TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

AFH	Adaptive Frequency Hopping
ARQ	Automatic Repeat-reQuest
ASK	Amplitude-shift keying
AWGN	Additive white Gaussian noise
BE	Best Effort
BER	Bit Error Rate
BPSK	Binary Phase Shift Keying
CDMA	Code Division Multiple Access
CLKN	Clock Native
CPFSK	Continuous-phase frequency-shift keying
CRC	Contrôle par Redondance Cyclique
EDF	Earliest Deadline First
FEC	Forward Error Correction
FEQ	Fair and Effective Queueing
FFH	Fast Frequency Hopping
FH	Frequency Hopping
FHSS	Frequency Hopping Spread Spectrum
FIFO	First In First Out
FSK	Frequency-Shift Keying
FTP	File Transfer Protocol
HCR	High Capacity Radio
HPF	Highest Priority First
ISM	Industrial Scientific and Medical
MAC	Media Access Control
MFSK	Multiple Frequency-Shift Keying
MHz	MegaHertz
MLLO-FH	Multi-Level Locally Orthogonal - Frequency Hopping
MPEG	Moving Pictures Experts Group
MRR	Minimum Reserved Rate

NAK	Negative-AcKnowledge
nrtPS	non real-time Polling Services
OFDM	Orthogonal Frequency-Division Multiple
OFDMA	Orthogonal Frequency-Division Multiple Access
OQPSK	Offset quadrature phase-shift keying
PER	Paquet error rate
PS	Phase-shift keying
QoS	Qualité of Service
QPSK	Quadrature Phase Shift Keyed
RR	Round Robin
rtPS	real-time Polling Services
SCDC	Service Class Downlink Scheduling
SFH	Slow Frequency Hopping
SJF	Shortest Job First
SRT	Shortest Remaining Time
UBAFH	Utility Based Adaptive Frequency Hopping
UGS	Unsolicited Grant Service
WFQ	Weighted Fair Queuing
WiMAX	Worldwide Interoperability for Microwave Access
WLAN	Wireless Local Area Network
WRR	Weighted Round Robin

INTRODUCTION

Les réseaux ad hoc se présentent comme une solution de choix, par rapport aux réseaux classiques, lorsqu'il est question d'installer, dans de brefs délais, un réseau sans fil peu couteux et temporaire. Notamment, ils ne demandent pas d'infrastructures fixes pour fonctionner. Les nœuds mobiles gèrent leurs interconnexions de façon dynamique même dans des configurations géographiques aléatoires et changeantes dans le temps. Les réseaux ad hoc sont donc d'un grand intérêt pour les réseaux militaires fonctionnant dans des zones hostiles. L'intérêt que suscitent les réseaux ad hoc pour le domaine militaire donne lieu à plusieurs défis pour les chercheurs.

Dans les réseaux militaires, la protection de l'information et l'atteinte d´une qualité de service, "Quality of Service", QoS, acceptable demeurent toujours des défis importants (Lagoutte, 2000). Les réseaux militaires sont appelés à fonctionner dans des zones hostiles, faisant ainsi face à de diverses attaques telles que l'interception et le brouillage de l'information. En plus de ces attaques, l'environnement de fonctionnement n'est pas connu d'avance. La question se pose donc ainsi : comment sécuriser l'information tout en offrant une bonne QoS dans les réseaux ad hoc? Plusieurs approches proposées dans la littérature utilisent des techniques d'ordonnancement à sauts de fréquence dans les communications en mode ad hoc. Ces techniques semblent être prometteuses.

La technique de saut de fréquence, "frequency hopping", consiste à changer de fréquence de porteuse plusieurs fois, suivant une séquence connue de l'émetteur et du récepteur, au cours d'une même communication. Ces changements de fréquence peuvent se produire sur une large bande. L´ordonnancement, "scheduling", dans ce contexte consiste à allouer des tranches de temps-fréquence aux multiples usagers de sorte que l'utilisation du médium soit grande. Il faut donc brouiller plusieurs fréquences pour espérer brouiller une communication, ce qui est assez couteux. Pour intercepter l'information, il faut connaitre l'ordre des sauts. Ce qui n'est pas évident tant qu'il n'est pas rendu public. En améliorant ces techniques, nous pouvons avoir de meilleures latences et une meilleure QoS.

Ce document présente notre approche pour améliorer la QoS, la latence et le niveau de sécurité dans les réseaux ad hoc. Nous proposons une méthode d'ordonnancement qui utilise des sauts de fréquence de taille et d'intervalles de temps variables.

Le réseau considéré est constitué de nœuds capables de communiquer en mode "full duplex", soit des communications bidirectionnelles simultanées, et d'effectuer des sauts de fréquence de façon rapide.

Problématique

Dans les réseaux mobiles ad hoc, la garantie d'un bon débit réel et de faible latence reste un défi (Mo et al., 2010). Ce défit est encore plus important lorsque les réseaux ad hoc doivent être utilisés dans des applications militaires. Les applications militaires opèrent entre autres dans des zones hostiles où elles sont souvent menacées par la guerre électronique. Les tentatives de brouillage sont très fréquentes, et peuvent conduire à des erreurs binaires importantes. Si les bits erronés sont irrécupérables, des tentatives de retransmissions s'imposent, ce qui augmente la latence qui à la base était un défi pour les réseaux ad hoc.

Pour résoudre ces problèmes, plusieurs travaux proposent des approches différentes. Pour diminuer le taux d'erreur binaire, la littérature présente diverses méthodes à sauts de fréquence parmi lesquelles le Frequency Hopping Spread Spectrum (FHSS). Ces méthodes, conçues à la base pour empêcher l'écoute distante, ont connu beaucoup d'améliorations visant à les rendre plus robustes face aux guerres électroniques. Parmi les améliorations, nous pouvons citer l'Adaptive Frequency Hopping (AFH), qui en plus de faire des sauts de fréquence, évite les canaux brouillés par d'autres systèmes utilisant la même bande. Grâce à l'AFH, le taux d'erreur binaire est réduit par rapport au FHSS. Mais à part les systèmes utilisant la même bande que l'AFH, des interférences temporaires sont présentes. Alors, l'AFH fait constamment un balayage de ses fréquences, ce qui alourdit le réseau. Pour pallier à ce problème et aussi éviter une utilisation inutile de la ressource dans l'envoi des paquets de contrôle, l'UBAFH, «Utility Based Adaptive Frequency Hopping », est proposée. Elle se base sur l'historique des pertes binaires dans chaque canal pour déterminer s'il faut transmettre dans ce canal ou pas. Cette solution offre un faible taux d'erreur binaire et une

meilleure latence par rapport à l'AFH. Même avec ces améliorations, les techniques à sauts de fréquence restent vulnérables face aux brouilleurs redoutés comme par exemple le brouilleur à large bande et le brouilleur suiveur.

D'autres auteurs résolvent le problème autrement et proposent une solution hybride des sauts de fréquence avec des étalements par code, FH-CDMA. Ils espèrent tirer profit de la robustesse des étalements par code, dans des environnements très bruités. Mais cette approche ne garantit pas une bonne QoS pour les flux temps réel. Car, améliorer la latence ne dépend pas seulement de l'amélioration du BER, mais aussi du choix d'un bon algorithme d'ordonnancement.

Dans la plupart des techniques citées ci-dessus, les ressources sont déjà subdivisées et leur allocation nécessite un ordonnancement efficace. La plupart des techniques d'étalement à sauts de fréquence présentées utilisent les algorithmes d'ordonnancements standards pour faire l'allocation des ressources. Parmi ces algorithmes d'ordonnancement, nous pouvons citer deux qui sont couramment utilisés : Weighted Round Robin, WRR, et Weighted fair queuing, WFQ (Tata, 2009). Ces algorithmes, pour assurer une bonne qualité de service aux différents flux en attente, utilisent des mécanismes de calcul de priorité. Ces priorités sont parfois dynamiques car, dépendent de l'état de la file en attente. Le fait que les ressources ne soient pas subdivisées de façon synchrone avec les paramètres de priorité peut causer des pertes de ressources ou des pertes de paquets. Par exemple lors de la mise en trame d'un paquet, si la longueur de ce dernier est insuffisante pour remplir la trame, le système peut attendre la prochaine rafale. Si le délai d'attente de la rafale dépasse le temps d'attente autorisé pour ce paquet, soit il est envoyé dans une trame incomplète, ce qui cause une perte de la ressource, soit il est perdu.

Un autre problème rencontré dans les solutions existantes est la sous utilisation de la ressource. En effet, lorsqu'un nœud libère son canal et que le nombre de canaux disponibles est plus grand que le nombre de file en attente, la ressource libérée est inusitée. Pendant ce temps, des paquets sont perdus dans des files, soit parce que ces dernières sont pleines, soit parce que le délai d'attente est dépassé. Ce problème est généralement présent dans les files

de faibles priorités. Des travaux comme ceux de (Tata, 2009) proposent un algorithme basé sur la courtoisie des files prioritaires pour limiter les paquets perdus dans les files de faibles priorités. Dans ces travaux, les pertes ont été réduites sans pour autant être nulles. L'amélioration de la latence reste alors un défi. Notre but dans ce travail sera alors d'améliorer la latence du système.

Objectif du mémoire

Notre recherche vise à développer une nouvelle méthode d'ordonnancement offrant de faibles latences par rapport aux solutions existantes. Pour atteindre cet objectif, nous pensons améliorer l'utilisation de la ressource en limitant le problème du remplissage, et en utilisant la ressource libérée pour les files encore en service. Nous essayons tout au moins de maintenir, ou au mieux d'améliorer, le taux d'erreur binaire des solutions existantes, ce qui diminuera le taux de retransmission des paquets. Ensuite l'algorithme d'ordonnancement proposé doit être assez flexible pour prendre en compte les pertes de ressources enregistrées dans les solutions précédentes. Pour cela, nous avons émis quelques hypothèses de départ.

Hypothèses de départ

Pour diminuer le taux d'erreur binaire, l'utilisation des sauts de fréquence semble prometteuse car, avec le principe de l'AFH, on évite de faire des sauts sur les canaux bruités. Cette approche offre une flexibilité dans l'utilisation de la bande. De plus, grâce aux sauts de fréquence de courtes durées, les interférences multi trajets sont évitées. En considérant que les interférences peuvent cibler des fréquences spécifiques pour y concentrer toute leur puissance, et que le brouilleur suiveur a besoin de détecter le signal, nous prévoyons qu'un étalement à sauts de fréquence incluant des débits variables combinés à des puissances d'émissions variables et une disposition aléatoire sur l'échelle des fréquences rendra le brouillage plus difficile. Ainsi, la probabilité d'erreur peut être améliorée.

Pour maximiser l'utilisation de la bande passante et ainsi améliorer la latence, nous proposons que la durée des sauts soit variable, tout en restant assez petite. Cette variation

permettra d'utiliser de façon plus efficace les priorités qui sont calculées par l'algorithme d'ordonnancement choisi pour chaque trafic.

En faisant les variations de débit et d'intervalle de temps de façon contrôlé, et en fonction des priorités calculées, la latence devrait être améliorée. L'étude de ces hypothèses et les résultats des simulations seront présentés suivant le plan ci-après.

Plan du rapport

Ce travail sera présenté en quatre chapitres. Dans un premier chapitre, nous présenterons les méthodes d'ordonnancement à sauts de fréquence traditionnelle et leurs vulnérabilités. Nous y présenterons aussi les techniques d'ordonnancement les plus utilisées. Le second chapitre comportera l'étude théorique des différents éléments à prendre en compte pour faire un ordonnancement. Les hypothèses y seront également traitées. Après avoir traité les hypothèses, nous présenterons nos approches dans le chapitre 3. Les simulations de nos solutions seront présentées dans le chapitre 4. Dans ce même chapitre 4, toutes les approches seront comparées grâce à des métriques de comparaison qui seront présentées. Enfin, en conclusion, nous ferions un récapitulatif des objectifs atteints, les limites rencontrées, ainsi que les défis pour les futurs travaux.

Contribution

Dans ce mémoire, nous proposons un ordonnancement à sauts de fréquence permettant d'avoir une latence de service plus faible que celle du FHSS. L'élément nouveau est l'aspect variable que nous proposons dans les sauts de fréquence. Grâce à une flexibilité dans la subdivision des ressources existantes, en termes de bande passante, nous offrons aux différents nœuds un ordonnancement qui se rapproche mieux de la demande de leurs files et nous limitons ainsi les problèmes de remplissage rencontrés lors de la mise en trame. L'amélioration de la latence permet de diminuer le taux de perte de paquets dû à des délais d'attente excessifs. Aussi, dans le but de préserver le même rapport signal sur bruit que dans la courbe théorique du BER du FHSS, nous faisons varier la puissance du signal en fonction du débit offert. En déterminant l'énergie consommée pour l'envoi, par le produit de la

puissance et de la latence de service, nous enregistrons une économie d'énergie des sauts de fréquence variables par rapport au FHSS.

CHAPITRE 1

L'ORDONNANCEMENT À SAUTS DE FRÉQUENCE

1.1 Introduction

Un système de communication sans fil est représenté en schéma bloc à la figure 1.1.

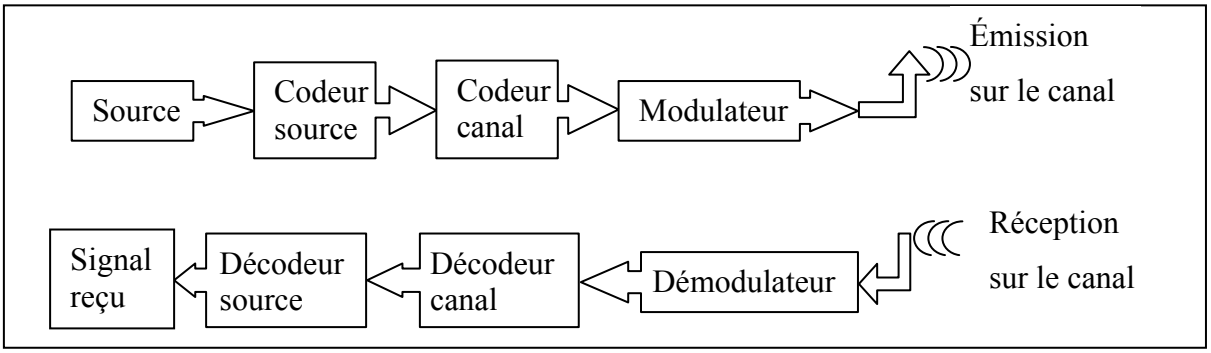

Figure 1.1 Schéma synoptique d'une chaîne de transmission
Reproduite et adaptée avec l'autorisation de (Holmes, 2007)

Tous ces blocs sont importants pour avoir une communication sans fil. La source génère l´information utile à envoyer qui, si elle est numérique, forme une suite binaire de '1' et de '0'. Cette information parvient au codeur source, qui la code suivant le format correspondant au type d´information (audio, vidéo, etc.). Le codage canal, quant à lui, s'occupe d´associer à chaque impulsion '1' ou '0', un signal qui est la réponse impulsionnelle du canal. Le modulateur transforme l'information pour transmettre des symboles constituant un signal électrique concret. Une fois cette information modulée, elle est acheminée au canal qui, la transporte au récepteur. Une fois au récepteur, l'information suit le chemin inverse à celui de l'émetteur pour être décodée. Le même canal de transmission peut être partagé par plusieurs équipements. Alors, il peut avoir des interférences entre différents usagers. Pour éviter ce problème, le canal est alloué à chaque utilisateur par une allocation des ressources temporelle, fréquentielle et même spatiale. Pour une allocation temporelle, on utilise souvent un algorithme dit d'ordonnancement, « scheduling », qui est implémenté au niveau de la

couche MAC, « Medium Access Control ». En plus de l'ordonnancement, pour rendre le signal plus robuste aux interférences, des techniques d'étalement de spectre sont utilisées. Les plus rencontrés dans le domaine militaire sont les ordonnancements à sauts de fréquence car, ils combinent sécurité et robustesse face à plusieurs types d'interférences.

L'allocation du canal peut être statique ou dynamique. L'allocation statique consiste à allouer des ressources aux usagers de façon fixe. Cette technique est moins efficace car lorsqu'un usager n'a rien à transmettre, la ressource est gaspillée. L'allocation dynamique, comme son nom l'indique, n'est pas fixe et permet de contourner ce problème. Elle se fait suivant les ressources disponibles. Ces ressources peuvent varier suivant plusieurs paramètres, comme les techniques d'étalement utilisées. L'allocation des ressources se fait par un serveur voulant servir un ou plusieurs clients. Dans ce cas, le partage des ressources se fait suivant les types de trafic à gérer. Plusieurs algorithmes proposent différentes façons de gérer les trafics, ou files d'attente, des usagers. Le bon choix d'un ou de plusieurs de ces algorithmes permet d'avoir une bonne QoS.

Dans les réseaux militaires, la qualité de service requise est très exigeante. Parmi les systèmes militaires offrant une bonne QoS sur le marché, nous pouvons citer ceux utilisant les radios de haute capacité, « *High Capacity Radio* », HCR, d'Ultra. Celles-ci constituent la cible de ce travail.

- l'HCR d'Ultra(Rackspace, 2012)

  L'HCR est un des derniers produits d'Ultra. Elle est conçue pour offrir de hautes capacités dans des systèmes de communication tactique sans fil. Elle opère dans la bande 4400-5000 MHz (Rackspace, 2012). Le logiciel défini pour les radios HCR supporte différentes formes d'ondes dépendamment du besoin et permet d'avoir des débits pouvant aller jusqu'à 150 Mb/s.

- les exigences de la QoS

  Les équipements HCR d'Ultra traitent des informations très sensibles. Ainsi, ils sont susceptibles d'être confrontés, comme tout équipement militaire, à des mesures de guerre électronique. Pour offrir de bons résultats, la QoS peut être implémentée à plusieurs niveaux dans les communications. Un état de l'art des quelques exigences en QoS se trouve dans les travaux de (Tata, 2009). Entre autres, nous avons les aspects tels que :

  - la disponibilité du service : c'est le temps effectif d'utilisation des ressources réseau par rapport au temps total qui est alloué au client. Ce temps est directement lié au contrôle d'admission;

  - débit binaire effectif, « *Data throughput* » : c'est la quantité d'information qui peut être transmise pendant une communication. Elle dépend de plusieurs paramètres tels que le taux d'erreur binaire, le type de modulation, l'algorithme de gestion du trafic, la bande passante allouée, etc.;

  - le délai : qui comprend le temps de transmission de l'information y compris celui nécessaire pour l'établissement de la connexion;

  - le taux de perte de paquet : c'est le rapport entre les paquets perdus et ceux émis;

  - la gigue : c'est la variation de délai entre les informations transférées. Elle doit être faible pour les trafics en temps réel.

En plus de ces aspects, nous ajouterons la sécurisation de l'information puisqu'il s'agit ici des réseaux militaires. La sécurité peut être implémentée au niveau du cryptage des données. Pour cela Ultra utilise l'algorithme de chiffrement AES 256 bits (Ultra Electronics TCS, 2011b). Mais elle peut être aussi implémentée dans la technique d'étalement du spectre. Ultra utilise deux techniques d'étalement connues pour leurs multiples avantages face aux attaques de plusieurs sortes. Nous pouvons citer les sauts de fréquence adaptatifs (Ultra Electronics TCS, 2011a) et l'OFDMA conforme à la norme WIMAX(Ultra Electronics TCS, 2011c). Nous nous intéresserons ici aux sauts de fréquence.

Dans la suite de ce chapitre, nous présenterons les techniques d'étalement à sauts de fréquence existant et leur fiabilité. Le spectre ainsi étalé revient en une subdivision de la ressource qui sera allouée suivant des algorithmes d'ordonnancements qui seront présentés par la suite.

1.2 Les techniques d'étalement de spectre à sauts de fréquence

La première technique à sauts de fréquence a été inventée pendant la Seconde Guerre mondiale par Hedy Lammar (co-inventrice)(Tanenbaum et Wetherall, 2011). Elle est très prisée dans le domaine militaire pour son efficacité face à divers types d'interférences. Cette technique est connue sous le terme FHSS (Frequency Hopping Spread Spectrum) ou encore FH (Frequency Hopping). Il existe deux types de sauts de fréquence : le saut de fréquence rapide, « *Fast Frequency Hopping* », FFH, et le saut de fréquence lent, « Slow Frequency Hopping », SFH. Cette rapidité se définit par rapport au nombre de sauts par seconde. Par exemple dans les HCR, 2000 sauts sont faits par seconde dans la technique actuellement utilisée (Ultra Electronics TCS, 2011a). Conçues initialement pour faire face aux attaques de l'ennemi dans des zones hostiles, les FH sont menacés par de nouvelles formes d'attaques. Depuis lors, elles ont connu plusieurs améliorations visant à les rendre toujours plus robuste face aux nouveaux types d'attaques. Ainsi, des travaux ont donné lieu à plusieurs autres approches dérivées des FH comme par exemple l'AFH, le FH-CDMA.

1.2.1 Le FHSS, “Frequency Hopping Spread spectrum”

Le FHSS est une technique d’étalement de spectre par saut de fréquence. Elle consiste à étaler l’information sur tout le spectre disponible. Pour ce faire, des sauts de fréquence sont opérés à des intervalles de temps prédéterminés. La bande passante de chaque saut est fixe et connue d’avance, voir figure 1.2.

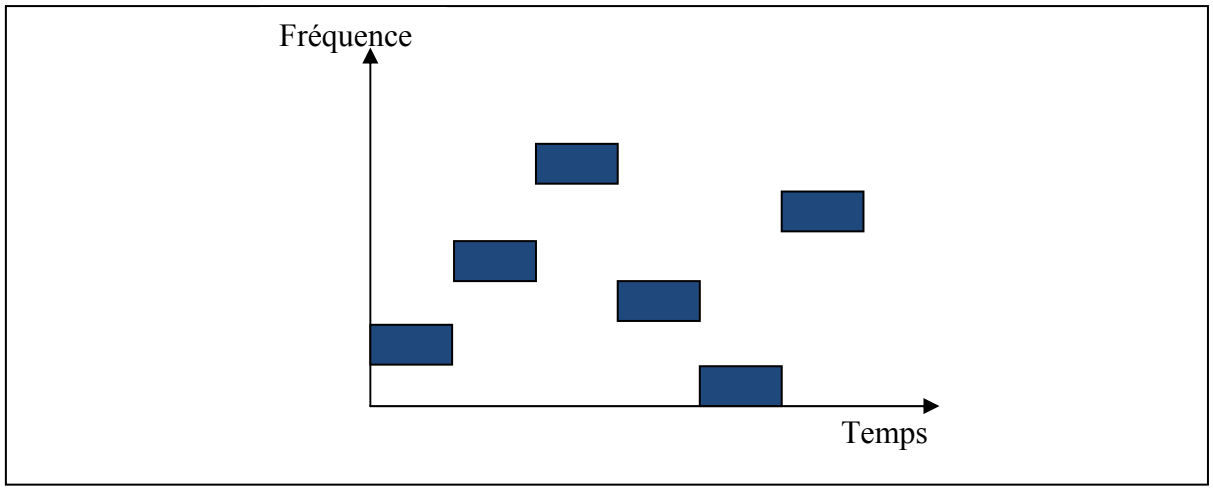

Figure 1.2 Transmission en mode FHSS

Cette technique est utilisée dans certains standards tels que le WLAN FHSS et le Bluetooth 802.15.1. Dans le Bluetooth, la bande est subdivisée en 79 canaux distincts de 1MHz chacun. Ces canaux couvrent toute la bande disponible sans se chevaucher. La rapidité des sauts est connue d’avance et est fonction de l’équipement utilisé. Par exemple dans la WLAN, « Les équipements FHSS doivent changer de canal au minimum 2,5 fois par seconde, soit 400ms ou moins, et chaque saut doit être espacé du précédent d’au moins 6 fréquences (6MHz)» (Roshan et Leary, 2004).

Étant donné que le canal change dans le temps, il est important que tous les dispositifs soient synchrones pour se retrouver sur le bon canal au bon moment. La synchronisation des sauts se fait par l’intermédiaire d’un maître qui gère le réseau. Par exemple dans les réseaux

Bluetooth, tous les dispositifs, y compris le maître, disposent d'une horloge native (CLKN). Pour se synchroniser au maître, les esclaves doivent s'assurer que leur CLKN est en phase avec celui du maître. Afin de maintenir un bon alignement, tous les esclaves d'un réseau effectuent continuellement une synchronisation sur l'horloge de leur maître. À chaque réception de paquet, il y a synchronisation(Baumgartner et Racloz, 2002). La ressource ainsi partagée est allouée aux utilisateurs suivant certains critères. Ces critères seront définis dans la suite du chapitre dans la partie ordonnancement.

