud collaboratif devrait choisir des valeurs pour sa fenêtre de contention de façon uniforme dans l'intervalle discret $[0..CWMin[$. Si un même paquet de données est retransmis à cause d'un échec de transmission, l'intervalle devient $[0..2 \cdot CWMin[$, car la borne supérieure de l'intervalle pour choisir *CW* double à chaque échec de transmission. En connaissant le nombre de retransmissions d'un même paquet de données, il est possible de ramener tous les intervalles à $[0..CWMin[$.

En collectionnant un nombre suffisant de choix de *CW* et en tenant compte du nombre de retransmissions, il est possible d'évaluer la valeur *CWMin* que le noeud a utilisée. Si cette valeur est plus petite que la valeur par défaut (soit 32 pour l'amendement 802.11b), alors le noeud est considéré probablement égoïste, mais il pourrait ne pas l'être. C'est pourquoi qu'un facteur de fausses détections des noeuds égoïstes est défini pour éviter la détection des noeuds considérés faussement détecté comme égoïste. Incidemment, plus un noeud choisit une valeur de *CWMin* beaucoup plus petite que la valeur par défaut, plus rapidement le noeud est confirmé égoïste avec certitude.

2.1.3 Méthode *CUSUM*

Cardenas *et al.* (2009) comparent leur stratégie de détection de noeuds égoïstes avec celles de DOMINO (Raya *et al.*, 2006) et de *SPRT* (Radosavac *et al.*, 2005), la méthode *SPRT* étant présentée à la section 2.2.4. En prenant seulement la partie des tests, dans DOMINO, où les périodes de retrait sont mesurés, ils démontrent la supériorité de la stratégie *SPRT* par rapport à celle de *DOMINO*. Malgré les mauvaises performances de DOMINO mais à cause de sa simplicité par rapport à *SPRT*, ils proposent une modification à DOMINO.

Avec la stratégie de somme cumulative non paramétrique (*Nonparametric Cumulative Sum* ou *CUSUM*) appliquée à *DOMINO*, les résultats pour détecter un noeud égoïste sont pondérés

selon le degré de malice d'un noeud. Avec la stratégie DOMINO originale, quand un test détecte un comportement égoïste, alors un compteur est incrémenté une fois. À la suite d'une série de tests, si ce compteur dépasse un certain seuil, alors le noeud sera considéré comme égoïste. En appliquant la stratégie *CUSUM* à DOMINO, si un test détecte un comportement égoïste, alors ce même compteur est incrémenté d'une valeur qui variant en fonction du degré de malice du noeud. Malgré tout, la stratégie *SPRT* est meilleure que la stratégie *CUSUM* appliquée à DOMINO, car elle est optimale.

2.2 Solutions appliquées aux noeuds

Les solutions présentées dans cette section sont conçues pour être appliqués aux noeuds dans un réseau en mode ad hoc. Dans certains cas et avec certaines modifications, ces solutions peuvent être appliquées dans un réseau en mode infrastructure, notamment au point d'accès.

2.2.1 Solution avec prescription des valeurs de *CW* à utiliser

La détection des noeuds égoïstes faite par Kyasanur et Vaidya (2005) se base sur les valeurs de *CW* observées. L'application de la solution nécessite la modification de la norme 802.11. Les noeuds collaboratifs doivent d'abord détecter les noeuds égoïstes avant d'appliquer des correctifs. Ces correctifs permettent de réduire les effets néfastes des noeuds égoïstes.

En supposant l'envoi d'un paquet de données d'un émetteur à un récepteur, le récepteur choisit la valeur du prochain *CW* que l'émetteur doit appliquer. Cette nouvelle valeur de *CW* est appliquée avant l'envoi du prochain paquet de données. Cette valeur prescrite est envoyée à l'intérieur d'un paquet *CTS* ou *ACK*. Si après avoir reçu cette valeur, l'émetteur du paquet original n'applique pas la valeur prescrite, alors le noeud est considéré comme ayant mal agi. La détection s'effectue en comparant la valeur prescrite et la période de retrait observée. Dans le cas d'un noeud égoïste, cette période observée est plus courte que la valeur prescrite.

Si un noeud a une période de retrait plus courte que celle spécifiée, ceci n'indique pas nécessairement que le noeud a mal agi. À cause de la configuration du réseau ou des mauvaises conditions de communication qui peuvent survenir sporadiquement sur le support, il est possible qu'un noeud ne détecte pas l'occupation du support pour un certain temps. Si un noeud détecte l'occupation du support alors qu'un autre ne détecte pas son occupation, alors un noeud restera en attente alors que l'autre va écouler sa période de retrait. Par la suite, le noeud qui a écoulé sa période de retrait va envoyer un paquet de données plus rapidement que prévu. De son point de vue, en écoulant légitimement sa période de retrait, il est détecté par les autres noeuds comme ayant mal agi.

Lorsqu'un noeud a été détecté comme ayant mal agi, il faut attendre d'avoir atteint un pourcentage minimal de détections de cet agissement égoïste par rapport au nombre total de détections effectué sur ce noeud avant de considérer ce noeud comme réellement égoïste. Dans ce cas, pour pallier au fait que le noeud égoïste a attendu moins longtemps que sa valeur prescrite, une valeur plus grande de *CW* lui est attribuée pour le prochain envoi. Cette solution suppose que le noeud égoïste est suffisament collaboratif pour appliquer les valeurs de *CW* prescrites. Un noeud vraiment égoïste pourrait simplement ignorer les valeurs qui lui sont assignées et il pourrait toujours choisir des valeurs près du minimum fonctionnel.

2.2.2 Régression linéaire

Une méthode de détection des noeuds égoïstes par régression linéaire est proposée par Hamieh *et al.* (2009). L'algorithme original est conçu pour fonctionner dans les réseaux 802.11 en mode ad hoc, mais il peut être adapté pour être utilisé dans les réseaux en mode infrastructure. La méthode fonctionne, car elle suppose que deux noeuds communiquent ensemble avec des débits identiques et si chacun d'eux respecte la norme 802.11, alors la durée des communications pour chacun des deux noeuds devrait être semblable à celle de l'autre.

Si deux noeuds accèdent au support de transmission pendant des durées semblables, alors la corrélation entre les différentes observations des temps d'accès au support provenant des deux noeuds sera grande. Si les temps d'occupation du support observés suivent des variables aléatoires, alors deux noeuds auront des variables aléatoires avec des distributions différentes si un des deux noeuds est égoïste. De la même façon, si deux noeuds choisissent leurs valeurs de *CW* de la même façon, alors les variables aléatoires auront une forte corrélation.

Une période d'initialisation sans la présence de noeuds égoïstes doit être présente au début pour calculer un seuil initial. Ensuite, en utilisant les temps d'accès au support comme statistiques, la régression linéaire est calculée à intervalles réguliers pour chaque paire de noeuds. À chaque intervalle, les régressions calculées sont comparées au seuil initial. Si à trois reprises, la régression calculée pour un noeud en particulier dans une paire de noeuds est plus grande que le seuil, alors ce noeud est considéré comme égoïste. Par contre, ce calcul ne fonctionne pas si deux noeuds d'une même paire sont égoïstes en même temps. Dans cette situation, un noeud confirmé collaboratif doit être utilisé pour vérifier si un noeud fait partie d'une paire où les deux noeuds sont égoïstes.

2.2.3 Borne inférieure variable

Une façon de détecter les noeuds égoïstes dans un réseau 802.11 est de modifier la norme elle-même. Avec une petite modification chez les noeuds collaboratifs, il est possible de rester compatible avec la norme originale tout en ayant la possibilité de détecter facilement les noeuds égoïstes. Une telle façon est de modifier la méthode *Binary Exponential Backoff* (*BEB*) pour la transformer en méthode de *Predictable Binary Backoff* (*PRB*).

La méthode *BEB* est utilisée pour calculer la valeur de la fenêtre de contention *CW*. À chaque échec de transmission, la borne supérieure de l'intervalle est doublée. La nouvelle valeur de *CW* est choisie dans l'intervalle discret $[0..2 \cdot CWMin]$. La valeur supérieure de l'intervalle est

bornée par *CWMax*. Si la borne inférieure était toujours fixée à 0, tous les noeuds pourraient utiliser, en fin de compte, des valeurs de *CW* à 0 et les noeuds égoïstes pourraient utiliser cette valeur plus souvent que les autres noeuds. En modifiant la borne inférieure de cet intervalle, à chaque transmission réussie ou à chaque échec de transmission, pour autre chose que 0 tel que décrit par Guang *et al.* (2008), la méthode *PRB* permet de prédire plus précisément les valeurs *CW* choisies par les noeuds en restreignant l'intervalle dans lequel les choix de *CW* s'effectuent.

En estimant la valeur du *CW* qu'un noeud choisit et en comparant cette valeur aux valeurs déterminées par les intervalles provenant de la méthode *PRB*, un noeud ayant fait un choix de *CW* trop petit est détecté comme égoïste. De plus, la borne inférieure est choisie, à chaque fois, de façon à maximiser l'équité entre les noeuds du réseau, dans la transmission des données. Si un noeud égoïste qui applique la méthode *PRB* choisit toujours, comme valeur de *CW*, la plus petite valeur possible de ces nouveaux intervalles, alors il ne sera pas détecté comme égoïste. En revanche, l'avantage de ce dernier serait mince et son impact sur les autres noeuds serait réduit. L'étendue du nouvel intervalle dans lequel le choix de *CW* s'exécute est alors réduite par rapport à la norme originale. Le fait que l'intervalle ne soit que réduit fait que la méthode *PRB* respecte la norme originale. Du même fait, les noeuds égoïstes sont facilement détectés si tous les noeuds, sauf les noeuds égoïstes, appliquent la méthode *PRB*.

2.2.4 Méthode *SPRT*

Il peut arriver parfois qu'un noeud puisse donner l'impression qu'il ne respecte pas le protocole à cause de problèmes qui peuvent survenir sur le support de transmission, comme des interférences ou des signaux trop faibles. La solution présentée par Radosavac *et al.* (2005) tient compte de ces phénomènes. Elle est conçue pour les réseaux en mode ad hoc mais elle pourrait être applicable pour les réseaux en mode infrastructure. D'abord, des données sont

recueillies par des noeuds observateurs. Ces observations contiennent des données qui servent à évaluer indirectement les valeurs des périodes de retrait utilisées par les autres noeuds.

Un test du type *Sequential Probability Ratio Test* (*SPRT*) est utilisé pour amasser un nombre variable d'observations jusqu'à ce qu'une tendance claire se dessine. Cette méthode vise à confirmer l'un de ces deux hypothèses, la présence ou l'absence de noeuds égoïstes. Avec un nombre variable d'observations, une tendance plus ou moins claire peut se dégager des observations. Cette tendance est la confirmation d'une des deux hypothèses. Par exemple, un noeud très égoïste qui utilise des valeurs de *CW* toujours près du minimum fonctionnel est détecté plus rapidement qu'un noeud qui n'utilise ces types de valeurs que sporadiquement. Comme pour la plupart des algorithmes de détection de noeuds égoïstes, il peut arriver qu'il y ait une fausse détection d'un noeud égoïste ou qu'un noeud égoïste passe inaperçu. Des probabilités de fausses détections ou de détections manquées sont définies. Des paramètres définissant les conditions d'interférence et de sensibilité de l'algorithme pour détecter les noeuds égoïstes sont également définis. À la suite de la détection d'un noeud égoïste, les noeuds observateurs peuvent avertir les autres noeuds.

2.2.5 Analyse stochastique des durées entre les paquets et méthode *SPRT*

Sans calculer directement les valeurs de la fenêtre de contention utilisées avant l'envoi de chaque paquet, la mesure des durées entre deux paquets consécutifs envoyés par un noeud peut nous donner de très bons indices sur les valeurs de *CW* utilisées.

L'article de Rong *et al.* (2006) développe un modèle stochastique basé sur l'article de Bianchi (2000) décrivant la distribution théorique des durées entre deux envois de paquets consécutifs. Un degré de malice est incorporé aux calculs dans le but de simuler un noeud plus ou moins égoïste quant à ses valeurs de *CW* choisies par rapport à la norme.

Les probabilités de collisions, c'est-à-dire la probabilité que deux noeuds envoient chacun un paquet simultanément, et les probabilités de transmission, c'est-à-dire la probabilité qu'un noeud envoie un paquet à tout instant, sont décrites en théorie. Des vérifications de leurs résultats théoriques par des simulations ont été effectuées pour montrer qu'ils s'approchent de la réalité. Les résultats montrent que les noeuds égoïstes ont une plus forte probabilité que les noeuds collaboratifs d'avoir de courts délais entre l'envoi de deux paquets consécutifs. De cette façon, les noeuds égoïstes peuvent envoyer des données avec un débit plus grand que les noeuds collaboratifs. Ensuite, pour détecter les noeuds égoïstes, le test *SPRT* est utilisé.

En observant un délai entre l'envoi de deux paquets de données consécutifs provenant d'un même noeud, le test *SPRT* permet de vérifier deux hypothèses. Il peut confirmer que le noeud est égoïste, confirmer que le noeud n'est pas égoïste ou ne confirmer ni l'une ni l'autre des deux hypothèses. S'il est impossible de confirmer une des deux hypothèses, alors des observations supplémentaires sont effectuées jusqu'à ce qu'une des deux hypothèses soit confirmée. Malgré tout, il peut arriver de confirmer à tort une des deux hypothèses. Les probabilités de détections manquées et de fausses détections sont incluses dans les algorithmes présentés.

2.2.6 Test de Kolmogorov-Smirnov

Lopez Toledo et Wang (2007) utilisent le test de Kolmogorov-Smirnov pour détecter les noeuds égoïstes. Comme pour le test *SPRT*, deux hypothèses y sont testées : ou le noeud est égoïste ou le noeud respecte le protocole. Le test de Kolmogorov-Smirnov utilise, pour un noeud donné, le nombre d'intervalles libres entre deux transmissions réussies pour déduire les valeurs utilisées dans les fenêtres de contention.

Lorsqu'un noeud envoie deux paquets de données consécutivement, un certain nombre d'intervalles libres est observé entre les deux paquets. Un intervalle est un temps fini prédéterminé et égal pour tous les noeuds. Un intervalle est dit libre pendant la durée où le noeud ne transmet

rien. Il est à noter que, pendant un certain nombre d'intervalles libres d'un noeud donné, il est possible que les autres noeuds transmettent des paquets. Il est noté qu'un noeud égoïste qui a réduit ses valeurs de *CW* a normalement moins d'intervalles libres après une transmission réussie qu'un noeud respectant la norme.

D'une part, à partir des échantillons de nombre d'intervalles libres d'un noeud donné, une fonction de répartition empirique est calculée. D'autre part, pour un noeud respectant la norme, la fonction de répartition du nombre d'intervalles libres est calculée. Avec les deux fonctions de répartition, le test de Kolmogorov-Smirnov est effectué afin de déterminer si les données recueillies ont été significatives. Si les données sont significatives, alors il est possible de confirmer si un noeud est égoïste ou non. Comme dans toutes les stratégies décrites jusqu'à maintenant, il est possible d'avoir de fausses alarmes positives et de fausses alarmes négatives.

2.3 Théorie des jeux

Cette section regroupe des solutions basées sur la théorie des jeux. Puisque dans les réseaux 802.11, tous ont intérêt à collaborer, il s'agit de jeux à somme non-nulle.

2.3.1 Protocoles basés sur la méthode d'accès *CSMA/CA*

Čagalj *et al.* (2005) utilisent la théorie des jeux pour comprendre et résoudre le problème de la présence de noeuds égoïstes dans les protocoles basés sur la méthode d'accès *CSMA/CA*, tels que la norme 802.11. Ils déterminent que les noeuds peuvent tirer plus ou moins d'avantage à réduire leurs valeurs déterminant leurs fenêtres de contention de façon différente selon les stratégies utilisées. Si des noeuds décident simplement de réduire leurs valeurs de *CW* au minimun fonctionnel, il y a un grand bénéfice pour ces noeuds et des pertes pour les autres noeuds. Il s'agit d'un équilibre de Nash, où seuls les noeuds égoïstes sont gagnants. En revanche, il est préférable que les noeuds décident de changer leurs valeurs de *CW* non pas au minimum fonc-

tionnel mais à des valeurs où tous pourraient tirer un gain. Il s'agit d'atteindre l'optimum de Pareto pour les débits de données tranférés.

Pour convaincre un noeud de ne pas simplement réduire au minimum fonctionnel ses valeurs de *CW*, les noeuds égoïstes doivent d'abord être détectés. En comparant les débits de données transférées des différents noeuds, un noeud dépassant la moyenne des débits de 10% est considéré comme égoïste. Après la détection d'un noeud égoïste, ce noeud est pénalisé pour éviter qu'un noeud ne réduise ses valeurs de *CW* au minimum fonctionnel. Le noeud est pénalisé en voyant ses paquets de données transférées brouillés par les autres noeuds. Le brouillage des paquets vise à le convaincre de ne pas trop réduire ses valeurs de *CW* sans quoi, ses paquets transmis seront brouillés. Un noeud qui remarque que ses paquets de données transférées sont brouillés adaptera ses valeurs de *CW*. En augmentant graduellement ses valeurs de *CW*, il voit éventuellement que ses paquets ne sont plus brouillés.

Pour un noeud égoïste, l'incitatif de gain rapide en diminuant drastiquement ses valeurs de *CW* est négativement balancé par la punition qui lui est infligée. Il ne doit donc pas trop réduire ses valeurs de *CW* et doit les maintenir à des niveaux acceptables pour les autres noeuds. Si tous les noeuds appliquent cette même stratégie, l'optimum de Pareto devrait être atteint lorsque les débits sont répartis de façon équitable entre les noeuds. De plus, si tous appliquent cette stratégie, cela prévient l'effondrement du réseau, car les noeuds égoïstes ne peuvent pas tirer d'avantages des autres noeuds.

2.3.2 Rôle du point d'accès dans les réseaux avec noeuds égoïstes

Le rôle du point d'accès pour contrôler les noeuds égoïstes dans les réseaux en mode infrastructure est décrit par Giarrè *et al.* (2009). Ils s'intéressent aux trafics en amont, des noeuds clients vers le point d'accès, et aux trafics bidirectionnels. Il est à noter que l'article fait référence aux probabilités de transmission plutôt qu'aux valeurs de *CW*. Or, il y a une relation

entre une valeur de *CW* et la probabilité de transmission. Plus la valeur de *CW* est petite, plus grande est la probabilité de transmission.

Dans un réseau où il n'y a pas de mécanisme pour punir les noeuds égoïstes, les probabilités de transmissions atteingnent l'équilibre de Nash avec au moins un de ces noeuds, à un débit nul. Pour pallier à ce problème, un mécanisme de punition contre les noeuds égoïstes est nécessaire pour éviter qu'un noeud soit toujours égoïste. Un noeud toujours égoïste a une probabilité de transmission de 1, alors que tous les autres noeuds de ce réseau auront une probabilité de transmission de 0. Dans le cas du trafic en amont, la méthode de punition du point d'accès est l'omission de l'envoi de paquet *ACK* à la réception d'un paquet de données. Ceci provoque l'échéance du temps d'attente pour la réception du paquet *ACK* chez le noeud égoïste, et éventuellement, un temps d'attente plus long pour sa prochaine transmission. En adaptant correctement les punitions, c'est-à-dire le nombre d'omissions des paquets *ACK* à la réception des paquets de données, l'optimum de Pareto est atteint.