Les FH n'offrent pas que des avantages. Un problème qui est rencontré avec les sauts de fréquence est l'interférence avec d'autres technologies partageant la même bande. Ceci est dû au fait que les sauts se font sur toute la bande. Tel est le cas pour le Bluetooth 802.15.1 et le WLAN 802.11b qui se partagent la même bande ISM (2400 à 2483,5 MHz). Pour pallier à ce problème, l'Adaptive Frequency Hopping (AFH) a été proposé.

1.2.2 L'AFH, « Adaptive frequency hopping »

L'AFH a un fonctionnement presque identique au FHSS utilisé dans le Bluetooth 802.15.1. La différence est que dans l'AFH, les sauts se font de façon plus intelligente. Dans les environnements où le Bluetooth est voisin du WLAN 802.11b, l'AFH évite de faire des sauts sur les fréquences utilisées par le WLAN. Pour s'y prendre, il tient compte de ses paquets erronés pour supprimer certaines fréquences de sa gamme de fréquences.

Pour choisir sa plage de saut, il estime dans un premier temps le canal. Il envoie donc des paquets sur chacune des fréquences possibles et déduit le taux d'erreur binaire (BER) sur chaque fréquence, voir tableau 1.1. Il note ensuite chacune de ces fréquences avec une mention 'Bonne' ou 'Mauvaise'.

Tableau 1.1 Table de statut des fréquences
Tiré de (Golmie, Rebala et Chevrollier, 2003)

Statut	**Décalage de fréquence**	**Probabilité [perte de disponibilité]**
Bon	0	10^{-3}
Mauvais	1	0.75
Mauvais	2	1
Mauvais	3	0.89
	…	
Bon	76	10^{-4}
Bon	77	10^{-3}
Bon	78	10^{-3}

Un seuil de probabilité de perte est spécifié pour qualifier une fréquence. Notons qu'il n'est pas évident que tous les paquets erronés soient causés par la présence d'autres systèmes comme le WLAN. D'autres types d'interférences temporaires peuvent en être la cause, ce qui engendre une diminution inutile de la bande passante. Pour éviter ce genre de problème, l'opération de sélection des fréquences possibles de l'AFH se fait très régulièrement, ce qui peut alourdir le réseau. L'Utility Based Adaptive Frequency Hopping (UBAFH) (Stabellini et al., 2009) prend en compte la probabilité d'utilisation d'un canal pour déterminer la plage des sauts. Pour ce faire, il se base sur l'historique du, *Paquet error rate*, PER, de chaque canal. Cette estimation lui permet d'offrir un meilleur PER que l'AFH, et ainsi une meilleure latence.

Dans le but d'améliorer davantage l'efficacité des FH dans des environnements très bruités, des étalements hybrides combinant les FH et le CDMA sont proposés.

1.2.3 Les étalements hybrides à sauts de fréquence

Comme étalement hybride, (Elsner, Tanbourgi et Jondral, 2011) proposent le Multi-level locally orthogonal FH-CDMA (MLLO-FH-CDMA). MLLO FH-CDMA est une méthode d'accès qui, en plus de faire des sauts de fréquence, fait aussi des sauts de code CDMA. Elle est utilisée dans le domaine militaire pour mieux résister aux interférences et à l'écoute. Pour ses sauts de fréquence, elle utilise le même principe que celui de l'AFH. Cette technique permet une meilleure utilisation de la bande passante. Néanmoins, elle fait face au problème d'interférence mutuelle entre divers équipements. Ainsi les travaux de (Elsner, Tanbourgi et Jondral, 2011) proposent une approche visant à faire du multi couche, à l'image d'un étalement CDMA, grâce à des codes localement orthogonaux. Elle permet de réduire l'auto-interférence à grande échelle causée par la réutilisation de fréquence par plusieurs équipements militaires. Cette approche consiste à classer les nœuds selon leur probabilité d'interférence. Lorsque deux nœuds s'interfèrent entre eux, ils sont reclassés dans des ensembles différents et auront différents codes. MLLO FH-CDMA est utile dans les cas où le nombre de nœuds à desservir dépasse le nombre de canaux disponibles. Elle est bénéfique pour des réseaux formés de plusieurs grappes dans lesquels les interférences sont limitées. Une limite de ces travaux est que l'accès au canal en temps réel n'a pas été abordé. Les auteurs prennent pour acquis qu'en fonction de l'application, l'allocation en fréquence peut être aléatoire ou basée sur une réservation. Cette allocation peut être faite en combinant un ou plusieurs techniques d'ordonnancement de base.

1.3 Techniques d'ordonnancement

Elle se définit comme la technique d'allocation des ressources disponibles aux files en attentes. Il existe plusieurs types d'ordonnancements. Le choix d'un type d'ordonnancement par rapport à un autre est souvent fonction de la QoS à offrir. La QoS dépend des exigences de chaque trafic.

1.3.1 Exigences et types des files d'attente

Les files d'attente peuvent contenir un ou plusieurs types de trafics dont, certaines sont plus exigeantes que d'autres. Le tableau 1.2 présente quelques types de trafics avec leurs exigences.

Tableau 1.2 Exigences de chaque type de service
Tiré de (Tanenbaum et Wetherall, 2011)

Application	**fiabilité**	**délai**	**guige**	**Bande passante**
Courrier électronique	Haute	Faible	Faible	Faible
Transfert de fichier	Haute	Faible	Faible	Moyenne
Accès au Web	Haute	Moyenne	Faible	Moyenne
Session à distance	Haute	Moyenne	Moyenne	Faible
Audio à la demande	Faible	Faible	Haute	Moyenne
Vidéo à la demande	Faible	Faible	Haute	Haute
Téléphonie	Faible	Haute	Haute	Faible
Vidéoconférence	Faible	Haute	Haute	Haute

Ces différents services sont regroupés en divers types de flux. Ces regroupements sont souvent propres à chaque système. Par exemple dans le WiMAX, nous pouvons citer :

- service à concession non sollicitée, UGS (Unsolicited Grant Service) : il supporte les paquets de types temps réel, émis à intervalles réguliers. Ces paquets sont de taille fixe. Dans le WiMAX, la bande passante allouée à ces types de paquet correspond à leur besoin maximal et est souvent fixe;

- rtPS, real-time Polling Service : il s'occupe des paquets de type temps réel, de taille variable et qui vienne à des intervalles réguliers. Dans ce service, la bande passante est allouée selon le besoin minimal. Comme exemple de paquets servis en rtPS, nous pouvons citer les vidéos MPEG;

- nrtPS, Non Real-Time Polling Service : il sert les paquets moins exigeants aux délais et ayant des tailles variables. La bande passante est allouée selon le besoin minimal. Comme exemple, nous pouvons citer les paquets FTP, File Transfer Protocol;

- BE, Best Effort : il gère les paquets n'ayant aucune exigence en QoS, donc aucune allocation de bande.

La communication efficace de ces différents flux nécessite un algorithme d'ordonnancement adéquat dont les objectifs seront :

- maximiser l'utilisation de la ressource;

- présenter un temps de réponse acceptable;

- respecter l'équité entre les processus selon le critère d'ordonnancement utilisé.

1.3.2 Types d'ordonnancements

L'ordonnancement constitue le processus de sélection itérative des paquets à transmettre. Il existe deux types d'ordonnancements :

- ordonnancement préemptif : dans ce cas un processus en cours de service peut être interrompu si un flux de priorité plus élevée est prêt à être servi;

- ordonnancement coopératif : ici le processus en cours de service est maintenu jusqu'à la fin du temps qui lui est alloué même si entre-temps un flux de priorité plus forte est prêt à être servi. Ce flux de forte priorité attendra la fin du service en cours.

Les principaux algorithmes d'ordonnancements sont :

- FIFO, First In First Out : l'ordonnancement est fait suivant l'ordre d'arrivée des paquets. Ce processus gère tous les trafics comme s'ils faisaient partie d'une file unique;

- SJF, Shortest Job First : à priorité égale, cet ordonnanceur sert les paquets les plus courts en premier. Pour calculer la priorité il tient compte des temps de séjour dans la file. La priorité croit avec le temps d'attente;

- SRT, Shortest Remaining Time : elle est la version préemptive du SJF. L'algorithme calcule le temps de service restant du processus en court. Si ce temps est supérieur à celui prévu pour le traitement du nouveau processus, il interrompt le service en cours pour servir le nouveau processus;

- RR, Round Robin : le service se fait de façon circulaire. Chaque file est considérée avec la même priorité. L'algorithme choisit un temps pendant lequel il servira chaque file. Après ce temps il passe à la file suivante. Si une file finit d'être servie avant la fin de son intervalle de temps, l'algorithme passe à la file suivante;

- HPF, Highest Priority First : il alloue une priorité à chaque file. Cette priorité peut être dynamique ou non. Il sert alors à tour de rôle les paquets qui ont plus de priorité. Il peut être préemptif ou non. Lorsque toutes les files ont la même priorité il utilise la règle du FIFO;

- EDF, Earliest Deadline First : il sert les paquets dont le délai d'attente autorisé est proche de l'échéance. Il traite tous les paquets comme étant dans la même file;

- WFQ, Weighted fair queuing : fonctionne comme le RR juste qu'ici un poids est associé à chaque file. Lorsqu'il associe un poids à un paquet, il le range dans la file correspondante à ce poids. Il sert ensuite ces files de façon circulaire. Mais certaines files auront droit à plus de ressources, quand leur tour arrivera, si leur poids est élevé;

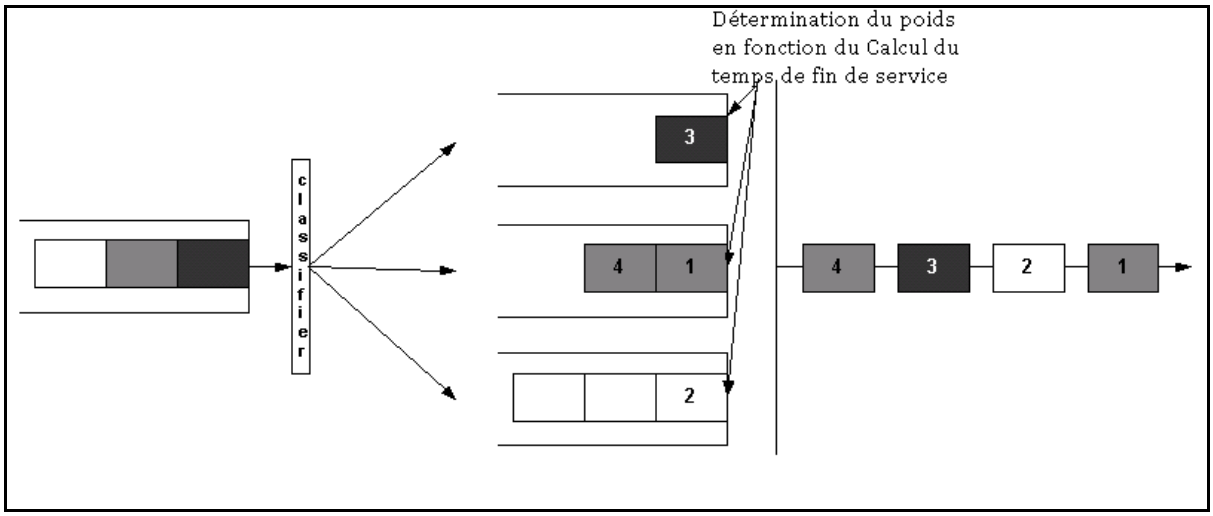

Figure 1.3 Algorithme WFQ
Adaptée de (Ungar, 1999)

- WRR, Weighted Round Robin : fonctionne comme le RR mais associe un poids à chaque flux et en fonction de ce poids associe un pourcentage de ressource à chaque flux. Il sert alors plusieurs files au même moment mais agit sur le pourcentage de ressource alloué.

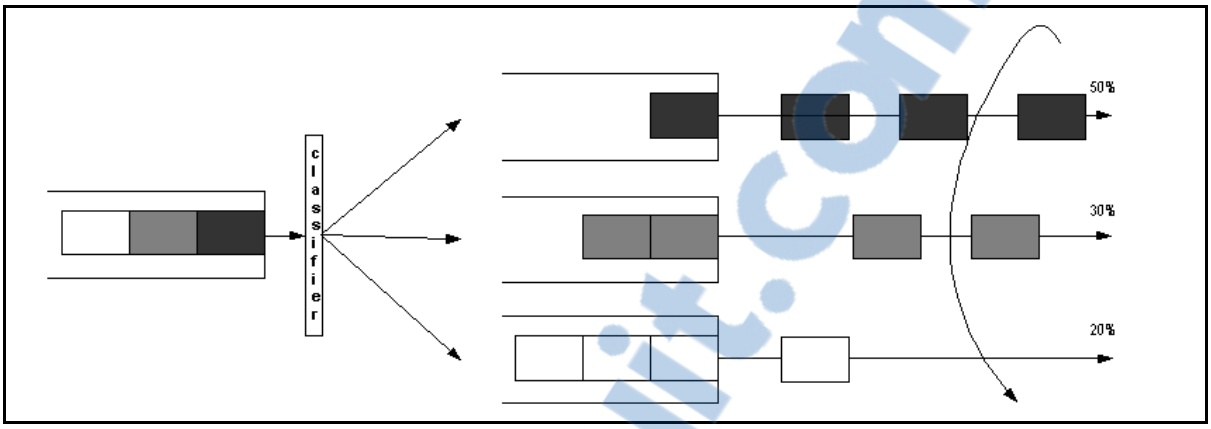

Figure 1.4 Algorithme WRR
Tirée de (Ungar, 1999)

Il existe plusieurs autres algorithmes qui se basent sur un ou plusieurs de ces principes de base. Nous pouvons donc citer :

- FEQ, *Fair and Effective Queueing*, est un algorithme de gestion de file d'attente contenant différents types de paquets. Il est présenté par les auteurs (Tata, 2009) pris dans (Xie, Chen et Wu, 2008). Dans cet algorithme, l'allocation de bande se déroule en deux phases. Dans la première phase, les files sont servies suivant l'algorithme WRR, "Weighted Round Robin" qui consiste à allouer une bande passante égale au MRR, "Minimum Reserved Rate" à chaque type de trafic. La valeur du MRR augmente avec les exigences des paquets. Les paquets n'ayant pas pu être servis dans la phase 1, et qui n'ont pas été perdu entre temps, passent en phase 2 où ils sont rangés dans une autre file d'attente unique. Cette file est servie suivant le principe EDF, "Earliest Deadline First", qui consiste à servir en priorité les paquets dont le délai d'attente toléré est presque échu. Cet algorithme favorise les paquets les plus exigeants au détriment des moins exigeants;

- SCDC, "Service Class Downlink Scheduling", présenté par les auteurs (Tata, 2009) pris dans (Thaliath et al., 2008), consiste à calculer un intervalle de temps spécifique à une trame donnée suivant l'exigence de sa classe de trafic. Il est proposé pour les liaisons

descendantes. Avec cette approche, les auteurs espèrent réduire le délai et améliorer le débit réel, "throughput", pour les trafics temps réel;

- un autre algorithme visant à améliorer le délai des files rtPS sans pour autant dégrader la QoS des flux nrtPS est présenté par (Tata, 2009) pris dans (Prasath, Fu et Ma, 2008). Leur solution consiste à subdiviser la quantité de bande totale requise aux deux files rtPS et nrtPS suivant un facteur d'ajustement. Ce facteur est choisi de sorte à favoriser les paquets rtPS en leur allouant plus de bandes lorsque nécessaire, ceci au détriment des paquets nrtPS. Car, les deux se partagent la même bande totale. Bien que les paquets nrtPS sacrifient leur bande passante, l'algorithme fait le partage de sorte à limiter la perte des paquets nrtPS due à un délai d'attente excédant la limite.

Un bref état de l'art de la plupart de ces algorithmes d'ordonnancement est proposé dans les travaux de (Tata, 2009). De cette étude, il ressort que l'algorithme WFQ, utilisé seul ou combiné à d'autres algorithmes, est présent dans la plupart des méthodes d'ordonnancement traitées. Toutefois, il n'est pas sans faille car, il peut causer des pertes de ressources dues à une sous utilisation de la bande allouée aux paquets prioritaires. Pendant ce même temps, les paquets de faible priorité peuvent souffrir d'un manque de bande.

L'EDF quant à elle n'est pas appropriée pour les cas non préemptifs, selon (Ekelin, 2006), cité par (Tata, 2009). Il existe plusieurs approches visant à résoudre ces faiblesses mais, pour (Tata, 2009) la quasi-totalité des propositions favorisent les trafics de forte priorité. Ainsi, l'auteur de (Tata, 2009) proposent un algorithme de courtoisie qui réduit le délai d'attente et le taux de perte des paquets de faible priorité. Leur approche, dit de courtoisie est basée sur un principe de calcul de priorité de transmission et du temps d'attente supplémentaire toléré pour les paquets de haute priorité. Selon ce calcul, les paquets de basse priorité peuvent être servis avant ceux de haute priorité si le temps d'attente des paquets prioritaires le permet. Cette solution améliore la latence des paquets de faibles priorités, bien qu'elle soit proposée pour des débits fixes. Nous émettons l'hypothèse qu'avec des paquets de taille variables, la

latence devrait être améliorée car le coefficient de courtoisie pourra être utilisé à son plein potentiel.

Ces algorithmes allouent les ressources suivant les subdivisions de bande préétablies au moment de l'étalement. Ils agissent juste sur le nombre et la durée des intervalles de temps à offrir à une file ou à une autre.

1.4 Conclusion

Dans ce chapitre, nous remarquons que les techniques d'étalement de spectre sont faites sans tenir compte de l'état des files. Les sauts de fréquences et les intervalles de temps ont des tailles prédéfinies. Lorsque l'un des algorithmes d'ordonnancement doit être utilisé pour gérer l'accès au canal, il agit seulement sur le nombre d'intervalles de temps à allouer aux files. Ce qui n'est pas toujours bénéfique pour le réseau. Dans le chapitre 2 nous examinons les différents facteurs influant sur la latence, que ce soit du taux d'erreur binaire jusqu'au taux d'utilisation du réseau.

CHAPITRE 2

ANALYSE MATHÉMATIQUE ET ÉTUDE DES HYPOTHÈSES

2.1 Introduction

Les techniques d'étalement à sauts de fréquence, une fois sur le terrain, sont confrontées à plusieurs limites. Ne pouvant réunir toutes les conditions du terrain dans un laboratoire, nous disposons des modèles mathématiques qui prennent en compte la plupart des critères à considérer. Ce chapitre présente les modèles mathématiques de ces critères. Ces modèles donnent une idée de ce qui se passera sur le terrain réel mais, ne garantissent pas que le fonctionnement théorique sera identique à celui pratique. Toutefois, cette étude nous permet de valider nos hypothèses afin d'en déduire les plus prometteuses.

Ainsi, parlant de critères, nous présenterons dans un premier temps la structure de nos paquets. Cette structure peut être identique à la structure du paquet dans une application pratique. Tel que nous l'avions vu dans le premier chapitre, les paquets subissent un codage canal. Nous présenterons alors certains codages utilisés dans les techniques de saut de fréquence traditionnelles. Le signal ainsi codé est émis sur le canal sous forme de symbole binaire. Nous parlerons brièvement des techniques de modulation de l'information en symbole binaire. Ensuite nous étudierons quelques causes d'erreurs binaires car, le signal une fois sur le canal peut subir toutes sortes d'interférences. Puis, nous parlerons de la latence, fonction du débit réel, qui lui à son tour est fonction de tous les paramètres pré cités. Nous étudierons nos hypothèses au fur et à mesure que nous présenterons les modèles et enfin, nous allons présenter les plus prometteuses dans la conclusion.

2.2 Structure du paquet

La structure du paquet dépend de la norme utilisée. Dans ce travail, nous ne nous focaliserons pas sur une norme spécifique car, les couches touchées par cette recherche sont la couche MAC et la couche PHYSIQUE. Néanmoins, nous supposerons une structure à nos

paquets pour la suite de ce travail. Il s'agira de la structure d'un paquet Bluetooth, tel qu'illustré à la figure 2.1. La seule modification apportée serait au niveau de la taille de la charge utile.

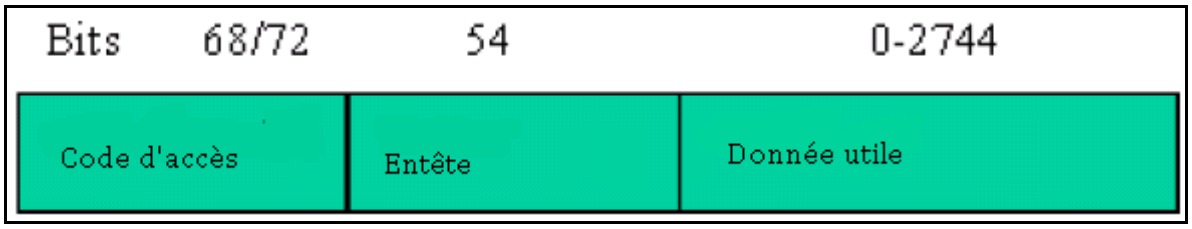

Figure 2.1 Structure d'un paquet Bluetooth
Adpatée de (Baumgartner et Racloz, 2002)

Nous avons choisi les paquets Bluetooth car, ils répondent déjà aux exigences des sauts de fréquence traditionnels. Avec une telle structure, nous tenons pour acquis que nos paquets disposent de champs suffisants pour permettre une bonne compréhension du message transmis. Dans la suite de ce travail ne nous intéressons pas au processus de signalisation. Nous supposons que chaque nœud connait la séquence des sauts. Toutefois, nous apporterons quelques modifications sans toucher aux champs qui servent à décoder le message. Ainsi, nous supposerons que la taille du code d'accès et celle de l'entête resteront fixes en tout temps. Les paquets de synchronisations ne seront pas étudiés dans cette rubrique car nous tenons pour acquis que les différents nœuds connaissent déjà la séquence des sauts. Pour protéger les données transmises nous proposons quelques techniques qui font déjà leur preuve dans les technologies existantes.

2.3 Technique de protection des données

Plusieurs techniques de protection des données sont détaillées dans la littérature. Mais nous proposerons quelques-unes en particulier celles utilisées dans Bluetooth. Nous nous sommes inspirés des techniques utilisées dans le Bluetooth parce qu'il constitue une technique à sauts de fréquence très prisée et bien documentée. Les techniques qui seront présentées ici sont : la

modulation, le FEC 1/3, le FEC 2/3 et l'ARQ. Notons que la technique utilisée pour protéger l'entête d'un paquet peut différer de celle utilisée pour protéger les données utiles.

2.3.1 Modulation

Sur le canal de transmission, les informations sont envoyées sous forme de symboles constituant un signal transmis. Cette technique consistant à transformer les bits en symboles s'appelle la modulation. Elle consiste à attribuer, pour chaque séquence de bits, une forme du signal prédéfinie. À la base, il existe trois types de modulations :

- modulation d'amplitude (ASK, Amplitude-shift keying);
- modulation de phase (PSK, Phase-shift keying);
- modulation de fréquence (FSK, Frequency-shift keying).

Il y existe d'autres modulations qui sont des combinaisons de ces modulations, ou encore des modulations dites non-linéaires. Nous nous intéresserons dans ce travail à la modulation de fréquence, encore dénommé MFSK, "Multiple Frequency-Shift Keying". Nous avons choisi la MFSK car, elle est souvent utilisée dans les sauts traditionnels comme par exemple le Bluetooth, et offre un bon taux d'erreur binaire dans un environnement de bruit Gaussien. Son principe est peu complexe. Il suffit d'associer à chaque symbole une fréquence donnée, voir figure 2.2. Pour éviter les interférences inter symboles, ces fréquences doivent être séparées d'au moins de π/T_d Hz (Smith, 1993). Avec T_d la durée d'un symbole.