Pour le trafic bidirectionnel, les auteurs supposent qu'il y a autant de trafic en amont qu'en aval et que les débits des trafics en aval sont égaux pour tous les noeuds clients. Si un noeud égoïste décide de transmettre un paquet de données en amont avec une probabilité de transmission de 1, alors le point d'accès provoque une collision dans tous les cas. De cette façon, le noeud qui transmet en amont doit diminuer sa probabilité de transmission s'il veut transmettre. Avec cette stratégie, tous les noeuds auront éventuellement des probabilités de transmission égales entre elles et plus petites que 1, où l'optimum de Pareto et l'équilibre de Nash sont atteints. Pour un réseau donné dans un contexte donné, cette probabilité de transmission est unique et identique pour tous les noeuds, c'est-à-dire qu'il y a une unique solution dans chaque cas et tous les noeuds auront la même probabilité de transmission.

2.4 Autres problèmes

Des problèmes autres que la manipulation du protocole par un noeud égoïste peuvent survenir. Cette section regroupe des problèmes qui font qu'un noeud peut paraître égoïste alors qu'en réalité il est collaboratif.

2.4.1 Effets des couches supérieures

Notons qu'il est possible pour un noeud d'avoir une part plus grande de la bande passante totale tout en respectant le protocole. Au niveau de la couche *MAC*, où la norme 802.11 est définie, il est possible que les noeuds soient collaboratifs, mais que leurs comportements s'apparentent à des noeuds égoïstes. En utilisant des applications très gourmandes en bande passante, un noeud peut nuire à ses pairs en monopolisant le réseau et empêcher les autres noeuds d'avoir une part équitable de la bande passante totale (Visoottiviseth *et al.*, 2010). Ceci peut survenir lorsqu'il y a des noeuds rattachés à un point d'accès et qui transfèrent beaucoup de données vers ce dernier, c'est-à-dire en amont. Cela peut limiter la bande passante d'un noeud qui veut simplement recevoir des données provenant du point d'accès. Une solution trouvée à ce problème est d'adapter les valeurs déterminant les fenêtres de contention, en particulier *CWMin*, afin que chaque noeud ait une part équitable de la bande passante totale. Même si la source du problème provient des couches supérieures, la solution au problème est appliquée au niveau de la couche *MAC*. Une solution s'apparentant à celle d'essayer de contrer un noeud égoïste impliquant la modification des paramètres par défaut de la norme 802.11 est élaborée.

2.4.2 Problèmes d'implémentation de la norme

D'autres problèmes moins communs peuvent survenir. Il est possible qu'un noeud soit détecté faussement comme étant égoïste parce qu'il existe certaines interfaces qui implémentent incorrectement la norme 802.11. Par exemple, le système DOMINO (Raya *et al.*, 2006) est déployé

et les noeuds du réseau ont un interface problématique tel que décrit par Bellardo et Savage (2003). Un noeud pourrait réinitialiser à 0 la valeur du *NAV* dans son interface lorsqu'il reçoit un paquet avec un champ *Duration* égal à 32767. En réinitialisant la valeur du *NAV*, le noeud au lieu de mettre en veille son interface radio pendant $32767\mu s$, n'attend pas ce délai et envoie immédiatement un paquet. Cet envoi hâtif de paquets peut être perçu comme un comportement égoïste si certaines des méthodes citées précédemment sont appliquées. En pratique, l'envoi d'un paquet de données avec la durée précise de $32767\mu s$ inscrite dans son champ *Duration* est peu probable, car cette durée est beaucoup trop longue en pratique.

CHAPITRE 3

DÉTERMINATION DE L'ÉTAT DES NOEUDS

Tel que vu dans le chapitre 2, il existe différentes façons pour détecter les noeuds égoïstes et réagir contre eux. La méthode présentée dans ce travail divise la méthode de détection et de réaction en trois étapes. Chaque étape réalise une partie du travail.

– la première étape est constituée du recueil des données. À cette étape, le nombre de paquets *ACK* reçus ou les durées de transmission des noeuds sont comptabilisés chez tous les noeuds implantant l'algorithme ;
– la deuxième étape est constituée du calcul des statistiques. À partir des données recueillies à l'étape précédente, des statistiques sont calculées dans le but de détecter les noeuds égoïstes ;
– la troisième étape est constituée de l'analyse des états. Les états des noeuds à partir des données et des statistiques recueillies précédemment, pour déterminer les réactions contre des noeuds égoïstes détectés, sont analysés. Les réactions programmées à cette étape sont exécutées par la suite.

3.1 Recueil des données

À cette étape, les données sur les différents comportements observés sur les noeuds du réseau sont recueillies durant chaque période d'observation. La fin d'une période d'observation survient lorsqu'un intervalle de temps fixe est écoulé ou lorsqu'un nombre de paquets fixe a été observé. Dans les circonstances présentes, les périodes d'observation sont déterminées selon une durée fixe ou selon un nombre total de paquets *ACK* observés.

3.1.1 Période d'observation à durée fixe

Au début d'une simulation, la durée des périodes d'observation est déterminée et demeure inchangée pour toutes les périodes d'observation. Ensuite, pour chaque noeud du réseau, des compteurs du nombre de paquets *ACK* reçus par chacun de ces noeuds sont définis. Par exemple, si le réseau comprend n noeuds, chaque noeud définit n compteurs. À noter que le noeud faisant les observations, dit noeud observateur, définit aussi un compteur pour lui-même dans le but de s'observer par rapport aux autres noeuds du réseau.

Pendant une période d'observation, le nombre de paquets *ACK* destiné pour chacun des noeuds est compté. Le début de la période de recueil des données et la durée de cette période étant identiques pour chacun des noeuds, le décompte est effectué de façon synchrone pour tous les noeuds. En pratique, la période d'observation est de quelques secondes. Cette période doit être assez longue pour obtenir une quantité suffisante de données. Ceci a pour but de calculer, à l'étape suivante, des statistiques significatives périodiquement.

3.1.2 Période d'observation avec nombre de paquets fixe

Un problème rencontré avec le recueil des données à durée fixe est la longueur de la période d'observation. Si une période d'observation est trop courte, alors il est possible qu'il n'y ait pas assez de données pour calculer des statistiques significatives. Ce problème peut survenir lorsque les débits de génération de données sont trop bas et que la taille des paquets est trop élevée. En choisissant des périodes de temps trop courtes, le phénomène d'iniquité de la norme 802.11 à court terme entre les noeuds est accentué (Koksal *et al.*, 2000). Également, la somme totale de tous les compteurs de paquets *ACK* à chaque période d'observation risque de varier d'une période d'observation à l'autre, ce qui n'est pas recherché. En fixant une période d'observation avec un nombre fixe de paquets *ACK* reçus, il est plus aisé de contrôler ces phénomènes. Puisque les paquets *ACK* sont émis par le point d'accès et que tous les noeuds peuvent observer

ces paquets, alors chaque noeud peut effectuer le même décompte total du nombre de paquets *ACK* échangé dans le réseau. En d'autres mots, le compteur global du nombre de paquets *ACK* devrait être identique pour tous les noeuds observateurs.

Pour définir une période d'observation avec un nombre de paquets fixe, en plus de définir un compteur de paquets *ACK* pour chaque noeud, chaque noeud initialise un compteur global du nombre de paquets *ACK* reçus au cours de la période d'observation. En fait, ce compteur global est la somme de tous les compteurs de paquet *ACK* pour chaque noeud. Pendant une période d'observation, ce compteur est incrémenté à la réception d'un paquet *ACK* peu importe sa destination. La période d'observation se termine lorsque le compteur global atteint un seuil déterminé à l'avance. En pratique, ce seuil doit être choisi de telle sorte que la durée d'une période d'observation soit au moins de quelques secondes, ceci afin de s'assurer de l'obtention éventuelle de statistiques significatives.

3.2 Calcul des statistiques

Après avoir compté les paquets *ACK* destinés à chacun des noeuds, des statistiques sont calculées. Le calcul de ces statistiques vise la détection de noeuds égoïstes. Après chaque période d'observation, la moyenne, notée μ, du nombre de paquets *ACK* reçus par les noeuds ainsi que l'écart-type σ sont calculés. Si l'écart-type calculé est suffisamment grand, les états des différents noeuds sont établis. Autrement, les états ne peuvent être établis correctement, car le niveau de saturation dans le réseau est insuffisant. La section 3.2.5 explique davantage le problème du niveau de saturation. Un état est une étiquette attribuée à un noeud indiquant si ce dernier respecte ou non la norme. Il y a un seul état par noeud et un noeud peut changer d'état d'une période d'observation à l'autre.

Soit N_i le nombre de paquets *ACK* destinés au noeud *i* pour une période d'observation donnée. À noter que ce compteur est présent chez tous les noeuds du réseau car tous font le même décompte.

L'état d'un noeud est établi en comparant N_i avec un seuil S_{Max}. Ce seuil est décrit à la section 3.2.2. À la fin d'une période d'observation, si N_i est plus grand que S_{Max}, l'état *CTR_HAUT* est attribué à ce noeud. Si N_i est plus bas qu'un autre seuil S_{Min}, seuil qui est fonction de S_{Max}, l'état *CTR_BAS* est attribué au noeud. Autrement, l'état *CTR_OK* est attribué au noeud, indiquant que N_i est entre S_{Min} et S_{Max}. Si le seuil S_{Max} est bien choisi, l'état *CTR_HAUT* est associé généralement à un noeud égoïste. Du moins, l'état *CTR_HAUT* est associé à un noeud qui a envoyé plus de paquets que ses semblables. Les états *CTR_BAS* et *CTR_OK* sont généralement associés aux noeuds collaboratifs mais la présence d'un ou plusieurs noeuds à l'état *CTR_BAS* peut indiquer la présence de noeuds égoïstes.

3.2.1 Choix du seuil

Le choix du seuil utilisé pour détecter la présence de noeuds égoïstes est primordial pour réduire les erreurs de détection. Si le seuil est trop bas, la probabilité de faux positif, c'est-à-dire un noeud détecté à tort comme égoïste, augmente. De façon semblable, si le seuil est trop élevé, la probabilité de faux négatif, c'est-à-dire un noeud égoïste non détecté, augmente.

Pour que l'algorithme fonctionne bien, le seuil doit être choisi correctement en fonction du nombre de noeuds dans le réseau, de la taille des paquets et du débit de génération du trafic de données. De ce fait, le choix du seuil doit être réévalué à chaque modification d'un paramètre du réseau. À noter que le choix du seuil est plus difficile en situation de faible saturation par rapport à une situation de forte saturation du réseau. Ceci est dû au fait qu'il est plus difficile de distinguer, dans ce cas, un noeud collaboratif d'un noeud égoïste. La section 3.2.5 discute de l'effet du niveau de saturation dans le réseau.

3.2.2 Seuil pour la détection des noeuds égoïstes

3.2.2.1 Motivation

Pour éviter la recherche par tâtonnement d'un nouveau seuil lorsqu'un paramètre du réseau est modifié, un algorithme par apprentissage a été développé. Dans les circonstances présentes, cet algorithme détermine un seuil approprié donnant moins de 1% de faux positifs. Le choix de ce pourcentage provient d'un compromis entre le nombre de détections de faux négatifs et de faux positifs.

D'une part, si ce pourcentage était trop petit, la probabilité d'avoir des faux négatifs augmenterait, c'est-à-dire que des noeuds réellement égoïstes passeraient, plus souvent, inaperçus. D'autre part, si ce pourcentage était trop grand, la probabilité d'avoir des faux positifs augmenterait, c'est-à-dire que des noeuds collaboratifs seraient, plus souvent, injustement considérés comme égoïstes. La détection de faux positifs arrivant inévitablement, un algorithme de correction contre les détections de faux positifs est mis en place par la suite. Cet algorithme est discuté à la section 3.2.4.

3.2.2.2 Algorithme par apprentissage et mise en situation

Quelques définitions de variables sont nécessaires pour la suite des choses. À noter que chaque noeud exécute le même algorithme.

- soit n le nombre de noeuds associés au point d'accès ;
- soit $N_{i,t}$ le compteur du nombre de paquets *ACK* destinés au noeud i au cours de la période d'observation t. Puisque les paquets *ACK* sont transmis par le point d'accès et que chaque paquet *ACK* contient, dans son en-tête, l'information sur son destinataire, tous les noeuds auront la possibilité de compter de façon identique le nombre de paquets *ACK* ;

- soit B le nombre total de paquets ACK observés au cours d'une période d'observation. Ce nombre B équivaut à la somme $\sum_{i=1}^{n} N_{i,t}$. Lorsque la somme atteint B, la période d'observation est complétée ;
- soit $M_t = \max\{N_{i,t} \mid 1 \leq i \leq n\}$ le compteur le plus élevé parmi tous les compteurs mesurés au cours de cette période d'observation t. À la fin de la période d'observation t, la valeur de M_t est calculée ;
- soit V le nombre de périodes d'observation effectuées ;
- soit $\mathcal{M}ax = max\{M_t \mid 1 \leq t \leq V\}$ le plus grand compteur M_t observé au cours des V périodes d'observation ;
- soit $\mathcal{M}in = min\{M_t \mid 1 \leq t \leq V\}$ le plus petit compteur M_t observé au cours des V périodes d'observation ;
- soit k un nombre dans l'intervalle $I = [\mathcal{M}in \ldots \mathcal{M}ax]$;
- pour chaque $k \in I$, définissons $C_k = |\{t \in [1 \ldots V] \mid M_t = k\}|$. En pratique, il est possible que certains compteurs C_k soit nuls.

La figure 3.1 illustre un exemple de répartition des valeurs M_t provenant d'une simulation avec $n = 5$ noeuds, le nombre de paquets ACK reçus $B = 5000$ par période d'observation et le nombre de périodes d'observation $V = 1000$. Bien entendu, il n'y a aucune valeur de M_t plus petite que la valeur $B/n = 1000$.

Pour terminer, soit $D_k = \sum_{\ell=\mathcal{M}in}^{k} C_\ell$, $\forall k \in I$. Par définition, $D_{\mathcal{M}ax} = V$. Ces valeurs définissent la fonction de distribution cumulative des valeurs M_t. La figure 3.2 illustre cette fonction non-décroissante pour l'exemple présenté à la figure 3.1.

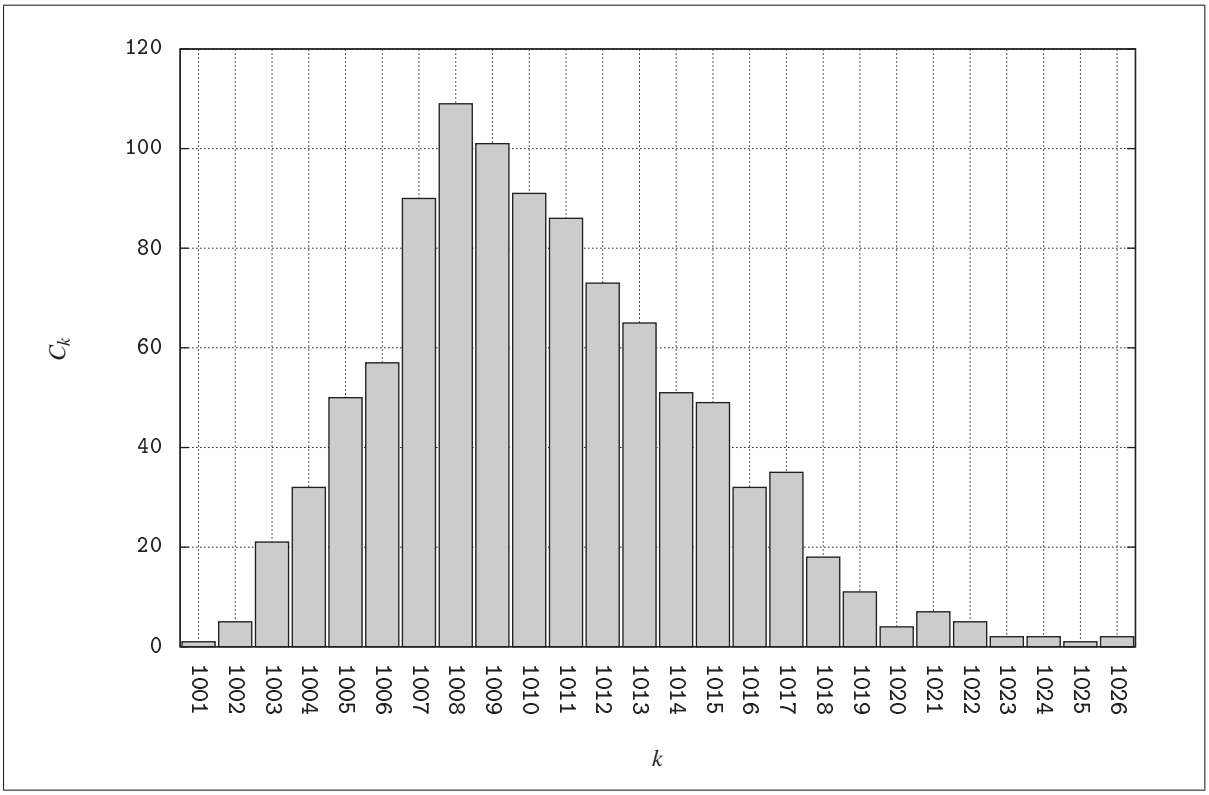

Figure 3.1 Histogramme de la distribution des valeurs M_t dans l'intervalle $[\mathcal{M}in..\mathcal{M}ax]$

3.2.2.3 Sans interpolation

Soit p, $0 < p < 1$, une fraction du nombre de périodes d'observation où un faux positif est détecté. En pratique, ce nombre est de l'ordre de quelques pourcents.