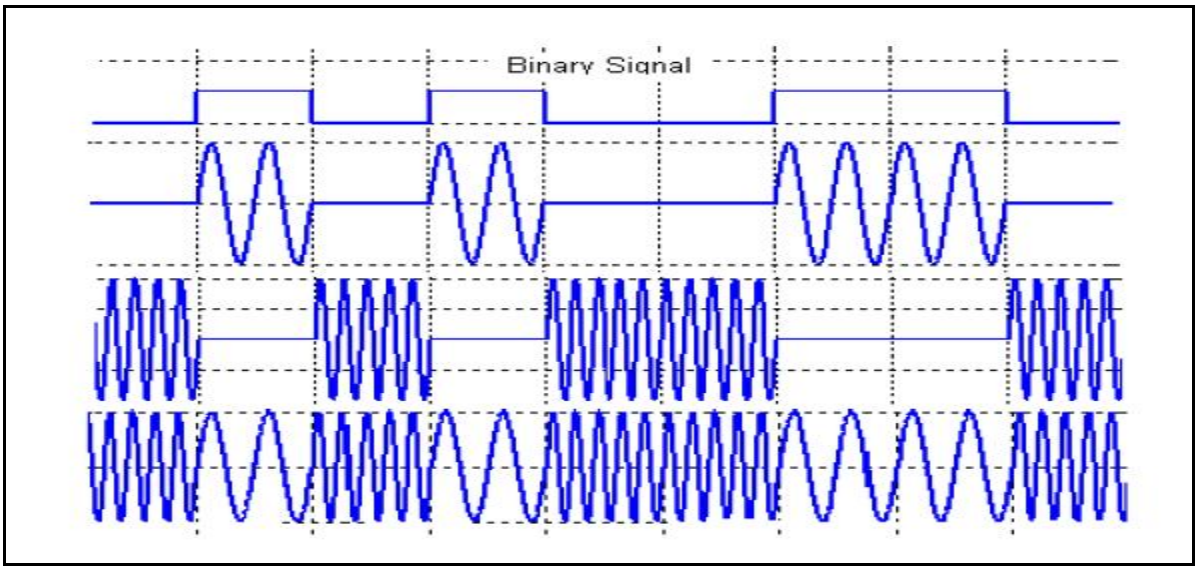

Figure 2.2 Modulation MFSK
Tirée de (Implustit, 2012)

À la figure 2.2, on illustre la modulation de symbole portant un seul bit. Au bit ‘0’, on associe une fréquence f_0 différente de celle f_1, associée au bit ‘1’. La première ligne représente le signal binaire à transmettre. La ligne 2 représente la fréquence correspondant au bit ‘1’ et la ligne 3, celle du bit ‘0’. La ligne 4 est la somme de ces deux signaux et représente aussi la forme du signal analogique qui sera transmis sur le canal.

Même avec la modulation, le signal n’est pas exempt des perturbations pouvant causer des pertes d’informations. Ce taux d’erreur binaire ou encore, "*Bit Error Rate*", BER est à minimiser pour avoir une meilleure communication.

2.3.2 Les codages

Il existe plusieurs types de codages. Nous avons le FEC 1/3, le FEC 2/3 et l’ARQ. Notons que les FEC sont des codes correcteurs d’erreurs et l’ARQ permet la retransmission.

- **FEC 1/3**

  Cette technique consiste à répéter trois fois chacun des bits. Le récepteur prend une décision en tenant compte des bits majoritaires. La figure 2.3 illustre cette répétition de bit.

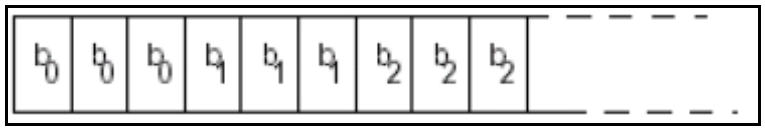

Figure 2.3 Codage FEC 1/3
Tirée de (Baumgartner et Racloz, 2002)

Par exemple, lorsque cette technique est utilisée dans l'entête du paquet en figure 2.1, si nous enlevons ce codage, le nombre de bits effectifs de l'entête serait de 54/3=18 bits. Donc la taille réelle de l'entête est de 18 Bits.

- **FEC 2/3**

  Elle utilise un code de Hamming qui génère 15 bits pour un mot de 10 bits. Le polynôme générateur est :

$$g(D) = (D+1)*(D^4 + D + 1) \tag{2.1}$$

  Ce code a une distance de 4. Il corrige 1 erreur ou il en détecte 2 (Baumgartner et Racloz, 2002). Il est généré par un registre à décalage de 5 bits.

- **Technique d'ARQ**

  Elle consiste à renvoyer un paquet protégé par le CRC, "Contrôle par Redondance Cyclique", lorsque le récepteur reçoit un NAK ou ne reçoit aucun acquittement. Ce renvoi peut causer des problèmes de doublon lorsqu'un acquittement est perdu. Chaque paquet est muni d'un bit de séquence pour éviter les problèmes de doublons.

2.4 Les interférences

Aussi appelées brouillages, elles consistent en un processus générant un signal qui détériore de façon plus ou moins importante les performances d'une liaison. Il existe deux familles de brouillage (Holmes, 2007), le brouillage malicieux et le brouillage non malicieux décrits ci-dessous.

2.4.1 Brouilleurs non malicieux

Ce sont des brouillages non intentionnels. Leurs causes peuvent être de plusieurs sortes. Il existe trois grands types de brouilleurs non malicieux.

Le premier est le brouillage mutuel causé par des équipements fonctionnant dans un même voisinage fréquentiel. Ce problème est réglementé par la normalisation faite par chaque pays pour les fréquences à utiliser par chaque application. Lorsque deux applications distinctes sont autorisées à se partager la même bande, c'est parce que l'institut de normalisation juge que le signal de chacune ne sera pas dégradé de façon significative.

Une autre forme d'interférence est celle rencontrée dans le CDMA où plusieurs équipements se partagent la même bande. Grâce à l'orthogonalité ou la faible corrélation des codes, le signal est dégradé de façon contrôlée.

Nous avons aussi l'interférence multi-trajets. Elle est due à la réflexion causée par la géographie de l'environnement de propagation du signal. Cette réflexion peut engendrer un léger retard du signal réfléchi par rapport au signal direct. Le signal reçu est la résultante du signal direct et de ceux qui ont subi une réflexion. Les sauts de fréquence rapides peuvent limiter l'impact de ce problème. Lorsque les différences de chemins sont faibles, par rapport à la durée d'un symbole, on parle de "flat fading". Cette interférence peut causer une dégradation considérable du signal original.

2.4.2 Brouilleurs malicieux

Par malicieux, on considère tout brouillage causé de façon intentionnelle. Ce brouillage vise à augmenter le taux d'erreur binaire dans le réseau ciblé. Les réseaux militaires en sont souvent victimes. Il existe 7 types de brouillages intentionnels (Holmes, 2007).

2.4.2.1 Brouilleur à large bande

Il émet un signal semblable à un bruit blanc gaussien se limitant à toute la bande passante du système, voir figure 2.4. Il est le brouilleur le moins efficace car, il a une faible densité spectrale. Soit :

$$N_{0j} = J / W \tag{2.2}$$

Avec J la puissance de brouilleur et W la largeur de la bande brouillée.

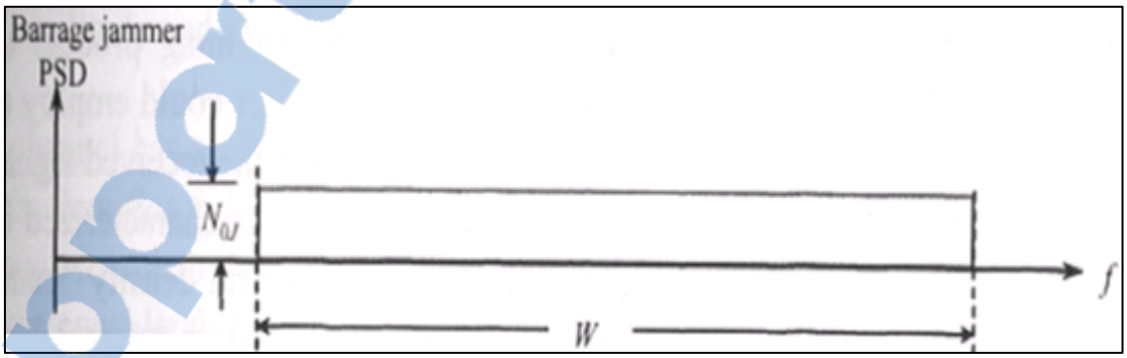

Figure 2.4 Représentation du brouilleur à large bande
Tirée de (Holmes, 2007)

L'impact du brouilleur à large bande sur le signal dépend du type de modulation utilisé. Il est représenté par la probabilité d'erreur du signal. Nous considérons la modulation PSK car, elle intervient dans les FH hybrides, et la modulation MFSK, souvent utilisée dans les sauts de fréquence.

- **brouilleur à large bande : Modulation par phase, PSK**

  Souvent utilisée dans les étalements hybrides, sa probabilité d'erreur ne dépend pas de l'ordre de modulation. Elle est la même pour le BPSK, le QPSK, et l'OQPSK, et est donnée par l'équation (2.3).

$$P_b = Q\left[\sqrt{\frac{2}{\frac{\alpha N_0 R_b}{P} + \frac{\alpha J}{P}\frac{R_b}{W}}}\right] \tag{2.3}$$

  Avec :

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy \tag{2.4}$$

  Ce qui revient après changement de variable à la fonction d'erreur complémentaire « *erfc* » définie par :

$$erfc(z) = \frac{2}{\sqrt{\pi}} \int_z^\infty e^{-\varsigma^2} d\varsigma \tag{2.5}$$

  En posant

$$\varsigma = \frac{y}{\sqrt{2}} \tag{2.6}$$

  Nous avons

$$d\varsigma = \frac{1}{\sqrt{2}} dy \tag{2.7}$$

Alors :

$$erfc(z/\sqrt{2}) = \frac{2}{\sqrt{2\pi}} \int_{z/\sqrt{2}}^{\infty} e^{-y^2/2} dy \tag{2.8}$$

Ce qui veut dire que :

$$Q(z) = \frac{1}{2} erfc(z/\sqrt{2}) \tag{2.9}$$

Où :

N_0 : l'intensité du bruit thermique

R_b : le débit binaire

P : la puissance du signal reçu

J : la puissance du brouilleur

W : la largeur de la bande brouillée

α : un facteur de réduction de la densité spectrale du bruit lors du désétalement à cause de l'effet miroir. Il peut être choisi fixe pour faire les calculs, soit par exemple $\alpha = 1$.

La fonction « ***erfc »*** étant une fonction décroissante, diminuer le BER revient à augmenter « z ». Les termes sur lesquels nous pouvons agir pour augmenter « z » sont R_b et P. Pour augmenter « z » nous pouvons soit augmenter la puissance, diminuer le débit R_b, ou faire les deux en même temps.

- **brouilleur à large bande : MFSK**

Souvent utilisée dans les étalements à sauts de fréquence, la probabilité d'erreur de la modulation MFSK est donnée par l'équation (2.10).

$$P_b = \frac{1}{2(M-1)} \exp\left[\frac{-kE_b}{2N_0^{'}}\right] \sum_{n=2}^{M} \binom{M}{n} (-1)^n \exp\left[\frac{kE_b(2-n)}{2nN_0^{'}}\right] \tag{2.10}$$

Avec

k : taille d'un symbole

M : nombre de symboles possibles

E_b : énergie par bit

$N_0^{'}$: la somme de la densité spectrale du bruit effectif et du bruit dû au brouilleur tel que,

$$\frac{E_b}{N_0^{'}} = \frac{1}{\frac{N_0 R_b}{P} + \frac{JR_b}{PW}} \tag{2.11}$$

Dans l'équation (2.10) nous avons :

$$\binom{M}{n} = C_M^n = \frac{M!}{n!(M-n)!} \tag{2.12}$$

L'équation (2.10) est fonction du débit, de la puissance et de l'ordre de modulation M. Pour diminuer cette probabilité, nous avons le choix entre diminuer le débit binaire, ou augmenter la puissance du signal.

2.4.2.2 Brouilleur à bande partielle

Il est presque identique au brouilleur à large bande, à la différence qu'il concentre son énergie sur une partie de la bande et non sur la totalité, voir figure 2.5.

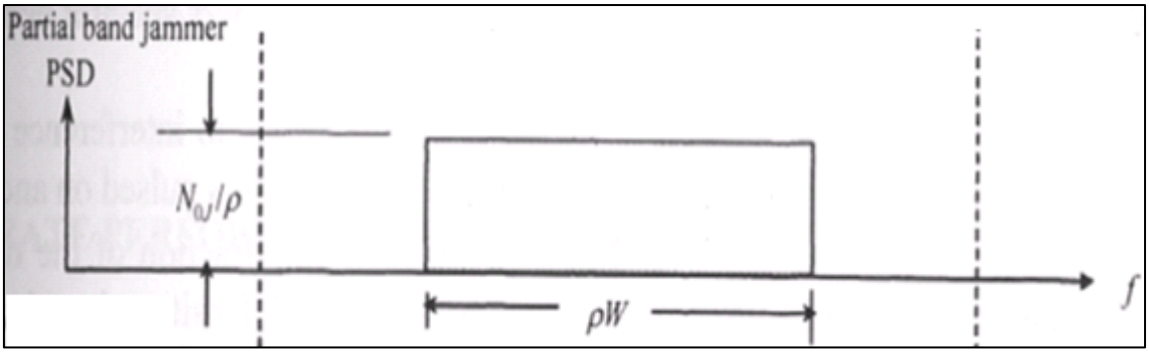

Figure 2.5 Représentation du brouilleur à bande partielle
Tirée de (Holmes, 2007)

En notant ρ la portion de la bande W brouillée, ρ est compris dans l'intervalle [0 1], nous avons comme densité spectrale :

$$N_{0J}^{\rho} = J/(\rho W) = N_{0J}/\rho \tag{2.13}$$

Il y a plus de chances d'avoir le signal du bruit qui dépasse celui du signal réel. Une partie de la bande étant brouillée, un premier avantage de faire les sauts de fréquence est qu'il y a des chances que certains sauts ne soient pas affectés. Si nous dénotons par ρ le pourcentage de bande brouillée, alors la probabilité qu'un saut soit totalement brouillé est de ρ, et celle qu'un saut ne soit pas brouillé est de (1-ρ). En appliquant ces probabilités à la probabilité P_b d'une bande brouillée, nous avons la formule de probabilité d'erreur pour un système FH-MFSK face à un brouilleur à bande partielle qui est :

$$P_{pJ} = (1-\rho)P_b\left(\frac{E_b}{N_0}\right) + \rho P_b\left(\frac{E_b}{N_0 + J/(\rho W)}\right) \tag{2.14}$$

Avec :

$$E_b = \frac{P}{R_b} \tag{2.15}$$

Et le P_b est l'expression dans l'équation (2.10) si c'est une modulation MFSK, ou encore l'équation (2.3) si c'est une modulation par phase.

Les facteurs qui influent sur cette probabilité d'erreur sont les mêmes que dans les équations (2.3) et (2.10).

2.4.2.3 Brouilleur à une porteuse

Toute la puissance du brouilleur est concentrée sur une seule fréquence, voir figure 2.6.

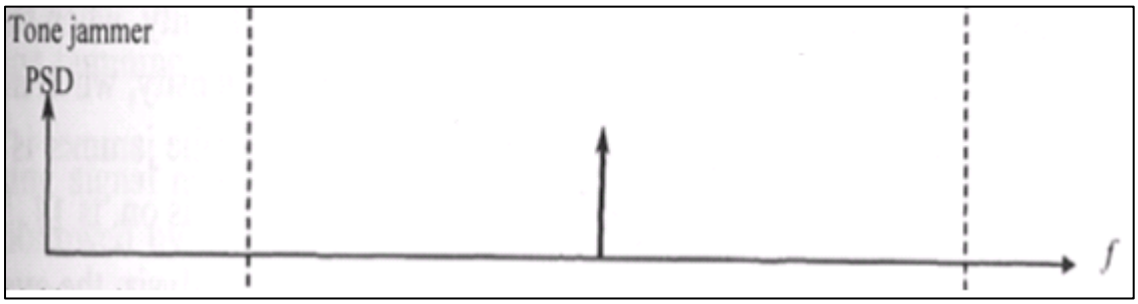

Figure 2.6 Spectre du brouilleur à une porteuse
Tirée de (Holmes, 2007)

Une fréquence est fortement brouillée à la fois et la densité spectrale du brouilleur est la plus grande en cette fréquence. Dans le « *direct sequence spread spectrum* », il y a une fréquence à laquelle la densité spectrale du bruit est plus grande que celle du signal et, lorsque le brouilleur est à cette fréquence le BER peut être très grand. Dans les FH, il existe plusieurs fréquences où la densité spectrale est forte donc les pertes sont limitées.

2.4.2.4 Brouilleur multi-porteuse

Il fonctionne comme le brouilleur à une porteuse, à la différence que le bruit est émis sur plusieurs impulsions. La puissance par impulsion est donc *1/N* de la puissance totale du brouilleur. Avec *N* le nombre total d'impulsions. Ceci est illustré à la figure 2.7.

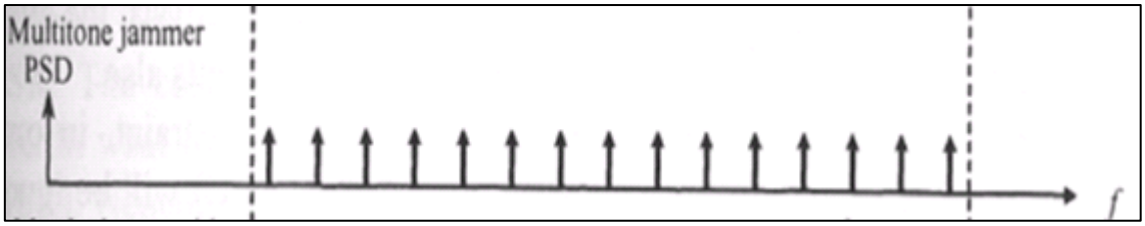

Figure 2.7 Spectre du brouilleur multi-porteuse
Tirée de (Holmes, 2007)

Dans cette rubrique, nous prenons pour hypothèse que le brouilleur connait parfaitement la taille et l'ordre des sauts. Une seule porteuse est brouillée par bande de fréquence MFSK. À chaque saut de fréquence, un des *M* symboles possibles du MFSK peut être transmis. Chaque symbole est espacé de R_{sy} Hz qui n'est rien d'autre que le débit symbole. On en déduit que la bande passante associée à chaque saut est :

$$W_m = M * R_{sy} / \log_2(M) \tag{2.16}$$

Soit « *n* » le nombre de porteuses brouillées. Si la puissance d'une porteuse du brouilleur est beaucoup plus faible que la puissance de réception du signal émis, alors le brouilleur n'aura aucun effet. Il faut donc que les porteuses aient une puissance presque égale ou supérieure à celle reçue par le récepteur pour pouvoir brouiller le signal. Alors, « *n* » est approximativement choisi grâce à la formule ci-dessous (Holmes, 2007) :

$$n \approx \left\lfloor \frac{J}{P} \right\rfloor \tag{2.17}$$

Avec *J* la puissance totale du brouilleur, et *P* la puissance de réception.

$\lfloor x \rfloor$: signifie le plus grand entier naturel inférieur ou égal à *x*.

Il est démontré que lorsque le signal MFSK est orthogonal, et que "*n*" est presque égal au rapport *J/P*, la probabilité d'erreur binaire est (Holmes, 2007) :

$$P_b = \begin{cases} 0.5 & \left(\frac{P*W}{J*R_b}\right) < M/\log_2(M) \\ \left(M/2*\log_2(M)\right)\left(\frac{J*R_b}{P*W}\right) & \frac{M}{\log_2(M)} \le \frac{P*W}{J*W_m} \le W/R_b \\ 0 & W/R_b < \frac{P*W}{J*R_b} \end{cases} \tag{2.18}$$

Pour espérer une amélioration du P_b dans ce cas, il faut idéalement se retrouver dans la condition où P_b=0. Pour cela, la variation de la valeur du R_b n'a aucune influence car, après simplification, pour R_b toujours positif, nous avons :

$$W/R_b < \frac{P*W}{J*R_b} \Rightarrow W < \frac{P*W}{J} \tag{2.19}$$

Nous pouvons toutefois agir sur le terme :

$$\left(\frac{P*W}{J*R_b}\right) < M/\log_2(M) \tag{2.20}$$

En diminuant le débit R_b, nous pouvons éviter la condition extrême où, la probabilité d'erreur binaire P_b=0.5. Mais une fois encore cette variation ne sera efficace que si nous connaissons la puissance du brouilleur, ce qui n'est pas souvent évident.

Hormis les deux cas précédents où la probabilité d'erreur reste fixe, nous avons la condition :

$$\frac{M}{\log_2(M)} \le \frac{P*W}{J*W_m} \le W/R_b \tag{2.21}$$

Dans le cas précédent, la probabilité d'erreur devient :

$$P_b = \left(M \Big/ 2 * \log_2(M) \right) \left(\frac{J * R_b}{P * W} \right) \tag{2.22}$$

P_b dans l'équation (2.22) a l'allure d'une droite affine. Une diminution du BER revient soit à garder un débit constant faible, ou soit à augmenter la puissance ou encore à maintenir un ordre de modulation faible.

2.4.2.5 Brouilleur adapté

Il suit un processus gaussien aléatoire pour produire un bruit ayant une densité spectrale correspondant à celle de la puissance du signal émis voir figure 2.8. Il est très difficile de corriger les erreurs causées par ce type de brouilleur.

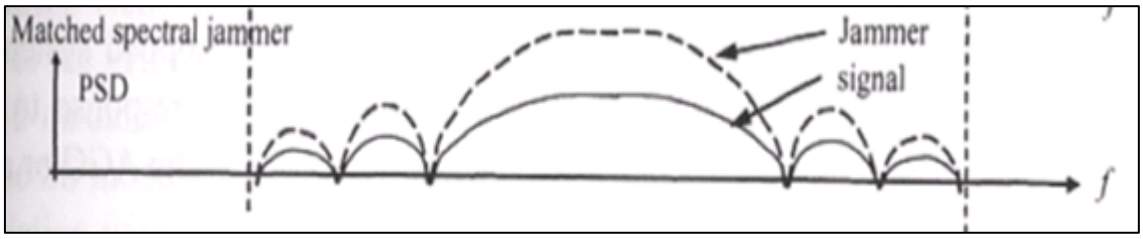

Figure 2.8 Spectre du brouilleur adapté
Tirée de (Holmes, 2007)

2.4.2.6 Brouilleur impulsif

Il envoie, de façon périodique, des bruits sous forme d'impulsions. Il reste allumé pendant une partie du temps et reste éteint le reste du temps. Ces impulsions peuvent couvrir une partie ou la totalité de la bande. Soit ρ ce pourcentage de bande couverte. Lorsque le brouilleur est allumé, sa densité spectrale est de $J/(\rho W)$ et elle est égale à zéro quand il est

éteint. Ce fonctionnement rend difficile la localisation du brouilleur. Il a la même probabilité de brouillage que celle du brouilleur à bande partielle (Holmes, 2007).

2.4.2.7 Brouilleur répéteur

Ce brouilleur intercepte le signal émis puis le retransmet à la même fréquence après l'avoir modifié. Ce type de brouillage accuse un retard par rapport au signal original. Dans les communications à sauts de fréquence lents, il augmente considérablement le BER. Dans les réseaux militaires, les techniques de FH rapides sont utilisées pour y résister. Si le retard dépasse la taille d'un saut, dans le cas des FH, le brouilleur n'aura pas d'effet sur le signal original car le récepteur aurait déjà changé de fréquence d'écoute. Les auteurs (Felstead, 2009) le présentent comme un brouilleur très redouté par les FH. Ils suggèrent de faire 10 000 sauts par seconde pour éviter l'effet de ce brouilleur, ce qui est difficile à faire avec les radios. Pour contourner cette limite, les auteurs (Nguyen Vo, Bellemare et Forté, 2012) proposent la modulation CPFSK-OFDM pour les *Fast FH*, car selon eux, faire 10000 sauts par seconde revient à envoyer un seul symbole OFDM par saut. La limite de l'OFDM est qu'elle ne permet pas une grande mobilité à cause des fréquences d'offset. Donc son application sera difficile dans les réseaux ad hoc mobiles. À défaut de faire 10000 sauts par seconde, nous pouvons voir le problème autrement. En effet, puisque le brouilleur répéteur, pour détecter les sauts de fréquence se sert de la radiométrie, il faut varier la puissance de chaque saut. Ainsi, le brouilleur ne pourra pas facilement accumuler les statistiques nécessaires à la prédiction des sauts.

Le radiomètre détecte le signal à partir de son énergie. Pour une détection optimale des sauts de fréquence, le détecteur écoute sur chacun des canaux possibles pendant la durée de l'intervalle de temps (Holmes, 2007). Cela demande que la subdivision de la bande totale soit connue du détecteur et ne change pas. Aussi pour détecter le signal, il doit connaitre la puissance totale du bruit, y compris le bruit thermique. Si ce bruit varie, le détecteur aura des difficultés à déduire la puissance du signal émis. En présence d'interférence, ce détecteur peut fausser sa détection.

Le bruit thermique et la puissance des interférences ne sont pas contrôlés par le système attaqué. Par contre, pour rendre la tâche difficile à ce détecteur, en faisant varier la taille des sauts et des intervalles de temps, le détecteur mettra un temps supplémentaire pour préparer tous les canaux sur lesquels il doit écouter. Ce qui rend ce détecteur inefficace dans les configurations changeantes.

2.5 Étude de la latence

La latence est définie comme la somme du temps qu'un usager passe dans la file d'attente et de son temps de service. Ce temps est fonction du débit réel et de la l'algorithme d'ordonnancement utilisé. Le débit réel à son tour est fonction comme nous l'avons vu plus haut de plusieurs paramètres dont le BER, tandis que le choix du meilleur algorithme d'ordonnancement peut différer d'un système de file d'attente à un autre.

2.5.1 Débit réel

Le débit réel ou débit effectif est la quantité d'information utile arrivée à destination en une seconde. Son calcul est très complexe car, tient compte des bits perdus, du codage, de la taille de l'entête et des retransmissions des paquets erronés. Mais ce calcul est aussi fonction de la capacité théorique du lien.

La capacité théorique ou encore, débit binaire théorique d'un lien représente le nombre maximal de bits pouvant être transmis en une seconde. Ce débit est fonction du type de canal choisi. Ainsi pour un canal AWGN, Claude Shannon (Holmes, 2007) définit la capacité du lien comme :

$$C = W \log_2(1 + S/N) \text{ bits/s} \tag{2.23}$$

Avec W la largeur de la bande du canal, S la puissance du signal en Watts et N la puissance totale du bruit dans le canal en Watts.

Ce débit ne peut être atteint. Alors pour évaluer un lien, nous pouvons déterminer l'efficacité de la bande passante connaissant la quantité d'information transmise. Dans (Holmes, 2007), elle se calcule grâce à l'équation (2.24).

$$\eta = \frac{Débit\_réel}{L\arg eur\_bande\_passante} \tag{2.24}$$

Le débit réel d'un lien est donné par l'équation (2.25).

$$Débit\_réel = \frac{\sum Bit\_transmis_{paquet}}{Durée\_transmission} \tag{2.25}$$

Avec $Bit\_transmis_{paquet}$: le nombre de bits réellement transmit par paquet.

La latence ne dépend pas que du débit réel, mais aussi de la façon d'ordonnancer ce débit aux différentes files.

2.5.2 Système de file d'attente

Un système de file d'attente peut contenir une ou plusieurs files d'un ou de plusieurs types. Lorsque ces files doivent se partager le même canal, le mécanisme de prise de décision pour l'accès au canal est basé sur les calculs et les classifications issus de la technique d'ordonnancement. L'efficacité d'un algorithme d'ordonnancement peut varier d'un système de file d'attente à un autre. Mais dans la littérature une des techniques d'ordonnancement est plus souvent utilisée, il s'agit du WFQ, présenté dans la section 1.3.2 de ce mémoire. Cette technique est efficace dans les cas où plusieurs types de files se partagent le même canal. Le système fait un service à tour de rôle. À chaque passage par une file, il prend un nombre d'octets qui est fonction de la priorité de cette dernière (Tanenbaum et Wetherall, 2011). Il est avantageux car, en faisant un service basé sur les octets et les règles de priorités il assure une bonne équité du service. Dans un étalement en saut de fréquence, où nous désirons servir

plusieurs usagers, cet ordonnanceur peut servir à la classification des files au niveau de chaque usager, mais il ne peut pas être utile pour gérer l'accès aux ressources. Car un même saut de fréquence ne peut être utilisé par deux usagers au même moment. Aussi envoyer un octet par saut peut conduire à un gaspillage de la ressource. Si les sauts sont de courtes durées, les radios ne pourront pas correctement se synchroniser. Alors, le partage de ressource entre usagers peut se faire suivant l'algorithme WRR présenté dans la section 1.3.2 car, au lieu de faire un service par octet, il le fait par paquet tout en suivant la même règle de priorité que le WFQ pour choisir le paquet à envoyer.

2.5.2.1 Analyse du WFQ et du WRR

Supposons un usager ayant *N* flux, de différents types ou non, dans lesquelles les arrivées suivent une loi de Poisson. Ces flux seront rangés en une seule file suivant le WFQ. La figure 2.9 illustre le fonctionnement du WFQ.

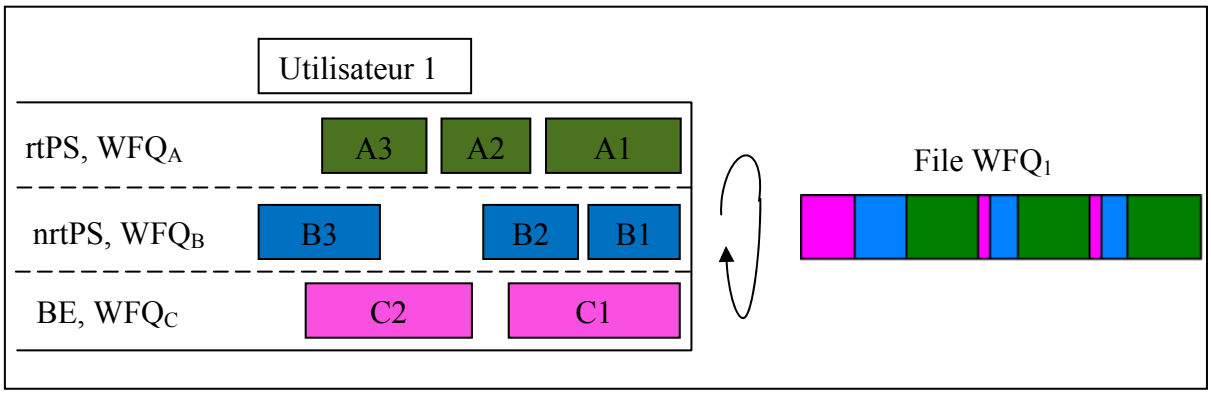

Figure 2.9 Classification WFQ du trafic chez un utilisateur

Dans cette figure, WFQ_i représente le poids de la file. Il définit le nombre d'octets à prendre dans cette file par tour. Ce poids tient compte aussi du temps d'attente permis pour chaque type de flux. En supposant que le service se fait automatiquement, d'après (Tanenbaum et Wetherall, 2011) l'instant de fin du paquet «*i*» est donné par l'équation (2.26).

$$F_i = \max(A_i, F_{i-1}) + L_i / WFQ_{file} \tag{2.26}$$

Avec A_i l'instant d'arrivée, F_i l'instant de fin et L_i la longueur du paquet i et WFQ_{file} est le poids de la file contenant le paquet i.

Les files WFQ ainsi obtenues pour chaque utilisateur constituent les files qui seront servies par les sauts de fréquence. Chaque file a une priorité qui varie constamment en fonction de son contenu. Dans les sauts de fréquence standard, la subdivision des ressources est déjà prédéfinie. Lorsqu'il y a assez de ressources pour toutes les files, l'ordonnancement consistera à offrir des sauts à chaque usager. Le nombre d'intervalles de temps alloués peut varier pour certains paquets comme il est par exemple le cas dans le Bluetooth avec les paquets DM3 (3 slots) et DM5 (5 slots) (Baumgartner et Racloz, 2002). La figure 2.10 illustre le service résultant pour les sauts de fréquences standards.

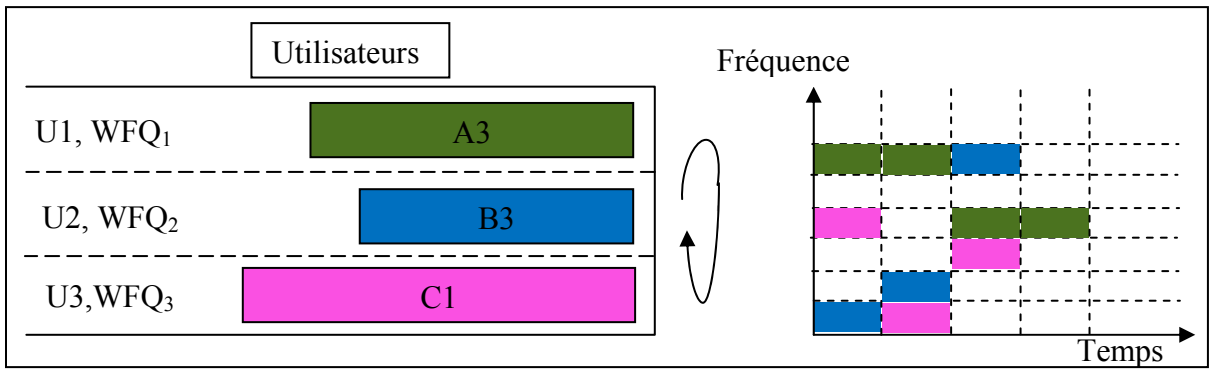

Figure 2.10 Allocation de ressource suivant le WFQ dans les sauts standards

Lorsqu'il y a des ressources disponibles, un tel ordonnancement ne pose pas de défis. Le service se fait au niveau de chaque file suivant le FIFO. L'algorithme servira toutes les files selon leur besoin en bande. La durée de service d'une file est donnée par l'équation (2.27).

$$Temps\_service_{file\_i} = \left\lceil \frac{L_{file\_i}}{Taille_{paquet\_FHSS}} \right\rceil * T_h \tag{2.27}$$

Avec :

$Taille_{paquet\_FHSS}$: la taille en bits d'un paquet FHSS

$\lceil x \rceil$: le plus petit entier naturel supérieur à x

L_{file_i} : la longueur de la file i

T_h : la taille d'un intervalle de temps dans le FHSS

L'opérateur $\lceil x \rceil$ est utilisé parce que la taille de la file n'est pas forcément un multiple de la taille du paquet. À la fin du service, la charge restante a le choix d'attendre la prochaine rafale si son délai d'attente le permet. Dans le cas où ce délai est échu, soit cette charge est envoyée dans une trame incomplète, donc subit un remplissage (Tanenbaum et Wetherall, 2011), soit elle est perdue après que son délai d'attente soit dépassé.

Le problème de remplissage cause une perte de la ressource. Pour limiter la perte, l'intervalle de temps peut être choisi très petit. Une autre option est d'ajuster la taille de l'intervalle de temps pour l'envoi du dernier paquet. Si cet ajustement se fait à la juste mesure de la charge à envoyer, le temps de service se calculera par l'équation (2.28).

$$Temps\_service_{file\_i} = \frac{L_{file\_i} - L_{dernier\_paquet\_ajusté}}{Taille_{paquet\_FHSS}} * T_h + T_{ajusté} \tag{2.28}$$

Avec $L_{dernier\_paquet\_ajusté}$ qui représente la taille en bits du dernier paquet d'une file. Cette taille est ajustée selon la quantité d'information restante dans une file.

Ce temps de service est légèrement plus petit que celui de l'équation (2.27) car, la longueur du dernier paquet dont la taille a été ajustée est supérieure à la longueur d'un paquet normal sans être supérieur à deux fois ce dernier. Alors, le temps ajusté sera inférieur à $2*T_h$. Dans une configuration où ce cas est souvent rencontré, les gains en ressources peuvent se faire sentir.

Supposons maintenant qu'une ou plusieurs files en attentes aient connu des rafales importantes de paquets, et que le débit maximal offert sur un lien ne soit pas suffisant pour éviter que certains paquets ne se perdent. Alors, le problème est lié au dimensionnement du réseau. C'est un phénomène connu qui est souvent représenté dans la littérature par un seau qui se remplit (Tanenbaum et Wetherall, 2011). Le bas de ce seau est percé et l'eau coule à son débit maximal. Si le débit de l'eau qui rentre dépasse celui de l'eau qui sort alors il va falloir augmenter le diamètre de la partie percée du seau, si possible, pour espérer répondre à cette augmentation de débit. Le système à sauts de fréquence traditionnel peut être assimilé à l'exemple du seau. Par exemple dans le Bluetooth, la bande est subdivisée en 79 canaux fixes de 1MHz de large. Si une file contient plus de paquets qu'il n'en faut, les sauts continueront à être de 1MHz de large car, le système n'est pas flexible à ce niveau. Régler ce problème revient à augmenter le débit, c'est-à-dire la bande passante, dans la mesure du possible. Les radios doivent être donc flexibles aux changements de largeur de bande.

De façon générale, l'augmentation de la bande passante, même lorsque la file n'excède pas son maximum est évidemment bénéfique au dépend de l'efficacité spectrale du système. Le temps de service sera réduit si la ressource est disponible, et le temps d'utilisation du système sera réduit, ce qui permet de mieux se préparer aux grandes rafales qui peuvent venir en tout temps. Cette approche sera aussi bénéfique dans le cas où N utilisateurs transmettent sur N canaux dans un système à sauts de fréquence comportant un maximum de N canaux. Si à un moment de la transmission, certains utilisateurs ont fini d'être servis, les canaux libérés seront redistribués aux utilisateurs encore en service. Cela diminue la latence de service du système.

Dans une configuration où il y a plus d'utilisateurs que de canaux disponibles, le service peut se faire suivant le WRR. Supposons un système dans lequel la priorité change fréquemment. Le tableau 2.1 illustre le problème. Le nombre de canaux disponibles est de 1 et nous avons 6 files à servir. Les files peuvent avoir des priorités distinctes. En supposant que chaque file dispose toujours de paquets prêts à être servis, le WRR alloue les intervalles de temps en fonction du poids de chaque file. L'ordre de service est aussi fonction de cette priorité. Pour

faciliter la présentation, la priorité de chaque file représente un pourcentage. Ce pourcentage symbolise le pourcentage de temps requis pour cette file en une seconde. Dans les sauts de fréquence standard, la base des temps est pré subdivisée en des intervalles de temps fixes. Lorsque le pourcentage de temps correspondant à une file n'est pas un multiple de l'intervalle de temps, le nombre d'intervalles alloués à cette dernière est arrondi à l'ordre inférieur ou à l'ordre supérieur, dépendamment des exigences de la file.

Tableau 2.1 Partage d'un canal unique par plusieurs files selon WRR

Numéro de file	**Poids de la file (%)**	**Ordre de service**	**Nombre d'intervalle de temps alloué**
1	33	1	3
2	10	4	1
3	18	3	2
4	7	6	1
5	22	2	2
6	10	5	1

Le nombre d'intervalles de temps est calculé en considérant que 10 intervalles de temps correspondent à 1sec et que les pourcentages symbolisent le pourcentage de temps qu'une file occupe le canal en une seconde. Le nombre d'intervalles de temps est alors donné par l'équation (2.29).

$$Nbre_{\text{int}ervalle\_temps} \approx \frac{Poids_{file}}{10} \tag{2.29}$$

Ce nombre n'est pas toujours un entier mais nous l'avons arrondi à l'entier le plus proche. Ainsi, il peut arriver que certaines files soient servies en dessous de leur exigence pendant

que d'autres sont servies au-dessus de leur exigence. Cette approximation peut donc favoriser certaines files au détriment des autres, on parlera alors d'un problème d'équité dans le service. Le tableau 2.2 en donne une illustration. Dans ce tableau, la perte d'équité représente le nombre d'intervalles de temps en trop ou en moins dont a bénéficié chaque file. Lorsque nous faisons la somme de ces pertes, nous obtenons la valeur nulle. Alors, dans une approche où les intervalles de temps ne sont pas pré subdivisés, nous pouvons offrir à chaque file un service à sa juste valeur et ainsi offrir une meilleure équité dans le service.

Tableau 2.2 Problème d'équité de service dans WRR

Numéro de file	**Poids de la file (%)**	**Nombre d'intervalle de temps alloué**	**Poids correspondant à l'intervalle de temps (%)**	**Perte d'équité de service (%)**
1	33	3	30	-3
2	10	1	10	0
3	18	2	20	+2
4	7	1	10	+3
5	22	2	20	-2
6	10	1	10	0

Certaines files reçoivent un service en dessous de leur priorité et d'autres au dessus de leur priorité. Dans certains systèmes, les paquets les plus exigeants sont servis à l'intervalle de temps plus long. Ceci se fait au détriment des paquets de faibles priorités. Ce problème est souvent rencontré entre les paquets rtPS et les paquets nrtPS ou encore les paquets BE. Une bonne équité revient donc à servir chaque file suivant sa demande en bande. Pour ce faire, la ressource ne doit pas être subdivisée à l'avance. L'idéal serait de faire la subdivision en fonction des files en attente.

Certains travaux comme ceux de (Kadoch et Tata, 2010; Tata, 2009) améliorent l'équité en maintenant la QoS pour les paquets prioritaires rtPS. Ils exploitent un principe de courtoisie. Tel que présenté dans la section 1.3.2, si un paquet de forte priorité peut encore attendre un intervalle de temps sans être perdu ni affecter la QoS, alors un paquet de faible priorité est servi à sa place. Cet algorithme, bien qu'il ait été proposé pour le WiMAX, peut bien s'appliquer aux systèmes à sauts de fréquence. Dans un étalement à sauts de fréquence, le temps de courtoisie sera comparé au T_h, le temps de courtoisie effectif sera alors :

$$Temps_{courtoisie} = \left\lfloor \frac{Temps_{courtoisie\_théorique}}{T_h} \right\rfloor * T_h \tag{2.30}$$

Dans une configuration où l'intervalle de temps peut être choisi de façon dynamique, ce temps de courtoisie peut correspondre au temps de courtoisie théorique. La QoS des files de faibles priorités sera alors davantage améliorée.

2.6 Conclusion

De cette analyse, il ressort que la variation de la taille des paquets et du débit permet d'offrir un service de moindre latence que les sauts de fréquence standards, par exemple le FHSS, dans lesquels la taille des paquets et le débit sont fixes. Elle augmente l'utilisation de la ressource et permet une meilleure équité. Cela valide nos hypothèses de départ. Dans le chapitre 3 nous proposons alors notre approche basée sur des paquets de tailles et de débits variables.

CHAPITRE 3

SAUTS DE FRÉQUENCE AVEC DES TAILLES DE PAQUET VARIABLES

3.1 Introduction

Nous proposons, dans ce chapitre, une nouvelle méthode d'ordonnancement rapide, sécuritaire et fiable pour les réseaux militaires opérants dans des zones hostiles. Notre approche est issue de l'analyse des hypothèses faite précédemment. Cette analyse nous montre qu'il est bénéfique de faire varier la taille des paquets. Cette variation combinée aux sauts de fréquence et à un algorithme d'ordonnancement adéquat, permet respectivement de rendre la détection difficile à l'ennemi et d'offrir une bonne latence de service. Mais pour avoir une telle flexibilité dans les réseaux sans fil, il faut disposer des radios assez puissantes et flexibles, comme les HCR d'Ultra. Toutefois, des contraintes s'appliquent aux HCR.

3.2 Contraintes du réseau

Les HCR fonctionnent dans la bande 4400-5000 MHz. Ils peuvent communiquer en mode Full Duplex en écoutant sur un maximum de 100MHz et en émettant sur un maximum de 20MHz. Les radios doivent pouvoir changer de fréquence de façon rapide. Les réseaux militaires étant destinés aux zones hostiles, ils ne disposent pas de beaucoup de temps pour l'installation ni d'infrastructures fixes. Les réseaux ad hoc semblent donc les plus appropriés et plus rapides à déployer. Les nœuds se regroupent en grappe et désignent un nœud maître qui aura la mission de les interconnecter. Le nœud maître doit avoir une puissance de calcul satisfaisante car il doit ordonnancer les trafics en provenance des nœuds. Dans une grappe, le maître (clusterhead) peut faire du point à multipoint pour communiquer avec ses esclaves.

3.3 Algorithme d'ordonnancement à sauts de fréquence avec des paquets variables

Faire un étalement du spectre par sauts de fréquences n'est pas une nouveauté. Mais rendre la taille de ces sauts variables en est une. Nous avons résolu le problème de variation du débit par la variation de la largeur de bande passante allouée à chaque nœud pendant un intervalle de temps, qui est aussi variable. Nous proposons deux solutions. Une qui fait une subdivision totalement aléatoire et l'autre qui intègre des règles de priorité dans la subdivision de la bande. La première solution cible les trafics dans lesquels toutes les files ont la même priorité, tandis que la seconde peut prendre en compte différents types de trafics. Dans le premier algorithme, les ressources sont allouées selon la charge présente dans les files. Cet algorithme fonctionne comme le WRR dans un premier temps puis après, peut adopter le principe de l'EDF grâce à un algorithme dit correcteur. Dans la seconde approche, pour allouer la ressource aux trafics en attente, nous proposons une version modifiée de l'algorithme de courtoisie de (Tata, 2009). Cet algorithme a été adapté à notre technique d'étalement de spectre.

3.4 Conception de la première version de l'algorithme d'ordonnancement

La première solution peut être subdivisée en deux grandes phases telles que présentées dans la figure 3.1.

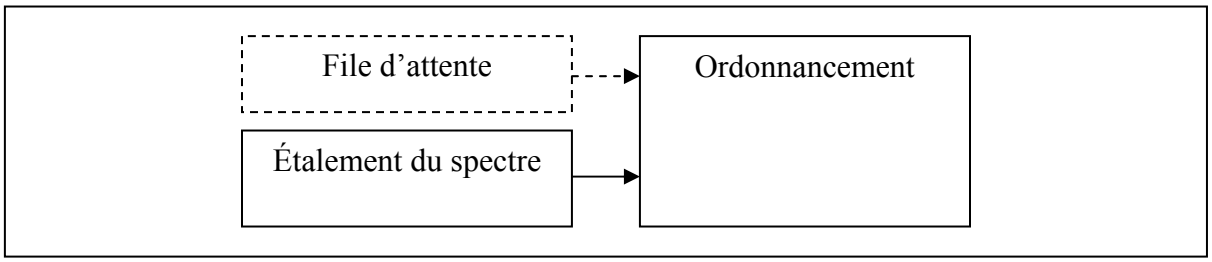

Figure 3.1 Schéma en bloc de l'algorithme 1

L'étalement du spectre regroupe tout le processus de subdivision de la ressource disponible. Dans notre cas, cette subdivision offrira des ressources de tailles variables. La section 3.4.1 présente notre technique d'étalement.

L'ordonnancement spécifie la façon dont les ressources disponibles seront allouées aux files d'attente. Il tient compte de l'état de la file d'attente pour prendre ses décisions. Il est basé sur certaines techniques d'ordonnancement existantes.

La file d'attente est en pointillés car elle n'intervient pas dans notre solution. Elle fournit plutôt les informations utiles pour que l'ordonnanceur fasse la prise de décision.

3.4.1 Étalement de spectre

3.4.1.1 Partage de la bande totale aux grappes

Dans un premier temps, la bande totale disponible est subdivisée pour chaque grappe. Dans une configuration où nous avons des grappes voisines, pour limiter les interférences, chaque grappe fonctionnera dans une bande différente de celle de son voisin. Dans notre cas, la bande totale disponible a une largeur de 600MHz comprise entre 4400 et 5000 MHz. Cette bande sera subdivisée entre les grappes. Rappelons que les HCR considérées ne peuvent écouter plus de 100MHz à la fois. Alors, nous limitons la taille maximale de la bande allouée à chaque grappe à un maximum de 100MHz. Pour distribuer la ressource à la grappe, nous suivons l'algorithme en figure 3.2.

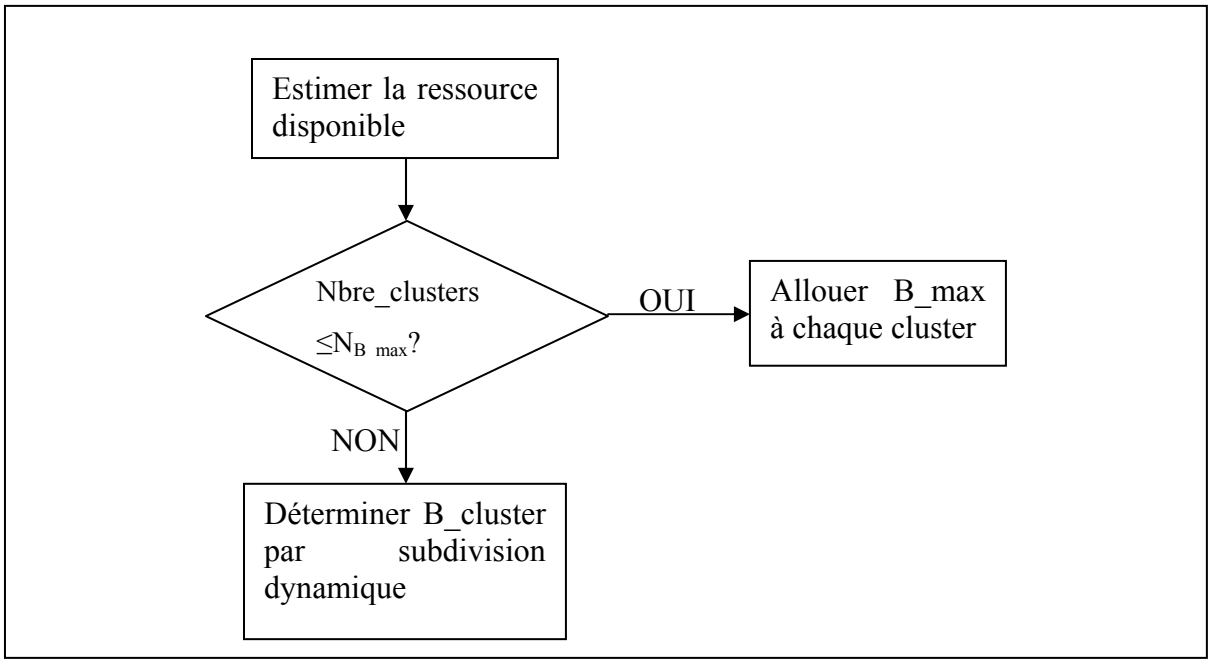

Figure 3.2 Organigramme de la subdivision de bandes aux grappes

L'estimation de la ressource disponible consiste à déterminer la bande passante restante non utilisée qui peut être partagée entre les grappes. Dans notre cas, si aucune partie de la bande n'est utilisée par une grappe au moment de l'estimation, la bande passante restante non utilisée aura une largeur de bande totale, *B_totale*, égale à 600MHz, soit comprise entre 4400 et 5000 MHz.