Partitionnons l'ensemble $E = \{M_t \mid 1 \leq t \leq V\}$ contenant V éléments en deux ensembles E_0^p et E_1^p tel que :

1. $E = E_0^p \cup E_1^p$.
2. $E_0^p \cap E_1^p = \emptyset$
3. $\left|E_1^p\right| = \lfloor p \cdot |E| \rfloor = \lfloor p \cdot V \rfloor$
4. $max\ E_0^p \leq min\ E_1^p$

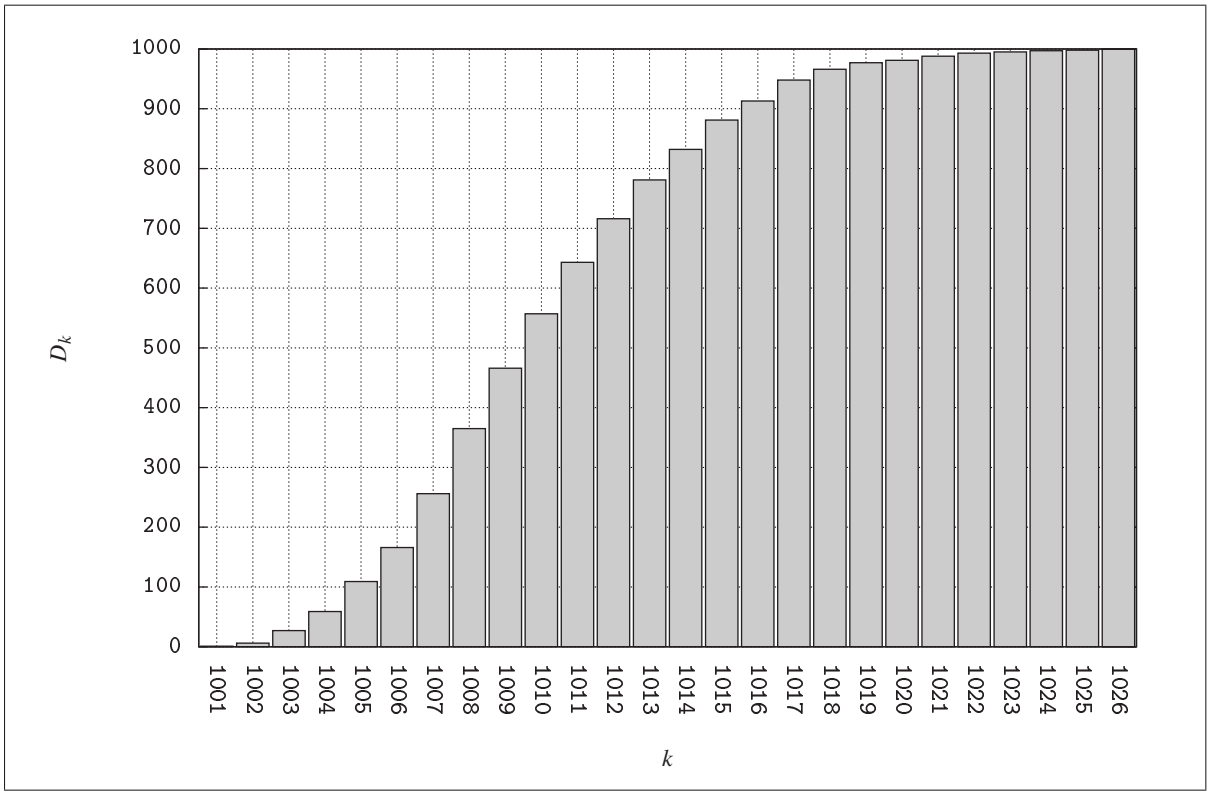

Figure 3.2 Histogramme de la distribution cumulative des valeurs M_t dans l'intervalle $[\mathcal{M}in..\mathcal{M}ax]$

Intituivement, cette partition regroupe les $\lfloor (1-p)\cdot|E|\rfloor$ plus petits éléments de E dans un ensemble E_0^p et les $\lfloor p\cdot|E|\rfloor$ plus grands éléments de E dans l'autre ensemble E_1^p. Cette partition peut nous servir à définir le seuil $S_{\mathcal{M}ax}$ comme suit.

Soit le seuil $S_{\mathcal{M}ax} = min\ E_1^p$. De ce fait, le nombre de M_j plus grand ou égal à ce seuil $S_{\mathcal{M}ax}$ est de $\left|E_1^p\right|$ ou $\lfloor p\cdot V\rfloor$. Ce nombre correspond au nombre de faux positifs en l'absence de noeud égoïste.

3.2.2.4 Interpolation linéaire

En traçant la fonction de distribution cumulative (voir figure 3.2) des différents D_k en fonction des k, $\forall k \in I$, une fonction croissante en escalier est obtenue. Puisqu'il est possible que le nombre d'éléments dans E soit plus petit que $1/p$, alors le seuil $S_{\mathcal{M}ax}$ risque d'être imprécis. Le $(1-p)^{ieme}$ centile de l'ensemble E n'est pas définie exactement si $|E| < 1/p$.

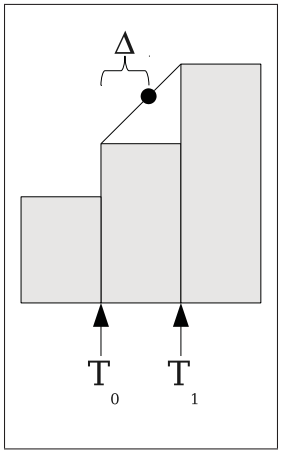

Figure 3.3 Représentation de l'interpolation linéaire

- soit $T_0 = D_{S_{\mathcal{M}ax}}$ la somme des compteurs jusqu'au seuil $S_{\mathcal{M}ax}$;
- soit $T_1 = D_{S_{\mathcal{M}ax}+1}$ la somme des compteurs jusqu'au compteur suivant $S_{\mathcal{M}ax}$. Si D_{T_1} n'est pas défini, alors $T_1 = T_0$;
- soit $\Delta = (V(1-p) - T_0)/(T_1 - T_0)$ la fraction du segment de la fonction entre T_0 et T_1 où devrait se retrouver exactement le $(1-p)^{ieme}$ centile de l'ensemble E. Si $T_1 = T_0$, alors $\Delta = 0$.

Le seuil recherché est $S_L = S_{\mathcal{M}ax} + \Delta = T_0 + \Delta$. La figure 3.3 montre un exemple d'un histogramme à 3 barres où Δ est non-nul.

3.2.2.5 Détermination de la constante multiplicative

Les seuils S_{Max} et S_L calculés précédemment ne sont pas utilisés directement pour déterminer si un noeud est égoïste ou non. Ils sont utilisés afin de déterminer un seuil S de la forme $\mu + b\sigma$ où b est une constante multiplicative de la forme $b = (S - \mu)/\sigma$.

Pour déterminer b de façon empirique, des simulations avec un grand nombre de périodes d'observation sans noeud égoïste sont effectuées. À chaque période d'observation, la valeur M_t, soit le nombre de *ACK* reçus le plus élevé, est enregistré. À la fin de la simulation, la moyenne μ, ainsi que l'écart-type σ, des échantillons de M_t sont enregistrés. En même temps, les seuils S_{Max} et S_L sont calculés. En substituant S par ces deux seuils dans la formule $b = (S - \mu)/\sigma$, la constante multiplicatrice b est obtenue.

La constante multiplicative est ensuite utilisée pour déterminer un seuil pour détecter les noeuds égoïstes. Au début d'une simulation, une période d'apprentissage sans noeud égoïste est executée pour calculer μ et σ. À la fin de la période d'apprentissage, le seuil de détection des noeuds égoïstes S est calculé en fonction de μ, σ et b. Les valeurs μ et σ sont obtenues pendant la période d'apprentissage, la constante multiplicative b reste fixe pour toutes les simulations.

Les tableaux 3.1 et 3.2 compilent les résultats provenant de ces simulations. Ces simulations utilisent des paramètres variés pour vérifier l'invariabilité de b. Le nombre de noeuds n lors des simulations varie entre 5, 10, 15 et 20. Six débits de trafic différents sont générés pour chaque nombre de noeuds et trois tailles différentes de paquets ont été utilisées. Chaque colonne des tableaux est donc le résultat de 24 simulations. Pour le recueil de statistiques, le nombre de paquets *ACK* comptés par périodes d'observation est fixé pour obtenir en moyenne 100 paquets par noeud. Autrement dit, chaque période d'observation se termine lorsque le compteur global du nombre de paquets *ACK* reçus atteint $100 \cdot n$. Le nombre de périodes d'observation par simulation est fixé à 5000.

Tableau 3.1 Détermination de la constante multiplicative b avec $S_{\mathcal{M}a\chi}$

Taille des paquets	**1000**	**2000**	**3000**
Moyenne des b	2.50	2.55	2.54
Écart-type des b	0.19	0.24	0.25

Tableau 3.2 Détermination de la constante multiplicative b avec S_L

Taille des paquets	**1000**	**2000**	**3000**
Moyenne des b	2.75	2.73	2.71
Écart-type des b	0.19	0.16	0.16

3.2.2.6 Justification de la constante multiplicative

Au cours d'une simulation, les comportements des noeuds sont normalement prévisibles. Les valeurs mesurées au cours des différents périodes d'observation, par exemple les M_t, sont normalement semblables d'une période d'observation à l'autre. Par contre, il peut arriver parfois que le comportement d'un noeud dévie sporadiquement de ses semblables sans raison particulière, qu'il y ait présence d'un noeud égoïste ou non. Ce problème sporadique provient probablement d'une erreur dans le code du simulateur.

Par exemple, lorsque des valeurs M_t sont calculées avec un grand nombre de périodes d'observation, il est possible qu'à certaines simulations, une ou plusieurs de ces valeurs M_t soient beaucoup plus élevées que les autres. Pour cette raison, utiliser directement le seuil $S_{\mathcal{M}a\chi}$ pour détecter un noeud égoïste peut donner de mauvais résultats car la valeur $S_{\mathcal{M}a\chi}$ est déterminée à partir d'une valeur de M_t parmi les plus élevés de l'ensemble des M_t.

L'exemple qui suit illustre la détermination d'un seuil de détection des noeuds égoïstes anormalement élevé provoquée par certains comportements imprévisibles sporadiques des noeuds.

Soit une simulation, sans noeud égoïste, avec 100 périodes d'observation ayant pour but de déterminer le seuil $S_{\mathcal{M}ax}$ avec la méthode décrite à la section 3.2.2.3. Soit la fraction p du nombre de périodes d'observation où un faux positif est détecté fixée à 1%. Si de ces 100 valeurs de M_t recueillies, il y en a deux beaucoup plus élevées que les autres, alors la valeur $S_{\mathcal{M}ax}$ obtenue sera également beaucoup plus élevée que la plupart des valeurs M_t recueillies. Puisque la valeur $S_{\mathcal{M}ax}$ correspond au $(1-p)^{ieme}$ centile de l'ensemble des M_t, la valeur $S_{\mathcal{M}ax}$ correspond au 99^{ieme} centile de l'ensemble des M_t, soit une valeur beaucoup trop élevée dans le cas présent. Si cette valeur est utilisée pour détecter les noeuds égoïstes, alors il est possible que le nombre de faux négatifs soit anormalement élevé.

Pour amoindrir ce phénomène, les seuils sont déterminés avec la formule $\mu + b\sigma$ et non pas directement avec les méthodes présentées aux sections 3.2.2.3 et 3.2.2.4. Utiliser la moyenne μ des M_t permet de niveler les différences qui peuvent exister entre les différents M_t. De plus, à cause de certains de ces comportements imprévisibles, la détermination de constante multiplicative b est produite à l'aide d'un grand nombre de simulations.

3.2.2.7 Vérification de la constante multiplicative

Dans le but de tester la valeur de la constante multiplicative obtenue à la section précédente, d'autres simulations ont été effectuées. Les simulations s'effectuent sans la présence de noeuds égoïstes.

L'objectif est de s'assurer que le seuil $S = \mu + b\sigma$ ne génère pas plus de 1% de faux positifs. Les valeurs de μ et σ doivent être calculées pour chaque simulation. Au début de chaque simulation, une période d'apprentissage de 100 périodes d'observation est exécutée pour obtenir des valeurs de M_t. Le calcul de μ, σ, ainsi que du seuil S, s'en suit. Pour les 1000 périodes d'observation suivantes, de nouvelles valeurs de M_t sont observées. Si une de ces valeurs est plus grande que $S = \mu + b\sigma$, alors un compteur de faux positifs est incrémenté.

Tableau 3.3 Nombre de faux positifs pour 1000 observations

Taille des paquets	**Constante multiplicative**				
	2.5	**2.6**	**2.7**	**2.8**	**2.9**
1000	9.8	7.8	7.0	5.5	4.6
2000	16.3	12.0	10.1	8.7	6.7
3000	14.0	11.4	9.8	8.3	6.9

Le tableau 3.3 présente les résultats avec différents choix de b. En s'inspirant des valeurs présentées dans les tableaux 3.1 et 3.2, d'autres choix de b ont été également utilisées. Chaque résultat du tableau représente le résultat de 20 simulations avec des nombres de noeuds et débits différents. Puisque 1000 observations ont été effectuées et que le seuil de détection de faux positifs est fixé à 1%, alors le nombre moyen de faux positifs attendu devrait être de $1000 \cdot 1\% = 10$. Les résultats provenant des simulations sont très près de cette valeur. Le choix d'une bonne constante multiplicative dans le contexte présent serait fixé autour de 2.7.

3.2.3 Limitations de la constante multiplicative

La précision du calcul du seuil est importante dans les situations où la saturation du réseau est faible. Puisque le seuil est déterminé selon la constante multiplicative, la précision de la constante multiplicative a un impact sur la précision du seuil. Cette précision est importante car la différence de débit mesuré entre un noeud égoïste et un noeud collaboratif est moins grande en situation de saturation faible qu'en situation de saturation forte. En situation de saturation forte, un seuil plus grand que celui calculé à partir de la constante multiplicative définie précédemment donne de meilleurs résultats car le nombre de faux positif s'en trouve réduit.

Une constante multiplicative, et du même fait un seuil, trop petits dans une situation de saturation faible font que le nombre de détection de faux positifs sera anormalement élevé. Inverse-

ment, si la constante multiplicative est trop grande, alors le nombre de détection manquée, soit de faux négatifs, sera anormalement élevé.

De plus en situation de saturation faible, il est possible que le M_t, pour une période d'observation t donnée, ne soit pas toujours associé au noeud égoïste, ce qui n'est pas le cas en situation de saturation forte. À l'extrême, sans saturation, les écarts entre les nombre de paquets *ACK* comptés chez les différents noeuds sont très petits malgré la présence de noeud égoïste. Dans cette situation, le M_t peut appartenir à n'importe quel noeud et aucune constante multiplicative aussi précise soit-elle ne permet de détecter les noeuds égoïstes.

3.2.4 Détection de faux positifs

Il est toujours possible que le comportement d'un noeud égoïste s'apparente ponctuellement à celui d'un noeud égoïste. Pour cette raison, l'algorithme de détection ne fait pas la distinction entre les vrais positifs et les faux positifs. Pour remédier en partie à ce problème, un nouvel algorithme est déployé à la suite de l'algorithme de détection. Après un nombre donné de périodes d'observation, cet algorithme permet d'étiqueter les noeuds vraiment égoïstes, c'est-à-dire les vrais positifs. Malgré l'implantation de cet algorithme, il n'est pas possible d'affirmer hors de tout doute si un noeud détecté égoïste est un vrai positif ou un faux positif. Par contre, l'algorithme réduit les risques de commettre des erreurs en sachant que la détection des noeuds égoïstes génère toujours des faux positifs. L'essence de l'algorithme réside dans le fait que si un noeud est détecté trop souvent comme égoïste, alors il s'agit probablement d'un noeud égoïste.

3.2.4.1 Algorithme discrimant les vraies détections de noeud égoïste des fausses

Quelques définitions sont nécessaires pour la suite des choses :

Au début d'une simulation, une valeur p_s est définie. Cette valeur définit la proportion maximale tolérée du nombre de périodes d'observation où le noeud a été détecté comme égoïste par rapport au nombre total de périodes d'observation.

Après chaque période d'observation et pour chaque noeud du réseau, les valeurs suivantes sont mises à jour :

– soit n_m le nombre total de périodes d'observation où le noeud a été détecté comme égoïste ;
– soit n_t le nombre total de périodes d'observation effectué.

Alors $p_m = n_m/n_t$ est la proportion observée du nombre de périodes d'observation où le noeud a été détecté comme égoïste par rapport au nombre total de périodes d'observation.

En fixant p_s beaucoup plus grand que p, p étant la fraction recherchée du nombre de périodes où un faux positif est détecté, alors les noeuds collaboratifs ne seront probablement jamais considérés comme égoïstes. En pratique, la valeur de p_s est fixée à 5% et la valeur de p est fixée à 1%. Les noeuds toujours collaboratifs, avec un seuil provoquant la détection de faux positifs en moyenne 1% du temps, ne verront probablement jamais leurs proportions observées p_m dépasser p_s.

Avec cet algorithme, un noeud étiquetté égoïste a dû être détecté comme égoïste pendant plus de 5% des périodes d'observation. Or, il est supposé qu'un noeud vraiment égoïste agit malicieusement pendant plus que 5% des périodes d'observation. Autrement, si un noeud décide d'être volontairement égoïste moins de 5% des périodes d'observation, son gain relatif par rapport aux autres noeuds est faible. Dans cette situation, il n'y a pas lieu de déclarer ce noeud formellement égoïste.

3.2.5 Niveau de saturation minimal

Un niveau de saturation minimal doit être présent dans le réseau pour permettre la détection des noeuds égoïstes. En supposant que les noeuds génèrent du trafic avec les mêmes paramètres, dans un réseau non-saturé, calculer la valeur $\max\{N_i\}$, soit le plus grand nombre N_i de paquets *ACK* observé pour un noeud pour une période d'observation donnée, à chaque période d'observation ne permet pas de détecter les noeuds égoïstes. Dans cette situation, tous les noeuds auraient des valeurs semblables de N_i. Dans un réseau fortement saturé, si la valeur $\max\{N_i\}$ est beaucoup plus grande que toutes les autres valeurs N_i, la détection d'un noeud égoïste se fait alors sans ambiguïté.

La détection est plus compliquée dans un réseau faiblement saturé. Dans ce cas, la valeur $\max\{N_i\}$ n'est pas toujours associée à un noeud égoïste. Pour s'assurer une cohérence dans la détection des noeuds égoïstes, un niveau de saturation minimal est défini pour s'assurer que la valeur $\max\{N_i\}$ est bel et bien associée dans la majorité des cas à un noeud égoïste.

La variance entre les différents N_i mesurés au cours d'une période donne un bon indice sur le niveau de saturation. Si la variance est nulle, alors le réseau n'est pas saturé. De façon expérimentale, il a été établi que plus cette variance est grande, plus le niveau de saturation est grand. Le tableau 3.4 illustre cette tendance. Des simulations avec cinq noeuds, dont un égoïste, ont été effectuées avec différents niveaux de saturation. Pour chaque simulation, 1000 périodes d'observations sont effectuées pour chacune d'elles. Le nombre de paquets *ACK* recueilli est de 500 par période d'observation.

La première colonne du tableau est le nombre moyen de paquets perdus par secondes. Ces paquets perdus correspondent aux paquets qui ont été détruits parce qu'ils n'ont pas pu être transmis à cause de la congestion sur le support.

La deuxième colonne correspond au nombre de fois que le plus grand des N_i mesurés au cours d'une période d'observation, soit le $\max\{N_i\}$, correspond au noeud égoïste. Autrement dit, en supposant que le noeud égoïste soit d'indice m, c'est le compteur du nombre de fois où l'égalité $N_m = \max\{N_i\}$ est vérifiée. Parce qu'il y a 1000 périodes d'observation par simulation, alors ce nombre ne peut pas dépasser 1000.

La dernière colonne correspond à la moyenne des variances des N_i de chaque période d'observation. Plus exactement, la variance des N_i est calculée pour une période d'observation et la moyenne des variances est faite sur les 1000 variances obtenues lors des 1000 périodes d'observation.