La condition *"Nbre_clusters* $\leq N_{B\_max}$*?"* vérifie que le nombre de grappes en attente de recevoir la bande ne dépasse pas N_{B_max}. Avec N_{B_max} le maximum de grappe pouvant recevoir la bande passante maximale par grappe *B_max*. Dans notre cas *B_max*=100MHz et N_{B_max} se calcule grâce à l'équation (3.1).

$$N_{B\_\max} = \left\lfloor \frac{B\_totale}{B\_\max} \right\rfloor \tag{3.1}$$

Ainsi, nous avons comme valeurs numériques, N_{B_max}=6. Dans le cas où la condition précédente est remplie, chaque grappe reçoit une bande de largeur *B_max*. Lorsque cette condition n'est pas satisfaite, alors la subdivision dynamique par grappe est utilisée pour allouer les bandes aux grappes. Cette subdivision est détaillée ci-dessous.

Subdivision dynamique par grappe

Elle nous donne la largeur de bande pour la grappe sélectionnée. Dans le cas où le nombre de grappes dépasse N_{B_max}, l'allocation de la bande est fonction de la charge de chaque grappe. La détermination de la largeur de bande se fait comme suit :

- faire la somme de la charge Ch_i de toutes les grappes n'ayant pas encore de bande allouée. Nous entendons ici par charge, le nombre de nœuds prêts à transmettre dans la grappe.

Soit S_{charge} cette somme :

$$S_{charge} = \sum_{i=1}^{Max} Ch_i \tag{3.2}$$

Cette charge totale correspond à la totalité des bandes disponibles soit,

$$Bande_{restante} = B\_totale - \sum_{i=0}^{Cl_{ok}} Bcl_i \tag{3.3}$$

Avec Cl_{ok} le nombre de grappes ayant obtenu de bande passante.

La largeur de bande allouée à chaque grappe est maintenant déterminée par une règle de trois. Cette largeur se calcule en premier pour les grappes ayant plus de charges. Soit Bcl_i la largeur requise pour la grappe i et Ch_i sa charge.

$$Bcl_i = \frac{Bande_{restante} * Ch_i}{S_{charge}} \quad (3.4)$$

Le Bcl_i ainsi calculé doit satisfaire à une exigence : celle de ne pas dépasser *B_max*. Pour cela, lorsqu'à l'issu du calcul, le Bcl_i dépasse *B_max*, alors l'algorithme force $Bcl_i = B\_\max$ et passe à la grappe suivante.

Après ce calcul, nous disposons de la valeur du Bcl_i qui sera allouée à chaque grappe d'indice *i*. Une fois les bandes des grappes allouées, elles sont distribuées aux nœuds au sein de chaque grappe.

3.4.1.2 Subdivision de la bande passante aux nœuds de chaque grappe

Le partage de ressource au sein d'une grappe se fait de façon aléatoire. Les valeurs des créneaux de bande doivent satisfaire deux conditions :

- comprises dans l'intervalle $[W_{\min} \quad W_{\max}]$,
- la somme de tous les créneaux de bande doit être égale à Bcl_i.

Avec W_{min} et W_{max} les valeurs extrêmes des créneaux de bande. Dans notre exemple, elles prennent respectivement les valeurs 1MHz et 20MHz. Une valeur minimale est nécessaire pour s'assurer de toujours être plus grand qu'un certain débit. Ce débit limite est choisi en fonction des contraintes du système. La subdivision de la bande suit l'organigramme présenté en figure 3.3.

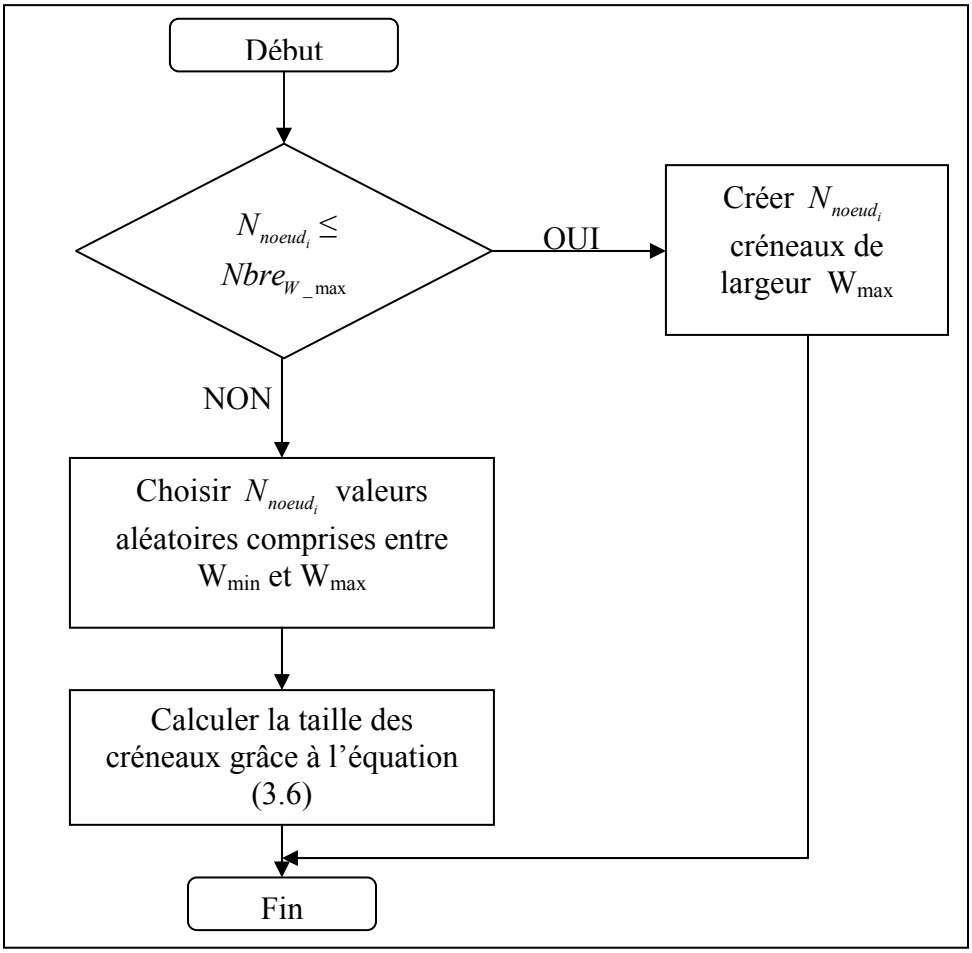

Figure 3.3 Organigramme de l'algorithme 1

Dans cet organigramme, N_{noeud_i} représente le nombre de nœuds ayant des paquets à transmettre dans la grappe d'indice *i*, et $Nbre_{W\_\max}$ le nombre de créneaux de bandes de taille maximale W_{max} qui peut être distribué dans la grappe *i*, on a :

$$Nbre_{W\_\max} = \left\lfloor \frac{Bcl_i}{W_{\max}} \right\rfloor \tag{3.5}$$

Si le nombre de nœuds en attente est inférieur ou égal à $Nbre_{W\_\max}$ alors la bande Bcl_i est subdivisée en autant de créneaux de longueur W_{max} qu'il y a de nœuds en attente. Dans ce cas, si le nombre de nœuds est strictement inférieur à $Nbre_{W\_\max}$, une partie de la bande est inutilisée. Ceci est dû au fait que les radios ne peuvent émettre sur plus de W_{max} à la fois.

Lorsque le nombre de nœuds en attente est supérieur à $Nbre_{W\_\max}$, alors l'algorithme propose des créneaux de taille aléatoire. Pour choisir ces tailles, il choisit N_{noeud_i} valeurs V_a entre W_{min} et W_{max}. Il somme toute les valeurs choisies puis utilise l'équation (3.6) pour déterminer la taille des créneaux. Soit W_j la taille du brin d'indice j.

$$W_j = \frac{V_j * Bcl_i}{\sum_{a}^{N_{noeud_i}} V_a} \tag{3.6}$$

Après ce calcul, si $W_j \geq W_{\max}$ alors il force $W_j = W_{\max}$ puis actualise la valeur de Bcl_i à $Bcl_i = Bcl_i - W_j$. Il passe ensuite au calcul de la longueur du prochain brin avec la nouvelle valeur de Bcl_i. Pour éviter que le système ne reprenne le calcul de certains créneaux, nous calculons en premier les créneaux de plus grande longueur, c'est-à-dire ceux correspondants aux plus grandes valeurs de V_a.

Une fois les longueurs des créneaux déterminées, ils sont disposés aléatoirement sur l'échelle des fréquences. Ensuite, une valeur d'intervalle de temps est choisie pour tous ces créneaux.

3.4.1.3 Choix de l'intervalle de temps

Le choix de T_h se fait de façon aléatoire entre les deux extrêmes T_{min} et T_{max}. Avant ce choix, l'algorithme s'assure que chaque nœud ayant une file en attente dispose d'une quantité d'information plus grande que le seuil correspondant à la plus grande taille que peut prendre un paquet à transmettre. Le calcul de la taille maximale du paquet est donné par l'équation (3.7) dans laquelle T_h et W_i prennent leur valeur maximale.

Si tous les nœuds respectent la condition précédente, l'ordonnanceur choisit de façon aléatoire la taille de l'intervalle de temps T_h entre T_{min} et T_{max}. Les temps T_{min} et T_{max} sont

prédéfinis. Dans ce travail nous avons choisi T_{min}=0.5 ms et T_{max}=2 ms. Le même intervalle de temps est utilisé par tous les nœuds en communication.

Dans le cas où la longueur de la file ayant moins d'information en attente est inférieure à la taille maximale que peut prendre un paquet, la subdivision de la bande se fait suivant l'algorithme correcteur qui est détaillé dans la section 3.4.3.

À partir des valeurs choisies, il calcule la taille des paquets puis les alloue aux usagers en commençant par attribuer les plus gros paquets aux usagers ayant le plus d'information à transmettre.

3.4.1.4 Taille théorique du paquet

À partir des valeurs choisies, la taille théorique des paquets est calculée en se basant sur la formule de la capacité du lien présentée dans l'équation (2.23) et la longueur de l'intervalle de temps T_h. Soit L_{paquet} ,

$$L_{paquet} = \lfloor T_h * C \rfloor \tag{3.7}$$

La figure 3.4 donne un aperçu de toutes les étapes précitées.

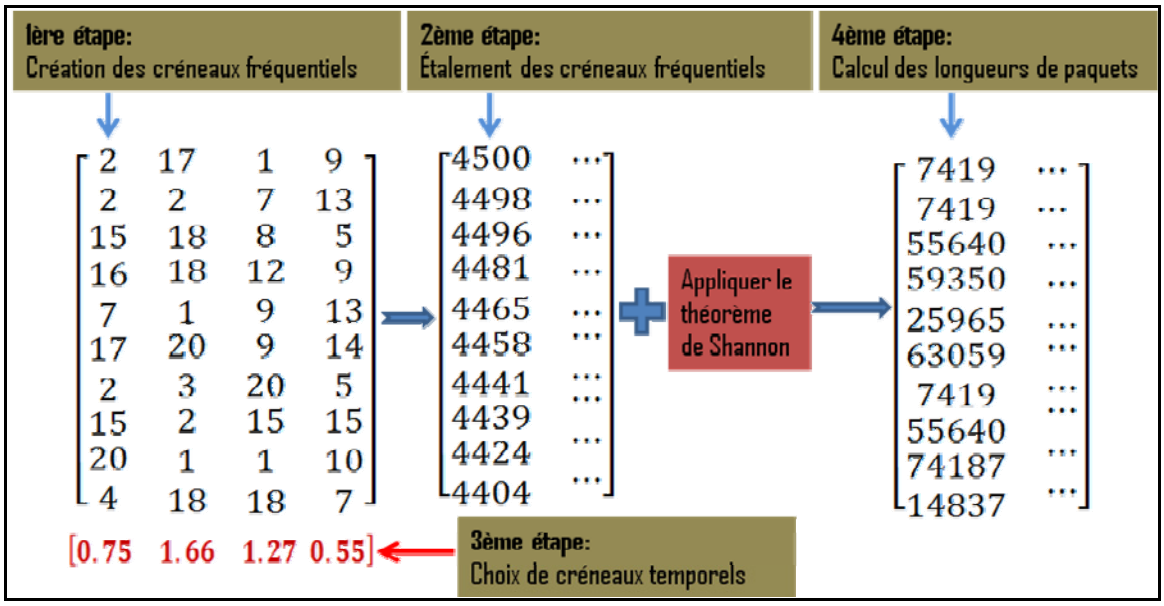

Figure 3.4 Subdivision de la ressource suivant l'algorithme 1

Dans cette figure, la première étape correspond au choix des créneaux de bande passante. La matrice à gauche contient quatre colonnes qui représentent chacune des séries de créneaux de bande passante. La somme des dix créneaux de bande est égale à 100MHz, car chaque grappe ne peut écouter plus de 100MHz à la fois. Les créneaux pris individuellement ne doivent excéder 20MHz, car chaque nœud ne peut émettre sur plus de 20MHz à la fois. Une fois la série de créneaux de bande choisie, la seconde étape consiste à la disposer sur l'échelle des fréquences. Alors, la matrice du milieu dans la figure 3.4 illustre la disposition des créneaux de bande pour une grappe opérant entre 4400MHz et 4500MHz. Après avoir fait la disposition sur l'axe des fréquences, l'algorithme choisit la taille de l'intervalle de temps correspondant à la série de créneaux de fréquence en cours. La taille de cet intervalle est choisie de façon aléatoire dans une plage connue d'avance. Dans la figure 3.4, la matrice correspondant à l'étape 3 propose quatre valeurs choisies de façon aléatoire dans la plage allant de 0.5ms à 2ms. Nous choisirons une de ces valeurs pour la série de créneaux de bande passante en cours. Lorsque la taille de chaque bande passante est connue et que la durée est connue, nous pouvons, à l'aide de la formule de Shannon déterminer la taille théorique des paquets. C'est ce qui est présenté dans la matrice la plus à droite dans la figure 3.4. Les

premières colonnes des matrices de la première étape (créneaux de bande passante) et troisième étape (intervalle de temps) sont utilisées pour déterminer la taille théorique des paquets. Une fois la taille des paquets calculée, il faut faire les associations avec chaque utilisateur. Cette association se fait par un ordonnanceur.

3.4.2 Ordonnanceur de Paquet

L'ordonnancement des paquets consiste à allouer les paquets ainsi formés aux nœuds en attente. Pour cela, il détermine la taille de la charge utile de chaque paquet. Cette taille dépend du codage canal, de la taille de l'entête et du code d'accès. Le codage canal n'étant pas l'objet de cette étude, nous supposerons que le codage FEC 1/3 est utilisé dans la suite de ce travail.

3.4.2.1 Architecture du réseau

Une représentation graphique de notre solution est présentée dans la figure 3.5. Le nœud maître représente le nœud #10. L'architecture du réseau représente une topologie en étoile. Les liens offerts entre le nœud maître et chacun de ses esclaves sont bidirectionnels. Le maître fait du point à multipoint pour communiquer avec ses nœuds esclaves pendant que ces derniers font du point à point pour communiquer avec lui.

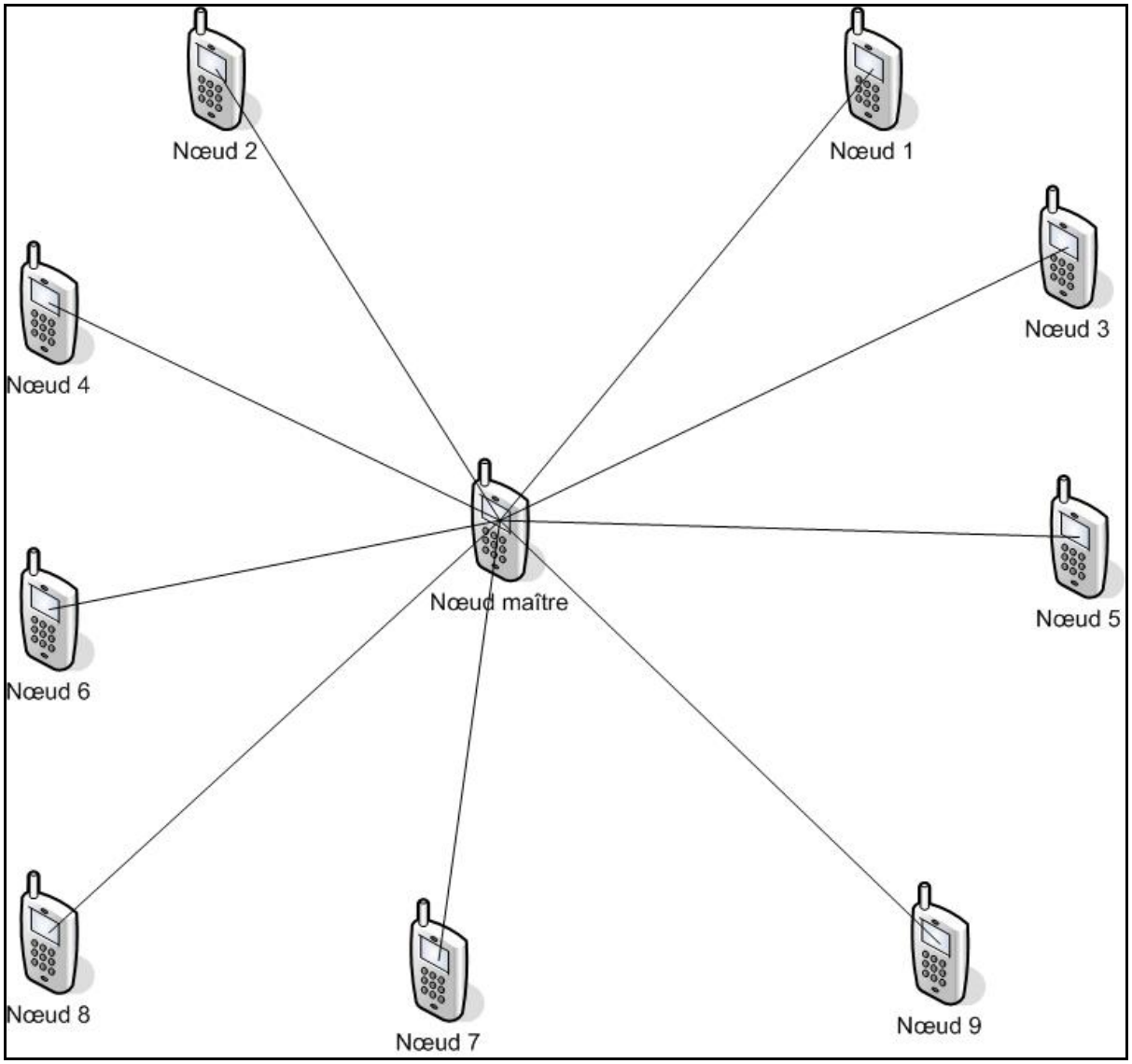

Figure 3.5 Architecture des liens offerts dans le réseau

3.4.2.2 Point à multipoint

Dans l'allocation, le nœud maître est considéré au même titre que les autres nœuds. Par contre vu qu'il sert de point de transit pour tous les messages dans sa grappe, il aura souvent une file d'attente plus importante que celle de ses esclaves. Ainsi, il se verra allouer une

bande plus importante. C'est par cette liaison que le maître informera les nœuds esclaves de l'ordre des sauts et de la durée d'un intervalle de temps.

3.4.2.3 Point à point

Lorsqu'un esclave veut communiquer avec son maître, il fait du point à point. Le maître peut écouter sur une bande totale de 100MHz. Étant donné que toute la bande de la grappe n'excède cette limite, il peut écouter tous ses nœuds au même moment. Les bandes passantes allouées aux nœuds ne se chevauchent pas. Donc, pour extraire le signal de l'utilisateur i, le maître peut utiliser un filtre. Connaissant l'emplacement et la taille des sauts de chaque nœud, il pourra filtrer chaque saut.

L'ordonnancement des paquets dans le point à point se fait au niveau du nœud maître. Dans la première solution, nous supposons que tous les nœuds disposent de trafic de même priorité. Alors, l'association des paquets se fait suivant la taille de chacune des files. Les files qui ont le plus de paquets en attente se verront attribuer les plus grandes ressources. Cette priorité peut varier au cours du temps car après chaque intervalle de temps elle est recalculée. Nous pouvons assimiler ce fonctionnement à celui du WRR. La figure 3.6 illustre le fonctionnement de cet ordonnanceur. La figure 3.6 est une suite à la figure 3.4. Les paquets formés dans la figure 3.4 sont classés pas ordre de grandeur dans la figure 3.6. Ce classement, représente la cinquième étape dans le fonctionnement du premier algorithme proposé, mais la première dans l'ordonnancement des paquets. Lorsque les paquets sont classés, la sixième étape consiste à classer les files en attente selon leur taille. Une fois le classement terminé, l'association se fait suivant l'égalité des rangs. La file qui occupe le premier rang dans la matrice des files des utilisateurs se verra attribuer le paquet le plus gros qui occupe aussi le premier rang dans la matrice des paquets.

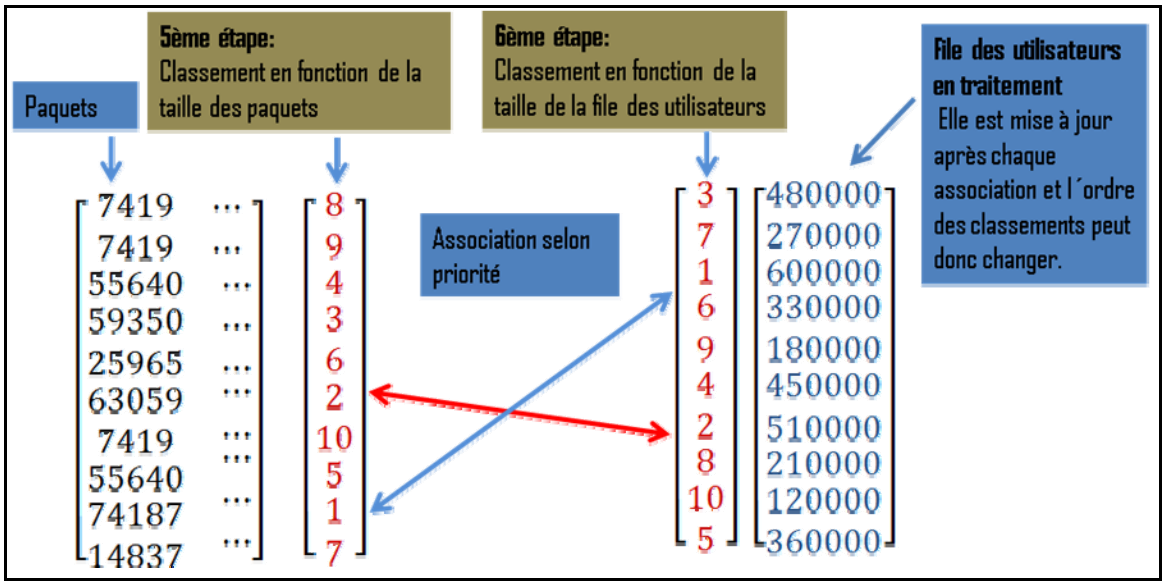

Figure 3.6 Ordonnancement des paquets de taille aléatoire

Cette solution fonctionne correctement jusqu'au moment où certaines files ne disposent pas d'assez de charges pour remplir un paquet. Alors, un algorithme correctif est proposé.

3.4.3 Correctif de l'approche

Une limite relevée dans l'approche précédente est la perte de bande passante lorsque la charge à envoyer est plus faible que la taille du paquet alloué. Cela se produit souvent lors du dernier envoi du nœud. Pour palier à cette faille nous proposons un algorithme correcteur. La figure 3.7 illustre les pertes notées.

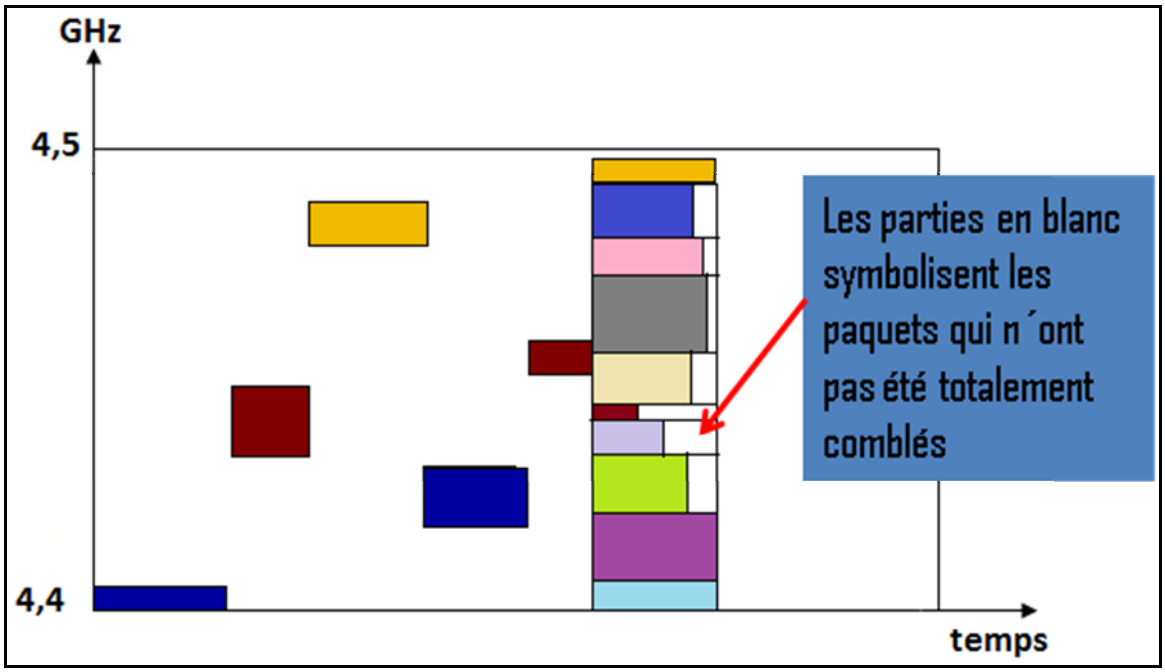

Figure 3.7 Problème de remplissage dans l'algorithme 1

Dans le cas où il aurait des nœuds dont la quantité d'information serait inférieure à la taille du plus grand paquet, le système tiendra compte de l'usager ayant le moins d'informations. La taille maximale d'un paquet est déterminée par l'équation (3.7) dans laquelle la bande passante est de 20MHz. Cette équation donne la taille du paquet s'il n'y a ni codage, ni entête. Pour avoir la taille de la charge utile, il faut déduire ces derniers. L'algorithme correcteur calculera la taille de la bande passante et de l'intervalle de temps requise pour l'envoi de cette information. Ce calcul est fait en s'assurant de ne pas avoir des tailles inférieures aux limites minimales choisies, dans le cas échéant il attribut les limites inférieures possibles, soit 1MHz. Pour déterminer les valeurs ajustées, il pose $T_h = T_{\max}$ puis déduit la largeur de la bande requise à partir de l'équation (3.7), connaissant déjà la taille du paquet à transmettre. Si cette bande est inférieure à 1MHz, alors il reprend la détermination en posant W=1MHz puis il détermine le T_h nécessaire pour envoyer l'information restante dans la plus petite file. Une fois ce calcul fait, il reprend le même processus pour tous les autres paquets dont la quantité d'information est inférieure à la taille du plus grand paquet possible, mais cette fois, il tiendra compte de la taille de l'intervalle de temps déjà calculé

pour le plus petit paquet. Il déterminera juste la largeur de bande requise pour la file indexée. Après cette opération, s'il reste de la ressource disponible, il la partagera de façon aléatoire aux autres usagers dont la taille de la fille dépasse la limite des plus grands paquets possible.

Une fois l'algorithme correcteur appliqué, nous avons la bande perdue dans la figure 3.7 qui devient celle de la figure 3.8.

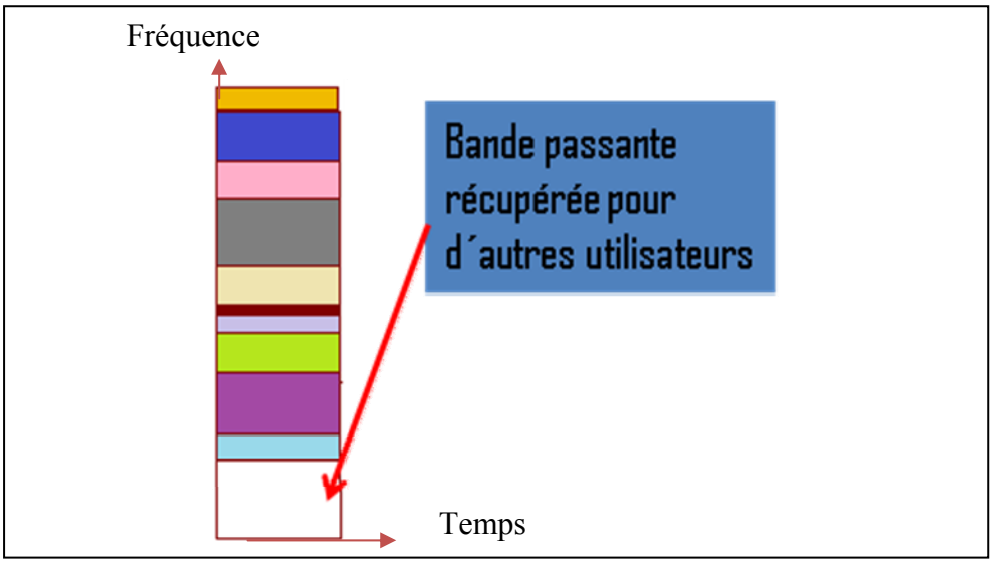

Figure 3.8 Récupération de la ressource perdue lors du remplissage

En récupérant de la bande, nous optimisons l'utilisation de la ressource dans notre solution. Ainsi, le temps de service est réduit. Toutefois, bien qu'étant peu complexe en termes de calcul, cette solution ne tient pas compte de plusieurs types de trafics. Alors, nous proposons un second algorithme un peu plus complexe mais qui prend en compte plusieurs types de trafics.

3.5 Conception de la deuxième version de l'algorithme d'ordonnancement

Parallèlement à la solution précédente, nous proposons une autre approche qui est plus flexible que la première. Cette approche se différencie par sa façon d'allouer la ressource lorsqu'il y a plus de $Nbre_{W\_max}$ utilisateurs en attente.

L'étalement du spectre se fait en même temps que l'ordonnancement et tient compte des informations de la file d'attente. Alors, le synoptique de cette solution ressemble à la figure 3.9.

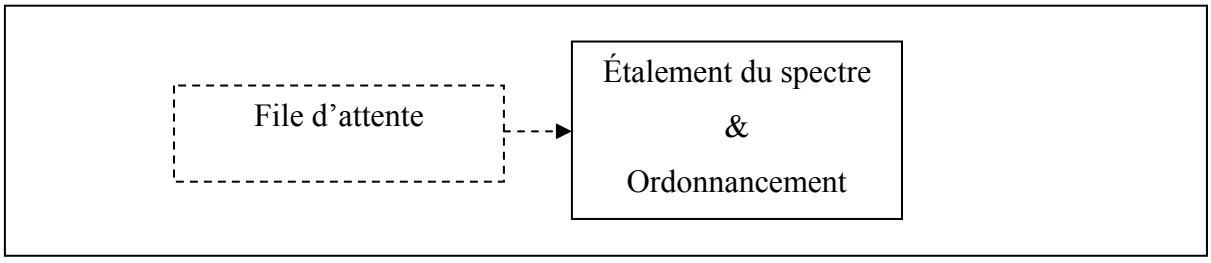

Figure 3.9 Schéma en bloc de l'algorithme 2

3.5.1 Étalement de spectre et ordonnancement

Pour subdiviser la bande de la grappe en des créneaux de bande pour chaque nœud, des règles de priorité sont appliquées. Dans un premier temps, s'il y a assez de ressources disponibles pour servir toutes les files, l'algorithme suit l'organigramme de la figure 3.10.

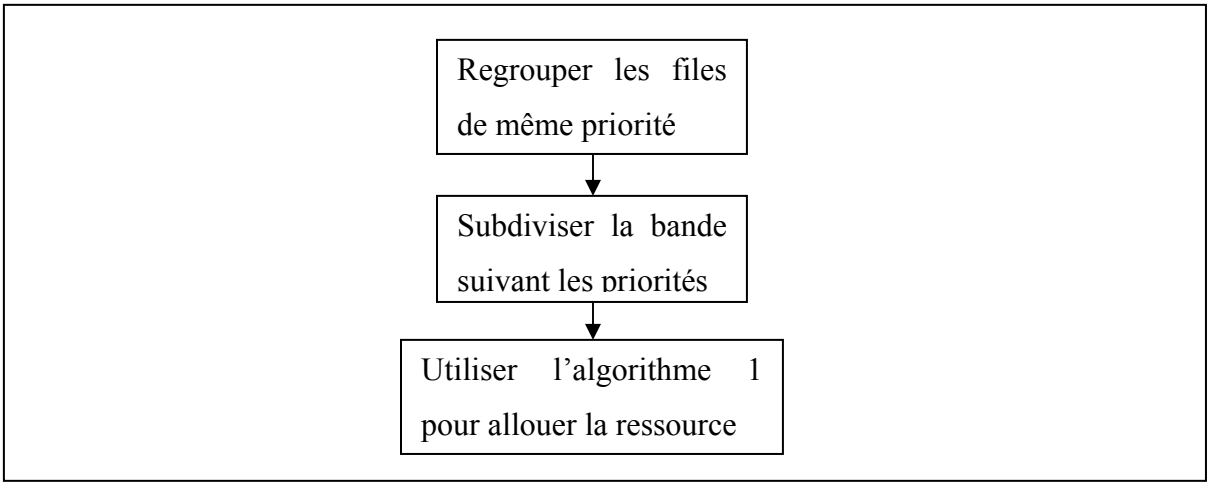

Figure 3.10 Fonctionnement algorithme 2 - ressource suffisante pour la file

Regrouper les files consiste à supposer que tous les trafics de même priorité appartiennent à la même file. Chacune de ces files a donc une priorité qui lui est allouée. La bande totale de la grappe est ensuite subdivisée en des sous bandes. Cette subdivision se fait suivant l'équation (3.8). Soit S_Bcli cette sous bande :

$$S\_Bcli = \frac{Bcl_i * Nbre_{file\_P} * P_{file}}{N_{noeud_i}} \tag{3.8}$$

Avec $Nbre_{file\_P}$ le nombre de files ayant pour priorité commune P_{file}. Ensuite l'algorithme 1 sera utilisé pour déterminer la taille des créneaux de bande à allouer à chaque nœud. Néanmoins, cet algorithme supposera que $Bcl_i = S\_Bcli$. L'allocation se fera pour chaque groupe de trafic de façon indépendante.

Dans le cas où la ressource existante serait insuffisante, alors le fonctionnement est totalement différent du précédent. Dans ce cas, aucune subdivision préalable de la bande ne se fait pour les paquets de même priorité. La subdivision de la bande totale de la grappe se fait en fonction de la priorité du trafic, du coefficient de courtoisie et de la taille des files. Il suit le processus en figure 3.11. Dans la figure 3.11, la bande déductible correspond à la bande passante qui peut être déduite de la bande minimum requise pour les trafics de grande priorité. La bande déductible se calcule grâce au coefficient de courtoisie (Tata, 2009), de telle façon que la QoS ne soit dégradée. Lorsque la bande déductible est calculée, nous pouvons déterminer la *bande_flux_prioritaire*, qui représente la bande théorique minimale requise pour les flux prioritaires moins la bande déductible.

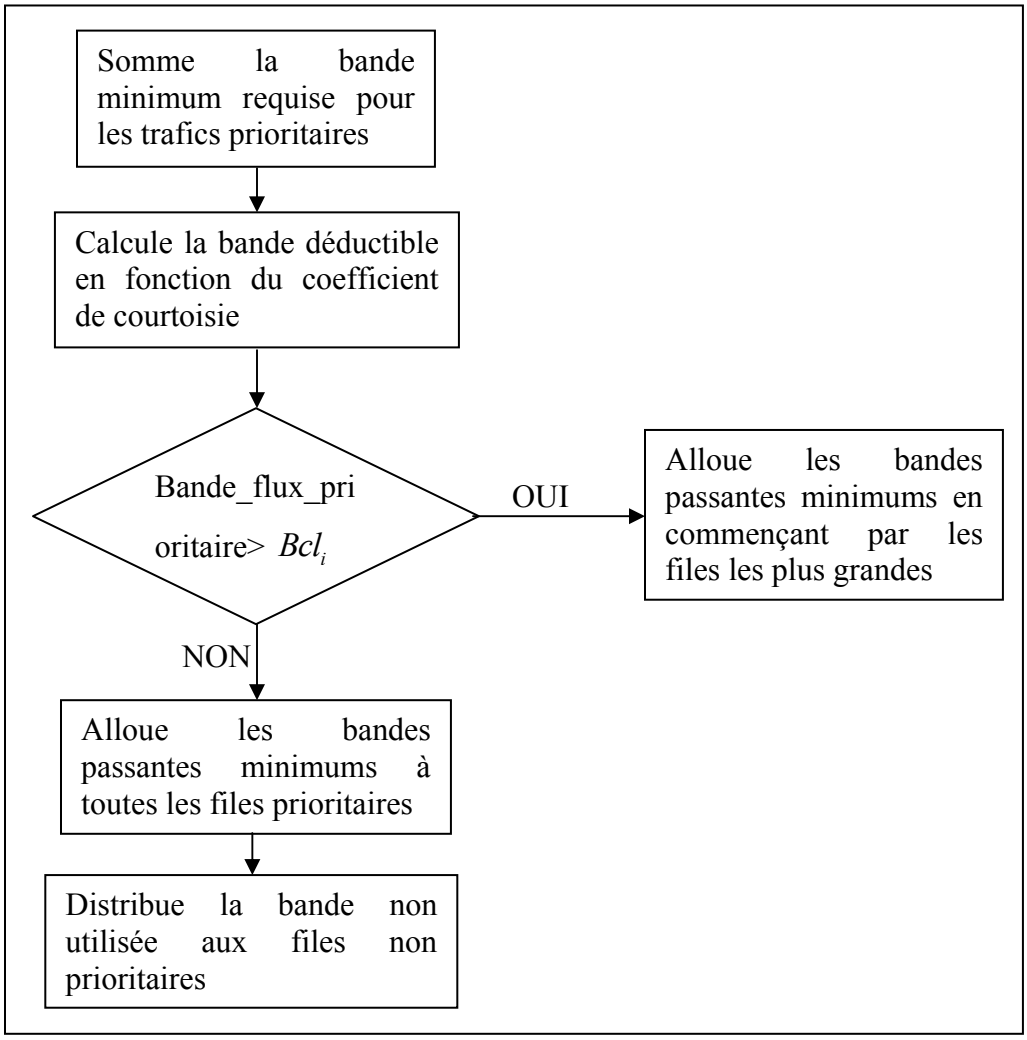

Figure 3.11 Organigramme de subdivision de la ressource - algorithme 2

Comme trafics exigeants, nous pouvons citer les trafics rtPS et nrtPS. Nous nous intéresserons d'abord aux trafics rtPS. Ces trafics exigent une bande minimale. Dans un premier temps, nous ferons la somme de la bande minimale requise pour tous les flux rtPS réunis. Nous calculerons le coefficient de courtoisie pour chacune de ces files. Pour chaque file, nous déterminerons le délai d'attente maximal qui est autorisé. Si ce délai est inférieur à la valeur T_{min} alors nous supposons que cette file doit être immédiatement servie. Dans le cas où ce délai dépasse T_{min}, cette file attendra le prochain service. L'intervalle de temps est

choisi entre $[T_{\min} \quad \xi]$, avec ξ le plus petit délai de courtoisie supérieur à T_{min} que peut tolérer les files prioritaires qui ne seront pas servies pendant ce tour de service.

Après cette étape nous connaissons le nombre de files prioritaires rtPS qui ne seront pas servies pendant ce tour. Nous déduisons alors leur bande minimale (avec la courtoisie) de la somme des bandes minimales calculées précédemment (sans courtoisie). Ainsi, nous avons la bande du flux prioritaire rtPS qui sera partagée aux flux rtPS devant être servis pendant ce tour.

Si cette bande rtPS est inférieure à Bcl_i, alors la bande disponible restante sera d'abord partagée aux flux nrtPS. Car, ceux-ci sont les paquets les plus exigeants après les paquets rtPS. Lorsque la bande restante est non nulle, nous reprenons le même processus que précédemment avec les paquets nrtPS. La différence ici est que nous connaissons déjà le temps de courtoisie minimum que doit tolérer une file pour qu'elle attende le prochain service. Ce temps reste égal à ξ calculer précédemment. Si après élimination des paquets courtois, il reste encore de la bande passante disponible supérieure à W_{min}, alors celle-ci est subdivisée aux trafics non prioritaires, comme le *Best Effort*, BE, suivant l'équation (3.9).

$$W = \frac{L_{file\_i} * Bande_{restante}}{\sum L_{file\_non\_prioritaires}} \tag{3.9}$$

Avec une telle formule, nous espérons garantir une bonne équité de service. Toutefois, le calcul est fait en premier pour les plus grandes files. Lorsqu'après le calcul de W, la valeur obtenue dépasse la bande W_{max} alors, on force cette dernière à cette valeur. On vérifie aussi si W est supérieur à W_{min}. Si la largeur du brin de bande W est inférieure au W_{min}, alors on force $W=Bande_{restante}$. Rappelons que $Bande_{restante}$ est la même que celle utilisée dans l'équation (3.9). L'opération se poursuit jusqu'à ce que $Bande_{restante}$ =0, ou que toutes les files non prioritaires soient servies. Si toutes les files non prioritaires sont servies et que la bande restante n'est pas égale à 0, alors cette bande résiduelle est redistribuée sur les paquets de

fortes priorités tels que rtPS grâce à la même équation (3.9) dans laquelle la somme des files non prioritaires sera remplacée par la somme des files de fortes priorités non servies.

Le calcul du coefficient de courtoisie, dans (Tata, 2009), est donné par :

$$coeff_{courtoisie(i-1)} = K_{i-1} - \left(\left(\lambda_{i-1} + \sigma_{\lambda_{i-1}} \right) * \left(Wq_{i-1} + \sigma_{Wq_{i-1}} \right) \right) - L_{i-1} \tag{3.10}$$

Où :

K_{i-1} : la taille de la file d'attente des paquets courtois,

λ_{i-1} : le taux d'arrivée moyen des paquets de classe d'indice *i-1* ,

$\sigma_{\lambda_{i-1}}$: la variance de λ_{i-1} ,

Wq_{i-1} : le temps d'attente moyen des paquets de classe C_{i-1} dans leur file,

$\sigma_{Wq_{i-1}}$: est la variance de Wq_{i-1} ,

Certains termes se déterminent comme suit :

$$Wq_{i-1} = \frac{\sum_{k=1}^{i-1} \frac{\rho_{k-1}}{\mu}}{(1-\sigma_{i-1}) * (1-\sigma_i)} \tag{3.11}$$

$$\sigma_r = \sum_{k=1}^{r} \rho_k < 1 \tag{3.12}$$

$$L_{i-1} = \frac{\lambda_{i-1} \sum_{k=1}^{n} \rho_k / \mu}{(1-\sigma_{i-1})(1-\sigma_i)} \tag{3.13}$$

$$\rho_i = \frac{\lambda_i}{\mu} \tag{3.14}$$

Avec :

λ_i : Le taux d'arrivé des paquets dans la file i

μ : Le temps de service moyen d'un paquet

Dans cette équation, tous les paquets ont le même temps de service moyen. Pour obtenir ce temps, le débit est calculé pour la largeur de bande minimum requise pour les trafics prioritaires. Nous considérons uniquement les trafics prioritaires car, le coefficient de courtoisie se calcule uniquement pour les flux prioritaires.

Nous avons aussi le temps de tolérance qui donne :

$$\xi_{i-1} = coeff_{courtoisie(i-1)} * \mu \tag{3.15}$$

Ce temps est calculé pour chaque file prioritaire. Lorsqu'il est supérieur à T_{min} alors, comme nous l'avions précisé précédemment, cette file attendra le prochain tour de service. Un tour de service dure pendant un intervalle de temps. Notons que dans notre solution, la loi de service n'est pas toujours fixe. Elle peut varier d'un flux rtPS à un flux nrtPS.

L'algorithme 2, tout comme l'algorithme 1, reprend le même processus après chaque intervalle de temps. L'algorithme correctif n'a pas besoin d'être utilisé dans cet algorithme 2. Un des profits de notre solution est d'améliorer la latence en faisant une allocation et une subdivision très dynamique de la bande. Ceci permet une meilleure utilisation du coefficient de courtoisie. Aussi en faisant des sauts de fréquence avec des tailles de paquets variables, nous maximisons l'utilisation de la ressource. Néanmoins, certaines conditions s'appliquent à la radio pour permettre un bon fonctionnement de cet algorithme.

3.6 Condition d'application

Les solutions proposées dans ce chapitre visent à améliorer la latence dans les réseaux ad hoc. En utilisant les sauts de fréquence, le bon fonctionnement de ces algorithmes dépend des caractéristiques de la radio.

Les radios doivent être assez flexibles pour permettre les sauts de fréquence variables. Cette condition est importante pour avoir une bonne synchronisation. Ces radios devront être rapides. Pour cela, elles doivent pouvoir changer de saut sans avoir de grand retard. Aussi, elles doivent être dotées d'une bonne puissance de calcul.

3.7 Étude mathématique des algorithmes

Mieux que les organigrammes, une étude mathématique permettra de voir les changements apportés par notre solution. Dans cette étude nous mettons en évidence la latence de service de chaque file.

3.7.1 Représentation du signal

Les sauts de fréquence étant de tailles variables, l'équation du signal sera différente de celle des sauts traditionnels. Avec une modulation MFSK, l'équation du signal sera :

$$s(t) = \sqrt{2P} \sum_{i=-\infty}^{\infty} p_{t_h}(t - \sum_{a=-\infty}^{i} T_{h_a}) \cos\left[\omega_i t + \theta_{d_i}(t) + \varphi_i\right] \tag{3.16}$$

Ce signal représente le signal en sortie de l'émetteur. Elle n'est rien d'autre que l'équation d'un signal FHSS-MFSK standard sans le facteur bruit (Holmes, 2007). Dans cette équation, le terme $\sum_{a=-\infty}^{i} T_{h_a}$ représente la somme de toutes les valeurs prises par T_h depuis le début de la communication jusqu'à l'instant i.

T_{h_a} varie de 0 à $T_{h_{i+1}}$ qui représente la nouvelle valeur prise par T_h

ω_i : une des valeurs de saut de fréquence. Elle change à chaque intervalle de temps

$\theta_d(t)$: la modulation de phase qui change à chaque T_{d_i} seconde (durée d'un symbole) pour prendre l'une des *M* fréquences de modulation possible. Nous supposons que la durée d'un saut est supérieure à celle d'un symbole. Cette durée T_{d_i} change à chaque valeur de *i*.

La différence entre cette équation et l'équation FHSS-MFSK standard est que certains termes qui étaient fixes deviennent des fonctions du temps.

3.7.2 Latence de service

Parler de latence de service revient à donner le temps passé par un nœud dans le système de file d'attente plus le temps qu'il a fallu pour le servir. Tel que présenté dans l'équation (3.16), l'expression du signal change après chaque intervalle de temps. Nous étudierons ici quelques cas. Dans ces algorithmes, la latence est améliorée en résolvant deux problèmes. Le premier est la récupération de la bande libérée et le second est la récupération faite par rapport au problème du remplissage.

3.7.2.1 1er cas : récupération bande libérée

Dans ce cas nous illustrerons le service dans un système FHSS versus un système à sauts variables. Nous supposons que le nombre d'utilisateurs est égal au nombre de file et que ce nombre est inférieur au nombre de canaux disponibles dans le FHSS. Ce même nombre d'utilisateurs est servi par les sauts de fréquence variable. Le temps d'attente avant le début des premiers paquets est supposé identique dans les deux situations. La figure 3.12 illustre le service.