Tableau 3.4 Comparaison du nombre de paquets perdus et des variances des N_i

Paquets perdus / s	**$\max\{N_i\}$ est le noeud égoïste**	**Variances des N_i**
0.5	462	0.592
1	556	0.950
2	775	2.058
5	984	8.514
10	1000	29.924
50	1000	577.954

Une nette tendance est observée. Le nombre de fois que $\max\{N_i\}$ correspond au noeud égoïste, ainsi que la moyenne des variances des N_i, s'élèvent en fonction de l'augmentation du niveau de saturation. En situation de très faible saturation, la fiabilité pour détecter les noeuds égoïstes est trop faible. Par exemple, à la première ligne du tableau, le noeud égoïste a obtenu le compteur N_i le plus élevé parmi tous les noeuds que pendant 46,2% des périodes d'observation. Il n'est donc pas possible de détecter des noeuds égoïstes en supposant que $N_m = \max\{N_i\}$ en situation de très faible saturation car les risques d'erreurs sont trop grands.

Il a été établi empiriquement que, pour assurer une détection relativement fiable des noeuds égoïstes, la variance des N_i lors d'une période d'observation doit au moins être de $4 \cdot n$, où n est le nombre de noeuds dans le réseau. Par exemple avec 5 noeuds, la variance observée doit au moins être de 20.

Également à partir du tableau 3.4, il est observé que si un noeud perd plus de 10 paquets par secondes, alors la détection des noeuds égoïstes peut s'effectuer de façon relativement fiable.

3.3 Analyse des états

Puisque les états de chaque noeud sont évalués après chaque période d'observation, une noeud peut réagir contre un noeud égoïste après chaque période. Dépendemment des configurations et des états des noeuds du réseau, l'analyse des états permet de programmer différents types de réactions contre un noeud égoïste. Un noeud peut avoir une réaction positive, neutre ou négative. De plus, il y a différentes façons d'analyser les états obtenus à l'étape précédente. Les différentes variantes de l'algorithme d'analyse sont décrits dans la section 3.3.3.

De plus à l'étape précédente, un noeud a déterminé les états de tous les noeuds du réseau, y compris pour lui-même. Un noeud peut donc analyser son propre état par rapport aux autres noeuds. En accord avec les états des autres noeuds mais aussi en accord avec son propre état, la programmation d'une réaction peut être effectuée de différentes façons.

3.3.1 Types de réaction

La réaction neutre est la plus simple : il s'agit en fait de l'absence de réaction. L'absence de réaction est généralement liée à la rencontre d'un noeud collaboratif.

La réaction négative est généralement liée à la détection d'un noeud égoïste. Cette réaction vise à punir ce dernier. Une réaction négative peut être la diminution des valeurs déterminant

sa fenêtre de contention ou le brouillage des paquets provenant du noeud égoïste. Les détails des implémentations de ces réactions sont décrits dans les sections 3.3.2.1 et 3.3.2.2 respectivement.

La réaction positive est généralement liée à la détection d'un noeud collaboratif ou un noeud qui semble trop collaboratif. Un noeud dit trop collaboratif est un noeud qui n'a pas eu la chance de transmettre autant que les autres noeuds. Si tous les noeuds génèrent du trafic avec les mêmes paramètres, il est anormal qu'un noeud transmettre moins que les autres noeuds. Une réaction positive peut être l'augmentation des valeurs déterminant sa fenêtre de contention. Le détail de l'implémentation de cette réaction est décrit dans la section 3.3.2.1.

3.3.2 Familles de réactions

Deux familles de réaction sont proposées. Il y a d'abord la famille dont les réactions modifient, en diminuant ou augmentant, les valeurs déterminant la fenêtre de contention. Cette famille de réaction modifie en fait les valeurs de *CWMin* et *CWMax*. L'autre famille est la réaction qui brouille les paquets provenant du noeud égoïste. Au début d'une simulation, le choix d'une famille de réaction doit être effectué. Ensuite, tous les noeuds du réseau doivent appliquer que les réactions de la famille choisie durant toute la durée de la simulation.

3.3.2.1 Modification des valeurs déterminant la fenêtre de contention

Un des buts de la norme 802.11 est de diviser en part égale la capacité totale du réseau parmi tous ses noeuds. Normalement, un réseau fonctionnant avec cette norme ne suppose pas la présence de noeuds égoïstes. Ici, il y a présence de noeuds ayant modifié leur implémentation de la norme dans le but d'en tirer un avantage. Il est supposé que les paramètres déterminant la fenêtre de contention, soit les paramètres *CWMin* et *CWMax*, ont été diminué chez les noeuds

égoïstes. Pour tenter de rétablir l'équilibre dans les débits de transmission chez les noeuds du réseau, les noeuds collaboratifs sont invités à modifier à leur tour ces deux paramètres.

Dans l'amendement 802.11b de la norme 802.11, les valeurs déterminant la fenêtre de contention sont fixées à 31 pour *CWMin* et 1023 pour *CWMax*. À partir des types de réactions programmées précédemment, trois réactions sont possibles. Si une réaction positive est programmée, des augmentations de *CWMin* et *CWMax* sont effectuées. Si une réaction négative est programmée, des diminutions de *CWMin* et *CWMax* sont effectuées. Autrement, une réaction neutre a été programmée et les valeurs déterminant la fenêtre de contention sont laissée inchangées. Dans tous les cas, la forme $2^n - 1$ pour des valeurs de *CWMin* et *CWMax* est respectée.

Si une réaction négative a été programmée, alors les valeurs déterminant la fenêtre de contention sont diminuées. À chaque période où une réaction négative est programmée, la valeur de *CWMin* est divisée par deux et la valeur de *CWMax* est divisée par quatre. Si plusieurs diminutions s'enchaînent, alors ces valeurs peuvent atteindre des planchers de 2 pour *CWMin* et *CWMax*. Ces valeurs correspondent au minimum fonctionnel et ne peuvent être diminuées davantage, autrement le réseau devient non fonctionnel.

Si une réaction positive a été programmée, alors les valeurs déterminant la fenêtre de contention sont augmentées. Dans cette situation, la valeur de *CWMin* est multipliée par deux plus un et la valeur de *CWMax* est multipliée par quatre plus un. Si les valeurs de *CWMin* ou *CWMax* sont à 2, alors elles sont immédiatement décrémentées de un après leur multiplication. Ceci a pour but de respecter la forme $2^n - 1$ pour ces deux valeurs. Leurs augmentations ne peuvent pas dépasser les valeurs par défaut définies par l'amendement 802.11b, c'est-à-dire 31 pour *CWMin* et 1023 pour *CWMax*.

3.3.2.2 Brouillage des paquets du noeud égoïste

Au lieu d'agir sur lui-même en modifiant les valeurs déterminant sa fenêtre de contention, un noeud collaboratif peut réagir plus directement contre la présence d'un noeud égoïste. À la suite de la programmation d'une réaction négative, un noeud collaboratif peut brouiller les paquets de données provenant d'un noeud égoïste. De cette façon, le noeud égoïste ne peut pas envoyer correctement ses paquets de données et ainsi, son débit de données transmis correctement s'en trouve diminué.

Le brouillage s'effectue lorsqu'un noeud collaboratif envoie simultanément un paquet contenant des données fictives en même temps qu'un noeud égoïste envoie un paquet contenant des données légitimes. À la réception simultanée de ces deux paquets, la distinction entre les deux paquets ne peut pas être effectuée au niveau du récepteur. Le récepteur, le point d'accès dans le cas présent, ne peut que rejeter le paquet contenant les données légitimes.

Le noeud égoïste ne peut pas savoir immédiatement si ses paquets de données sont brouillés. En supposant que le noeud égoïste ne dispose que d'une seule interface radio, si ce dernier transmet des données, alors il ne peut pas en recevoir en même temps. Par contre, le noeud égoïste peut détecter qu'un problème est survenu lors de l'envoi de ses paquets de données lorsque, à la suite d'un temps d'attente, il ne reçoit pas le paquet *ACK* normalement associé à la transmission réussie d'un paquet de données.

Dans les circonstances présentes, le brouillage des paquets fonctionne seulement lorsque des paquets de données de grande taille sont transmis. Si le noeud égoïste est caché, il n'est pas possible de brouiller les paquets de petite taille, car rien ne survient avant l'envoi d'un paquet de petite taille. Par contre, les paquets *CTS* provenant du point d'accès sont toujours reçus. Ils peuvent donc être utilisés pour récupérer des informations sur l'émetteur d'un paquet de

grande taille, car les paquets *CTS* sont transmis seulement lorsque des paquets de grande taille doivent être transmis.

Un paquet *CTS* contient le champ *Duration* indiquant le temps qui reste, pour la séquence du paquet de données et des paquets de contrôle, avant d'avoir complété le processus d'envoi d'un paquet de grande taille. Pour réduire les risques de provoquer une confusion au niveau des autres noeuds du réseau, le brouillage est effectué que pendant une fraction du temps inscrit dans le champ *Duration*. Ainsi, seule une partie du paquet de données est brouillé afin de réduire la durée du brouillage. C'est qu'il n'est pas nécessaire de brouiller pendant toute la durée d'envoi d'un paquet de données pour provoquer sa corruption au niveau du récepteur. Généralement, le brouillage pendant une fraction significative de la durée d'envoi du paquet de données est suffisant pour provoquer sa corruption.

La durée inscrite dans le champ *Duration* provenant d'un paquet *CTS* inclut les durées d'envoi des paquets *CTS* et *ACK*, de plusieurs durées d'attente *SIFS* et du temps de transmission du paquet de données. La durée d'envoi du paquet de données n'est alors qu'une partie de cette durée. Malgré une différence probable entre le débit de transmission des paquets de contrôle et le débit de transmission des paquets de données, il est supposé que la durée d'envoi du paquet de données est plus grande que *Duration/2*, l'autre moitié étant réservé aux temps d'attente et aux paquets de contrôle. De plus, il est supposé que la durée d'envoi du paquet *CTS*, additionné d'une durée d'attente *SIFS*, est inférieure à *Duration/4*. Cette durée représente le temps, à partir du début de la réception d'un paquet *CTS*, où il est probable qu'un paquet de données soit en cours de transmission. Le brouillage d'un paquet de données s'effectue alors après un délai d'attente de *Duration/4* après le début de la réception d'un paquet *CTS* et pendant une durée de *Duration/*2.

À noter que les délais de propagation des ondes radio n'est pas pris en compte pour déterminer le délai et la durée du brouillage. C'est que le délai et la durée du brouillage n'ont pas besoin

d'être d'une extrême précision pour simplement corrompre un paquet de données. De plus, un seul noeud, dit brouilleur, est nécessaire pour effectuer le brouillage.

3.3.2.3 Remarques

Il n'existe pas de réaction positive dans la famille qui brouille les paquets. C'est que la réaction positive usuelle associée à cette réaction négative est de ne simplement pas brouiller les paquets. Il s'agit alors de la même réaction qu'une réaction neutre.

Lorsque la famille modifiant les valeurs déterminant la fenêtre de contention est choisie, il n'est pas possible d'appliquer dans toute les circonstances la réaction négative ou la réaction positive. Par exemple, si les valeurs déterminant la fenêtre de contention sont déjà à leurs valeurs minimales, alors il n'est pas possible de diminuer davantage ces valeurs. Une réaction neutre prend la place de la réaction négative. De la même façon, si une réaction positive est programmée et que les valeurs déterminant la fenêtre de contention sont maximales, alors une réaction neutre remplace la réaction positive.

À noter qu'un noeud qui adopte les réactions de la famille qui modifient les valeurs déterminant la fenêtre de contention ne peut qu'agir sur lui-même. La raison est qu'un noeud n'a pas la possibilité de modifier ces valeurs chez les autres noeuds, il ne peut les modifier que sur lui-même. Par son action sur lui-même, il lui est possible de modifier les comportements des autres noeuds. De façon similaire, si la famille qui brouille les paquets est adoptée, alors que les paquets provenant des autres noeuds sont brouillés ; il serait ridicule qu'un noeud brouille ses propres paquets envoyés.

3.3.3 Algorithme d'analyse des états

3.3.3.1 Algorithme d'analyse des états 1

Cet algorithme d'analyse suppose la bonne volonté du respect de la norme des noeuds. Une réaction négative est programmée seulement lorsqu'un noeud est à l'état *CTR_HAUT*. Autrement, une réaction positive est programmée car l'algorithme suppose la bonne volonté. Les états *CTR_OK* et *CTR_BAS* sont ignorés. Les deux familles de réaction peuvent être programmées avec cet algorithme.

3.3.3.2 Algorithme d'analyse des états 2

Une première faiblesse de l'algorithme précédent, décrit dans la section 3.3.3.1, est sa tendance à faire alterner réaction négative et réaction positive lors de la détection d'un noeud égoïste. Généralement, à la détection d'un noeud égoïste, une réaction négative est programmée. À la suite de l'exécution de la réaction négative, ou d'une suite de réaction négative, le noeud égoïste voit son nombre de paquets transmis diminué. Son nombre de paquets transmis devient éventuellement semblable aux autres noeuds. Dans cette situation, ce noeud égoïste est détecté comme collaboratif. Si l'algorithme précédent est exécuté, une réaction positive est programmée. La réaction positive avantage à son tour le noeud égoïste et une nouvelle réaction négative est programmée. Une alternance entre réaction négative et réaction positive s'en suit. Également, une deuxième faiblesse est que l'algorithme précédent ignore un noeud qui est à l'état *CTR_BAS*.

Pour pallier ces faiblesses, contrairement à l'algorithme précédent où une réaction positive était programmée automatiquement, une réaction positive est programmée lorsqu'un autre noeud, autre que lui-même, est à l'état *CTR_BAS*. Si un noeud égoïste est détecté, alors l'état *CTR_HAUT* lui est attribué. Dans ce cas, une réaction négative est programmée. En ne pro-

grammant pas automatiquement les réactions positives, le phénomène d'alternance décrit dans le paragraphe précédent n'est plus observé.

De plus, un noeud évalue son état par rapport aux états des autres noeuds. S'il y a une présence simultanée de différents noeuds avec les états *CTR_HAUT* et *CTR_BAS* dans le réseau, c'est un indicateur de la présence simultanée de noeuds égoïstes et de noeuds trop collaboratifs. Dans cette situation, le noeud programme une réaction neutre si son propre état est *CTR_OK*. Par contre, si son état est *CTR_HAUT*, alors ceci indique qu'il adopte lui-même un comportement égoïste. Pour éviter de se faire punir par les autres noeuds, une réaction négative est programmée. De la même façon, si son état est *CTR_BAS*, alors une réaction positive est programmée.

À noter que seule la famille modifiant les valeurs déterminant la fenêtre de contention est programmable avec cet algorithme.

3.3.3.3 Algorithme d'analyse des états 3

Un autre problème de l'algorithme décrit dans la section 3.3.3.1 est constaté lorsqu'un noeud égoïste redevient collaboratif. Ce problème survient lorsque la famille de réactions modifiant les valeurs de fenêtre de contention a été choisie. Lorsqu'un noeud a détecté un noeud égoïste durant plusieurs périodes d'observation, il a appliqué une succession de réactions négatives. Il en résulte que le noeud effectuant l'analyse aura des valeurs *CWMin* et *CWMax* aux minima fonctionnels. Ces minima sont probablement égaux aux valeurs du noeud égoïste. Lorsque le noeud égoïste redevient collaboratif, ses valeurs de *CWMin* et *CWMax* sont remises aux valeurs d'un noeud respectant la norme, soit d'un noeud collaboratif n'ayant jamais eu de réaction négative.

Puisque tous les noeuds collaboratifs utilisent la même stratégie, alors tous auraient leurs valeurs *CWMin* et *CWMax* au minimum fonctionnel. Dans cette situation, ces noeuds collaboratifs vont se détecter comme égoïste les uns par rapport aux autres. Ceci cause une succession

de réactions négatives laissant les valeurs *CWMin* et *CWMax* au minimum fonctionnel pour toujours. Puisque tous les noeuds adoptent des comportements égoïstes, sauf le noeud anciennement égoïste qui est redevenu collaboratif, le noeud collaboratif conserve toujours l'état *CTR_BAS* et tous les autres noeuds conservent l'état *CTR_HAUT*.

Si cette situation d'interblocage est rencontrée, au lieu d'avoir une réaction négative telle que proposée par l'algorithme décrit dans la section 3.3.3.1, un noeud peut adopter une réaction positive. De cette façon, la suite de réaction négative est brisée et la situation d'interblocage est éliminée.

3.4 Variante des critères de détection

Les algorithmes présentés dans les sections 3.1, 3.2 et 3.3 utilisent des périodes d'observation basées sur les nombres de paquets *ACK* reçus. Ces périodes sont définies selon un nombre de paquets *ACK* reçus pour une durée fixe ou selon un nombre total de paquets *ACK* reçus.

Au lieu d'utiliser les nombres de paquets *ACK* reçus, il est possible de recueillir les durées de transmission sur le support des différents noeuds pour calculer les statistiques. Le recueil des durées de transmission permet l'utilisation de paquets de tailles variables, contrairement à des paquets de taille fixe lorsque les paquets *ACK* sont comptés. Il est supposé que tous les noeuds collaboratifs auraient des durées de transmission sur le support, par périodes d'observation, semblables. Un noeud égoïste se démarquerait de ces derniers en ayant une durée de transmission plus grande.

Mise à part la nature des données qui change, les principales étapes nécessaire à la détection des noeuds égoïstes sont les mêmes. Puisque les données recueillies ne sont pas les mêmes que dans les sections précédentes, l'étape du recueil des données est modifiée ainsi que les étapes pour calculer le seuil. Notamment, la détermination d'une constante multiplicative doit être faite avec des durées de transmission au lieu des nombres de paquets *ACK*. L'analyse des

états s'effectue de la même façon que précédemment. Seules les modifications, par rapport aux sections précédentes, sont décrites dans cette section pour permettre l'utilisation des durées de transmission comme données d'entrées pour détecter les noeuds égoïstes.

3.4.1 Génération des paquets de tailles variables

Les algorithmes recueillant les nombres de paquets *ACK* reçus utilisent des paquets de taille fixe pour calculer les statistiques. Dans le but d'éprouver différemment les algorithmes de détection de noeuds égoïstes, des paquets de taille variable sont utilisés avec les algorithmes recueillant les durées de transmission.

Pour générer le trafic de données de façon semblable pour tous les noeuds, la moyenne des tailles des paquets doit être fixé et doit être identique pour tous les noeuds. Autrement, un noeud collaboratif avec une moyenne plus grande serait détecté injustement comme égoïste. Dans cette optique, un générateur de paquets de taille variable est défini pour chacun des noeuds et pour chacun des noeuds, il est paramétré identiquement. En résumé, ce générateur produit des paquets dont la taille varie dans le temps mais dont la moyenne des tailles, prise sur une période de temps assez longue, est identique pour tous les noeuds.

Pour générer les différentes tailles de paquets, une taille minimale de paquets m et un écart maximal M, par rapport à la taille minimale, sont choisis. Les tailles de paquets sont alors entre m et $m+M$. Pour générer la taille d'un paquet, un nombre e est choisi entre 0 et M selon une loi de probabilité uniforme. La taille est alors définie par $m+e$. De cette façon, les paquets générés sont en moyenne de taille $m+(M/2)$.