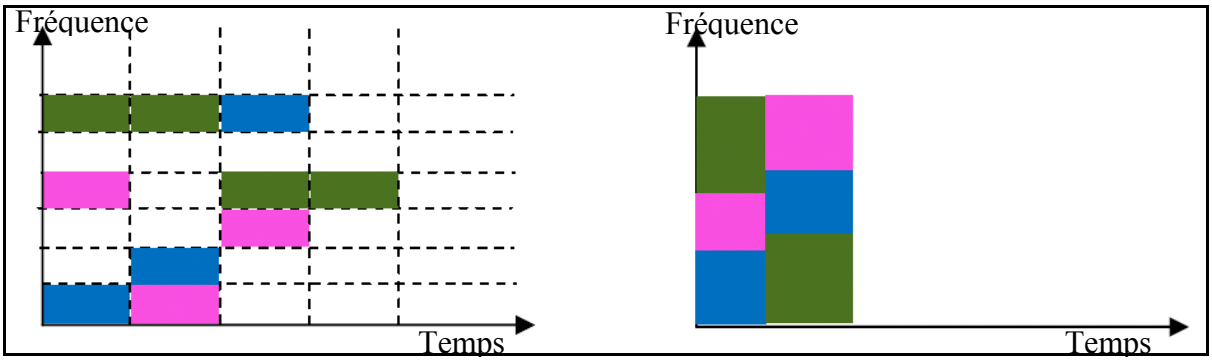

Figure 3.12 Récupération de la bande libérée - étalement FHSS vs FH variable

Dans la figure 3.12, le repère à gauche contient l'ordonnancement du FHSS. Les créneaux temporels et les créneaux fréquentiels sont de tailles fixes et connues d'avance. Nous avons des sauts inusités dans le cas où le nombre d'usagers en attente est inférieur au nombre de sauts possibles. Le repère de droite représente notre approche dans laquelle les sauts sont dimensionnés suivant le nombre et la taille de chaque file en attente. Cette subdivision permet d'utiliser toute la ressource disponible, alors le temps d'utilisation du système et la latence sont réduits. La réduction de la latence implique un temps de service des usagers présents dans notre approche plus faible que celui des usagers présents dans un ordonnancement FHSS. Dans la figure 3.12, nous remarquons que les sauts variables conduisent à un fort taux d'utilisation du système mais peuvent offrir une meilleure latence si les conditions le permettent.

3.7.2.2 2ème cas : remplissage

Dans une configuration où un paquet est incomplet, et que son envoi est imminent, le système fait du remplissage pour pouvoir l'envoyer. Cela cause une perte de ressource. Dans un système où les flux de grande priorité monopolisent la ressource pendant que les trafics de faible priorité attendent leur tour de service, la récupération des pertes de ressources dues au remplissage diminue aussi bien la latence de service des paquets de grande priorité que pour

les paquets de faible priorité qui attendent que la ressource se libère. La figure 3.13 illustre cette situation.

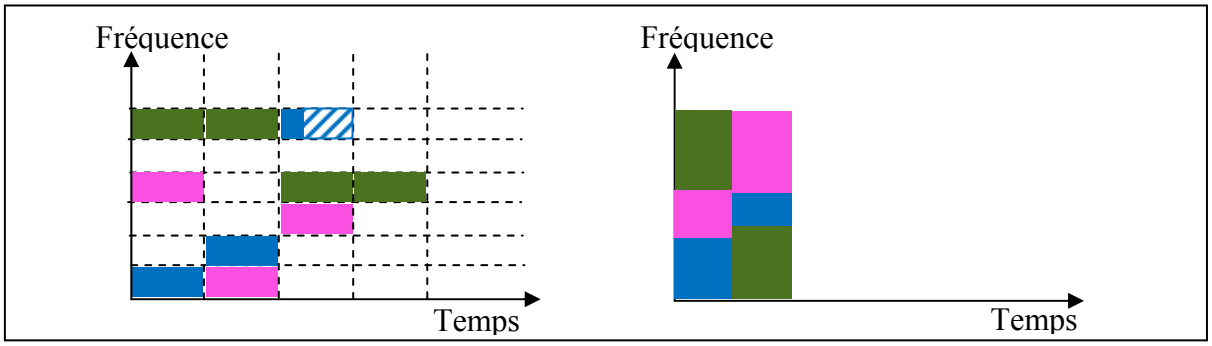

Figure 3.13 Remplissage - FHSS vs FH variable

Les pertes dans les FH variables ne sont pas nulles car si la taille du paquet est inférieure à la limite inférieure, il fait aussi du remplissage. Mais il peut arriver des situations où la correction évite totalement le remplissage.

Par contre, un point sur lequel le FHSS risque de rattraper son retard en latence est le BER car, les cas de retransmission peuvent être réduits pour le FHSS. Un étalement avec des débits variables a un BER plus élevé que le BER du FHSS dont la bande d'un saut est égale à la moyenne de ces débits. Dans le chapitre 2, nous avons vu quelques paramètres qui peuvent améliorer le BER. Nous ferons quelques simulations pour voir l'effet de chacun de ces paramètres et déduire les améliorations à apporter dans les FH-variables. Pour déterminer les probabilités d'erreurs dans les FH-variable, de légères modifications ont été apportées aux équations utilisées pour les sauts de fréquences traditionnels comme le FHSS.

3.7.3 Probabilité d'erreur moyenne du FH-variable

La probabilité d'erreur est le résultat d'une moyenne car, comme nous l'avons vu dans le chapitre 2, la probabilité d'erreur est fonction du débit R_b qui lui est variable. Ce débit étant

variable dans notre solution la probabilité d'erreur moyenne sera alors donnée par l'équation (3.17).

$$P_{b_{moy}} = \frac{1}{N_{ech}} \sum_{i=1}^{N_{ech}} P_{b_i} \tag{3.17}$$

Avec P_{b_i} la probabilité d'erreur correspondant au paquet *i* formé. Cette probabilité est choisie suivant le type d'interférence présent dans le canal. Elle peut être différente pour chaque paquet formé car, le débit peut changer d'un paquet à un autre. La probabilité moyenne est la moyenne de ces probabilités.

3.8 Conclusion

Tel que mentionné au début de ce chapitre, les réseaux militaires exigent de grandes qualités de service. Ils sont aussi confrontés à des problèmes de sécurité. Aussi, nous proposons un nouvel algorithme d'ordonnancement. Il consiste à bien utiliser des sauts de fréquence avec des tailles de paquet variables.

Faire l'ordonnancement avec des tailles de paquet variables n'est pas en soi nouveau car, dans le Bluetooth 802.1.15, la taille des paquets peut varier. Mais cette variation n'est que fonction du nombre d'intervalles de temps alloué au paquet. De plus, elle est prévisible car elle ne s'applique qu'à certains types de paquets connus d'avance. Notre approche quant à elle propose une variation plus complexe mettant à contribution autant l'intervalle de temps que la largeur de bande. En combinant ces variations à la technique d'ordonnancement des paquets nous obtenons de faibles temps de service par rapport aux techniques de saut de fréquence traditionnels.

CHAPITRE 4

RÉSULTATS ET SIMULATIONS

4.1 Introduction

Dans ce chapitre nous présentons les résultats obtenus de nos algorithmes après simulation. Les deux algorithmes seront ensuite comparés à la technique d'étalement à sauts de fréquence traditionnelle : le FHSS. Dans un premier temps, nous examinerons les performances de nos approches face au brouillage. Les taux d'erreurs binaires seront évalués. Ensuite, nous comparons la latence des systèmes utilisant le FHSS face à celle utilisant notre approche à sauts variables.

4.2 Environnement de simulation

Toutes nos simulations ont été faites avec Matlab. À l'aide de ce logiciel, nous avons implémenté les modèles mathématiques de chacun des systèmes. Ces modèles s'inspirent de l'étude faite dans les chapitres 2 et 3.

Pour la probabilité d'erreur binaire, l'équation change selon chaque brouilleur. Nous avons choisi les équations correspondantes aux sauts de fréquence utilisant la modulation MFSK. Les performances du FHSS sont directement données par ces équations. Étant donné que notre approche propose un étalement en saut de fréquence, les mêmes équations seront utiles, mais certains paramètres qui étaient fixes deviendront variables. Il s'agit du débit qui est fonction de la taille du saut et de l'intervalle de temps. Nous évaluerons en premier le *Bit Error Rate*, BER, dans nos approches.

Vu que notre étalement vient avec une technique d'ordonnancement, nous évaluerons la latence du système. Pour cela nous utilisons un système de files d'attente. Ce système est basé sur la loi de Poisson. Nous prévoyons servir un nombre de files ne dépassant pas la limite du système. Pour cela nous utilisons un système de 10 files, voir figure 4.1. Chaque

file à un taux d'arrivée λ_i. Ce taux augmente chaque seconde pour tendre vers le taux maximal de service du système.

Le taux maximal de service du système est calculé suivant l'équation (4.1).

$$Taux\_\max imal\_service = bande\_système * \log_2(1+\frac{S}{N}) \tag{4.1}$$

La bande de fréquence considérée est fixe pour les deux systèmes comparés, soit 100MHz. Les utilisateurs viennent dans la file avec un nombre aléatoire de paquets de tailles fixes, soit 3000 Bits. Si de nouveaux paquets viennent dans la file pendant qu'il n'y a pas de ressources disponibles, ils attendent leur tour de service. La fonction génératrice de la file est :

$$Nbre\_arrivé = poissrnd(\lambda,1,10) \tag{4.2}$$

Avec :

poissrnd : une fonction de Matlab qui génère des arrivées suivant le taux λ

Dans ce cas, le système génère 10 arrivées avec le même taux λ. Les nouveaux paquets sont classés suivant l'ordre d'arrivée dans une matrice conçue à cet effet. À chaque seconde, le taux d'arrivées λ augmente d'une valeur prédéfinie. Cette augmentation du taux d'arrivées permet une augmentation exponentielle des files en attentes.

Pour la seconde approche, le même système de file d'attente a été utilisé. Vu que la qualité de service est garantie pour les flux de hautes priorités tels que rtPS et nrtPS, nous n'avons pas implémenté leur flux. Pour émuler leur présence dans le système, nous faisons plutôt varier la bande disponible pour les flux de basse priorité. Cette variation suppose qu'il y a parfois des paquets courtois, ou que les files de haute priorité sont vides. La figure 4.1 illustre le système de file d'attente.

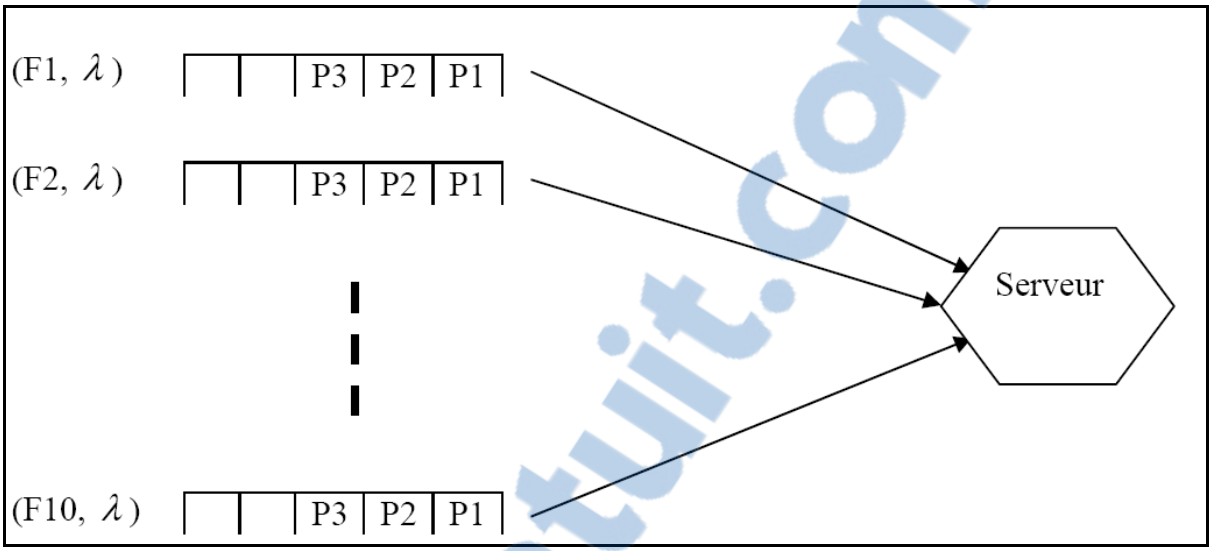

Figure 4.1 Système de file d'attente

Dans cette figure, (F_i, λ) dénote la file numéro 'i' et de taux d'arrivé λ. Le serveur ici aura pour rôle d'allouer des paquets de tailles variables à chaque usager. En plus d'être variable l'étalement de spectre sera à sauts de fréquence. Nos premières simulations portent sur le BER du système. Ce BER devrait être identique pour les algorithmes 1 et 2 car, les deux algorithmes ont justement en commun les débits variables. Ce sont ces débits variables qui induisent des BER aléatoires. La différence sera au niveau de la façon dont ces débits seront alloués. Selon l'allocation faite, la latence n'est pas forcément la même. Alors dans les simulations sur la latence, les deux algorithmes seront pris individuellement.

4.3 Simulation des performances de l'étalement par saut de fréquence

4.3.1 Taux d'erreur binaire

Dans cette sous-section, nous évaluerons le taux d'erreur binaire de notre approche en présence du brouilleur large bande. Les équations (2.10) et (3.17) sont utilisées pour cette simulation. Dans ces équations, les paramètres changent d'un mode d'étalement à l'autre. Le tableau 4.1 présente les paramètres généraux utilisés pour faire la simulation.

Tableau 4.1 Paramètres généraux de la simulation BER- brouilleur à large bande- MFSK

	FH-paquets de taille variable	**FHSS**
Taille d'un saut	Varie de1 à 20 MHz	10MHz
Durée d'un saut	Varie de 0.5 ms à 2 ms	1,25ms
Nombre de canaux	10	10
Largeur bande totale	100MHz	100MHz
Bande utilisée	4400-4500 MHz	4400-4500 MHz
Modulation	MFSK	MFSK

Avec la modulation MFSK dans l'équation (2.10) nous avons :

$$\frac{E_b}{N_0^{'}} = \frac{1}{\frac{N_0 R_b}{P} + \frac{JR_b}{PW}} \tag{4.3}$$

Dans la plupart des simulations rencontrées dans la littérature, l'ordre de grandeur du rapport $\left(\frac{E_b}{N_0^{'}}\right)_{dB}$ est de 0 à 15 dB (Holmes, 2007). Pour nos simulations, nous faisons varier ce rapport de -10 à 15 dB, pour prendre en compte les cas où la puissance du bruit est très élevée par rapport à la puissance du signal. Dans ce rapport le terme R_b est souvent supposé constant. Les termes qui peuvent varier sont :

- la puissance du signal, *P;*
- la puissance du bruit $N_{0;}$
- la puissance du brouilleur *J/W*.

Dans le FHSS, nous supposons que le débit, la puissance du signal et la puissance du bruit sont constants. Alors, varier le rapport signal sur bruit revient à varier la puissance du brouilleur.

Le but de cette simulation est de comparer le BER du FHSS à celui de notre approche. Mais vu que le rapport signal sur bruit est fonction du débit, si nous supposons deux débits différents et une puissance constante dans les deux approches, la comparaison ne sera pas juste, car le rapport signal sur bruit sera différent d'après l'équation (4.3). Il faut s'assurer d'être au même rapport signal sur bruit dans les deux solutions avant de pouvoir comparer leur BER. Pour cela, étant donné que dans notre approche le débit ne suit pas un ordre de variation prédéfini, nous faisons varier la puissance du signal en même temps que le débit varie.

Dans un premier temps, nous nous sommes fixé comme objectif d'atteindre le BER du FHSS. Pour cela, connaissant le rapport signal sur bruit désiré et le débit actuel dans notre approche, nous pouvons déterminer la puissance requise pour l'envoi du paquet en service. Cette puissance se calcule par l'équation (4.4).

$$P_{signal\_FH-\mathrm{var}i} = \frac{R_{b_{FH-\mathrm{var}i}}}{R_{b_{FHSS}}} P_{signal\_FHSS} \tag{4.4}$$

La puissance varie en même temps que le débit. P_{signal_FHSS} et $P_{signal\_FH-\mathrm{var}i}$ sont respectivement mis pour signifier la puissance du signal FHSS et la puissance du signal dans notre approche. La puissance du FHSS est supposée constante. Nous avons aussi $R_{b_{FHSS}}$, le débit dans le FHSS. Ce débit est constant. Le débit $R_{b_{FH-\mathrm{var}i}}$ représente le débit de notre approche est variable. Tant que la condition dans l'équation (4.4) sera respectée, notre approche garantit le même BER que celui du FHSS. La figure 4.2 montre le BER dans les deux approches.

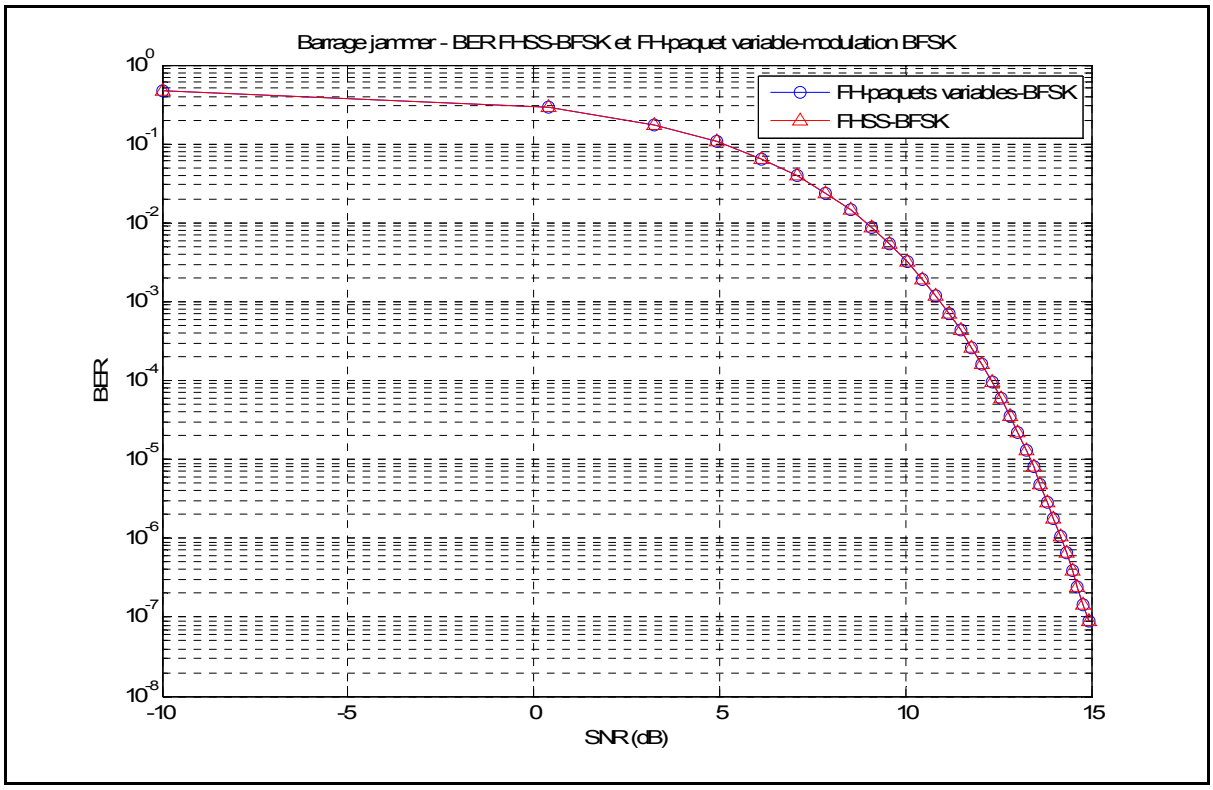

Figure 4.2 Brouilleur large bande-BER FHSS-BFSK vs FH-variable-BFSK

Dans la figure 4.2, les deux approches présentent le même BER, car utilisant l'équation (4.4) pour déterminer la puissance du signal dans notre approche, nous obtenons un rapport signal sur bruit, SNR, identique à celui du FHSS. Le rapport signal sur bruit est remplacé dans l'équation (2.10) pour les deux approches. L'équation (2.10) est utilisée parce qu'elle correspond à la probabilité d'erreur binaire dans une modulation MFSK à sauts de fréquence. Alors, pour avoir le même taux d'erreur binaire que dans le FHSS, les méthodes à débits variables proposées doivent varier la puissance du signal en fonction du débit alloué.

Dans notre approche, la puissance du signal peut être plus grande ou plus petite que celle d'un signal FHSS, si respectivement le débit des sauts variables est supérieur ou inférieur au débit FHSS. En nous intéressant à la puissance totale du système calculée par l'équation (4.5), nous remarquons qu'il y a une compensation lorsque la même bande passante est couverte par les deux systèmes en jeu, à savoir, notre approche à débit variable et le FHSS.

$$P_{signal\_FH-\text{var}i} = \frac{\sum R_{b_{FH-\text{var}i}}}{R_{b_{FHSS}}} P_{signal\_FHSS} \tag{4.5}$$

En supposant que la bande totale de la grappe est de 100MHz, et que toute la bande est subdivisée aux usagers, nous avons la puissance totale de notre approche qui donne :

$$P_{total\_signal\_FH-\text{var}i} = 10 * P_{signal\_FHSS} \tag{4.6}$$

Ceci est égal à la puissance totale des signaux FHSS lorsque la bande des 100MHz est partagée de façon identique entre 10 usagers. Dans la suite, nous supposerons que la puissance d'un signal FHSS est de 100mW.

$$P_{total\_FHSS} = 10 * P_{signal\_FHSS} \tag{4.7}$$

Nous avons donc une puissance identique lorsque la même bande est partagée dans le FHSS et dans les sauts variables. Mais en nous basant sur l'illustration dans la figure 3.12, le FHSS et les sauts variables ne couvrent pas toujours la même quantité de bande. Nous nous intéresserons alors à l'énergie nécessaire dans chacun des systèmes. Cette énergie est donnée par l'équation (4.8).

$$Energie_{signal} = Durée_{signal} * P_{signal} \tag{4.8}$$

L'énergie du signal dans nos approches et celle dans le FHSS sont obtenues après simulation dans le logiciel Matlab. La file d'attente de la figure 4.1 est servie suivant les algorithmes 1, 2 et le FHSS. Nous calculons au fur et à mesure l'énergie nécessaire pour l'envoi de chaque paquet. Ces énergies sont sommées dans chacun des systèmes pour déterminer l'énergie totale requise par chaque système pour servir la file en attente. La figure 4.3 montre les résultats obtenus.