Il n'est pas nécessaire de changer la taille des paquets à chaque nouveau paquet généré. Avec une taille moyenne des paquets utilisés en pratique relativement petite par rapport aux débits de données générés, plusieurs centaines ou milliers de paquets sont générés par secondes. Puis, les périodes d'observation sont de l'ordre de quelques secondes, donc des paquets de toutes

les tailles peuvent être reçus pendant ces périodes d'observation. En pratique, il est suffisant de changer la taille des paquets que fréquemment et non pas à chaque nouveau paquet de données générés.

Pour ces raisons, une durée constante d, où la taille des paquets restent inchangée, est définie. Lors de la simulation, à chaque bloc de durée d écoulé, la taille des paquets est changée.

3.4.2 Période d'observation avec des durées de transmissions constantes

La méthode pour recueillir les durées de transmission de chaque noeud est semblable à la méthode dénombrant les paquets *ACK* reçus, sauf pour la nature des données recueillies. À cause du problème des noeuds cachés, il n'est pas toujours possible de mesurer directement sur le support les durées de transmission de certains noeuds.

Lorsqu'un paquet de grande taille doit être transmis, des paquets de contrôle de type *RTS* et *CTS* sont d'abord émis. Ces paquets contiennent le champ *Duration*. Son contenu correspond au temps total de transmission comprenant le paquet de données, des paquets de contrôle ainsi que des différents temps d'attente sans activité. Si tous les noeuds utilisent les mêmes débits de transmission des paquets de contrôle et des paquets de données, alors les valeurs des champs *Duration* devraient être proportionnelles aux nombres d'octets contenus dans les paquets de données. Seuls les paquets *CTS* peuvent être utilisés pour récupérer la valeur du champ *Duration*, car les paquets *RTS* ne sont pas nécessairement reçus par tous à cause des noeuds cachés.

Au début d'une période d'observation, un noeud initialise un compteur de la durée de transmission sur le support pour chaque noeud du réseau. Lorsqu'un paquet *CTS* est reçu, le contenu du champ *Duration* est lu. Pour ne récupérer que la durée de transmission du paquet de données, la valeur du champ *Duration* est soustraite de deux durées *SIFS* et de la durée de transmission du paquet *ACK*. Le résultat est ensuite ajouté au compteur du noeud auquel le paquet *CTS*

est adressé. Après un certain moment, un compteur contient la durée de transmission sur le support, cumulée avec différents paquets de données transmis, pour un noeud donné.

Pour définir les périodes d'observation, chaque noeud définit un compteur global de la durée de transmission des paquets de données peu importe la source de ces paquets. Ce compteur global est en fait la somme des compteurs contenu dans un noeud tel que définis précédemment. Lorsque le compteur global atteint un seuil défini d'avance, la période d'observation est terminée. Puisque chaque noeud reçoit identiquement les paquets *CTS* transmis par le point d'accès et que chacun des noeuds exécute le même algorithme, tous les noeuds devraient terminer leur périodes d'observation au même moment.

3.4.3 Seuil pour la détection des noeuds égoïstes avec les durées de transmission

La méthode pour déterminer le seuil de détection des noeuds égoïstes avec les durées de transmission est similaire à la méthode présentée dans la section 3.2.2.2. Puisque les données recueillies sont des durées de transmission et non pas des compteurs de paquets reçus, alors de nouvelles variables doivent être définies. À noter que les variables semblables à celles de la section 3.2.2.2 ne sont pas définies dans la présente section.

– soit D_i la durée de transmission du noeud i pour une période d'observation donnée ;
– soit $max\{D_i\}$ la durée de transmission la plus élevée parmi tous les D_i au cours d'une période d'observation donnée ;

De façon similaire à ce qui est présenté à la section 3.2.2, une nouvelle constante multiplicative b_d doit être déterminée. Cette constante multiplicative permet éventuellement de calculer le seuil $S = \mu + \sigma b_d$ où μ et σ sont, dans le cas présent, la moyenne et l'écart-type des $max\{D_i\}$ mesurés au cours d'une période d'apprentissage. La période d'apprentissage regroupe l'observation d'un certain nombre de $max\{D_i\}$ avant l'arrivée de noeuds égoïstes. À noter que

la valeur b_d peut ne pas être identique à la valeur b de la section 3.2.2.5, car des durées de transmission sont maintenant mesurées et la taille des paquets est maintenant variable.

3.4.3.1 Détermination de la constante multiplicative

Avec le seuil de la forme $S = \mu + \sigma b_d$, la constante multiplicative b_d est de la forme $(S - \mu)/\sigma$. Pour calculer b_d de façon empirique, des résultats de simulations avec un grand nombre d'observations ont été utilisés pour calculer les statistiques μ et σ.

La technique pour déterminer la constante multiplicative est semblable à celle de la section 3.2.2.5. Par contre ici, μ et σ sont la moyenne et l'écart-type d'un certain nombre de $max\{D_i\}$ observés au cours de simulations sans noeud égoïste. De plus, le seuil S doit être substitué par le seuil S_d dans la formule $(S - \mu)/\sigma$.

Pour déterminer S_d, une simulation sans noeud égoïste avec un grand nombre de périodes d'observation est produite. À chaque simulation, un ensemble de $max\{D_i\}$ est recueilli. Dans le cas présent, l'ensemble de ces $max\{D_i\}$ est noté E. Si la fraction p de détection de faux positifs est fixée à 1%, le seuil S_d est assigné au 99^{ieme} centile des éléments de E.

Il est possible de fixer S_d sans interpolation quelconque car les paquets générés sont de tailles variables. Ici, les valeurs des *Duration* de chaque paquet *CTS* sont également variables. D'une période d'observation à l'autre, il est peu probable que deux D_i observés soient égaux. De ce fait, les $max\{D_i\}$ observés sont également différents.

Le tableau 3.5 illustre le résultat de simulations pour déterminer b_d. Plusieurs simulations sont produites pour obtenir une valeur b_d significative et universelle. Les paramètres suivants sont utilisés :

– le nombre de périodes d'observations est fixé à 1000 ;

– la durée cumulée de chaque période d'observation est fixée à $100 \cdot n$ millisecondes où n est le nombre de noeuds ;
– des paquets de tailles variables dont les moyennes sont de 1000, 2000 et 3000 octets sont générés. Le tableau regroupe, par colonne, les simulations avec des tailles variables moyennes identiques ;
– les différentes tailles de paquets générés ont un écart aléatoire maximal de $\pm$ 500 octets par rapport à la moyenne ;
– les tailles des paquets sont modifiées à chaque 10 millisecondes ;
– des réseaux avec 5, 10, 15 et 20 noeuds sont utilisés ;
– pour ces réseaux avec des nombres de noeuds différents, quatre débits différents sont utilisés ;
– la moyenne des b_d est le résultat de 16 simulations où la valeur $(S_d - \mu)/\sigma$ est calculée ;
– l'écart-type des b_d est l'écart-type des 16 valeurs de b_d obtenues.

Tableau 3.5 Détermination de la constante multiplicative b avec des périodes d'observations basées sur les durées de transmissions

Taille des paquets	**1000**	**2000**	**3000**
Moyenne des b_d	2.57	2.55	2.52
Écart-type des b_d	0.12	0.17	0.22

3.4.3.2 Vérification de la constante multiplicative

Le tableau 3.6 montre le nombre de faux positifs observé avec 1000 périodes d'observation. La fraction p du nombre de faux positif est fixée à 1%. Dans ces conditions, le nombre de faux positif attendu est de 10. La méthode est semblable à celle présentée à la section 3.2.2.7. Le nombre de périodes d'observation pour la période d'apprentissage est de 100. Ensuite, les valeurs μ et σ sont calculées. Puis le seuil $S = \mu + \sigma b_d$ est calculé. Finalement, 1000 périodes d'observation sont produites et le nombre de détection des faux positifs est compté.

Tableau 3.6 Nombre de faux positifs pour 1000 observations

Taille moyenne des paquets	**Constante multiplicative**					
	2.4	2.5	2.6	2.7	2.8	2.9
1000	16.6	13.9	11.3	9.1	7.4	6.1
2000	15.5	11.9	9.4	7.9	6.7	5.6
3000	10.6	8.1	6.7	5.7	4.9	4.2

CHAPITRE 4

RÉSULTATS EXPÉRIMENTAUX

À partir des algorithmes décrits précédemment, des simulations ont été produites pour en vérifier leur validité. Aussi, les simulations permettent de vérifier les différents comportements des algorithmes lorsque divers paramètres dans les réseaux sont utilisés, par exemple avec des nombres de noeuds différents ou des niveaux de saturation différents. En effectuant la détection d'un noeud égoïste et en réagissant contre ce dernier, il est possible de vérifier si l'application des algorithmes est bénéfique ou nuisible pour l'ensemble des noeuds du réseaux selon différentes situations.

4.1 Simulateur *ns-2*

Ns-2 est un logiciel de simulation conçu pour la recherche sur les réseaux informatiques (Issariyakul et Hossain, 2009). Il implémente plusieurs protocoles de communication des différentes couches du modèle *Open Systems Interconnection* (*OSI*), dont entre autres la norme IEEE 802.11 pour les réseaux sans fil. Le code source du simulateur est écrit en C++ et l'interface permettant d'interagir avec le simulateur est écrite en Tcl. Il est sous la license *GNU General Public License* (*GNU GPL*) et le code source est disponible gratuitement. Il est possible pour quiconque de modifier son code source dans le but de modifier ou d'implanter de nouvelles fonctionnalités.

Le travail de recherche de ce mémoire est basé sur la modification du code source de *ns-2*, plus particulièrement au niveau du module `Mac/802_11` qui implémente la couche *Media Access Control* (*MAC*) de la norme IEEE 802.11.

4.1.1 Minimum fonctionnel de la fenêtre de contention dans *ns-2*

Le nombre d'intervalles choisi pour la période de retrait est calculé à partir d'un nombre généré pseudo-aléatoirement auquel l'opération modulo est appliqué. Ce nombre d'intervalle est le nombre de fois que *SlotTime* est attendu pour la période de retrait, la durée de la période de retrait étant ce nombre d'intervalle multiplié par *SlotTime*. Soit A une valeur discrète choisie aléatoirement entre 0 et le plus grand entier supporté par le compilateur de *ns-2* $(MAXINT)$. Soit CW la valeur de la fenêtre de contention, soit une valeur entre *CWMin* et *CWMax*. Le nombre d'intervalle C d'une période de retrait quelconque est défini par l'équation $A \bmod CW$. Pour des raisons évidentes, CW ne peut pas être 0. De plus, si CW est égal à 1, alors le nombre d'intervalle choisi serait toujours 0. Si un noeud utilise ce nombre d'intervalle, les périodes de retraits serait nulles. Si plus d'un noeud utilise des périodes de retraits nulles, alors tous les paquets envoyés par les noeuds avec une fenêtre de contention nulle entreraient en collision car tous les envois provenant de ses noeuds seraient parfaitement synchronisés. Du moins, c'est ce qui a été constaté à l'intérieur du simulateur, probablement que cette constatation serait inexistante dans un réseau réel. Pour cette raison, la valeur minimale que peut prendre *CWMin* ou *CWMax* dans le simulateur dans le cadre de ce travail est fixée à 2.

4.2 Situation de saturation élevée

Les paramètres utilisés ont été ajustés pour simuler un trafic de données avec une saturation élevée. Le niveau de saturation est ajusté pour avoir une perte d'environ 50 paquets par noeuds par secondes lorsque tous les noeuds sont collaboratifs. À moins d'affirmation contraire, les réseaux simulés comportent 5 noeuds et la taille des paquets est de 1000 octets.

À la figure 4.1, le noeud N1 est d'abord collaboratif, comme tous les autres noeuds. Puis, N1 devient égoïste à partir de t=100s, soit 100s du début de la simulation, Ses valeurs de *CWMin* et *CWMax* sont mises à 2 à cet instant. Deux courbes sont présentes sur la figure, soit celle

du débit mesuré de N1 et une autre qui représente le débit mesuré moyen des noeuds N2, N3, N4 et N5, ces quatres noeuds restants collaboratifs tout au cours de la simulation. Cette figure illustre très bien l'impact d'un noeud égoïste sur un réseau. Lorsque N1 devient égoïste, son débit mesuré rejoint son débit de données généré à la source, alors que les autres noeuds subissent une perte dans leur débits mesurés.

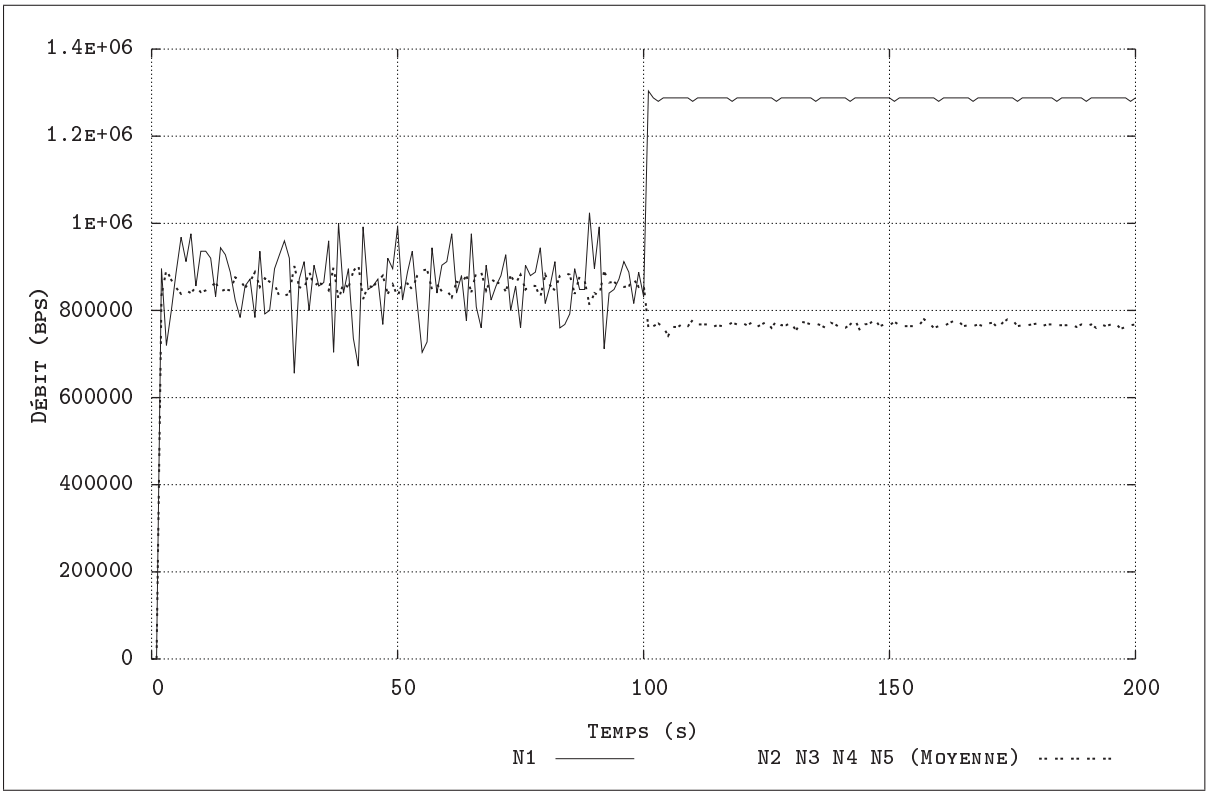

Figure 4.1 Impact du noeud égoïste sur le réseau

4.2.1 Changement des valeurs déterminant la fenêtre de contention

4.2.1.1 Avec l'algorithme 1

Dans le but de contrer la présence d'un noeud égoïste, les noeuds collaboratifs peuvent diminuer leurs valeurs *CWMin* et *CWMax* pour imiter le noeud égoïste, tel que décrit à la section 3.3.3.1. Avant de modifier ces valeurs, le noeud égoïste doit d'abord être détecté.

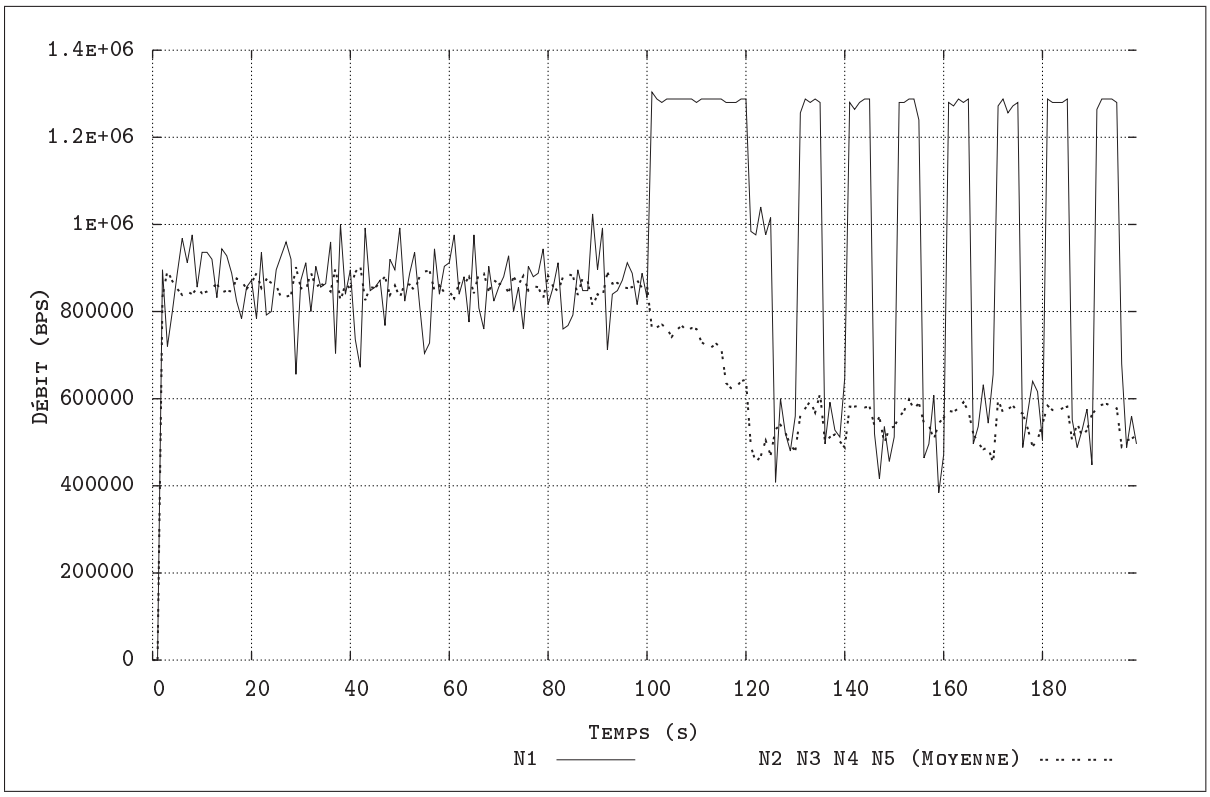

Figure 4.2 Changement de *CWMin* et *CWMax* - Algorithme 1

La figure 4.2 montre l'effet de l'application de l'algorithme d'analyse des états 1 (algorithme 1) en présence d'un noeud qui devient égoïste. Semblablement à la figure 4.1, le noeud N1 devient égoïste à t=100s. À cet instant, ses valeurs de *CWMin* et *CWMax* sont mises à 2. Les

autres noeuds restent collaboratifs tout au long de la simulation. Deux courbes sont illustrés, soit la courbe pour le noeud N1 et la courbe moyenne des quatre autres noeuds collaboratifs qui appliquent l'algorithme 1. Les périodes d'observation sont fixées à une durée de 5 secondes. Aussi, il n'y a pas de périodes d'observation pour déterminer un seuil car le seuil de détection des noeuds égoïstes, dans le cas des simulations avec une forte saturation, est fixé d'avance.

Malgré que les noeuds collaboratifs aient détecté le noeud égoïste à t=100s, la réaction contre le noeud égoïste n'est pas instantanée. Par la nature de l'algorithme 1, les valeurs déterminant la fenêtre de contention des noeuds collaboratifs vont être diminuées graduellement pour rejoindre éventuellement celles du noeud égoïste. Après t=100s, quatre périodes d'observation consécutives de diminution des valeurs *CWMin* et *CWMax* sont nécessaires avant de remarquer un effet sur le débit de données mesuré du noeud égoïste. À t=120s, la valeur de *CWMin* chez les noeuds collaboratifs est égale à 2 et celle de *CWMax* est égale à 3. Ces valeurs sont légèrement plus élevées que celles du noeud égoïste mais tout de même très près. Une perte dans le débit de données mesuré pour le noeud égoïste est mesuré à cet instant.

À t=125s, tous les noeuds ont des valeurs de *CWMin* et *CWMax* égales à 2. En fait, les noeuds collaboratifs adoptent temporairement le même comportement que le noeud égoïste N1. Les débits de données mesurés dans cette situation sont alors semblables pour tous les noeuds. Lorsqu'ensuite le nombre de paquets *ACK* mesuré devient semblable pour tous les noeuds, les noeuds appliquant l'algorithme 1 supposent que le noeud égoïste a disparu car tous les noeuds se verront attribuer l'état *CTR_OK*.

À t=130s, les valeurs *CWMin* et *CWMax* des noeuds collaboratifs sont augmentées, tandis que celles du noeud égoïste reste fixées à 2. Avec une différence dans les fenêtres de contention, le noeud égoïste revoit son débit de données mesurés augmenté, alors que les débits mesurés des noeuds collaboratifs diminuent légèrement. Par la suite, les noeuds collaboratifs redétectent N1 comme égoïste. Une alternance d'augmentations et de diminutions des valeurs de *CWMin*

et *CWMax* chez les noeuds appliquant l'algorithme 1 s'en suit. En conséquence, il est observé que le débit mesuré du noeud égoïste suit cette alternance.

4.2.1.2 Pause avec l'algorithme 1

L'application sans modification de l'algorithme 1, tel qu'illustré à la section 4.2.1.1, a pour effet de provoquer fréquemment une oscillation dans les débits de données mesurés. Cette oscillation permet au noeud égoïste de transmettre à son débit de génération de données durant une période d'observation sur deux dans la situation présente.

Au prix d'une réaction moins rapide si un noeud égoïste décide de redevenir collaboratif, les noeuds collaboratifs appliquant l'algorithme 1 peuvent appliquer une pause dans cette application lorsque leurs valeurs *CWMin* et *CWMax* deviennent minimales. En appliquant une pause de quelques périodes d'observation dans cette situation, cela permet à un noeud égoïste, s'il décide de ne jamais redevenir collaboratif, de ne pas avoir un débit de données mesuré égal à son débit de génération une période sur deux tel que précédemment. Par exemple, en pausant pendant une période, le noeud égoïste pourra transmettre à son débit de génération de données une période sur trois au lieu d'une période sur deux. En augmentant le nombre de période pausée, l'avantage d'un noeud à rester égoïste s'en trouve de beaucoup amoindri car le temps où le noeud égoïste est puni est augmenté.

La figure 4.3 montre une diminution du nombre d'oscillation lorsque les noeuds collaboratifs effectuent une pause. Cette pause, d'exactement une période d'observation, est faite dans l'application de l'algorithme 1 lorsque les valeurs *CWMin* et *CWMax* atteignent 2. Le tableau 4.1 illustre la généralisation de cette méthode avec différents nombres de périodes d'observation pausés. Plus le nombre de périodes d'observation pausés est grand, plus petit est le débit de données mesuré chez le noeud égoïste.

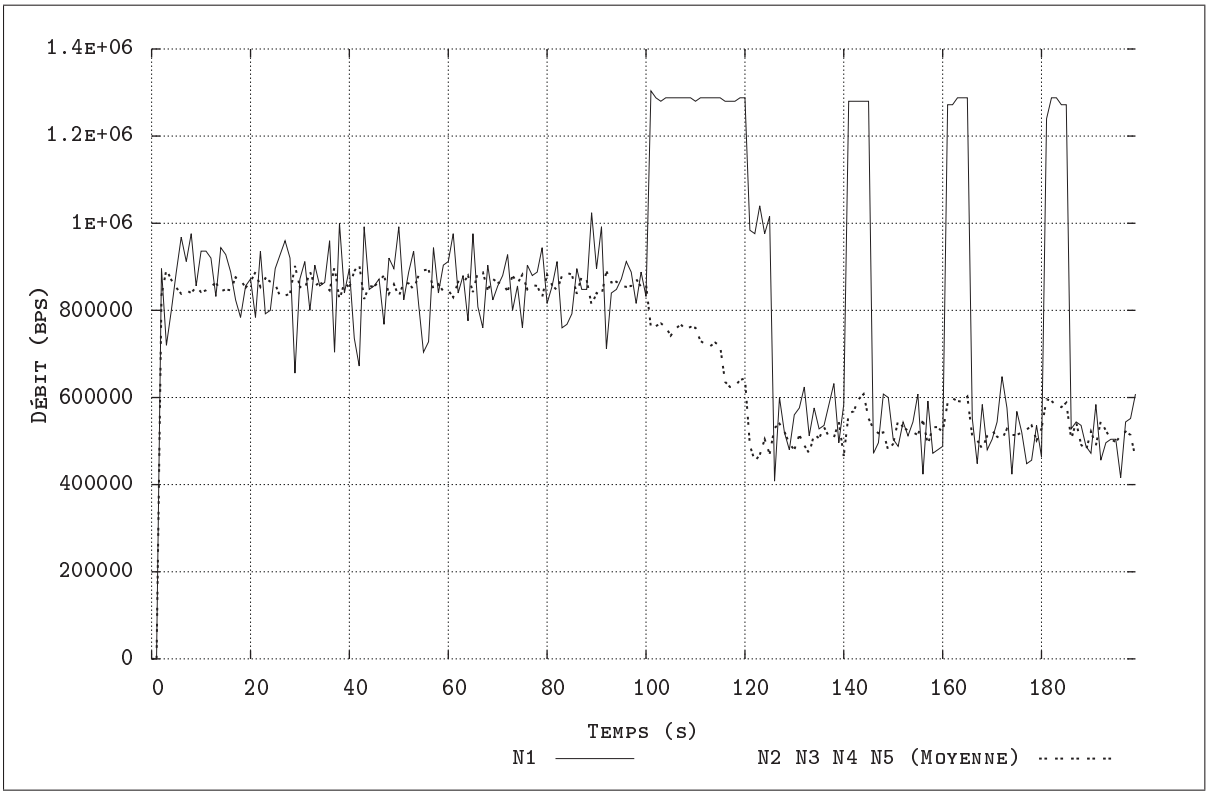

Figure 4.3 Changement avec pause dans les périodes d'observation

Tableau 4.1 Débits en fonction du nombre de périodes pausées

	Noeud égoïste		**Noeuds collaboratifs**	
Nombre de périodes pausées	**10-100s**	**100-500s**	**10-100s**	**100-500s**
0	862k	919k	861k	558k
1	862k	797k	861k	548k
2	862k	729k	861k	543k
5	862k	656k	861k	536k
10	862k	619k	861k	531k
20	862k	589k	861k	530k

4.2.1.3 Avec l'algorithme 2

Avec une configuration des paramètres identiques à celle de la section 4.2.1.1, la figure 4.4 est produite en remplaçant l'algorithme d'analyse des états 1 (algorithme 1) par l'algorithme

d'analyse des états 2 (algorithme 2). Parce que les états sont analysés différemment de l'algorithme 1, l'algorithme 2 ne produit pas d'oscillation dans les débits mesurés chez le noeud égoïste. En fait, l'algorithme 2 est conçu pour éviter l'augmentation automatique des valeurs *CWMin* et *CWMax* lorsque seulement des noeuds collaboratifs sont détectés dans le réseau, augmentation qui provoque des sauts dans les débits mesurés.

Les valeurs *CWMin* et *CWMax* restent minimales tant qu'un noeud égoïste reste égoïste. Si un noeud égoïste devient repenti, c'est-à-dire qu'il redevient collaboratif, ses valeurs *CWMin* et *CWMax* sont remises à celles spécifiées par la norme, soit 31 pour *CWMin* et 1023 pour *CWMax*. Pendant ce temps, les noeuds collaboratifs ayant appliqué l'algorithme 2 ont conservés leurs valeurs *CWMin* et *CWMax* à 2. À cet instant, au contraire d'une détection d'un noeud égoïste, les noeuds collaboratifs détecte le noeud repenti comme ayant transmis moins de paquets que les autres noeuds. Lorsque cette condition est rencontrée, les valeurs de *CWMin* et *CWMax* sont augmentées. La section 4.2.3 décrit un exemple de noeud repenti.

4.2.2 Brouillage des paquets du noeud égoïste

Pour contrer un noeud égoïste, le brouillage systématique de tous ses paquets transmis est une stratégie alternative. La figure 4.5 montre le résultat d'une simulation où le noeud N1 devient égoïste à t=100s. Lorsqu'un des noeuds collaboratifs choisi au hasard, soit N2, détecte la présence du noeud égoïste, il va brouiller systématiquement tous les paquets transmis par N1 pendant une période d'observation. À la période d'observation suivante, il observe la présence d'un noeud égoïste, d'où l'alternance entre débits faibles et débits élevés dans les mesures à chaque période d'observation.

L'application des algorithmes d'analyse des états provoquent, au lieu de la modification des valeurs de *CWMin* et *CWMax*, le brouillage des paquets provenant du noeud égoïste. Un seul

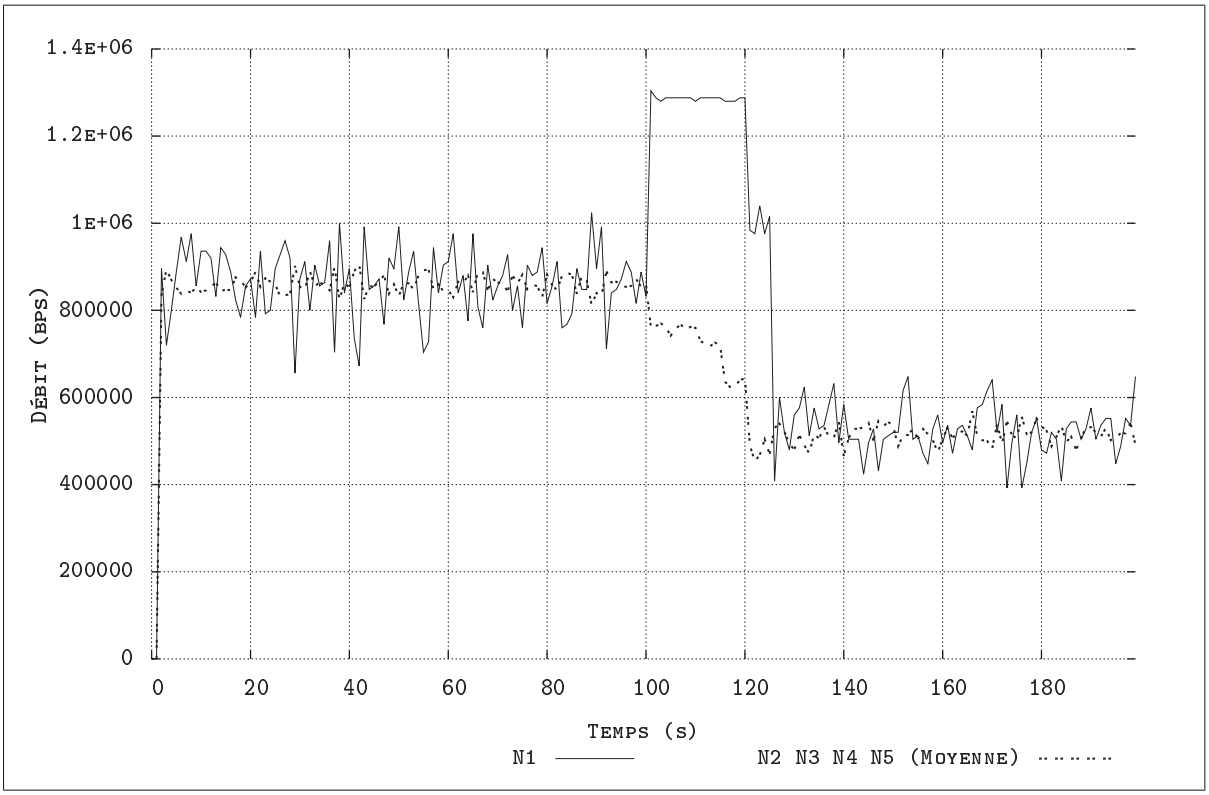

Figure 4.4 Changement de *CWMin* et *CWMax* - Algorithme 2

noeud collaboratif peut s'acquitter de la tâche de brouiller les paquets, car les résultats sont les mêmes si plusieurs noeuds brouillent les paquets du noeud égoïste.

Dans la configuration actuelle des paramètres du réseau, lorsque les paquets d'un noeud égoïste sont brouillés, les débits de données mesurés s'en trouvent de beaucoup diminués. En brouillant les paquets de données du noeud égoïste, les paquets *ACK* ne sont pas transmis. En ne recevant pas un paquet *ACK* pour un paquet de données transmis, une retransmission du paquet de données est faite. La valeur du paramètre *LongRetryLimit* étant fixée à 4 pour la transmission des grands paquets, alors un même paquet brouillé est retransmis à plusieurs reprises. La transmission successive de ces paquets, en conjonction avec des valeurs *CWMin* et *CWMax* petite, provoque la rupture des communications dans le réseau. Le noeud égoïste ne voit aucun

de ses paquets transmis correctement et les noeuds collaboratifs, voyant que le noeud égoïste accapare toujours le médium, ne peuvent jamais transmettre.

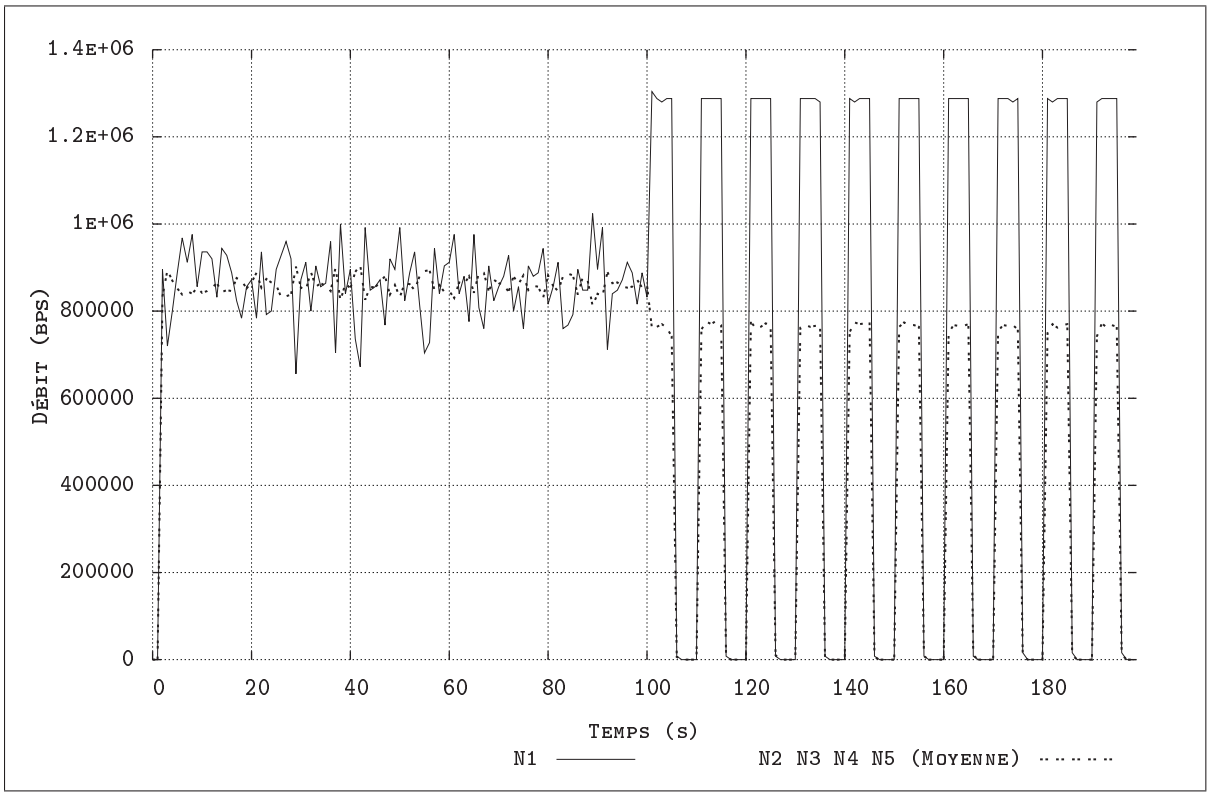

Figure 4.5 Brouillage des paquets provenant du noeud égoïste

4.2.2.1 Brouillage des paquets dans un réseau de 20 noeuds

En maintenant un niveau de saturation élevé, le brouillage des paquets d'un noeud égoïste est illustré à la figure 4.6. Le réseau simulé comprend un noeud qui devient égoïste à t=100s et 19 noeuds collaboratifs. Puisque la capacité du réseau est la même que lors des précédentes simulations, les débits de données générés des noeuds sont diminués pour permettre de garder un niveau de paquets perdus semblable aux simulations précédentes.

De la même façon que lors des précédentes simulations où les paquets du noeud égoïste sont brouillés, les débits de données mesurés lors des périodes de brouillage deviennent nuls chez le noeud égoïste. Par contre, la moyenne des débits des 19 autres noeuds diminue sans devenir nulle, contrairement à la simulation décrite à la section 4.2.2 où un nombre moins élevé de noeud est utilisé. Avec un nombre plus élevé de noeuds et un débit généré moins élevé par noeud, la fraction du temps total qu'occupe le noeud égoïste est plus petite que dans un réseau avec moins de noeuds. Malgré la retransmission des paquets brouillés chez le noeud égoïste, il reste une fraction assez élevée de temps libre, où le noeud égoïste n'occupe pas le médium, qui permet aux noeuds collaboratifs de transmettre.

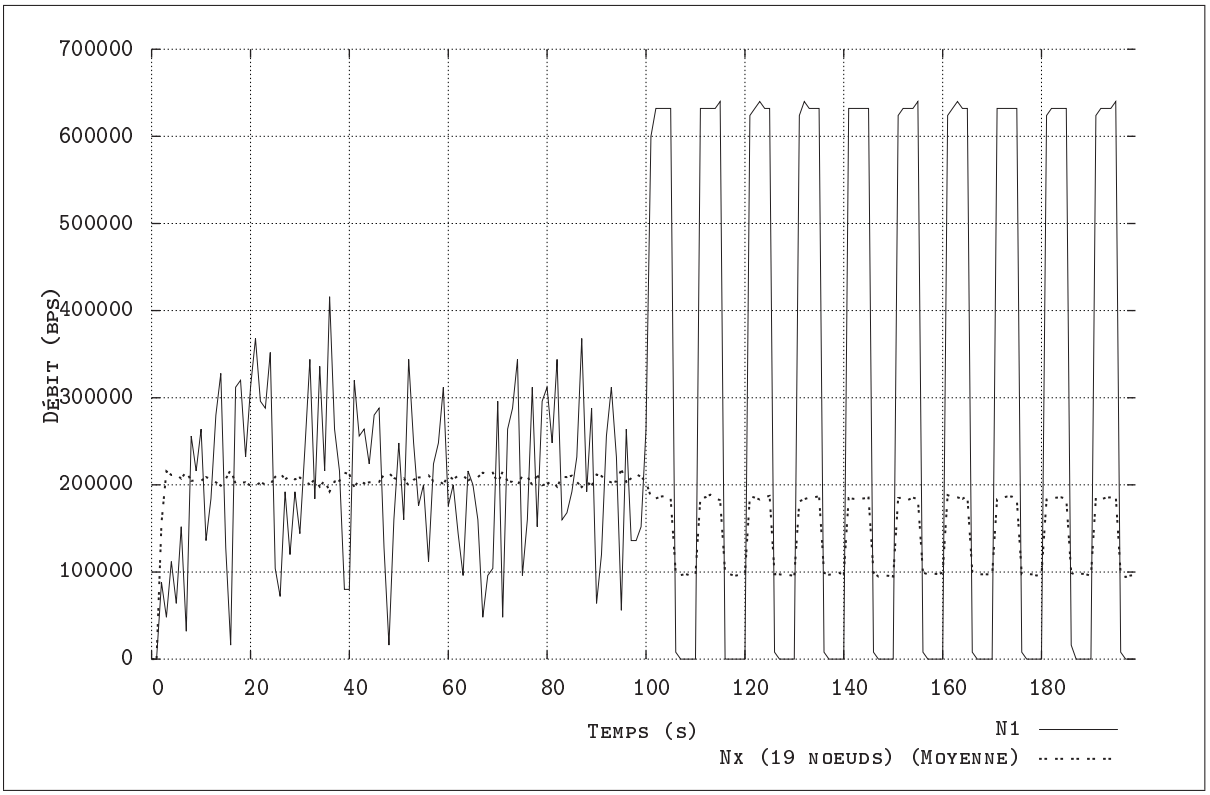

Figure 4.6 Brouillage des paquets avec 19 noeuds collaboratifs

4.2.3 Noeud égoïste repenti en saturation élevée

S'il est rationnel, un noeud égoïste peut redevenir collaboratif. Les raisons de se repentir peuvent provenir des conséquences de l'application des algorithmes présentés dans ce travail. Par exemple, un noeud égoïste détectant une déterioration de ses performances à cause de ses agissements égoïstes peut décider de redevenir collaboratif. Un tel noeud est dit repenti.

Lorsqu'un noeud est repenti, les noeuds appliquant les algorithmes doivent cesser de le considérer comme égoïste. Les noeuds collaboratifs doivent alors retrouver leurs paramètres *CWMin* et *CWMax* d'origine. Ainsi, tous les noeuds, même le noeud repenti, auront éventuellement les mêmes chances de transmettre.

La figure 4.7 montre l'example d'un noeud repenti. Une des courbes montre le noeud N1 qui est collaboratif jusqu'à t=100s, ensuite est égoïste pour les 100 secondes suivantes, puis il devient repenti à t=200s. L'autre courbe est la courbe de la moyenne des débits pour 4 noeuds collaboratifs appliquant l'algorithme d'analyse des états 1.

À t=200s, les noeuds appliquant l'algorithme 1, ne détectant plus la présence du noeud égoïste, augmentent graduellement leurs valeurs *CWMin* et *CWMax*. Pendant ce temps, le noeud repenti ayant des valeurs de *CWMin* et *CWMax* plus petites que les autres noeuds aura un débit mesuré plus petit que les autres noeuds durant quelques périodes d'observation. À t=215s, ces valeurs deviennent identiques pour tous les noeuds, même pour le noeud repenti. Si tous ont des valeurs égales, alors les débits mesurés pour tous les noeuds seront égaux.

4.2.4 Désynchronisation des périodes d'observation

La figure 4.8 illustre un exemple où les périodes d'observation sont désynchronisées. Dans les simulations précédentes, puisque tous les noeuds exécutent le même code, alors tous déclanchent la fin de leurs périodes en même temps, d'où la synchronisation des périodes d'ob-

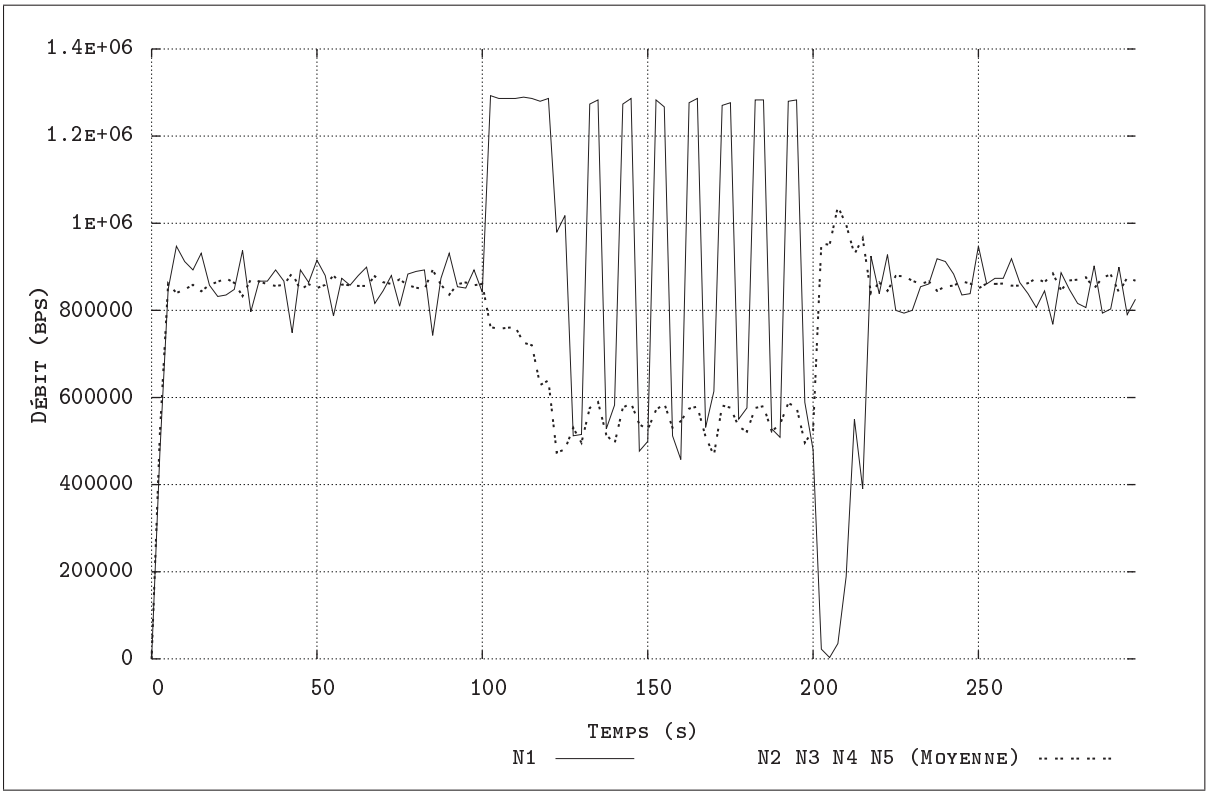

Figure 4.7 Noeud égoïste repenti - Algorithme 1

servation entre les noeuds. S'en suit la synchronisation des réactions contre un noeud égoïste détecté. Dans la simulation illustrée par cette figure, les périodes d'observation ont été volontairement désynchronisées entre les noeuds. Par contre, la durée des périodes d'observation est laissée identique pour tous les noeuds. Les paramètres utilisés sont exactement ceux utilisés lors de la simulation dans la section 4.2.1.1, hormis la désynchronisation des périodes d'observation.

4.2.5 Comparaison des stratégies en saturation forte

Le tableau 4.2 compile les résultats des différentes stratégies présentées précédemment en situation de saturation forte avec des réseaux de 5 noeuds. Le tableau 4.3 compile les mêmes

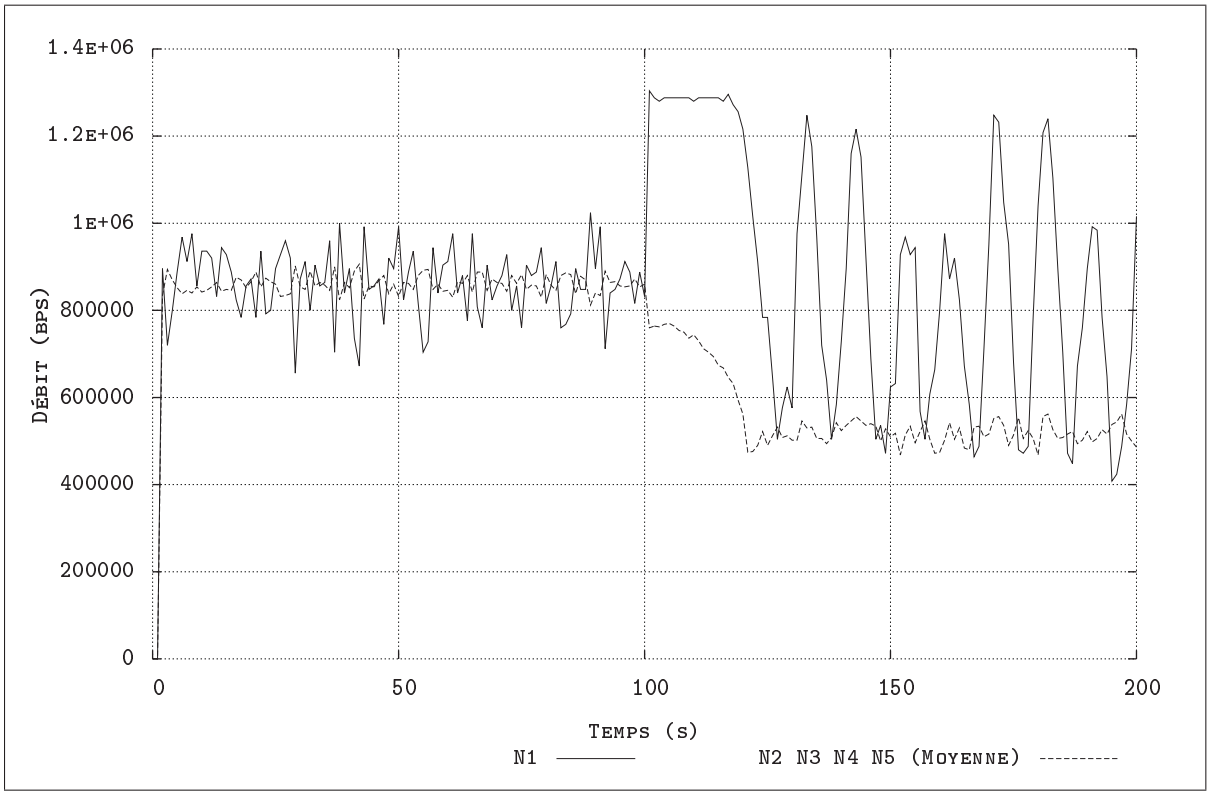

Figure 4.8 Désynchronisation des périodes d'observation

stratégies lorsque les réseaux ont 20 noeuds. Dans les deux cas, il y a un seul noeud qui devient égoïste dans le réseau, alors que tous les autres noeuds sont collaboratifs et appliquent les algorithmes présentés.

Dans tous les cas, réagir contre un noeud égoïste permet de diminuer son débit de données mesuré. De plus, augmenter le nombre de pauses dans l'application de l'algorithme quand les valeurs *CWMin* et *CWMax* deviennent minimales permet de nuire davantage au noeud égoïste. Dans le cas du brouillage des paquets dans les réseaux de 5 noeuds, la diminution du débit transmis pour les noeuds collaboratifs est très importante par rapport à la stratégie de changement des valeurs *CWMin* et *CWMax*. C'est le contraire dans le cas des réseaux à 20 noeuds, le

débit mesuré lorsque les paquets sont brouillés est supérieur à la stratégie modifiant les valeurs *CWMin* et *CWMax*.

L'analyse des résultats des tableaux 4.2 et 4.3 montre qu'avoir une seule stratégie contre un noeud égoïste n'est pas optimal. Dépendemment du nombre de noeuds dans le réseau, une stratégie doit être privélégiée par rapport à une autre pour éviter que les débits des noeuds collaboratifs descendent trop. Ces stratégies doivent en même temps nuire au maximum au noeud égoïste.

Tableau 4.2 Débits mesurés en saturation forte avec 5 noeuds

	Noeud égoïste		**Noeuds collaboratifs**	
Intervalle de temps (s)	**10-100**	**100-200**	**10-100**	**100-200**
Sans réaction	862k	1287k	861k	767k
Changement de *CWMin* et *CWMax* (Algo 1)	862k	972k	861k	578k
Changement avec pause pour 1 période	862k	874k	861k	574k
Changement avec pause pour 2 périodes	862k	817k	861k	564k
Changement de *CWMin* et *CWMax* (Algo 2)	862k	702k	861k	555k
Brouillage des paquets	862k	651k	861k	387k

Tableau 4.3 Débits mesurés en saturation forte avec 20 noeuds

	Noeud égoïste		**Noeuds collaboratifs**	
Intervalle de temps (s)	**10-100**	**100-200**	**10-100**	**100-200**
Sans réaction	213k	633k	206k	186k
Changement de *CWMin* et *CWMax* (Algo 1)	213k	374k	206k	109k
Changement avec pause pour 1 période	213k	321k	206k	106k
Changement avec pause pour 2 périodes	213k	277k	206k	104k
Changement de *CWMin* et *CWMax* (Algo 2)	213k	205k	206k	102k
Brouillage des paquets	213k	320k	206k	142k

4.3 Situation de saturation faible

En situation de saturation faible, le seuil doit être choisi avec plus de minutie. Dans cette situation, la différence entre le nombre de paquets envoyé par un noeud égoïste et un noeud collaboratif est plus faible qu'en situation de saturation forte. Choisir un seuil trop bas causerait une proportion de faux positifs trop élevée et choisir un seuil trop élevé causerait une proportion de faux négatifs trop élevée.

Pour cette raison, la constante multiplicative doit être utilisée pour déterminer le seuil dans le but de réduire le nombre d'erreurs commises dans la détection des noeuds égoïstes. Lorsque la constante multiplicative est utilisée, les périodes de recueil des données ne peuvent plus être déterminées selon une durée fixe. Ces périodes sont déterminées selon un nombre total de paquets *ACK* reçus.

Hormis l'utilisation de la constante multiplicative et du changement dans les périodes de recueil des données, les simulations et les algorithmes appliqués en situation de saturation faible sont les mêmes qu'en situation de saturation forte.

Le nombre de paquets pour avoir une faible saturation est défini à la section 3.2.5. Dans le cas présent, le débit est fixé pour obtenir en moyenne 10 paquets perdus par seconde lorsque tous les noeuds sont collaboratifs. Pour obtenir ce taux de perte dans un réseau de 5 noeuds avec des paquets de 1000 octets, le débit généré est fixé pour chaque noeud à 938 kbps alors qu'avec 20 noeuds, ce débit est fixé à 287 kbps.

Puisque le seuil est calculé avec la constante multiplicative, une période d'apprentissage doit être présente au début de chaque simulation. Avant qu'un des noeuds ne devienne égoïste, les données de 50 périodes d'observation sont recueillies pour obtenir des valeurs M_t. Puis, les calculs de μ et σ sont faits tel que décrit à la section 3.2.2. Le calcul du seuil s'en suit.

Dans un réseau de n noeuds, une période d'observation compte $250n$ paquets *ACK*. Avec ces paramètres, une période d'apprentissage dure un peu plus de 120 secondes.

Les simulations en situation de saturation faible ont toutes le même scénario. Dans un réseau de 5 noeuds, la période d'apprentissage est complétée avant t=150s, puis à cet instant, le noeud N1 devient égoïste. Dans un réseau de 20 noeuds, le noeud N1 devient égoïste à t=500s car la durée de la période d'apprentissage est plus longue.

4.3.1 Comparaison des stratégies en saturation faible

Avec des scénarios de simulation similaires à ceux décrits à la section 4.2, les tableaux 4.4 et 4.5 compilent des résultats avec des réseaux de 5 et 20 noeuds. Les allures des courbes obtenus lors de ces simulations sont semblables à celles obtenues lors des simulations en situation de saturation forte.

Tableau 4.4 Débits mesurés en saturation faible avec 5 noeuds

	Noeud égoïste		**Noeuds collaboratifs**	
Intervalle de temps (s)	**10-150**	**150-300**	**10-150**	**150-300**
Sans réaction	859k	938k	857k	845k
Changement de *CWMin* et *CWMax* (Algo 1)	859k	716k	857k	607k
Changement avec pause pour 1 période	859k	655k	857k	583k
Changement avec pause pour 2 périodes	859k	629k	857k	569k
Changement de *CWMin* et *CWMax* (Algo 2)	859k	601k	857k	569k
Brouillage des paquets	859k	280k	857k	218k

Dans tous les cas, les noeuds appliquant les algorithmes réagissant contre un noeud égoïste provoquent une perte de débit chez ce dernier. Dans le cas du réseau avec 5 noeuds, la stratégie du brouillage des paquets est particulièrement nuisible pour le noeud égoïste, mais aussi pour les noeuds collaboratifs. Par contre, les pertes de débit sont relativement limitée chez les noeuds

Tableau 4.5 Débits mesurés en saturation faible avec 20 noeuds

	Noeud égoïste		**Noeuds collaboratifs**	
Intervalle de temps(s)	**10-500**	**500-1000**	**10-500**	**500-1000**
Sans réaction	209k	287k	206k	203k
Changement de *CWMin* et *CWMax* (Algo 1)	209k	189k	206k	123k
Changement avec pause pour 1 période	209k	134k	206k	96k
Changement avec pause pour 2 périodes	209k	123k	206k	93k
Changement de *CWMin* et *CWMax* (Algo 2)	209k	105k	206k	87k
Brouillage des paquets	209k	123k	206k	179k

collaboratifs dans les réseaux de 20 noeuds. La stratégie de brouillage des paquets doit donc être privilégiée lorsque le nombre de noeuds est élevé. Si le nombre de noeuds dans le réseau est petit, alors réduire les valeurs de *CWMin* et *CWMax* est la meilleure stratégie.

4.3.2 Noeud égoïste repenti en saturation faible

L'algorithme 1 a une lacune lorsque qu'il est utilisé en situation de saturation faible, lorsqu'un noeud se repenti et lorsqu'il y a peu de noeuds dans le réseau. Les seuils de détection étant plus sensible qu'en situation de saturation forte, un noeud qui se repenti provoque une réaction non souhaitée. Lorsqu'un noeud se repenti, son nombre de paquets transmis devient momentanément très petit car les noeuds collaboratifs, appliquant les algorithmes, conservent leurs valeurs *CWMin* et *CWMax* à 2 alors que le noeud repenti remet ces valeurs à 31 et 1023. Parce que le nombre de paquets transmis par les noeuds collaboratifs dépasse le seuil dans cette situation, alors les noeuds collaboratifs se détecteront mutuellement comme égoïstes. Ceci fait que les valeurs *CWMin* et *CWMax* resteront à 2 suite à des réactions négatives en chaîne et donc, le noeud repenti conservera un débit très bas. La figure 4.9 illustre ce phénomène. La courbe du noeud N1 montre que ce dernier conserve un débit très bas même s'il s'est repenti.

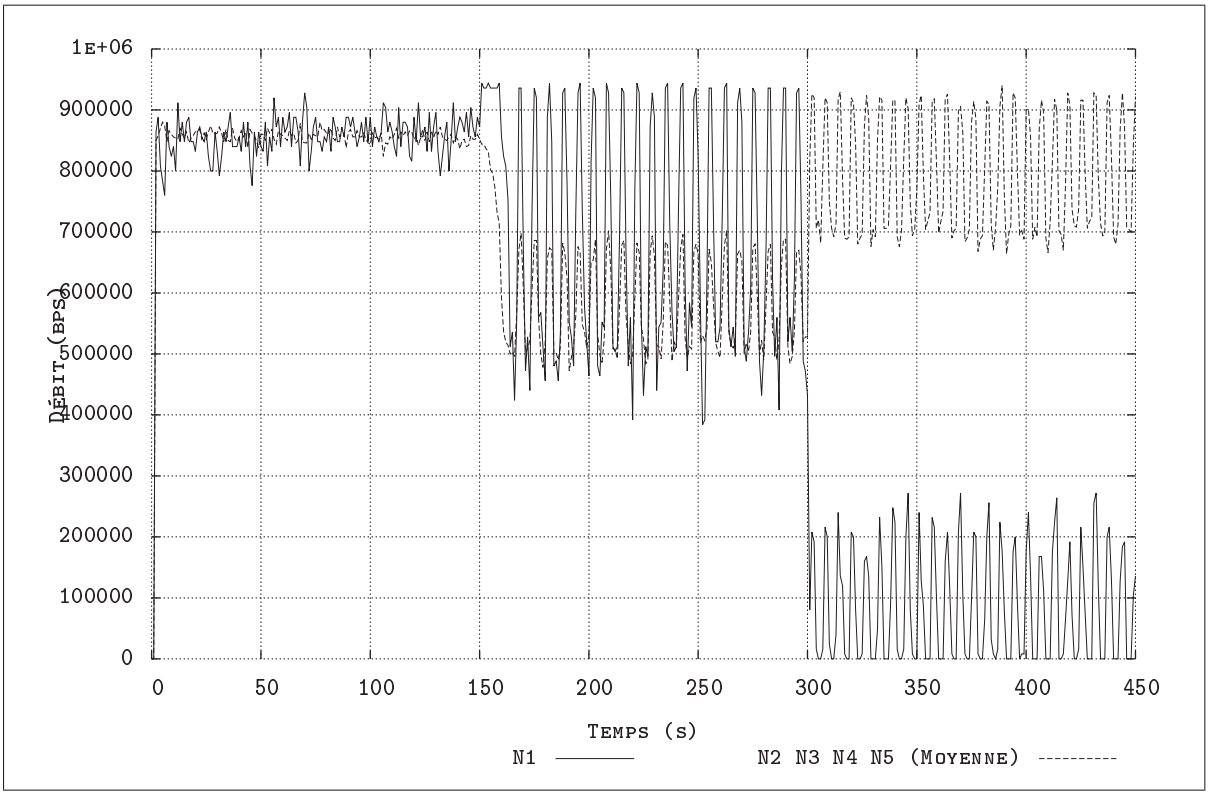

Figure 4.9 Lacune de l'algorithme d'analyse des états 1

L'application de l'algorithme d'analyse des états 3 corrige ce problème. Si trop de noeuds sont détectés comme égoïstes, malgré qu'en réalité il y en a un seul noeud vraiment égoïste, alors cet algorithme suppose que ceci a été provoqué par un noeud repenti. Dans ce cas, les valeurs *CWMin* et *CWMax* ne seront pas diminuées comme dans le cas de l'algorithme 1. La figure 4.10 illuste l'application de l'algorithme 3 où le phénomène décrit précédemment est corrigé.

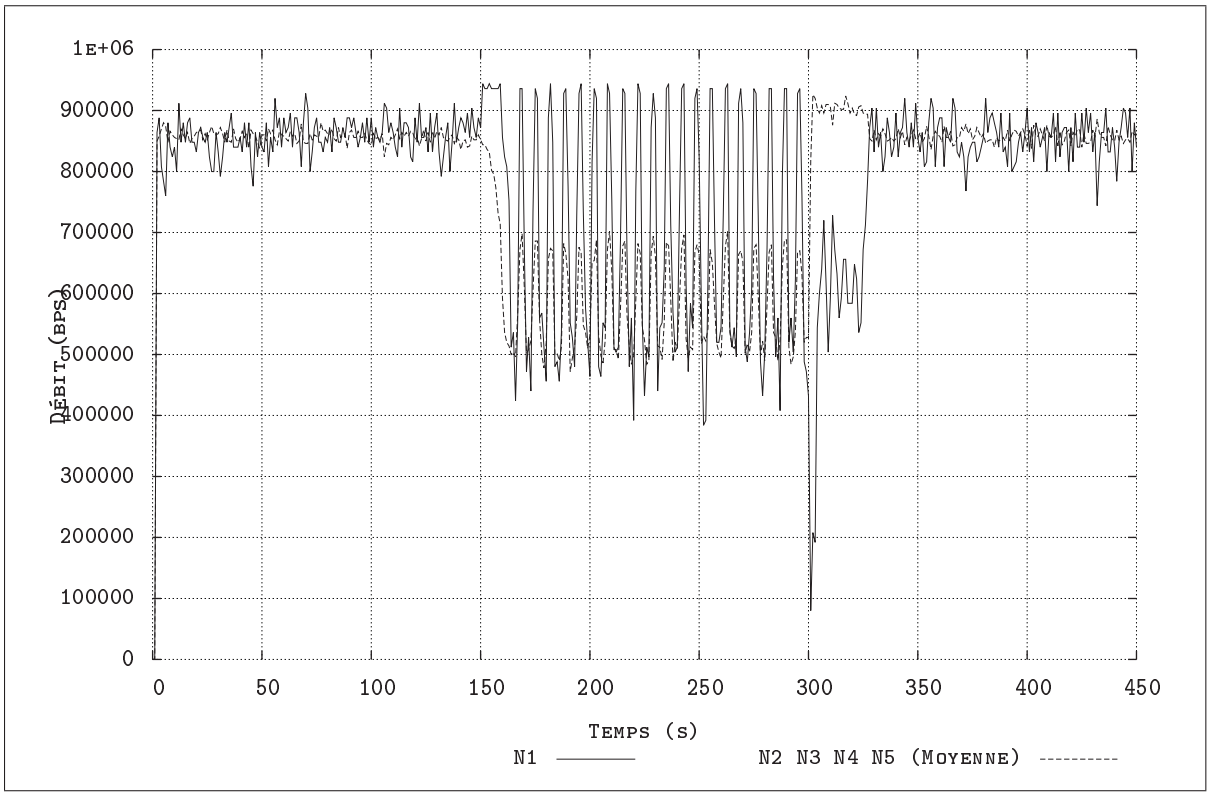

Figure 4.10 Correction par l'algorithme d'analyse des états 3

CHAPITRE 5

LIMITATIONS DE L'ÉTUDE ET FUTURS TRAVAUX

Les algorithmes présentés dans le cadre de ce travail ont été testés dans un environnement contrôlé. Plus particulièrement, utiliser un simulateur permet de fixer la plupart des paramètres définissant un réseau ainsi que des noeuds qui le compose. Malgré que le simulateur essaie d'imiter au maximum un réseau réel, tester les algorithmes dans un réseau réel pourrait assurer la validité externe des algorithmes. Rien par contre n'indique que les algorithmes présentés dans ce travail seraient totalement inadéquats dans un réseau réel.

Beaucoup de variables ne peuvent être contrôlés dans un réseau réel. Par exemple, plus d'un noeud égoïste pourrait se retrouver dans un réseau réel. Or, les algorithmes présentés dans ce travail suppose la présence que d'un seul noeud égoïste à la fois. De plus, le nombre de noeuds peut être variable dans le temps, c'est-à-dire que dans un réseau en mode infrastructure, les noeuds s'associent et se dissocient à différents moments. Une adaptation des algorithmes avec un nombre variable de noeuds dans le temps pourrait être l'objet d'une phase ultérieure de ce travail.

Les algorithmes ont été conçus dans l'optique que les noeuds génèrent du trafic à débit constant avec des paquets de type *UDP* exclusivement. Or dans un réseau réel, des noeuds vont parfois générer du trafic en rafales avec différents types de paquets. Pour augmenter la polyvalence des algorithmes, une adaptation des algorithmes à différents types de trafic doit être effectuée. De plus, les algorithmes supposent que seul du trafic en amont, c'est-à-dire des noeuds vers le point d'accès, est généré. C'est le seul type de trafic que les noeuds peuvent contrôler. Toutefois, il faudrait vérifier la réaction des algorithmes lorsque du trafic en aval est mélangé au trafic en amont, ce qui pourrait augmenter davantage la validité externe des algorithmes.

L'amendement IEEE 802.11e ajoute la qualité de service à la norme 802.11. Plus précisément, des classes de trafic de données sont définis par cet amendement. Aux différentes classes de trafic, différentes valeurs par défaut de *CWMin* et *CWMax* leur sont attribuées. Sans cet amendement, tous les paquets des différentes classes de trafic définissent leur fenêtres de contention avec les mêmes paramètres, soit 31 pour *CWMin* et 1023 pour *CWMax*. Par exemple en appliquant l'amendement, les paquets appartenant à la classe de trafic voix *AC_VO* a la valeur par défaut pour *CWMin* de 7 et la valeur par défaut pour *CWMax* de 15 (IEEE, 2005). Si les algorithmes fonctionnent dans un environnement où différentes classes de trafic sont présentes, alors une adaptation des algorithmes doit être effectuée. Autrement, le noeud générant du trafic voix serait considéré injustement comme égoïste.

En analysant les débits observés dans les tableaux 4.2, 4.3, 4.4 et 4.5, il arrive, lorsqu'un noeud égoïste est détecté, que réagir en modifiant les fenêtres de contention soit meilleur que de brouiller les paquets provenant du noeud égoïste. La situation inverse peut également exister. C'est que la meilleure réaction dépend du niveau de saturation et du nombre de noeuds dans le réseau. Caractériser avec plus d'exactitude, c'est-à-dire des nombres de noeuds précis et des niveaux de saturation précis, la meilleure réaction à adopter selon un éventail plus grand de situations pourrait être un objectif à atteindre dans un travail futur.

CONCLUSION

Il a été vu que la présence d'un noeud égoïste dans un réseau est nuisible pour l'ensemble des autres noeuds d'un réseau. De cette constatation, des algorithmes ont été développés dans le but de réagir à la présence d'un noeud égoïste. Cette réaction nuisible pour le noeud égoïste vise à le convaincre de se repentir, c'est-à-dire à remettre les paramètres définis par la norme IEEE 802.11. Dans le cadre de ce travail, ces paramètres sont ceux définissant la fenêtre de contention, c'est-à-dire *CWMin* et *CWMax*.

Si la présence d'un noeud égoïste entraîne les autres noeuds collaboratifs à devenir eux-mêmes égoïstes, cela n'apporte généralement pas de bons résultats. Il a été vu que si tous les noeuds devenaient égoïstes sur de longues périodes de temps, alors les débits mesurés seraient plus bas que si tous étaient collaboratifs. Pour cette raison, les réactions contre un noeud égoïste doivent être de courtes durées et doivent donner une chance à un noeud repenti de transmettre ses données.

Les seuils de détection pour un noeud égoïste sont difficiles à établir, car lorsqu'un paramètre réseau est modifié, ce seuil est modifié également. L'introduction d'une méthode de calcul des seuils, basée sur une constante multiplicative, permet d'éviter de chercher par tâtonnement un seuil à chaque fois. Plus particulièrement, la détection d'un noeud égoïste est basée sur le nombre de paquets *ACK* reçus pour chacun des noeuds. Après une période d'apprentissage composée de plusieurs périodes d'observation, des statistiques sont calculées sur ces nombres de paquets reçus. Un seuil est calculé par la suite pour détecter la présence des noeuds égoïstes. La précision de ce seuil dépend du nombre de statistiques utilisées, d'où la nécessité d'avoir un nombre suffisant de périodes d'observation dans la période d'apprentissage. Par la comparaison du seuil avec les différents compteurs de paquets *ACK*, des états sont attribués aux différents noeuds à chaque période d'observation.

Différents algorithmes d'analyse des états ont été testés. Par exemple, l'algorithme 1 ne donne aucune chance à un noeud repenti lorsque la saturation du réseau est faible et que le nombre de noeuds est petit, d'où l'application de l'algorithme 3 dans cette situation. Également, l'application sans modification de algorithme 1 provoque une oscillation dans les débits de données mesurés, d'où l'introduction de pause dans l'application de l'algorithme lorsque les valeurs *CWMin* et *CWMax* sont minimales. Pour éliminer complètement les oscillations, l'algorithme 2 à été développé.

Tel que constaté dans le développement des algorithmes, la détection des noeuds égoïstes dans un réseau n'est pas une chose simple, même dans un environnement simulé où la plupart des paramètres sont contrôlés.

RÉFÉRENCES BIBLIOGRAPHIQUES

J. BELLARDO et S. SAVAGE : 802.11 denial-of-service attacks : real vulnerabilities and practical solutions. *In Proceedings of the 12th conference on USENIX Security Symposium*, p. 2, 2003.

G. BIANCHI : Performance analysis of the IEEE 802.11 distributed coordination function. Volume 18, pp. 535–547, 2000.

S. BOYER, J.-M. ROBERT, C. ROUSSEAU et H. OTROK : A Novel Reputation-Based Tit-for-Tat Strategy for IEEE 802.11 CSMA/CA Protocol. *In Proceedings of the 9th Annual IEEE Consumer Communications and Networking Conference - Security and Content Protection*, 2012.

A.A. CARDENAS, S. RADOSAVAC et J.S. BARAS : Evaluation of Detection Algorithms for MAC Layer Misbehavior : Theory and Experiments. Volume 17, pp. 605–617, 2009.

L. GIARRÈ, G. NEGLIA et I. TINNIRELLO : The role of the Access Point in Wi-Fi networks with selfish nodes. *In Proceedings of the International Conference on Game Theory for Networks*, pp. 631–637, 2009.

A. GOKHALE : *Introduction to Telecommunications*. Delmar Cengage Learning, 2004.

L. GUANG, C.M. ASSI et A. BENSLIMANE : Enhancing IEEE 802.11 Random Backoff in Selfish Environments. Volume 57, pp. 1806–1822, 2008.

A. HAMIEH, J. BEN-OTHMAN, A. GUEROUI et F. NAIT-ABDESSELAM : Detecting Greedy Behaviors by Linear Regression in Wireless Ad Hoc Networks. *In Proceedings of the IEEE International Conference on Communications*, pp. 1–6, 2009.

IEEE : *IEEE Standard for Information Technology – Telecommunications and Information Exchange Between Systems – Local and Metropolitan Area Networks – Specific Requirements Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*. 1999.

IEEE : *Supplement to IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications : Higher-Speed Physical Layer Extension in the 2.4 GHz Band*. 2000.

IEEE : *IEEE Standard for Information technology – Telecommunications and information exchange between systems–Local and metropolitan area networks – Specific requirements Part 11 : Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 8 : Medium Access Control (MAC) Quality of Service Enhancements*. 2005.

T. ISSARIYAKUL et E. HOSSAIN : *Introduction to Network Simulator NS2*. Springer Science, 2009.

A. JARDOSH, K. RAMACHANDRAN, K. ALMEROTH et E. BELDING-ROYER : Understanding Congestion in IEEE 802.11b Wireless Networks. *In Proceedings of the 2005 Internet Measurement Conference*, pp. 279–292, 2005.

C. E. KOKSAL, H. KASSAB et H. BALAKRISHNAN : An Analysis of Short-Term Fairness in Wireless Media Access Protocols. *In Proceedings of ACM Sigmetrics*, pp. 118–119, 2000.

P. KYASANUR et N.H. VAIDYA : Selfish MAC layer misbehavior in wireless networks. Volume 4, pp. 502–516, 2005.

C. LIU, M. H. MACGREGOR et J. HARMS : Optimal Control of Spatial, Temporal and Bandwidth Contention in Wireless Ad Hoc Networks. *In Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, pp. 263–269, 2009.

A. LOPEZ TOLEDO et X. WANG : A Robust Kolmogorov-Smirnov Detector for Misbehavior in IEEE 802.11 DCF. *In Proceedings of the IEEE International Conference on Communications*, pp. 1564–1569, 2007.

S. RADOSAVAC, J.S. BARAS et I. KOUTSOPOULOS : A framework for MAC protocol misbehavior detection in wireless networks. *In Proceedings of the 4th ACM workshop on Wireless security*, pp. 33–42, 2005.

M. RAYA, I. AAD, J.-P. HUBAUX et A. EL FAWAL : DOMINO : Detecting MAC layer greedy behavior in IEEE 802.11 hotspots. Volume 5, pp. 1691–1705, 2006.

Y. RONG, S.-K. LEE et H.-A. CHOI : Detecting Stations Cheating on Backoff Rules in 802.11 Networks Using Sequential Analysis. *In Proceedings of the 25th IEEE International Conference on Computer Communications.*, pp. 1–13, 2006.

P. SERRANO, A. BANCHS, V. TARGON et J. KUKIELKA : Detecting selfish configurations in 802.11 WLANs. Volume 14, pp. 142–144, 2010.

V. VISOOTTIVISETH, A. TRUNGANONT et S. SIWAMOGSATHAM : Adaptive bandwidth allocation for per-station fairness on wireless access router. *In Proceedings of the International Symposium on Communications and Information Technologies*, pp. 238–243, 2010.

B. ZHOU, A. MARSHALL, W. ZHOU et K. YANG : A Random Packet Destruction DoS Attack for Wireless Networks. *In Proceedings of the IEEE International Conference on Communications*, pp. 1658–1662, 2008.

M. ČAGALJ, S. GANERIWAL, I. AAD et J.-P. HUBAUX : On selfish behavior in CSMA/CA networks. *In Proceedings of the IEEE Infocom*, 2005.