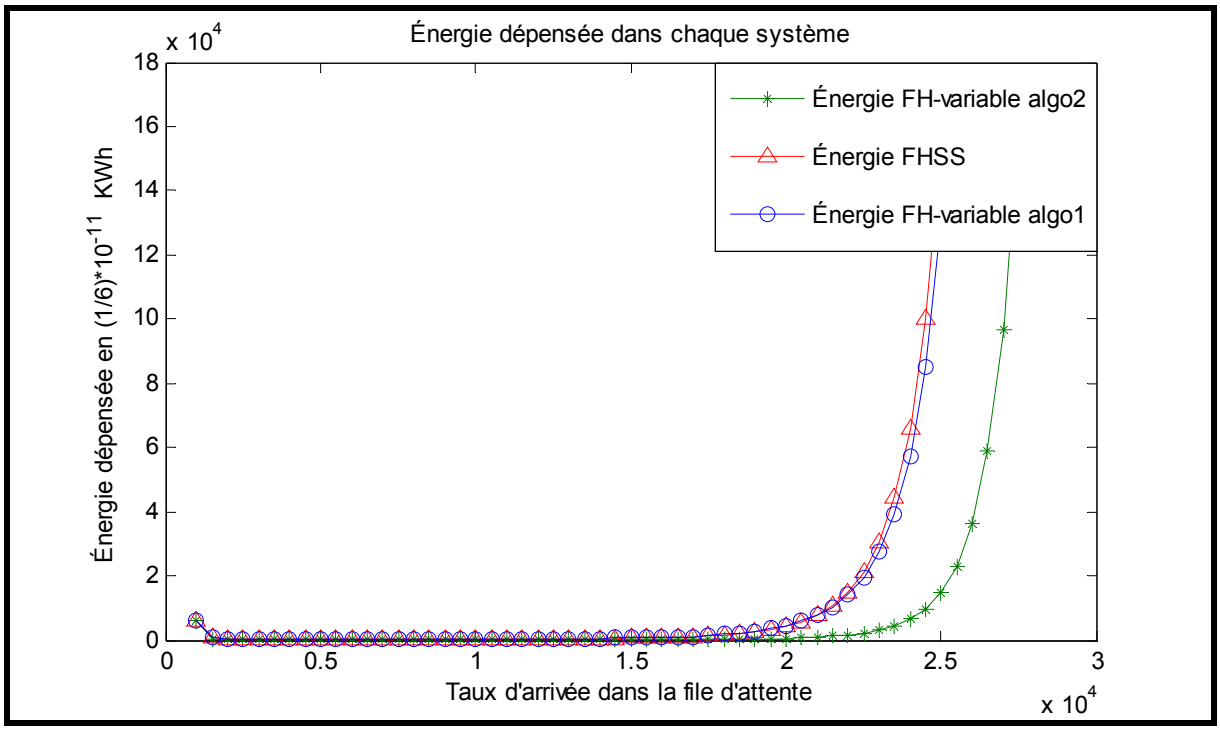

Figure 4.3 Comparaison de l'énergie dans les systèmes à sauts variables et dans le FHSS

Dans la figure 4.3, nous remarquons que, pour un même nombre de paquets, l'algorithme 1 et l'algorithme 2 dépensent moins d'énergie que le FHSS. En nous basant sur l'équation (2.15) qui nous donne l'énergie par bit, vu que cette énergie est constante dans toutes les approches, le seul paramètre qui peut causer une différence au niveau de l'énergie est la durée de service de la file d'attente dans les différentes approches. Néanmoins, en considérant le problème de remplissage présenté dans la section 2.5.2.1, une approche qui subit moins de perte de ressource due au remplissage dépensera moins d'énergie. Pour expliquer la différence observée au niveau de l'énergie, nous reprenons les mêmes simulations sur la file d'attente pour observer le nombre de bits perdus et la latence de chaque approche.

4.3.2 Ressource perdue

La figure 4.4 montre la ressource perdue en fonction du taux d'utilisation du système. Notons que ce taux d'utilisation est le rapport entre la quantité de ressources utilisées et la

capacité totale du système. Pour déterminer la ressource perdue, nous faisons la somme des bits ayant subi le remplissage dans chacune des approches.

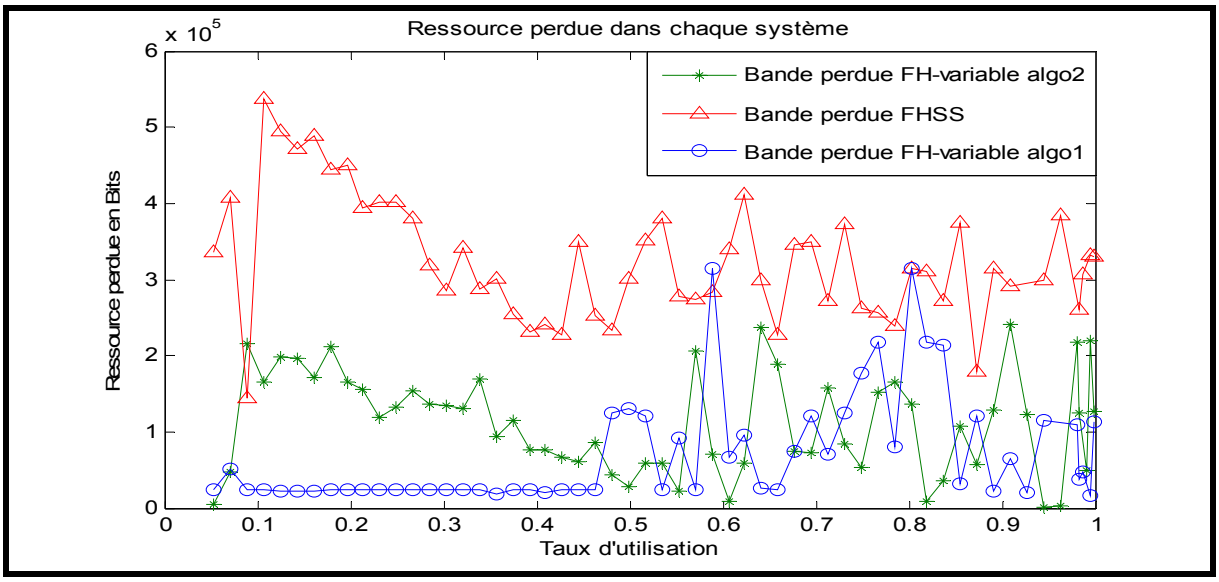

Figure 4.4 Comparaison des ressources perdues dans le FHSS vs les algorithmes à sauts variables

Les courbes représentées dans la figure 4.4 regroupent le nombre de bits perdus par l'effet de remplissage des paquets incomplets en fonction du taux d'utilisation du système. Nous remarquons que pour un taux d'utilisation faible, l'algorithme 1 perd moins de ressources que l'algorithme 2 et le FHSS. Cet avantage de l'algorithme 1 est dû au fait que l'algorithme de correction proposé pour le dernier envoi des files permet d'avoir des pertes de ressources presque nulles. L'algorithme 2 n'a pas un fonctionnement particulier lors de l'envoi des derniers paquets des files en attente. Par contre, sa subdivision de la ressource en fonction de la taille des files permet d'avoir moins de perte que le FHSS pour de faibles taux d'utilisation du réseau. Au fur et à mesure que le nombre de paquets à transmettre augmente, les corrections de l'algorithme 1 sont moins visibles, car la taille des paquets doit être supérieure à un certain seuil. Ce seuil est calculé en considérant la plus petite taille que peut prendre la bande passante et la plus petite taille que peut prendre l'intervalle de temps. Pendant que l'algorithme 1 perd de la ressource pour des taux d'utilisation élevés, l'algorithme 2 met à

profit sa subdivision contrôlée de la ressource pour tendre vers la demande requise à chaque file. Le FHSS quant à lui perd plus de ressources que les deux approches car, il ne dispose d'aucun moyen pour limiter les pertes dues au remplissage des paquets. De cette simulation il ressort que les sauts de fréquence variables permettent un ordonnancement plus adapté aux trafics.

4.4 La latence de service

4.4.1 Cas où le nombre d'utilisateurs est égal au nombre de canaux disponibles

Dans un premier temps, nous avons simulé la latence de service des algorithmes 1 et 2. Nous avons comparé ces latences à celle du FHSS. L'arrivée des utilisateurs dans la file suit la loi de Poisson telle que présentée au début de ce chapitre. Le système de file d'attente utilisé ici compte 10 files indépendantes, ce qui correspond au nombre de canaux FHSS. Dans cette simulation, le taux d'arrivée des utilisateurs augmente et tend vers le taux de service maximal que peut supporter le système. Ce taux de service correspond à la capacité de Shannon calculée par l'équation (2.23) dans laquelle la bande passante est égale à la bande totale disponible dans le système, soit 100MHz dans notre simulation. Il correspond au nombre de petits paquets de 3000 Bits qui peut être servi par le système en une seconde. À chaque seconde de nouveaux petits paquets arrivent dans la file des utilisateurs. Nous faisons le point des petits paquets en attente et de ceux en traitement après chaque seconde pour déterminer le nombre de paquets dans le système à un instant donné.

Ce principe a été utilisé pour déterminer le nombre de paquets présents dans nos deux approches utilisant un ordonnancement à sauts de fréquence à débit variable et dans le FHSS. Ainsi, nous avons la latence des trois systèmes qui est présentée dans la figure 4.5.

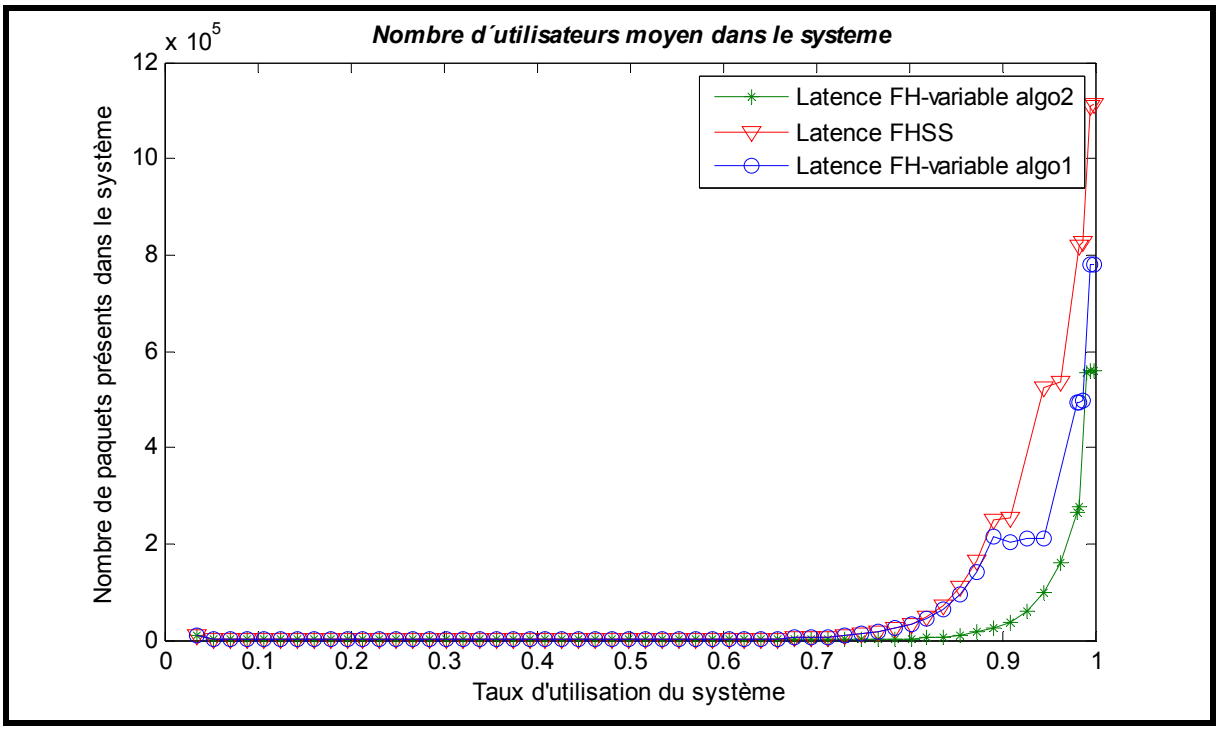

Figure 4.5 Comparaison de la latence des algorithmes 1 et 2 et de la latence du FHSS- bande de la grappe = 100MHz- 10 files en attente

Nous entendons par nombre de paquets présents dans le système, la taille de la file d'attente à un instant donné. Cette taille diminue au fur et à mesure que les paquets sont servis. Alors, plus il y a de paquets présents dans une file d'attente, plus la latence de service est grande. De cette figure, nos deux approches offrent une meilleure latence par rapport au FHSS, car pour un même taux d'utilisation du système la taille de la file d'attente dans nos approches est plus faible que dans le FHSS. Notons que, pour avoir un tel résultat nous avons supposé que la taille des files d'attente n'est pas un multiple de la taille des paquets FHSS. Cela implique qu'une partie de la bande perdue par remplissage avec le FHSS, peut être à la base de ce retard. Dans les sauts de fréquence variable, la ressource est toujours subdivisée en fonction du nombre de file en attente. Dans le FHSS la subdivision est fixe, soit de 10 canaux. Lorsqu'un canal se libère, le FHSS ne le redistribue pas aux files en cours de service. Alors, la ressource libérée dans le FHSS reste inusitée. Par contre, dans les deux algorithmes à sauts de fréquence variable proposés, lorsqu'un canal se libère, sa ressource est redistribuée

aux files en cours de service. Cela permet de maximiser l'utilisation de la ressource. Avec une telle utilisation de la ressource, la latence des deux algorithmes est inférieure à celle du FHSS. En considérant uniquement les deux algorithmes, nous remarquons que l'algorithme 2 a une latence plus faible que la latence de l'algorithme 1. Cette différence s'explique par la façon dont chacun de ces algorithmes alloue la ressource à ses utilisateurs. L'algorithme 2 subdivise la ressource selon la priorité de chaque file. Cette subdivision permet d'avoir une bonne équité dans le service des files. Par contre, l'algorithme 1 subdivise la ressource de façon aléatoire, et ceci indépendamment des files en attente. Il alloue les ressources ainsi subdivisées selon la priorité des files. L'algorithme 1 offre un service moins équitable par rapport à algorithme 2.

Dans la figure 4.5, pour un faible taux d'utilisation (inférieur à 70%), la différence de la latence entre les différentes approches est peu visible. Ceci veut dire que les utilisateurs sont servis aussitôt qu'ils viennent dans la file. Vu que les utilisateurs viennent dans la file après chaque seconde, alors nous pouvons dire que le temps de service est inférieur à une seconde donc aucun paquet n'attend dans la file. Lorsque le taux d'utilisation dépasse 70%, la courbe prend une allure exponentielle. Alors, les utilisateurs commencent à avoir un temps de service qui se rapproche de plus en plus d'une seconde. À partir de ce taux d'utilisation, le nombre de paquets en jeu devient important alors la différence entre les latences commence à se remarquer.

Alors, pour un faible taux d'utilisation du système, le choix d'un algorithme par rapport à l'autre est sans importance. Mais lorsque ce taux devient considérable, l'approche dans l'algorithme 2 permet d'avoir une meilleure latence et une meilleure économie d'énergie par rapport au FHSS et à l'algorithme 1. L'algorithme 1 a une latence et une dépense énergétique inférieures à celles du FHSS mais supérieures à celle de l'algorithme 2. En plus, grâce à la correction qui est faite pour le dernier envoi des files en attentes dans l'algorithme 1, nous enregistrons moins de pertes de ressources face au remplissage par rapport à l'algorithme 2 et au FHSS dans les cas de faible taux d'utilisation du système.

En visualisant les figure 4.5 et 4.4, nous pouvons dire que la latence est le paramètre qui a le plus d'influence sur l'énergie du système. La latence présentée dans la figure 4.5 correspond à un système dans lequel le nombre de canaux est égal au nombre de files en attente. Nous nous intéresserons maintenant à la latence dans le cas où le nombre de files est supérieur au nombre de canaux FHSS.

4.4.2 Cas où le nombre de files supérieures au nombre de canaux FHSS

Dans cette sous-section nous simulerons la latence des algorithmes 1 et 2 puis le FHSS avec un système de files d'attente contenant plus de files que le nombre de canaux FHSS. Nous supposons donc un système de file d'attente contenant 15 files. Tous les autres paramètres de simulation restent les mêmes que dans la simulation faite dans la sous-section 4.4.1. La figure 4.6 montre les résultats de la simulation.

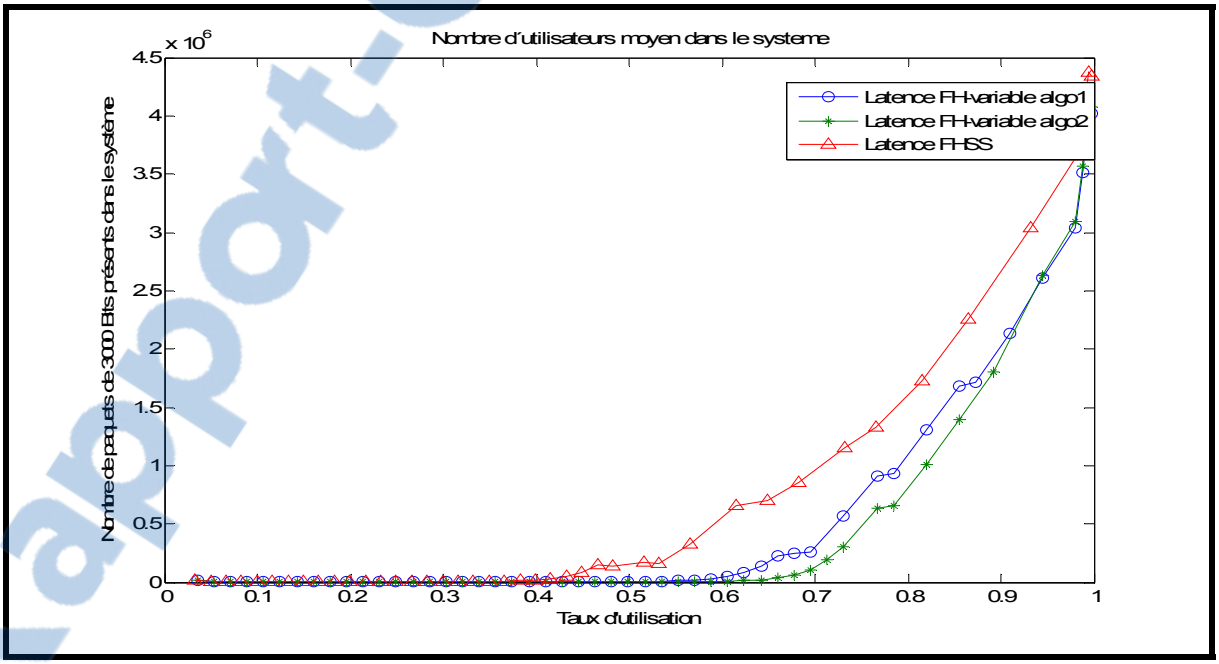

Figure 4.6 Comparaison de la latence des algorithmes 1, 2 et le FHSS dans un système où le nombre de files est plus grand que le nombre de canaux FHSS

Dans la figure 4.6, nous avons simulé les algorithmes 1, 2 et le FHSS dans un système où le système de files d'attente compte 15 files. La latence du FHSS est plus grande que la latence des algorithmes 1 et 2. Ceci est dû au fait que le système FHSS simulé ici contient 10 canaux. Lorsque les 15 files d'attente sont générées, seulement 10 sont traitées pendant que les autres attendent leur tour. Ceci cause une augmentation du nombre de files en attente de service. Par contre, dans nos deux algorithmes proposés, la ressource est subdivisée en 15 canaux, ce qui permet de servir toutes les files au même moment.

Sachant que la bande passante totale est la même dans les trois approches, soit de 100MHz, la latence élevée du FHSS s'explique aussi par la non-utilisation de toute la ressource disponible. En effet, lorsque 10 files sont en service, les 5 autres files restantes attendent. À la fin du temps de service des 10 files, si ce temps de service ne dépasse pas une seconde, alors le système servira 5 files tandis qu'en réalité ce même système peut servir 10 files. Alors, nous avons 5 canaux de libres. Ce problème n'est pas rencontré dans nos approches à sauts de fréquence variable, car la subdivision de la ressource est flexible. Il ressort de cette simulation que dans un environnement où le nombre de files à servir dépasse le nombre de canaux disponibles, les sauts de fréquence variables offrent une latence plus faible que celle du FHSS. La différence entre la latence dans les deux approches à sauts variables est due à la flexibilité de l'algorithme 2 par rapport à l'algorithme 1. Plus les sauts de fréquence variables sont flexibles ou encore équitables, moins nous accusons de temps de latence. Nous nous intéressons maintenant à savoir le comportement de chacune des approches dans un système où le nombre de file est inférieur au nombre de canaux.

4.4.3 Cas où le nombre de files est inférieur au nombre de canaux FHSS

Nous reprenons nos simulations en considérant le cas dans lequel le nombre de canaux FHSS est inférieur au nombre de files en attente. Chaque canal a une bande passante de 10MHz et une taille d'intervalle de temps de 125 ms. La figure 4.7 présente les résultats de la simulation.

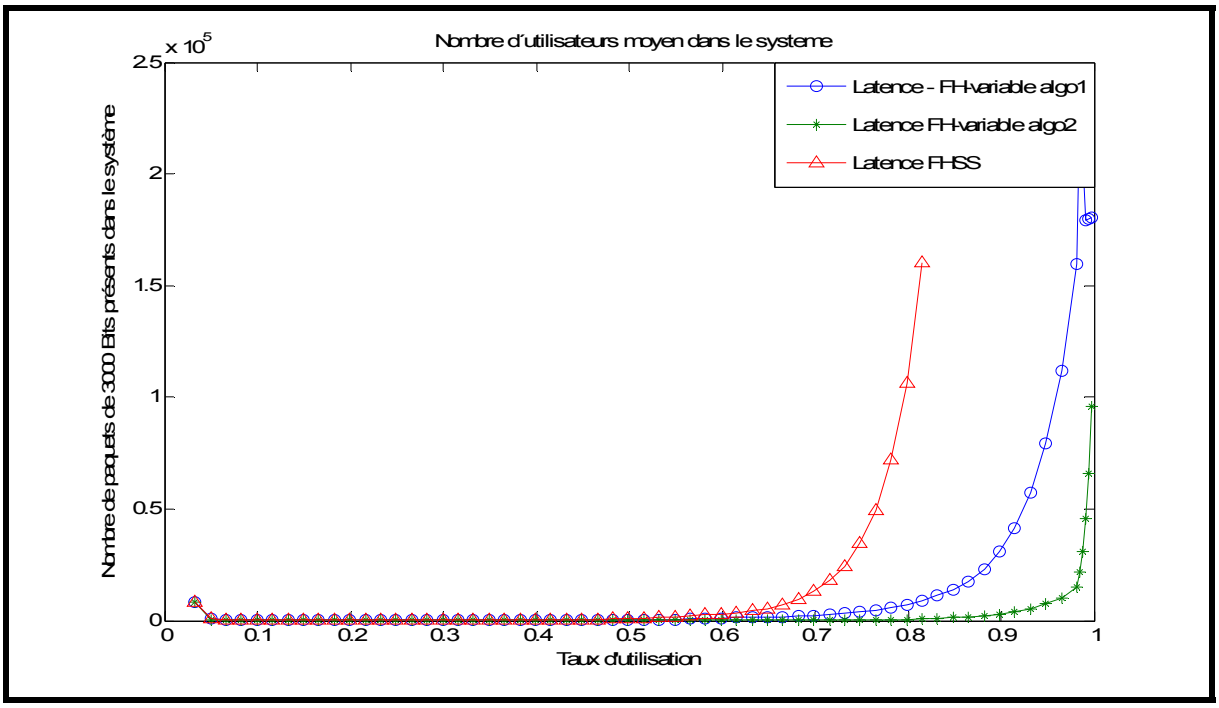

Figure 4.7 Comparaison – latence du FHSS vs saut de fréquence variable algorithmes 1 vs algorithme 2 dans un système où le nombre de files en attente est inférieur au nombre de canaux FHSS

Dans la figure 4.7, nous comparons la latence du FHSS et de nos deux algorithmes dans un système qui génère 8 files d'attente à chaque seconde. La latence du FHSS est plus grande que la latence des deux algorithmes proposés, car le service de 8 files d'attente entraine une perte de 20% de la ressource disponible soit, l'utilisation de 8 canaux sur les 10 disponibles dans le FHSS. Dans nos approches cette ressource n'est pas perdue, car toute la ressource disponible est subdivisée aux 8 files en attentes. Dans le FHSS, lorsque le temps de service des files en attente est inférieur à 1 sec, de nouvelles files ne sont pas encore générées, donc le FHSS de 10 canaux fonctionne comme un FHSS de 8 canaux. Dans le cas où le temps de service dépasse 1 sec, de nouvelles files s'ajoutent aux files en service et le FHSS utilise ses 2 canaux disponibles pour servir deux des nouvelles files d'attente. Les 6 autres files en attente pour le FHSS attendront leur tour de service. Les algorithmes utilisant les sauts de fréquence variable permettent d'avoir une faible latence, car ils utilisent constamment toute la ressource disponible. L'algorithme 2 a une latence plus faible que celle de l'algorithme 1,

car il offre un ordonnancement plus équitable que ce dernier. Grâce à cette simulation, nous pouvons conclure que pour un nombre de files d'attente inférieur au nombre de canaux dans le FHSS, la substitution du FHSS par les méthodes à sauts de fréquence variable permet d'avoir une faible latence. Toutes les simulations faites précédemment considèrent une bande totale fixe de 100MHz. Cette bande est multiple de la bande d'un canal FHSS. Sachant que la bande totale de la grappe n'est pas forcément multiple de la taille d'un canal FHSS, nous reprenons nos simulations pour une bande de 69MHz.

4.4.4 Cas pour une bande totale de la grappe de 69MHz

Les simulations précédentes supposent que la bande allouée à la grappe est un multiple de la bande d'un saut du FHSS. Dans une configuration où la bande d'une grappe est susceptible de changer, lorsque la bande disponible n'est pas un multiple de la taille d'un saut FHSS alors il y a perte de bande dans le FHSS. Par contre, nos solutions utilisent cette bande perdue grâce à leur flexibilité dans la subdivision et dans l'allocation. La figure 4.8 présente le résultat pour une bande disponible de 69MHz.

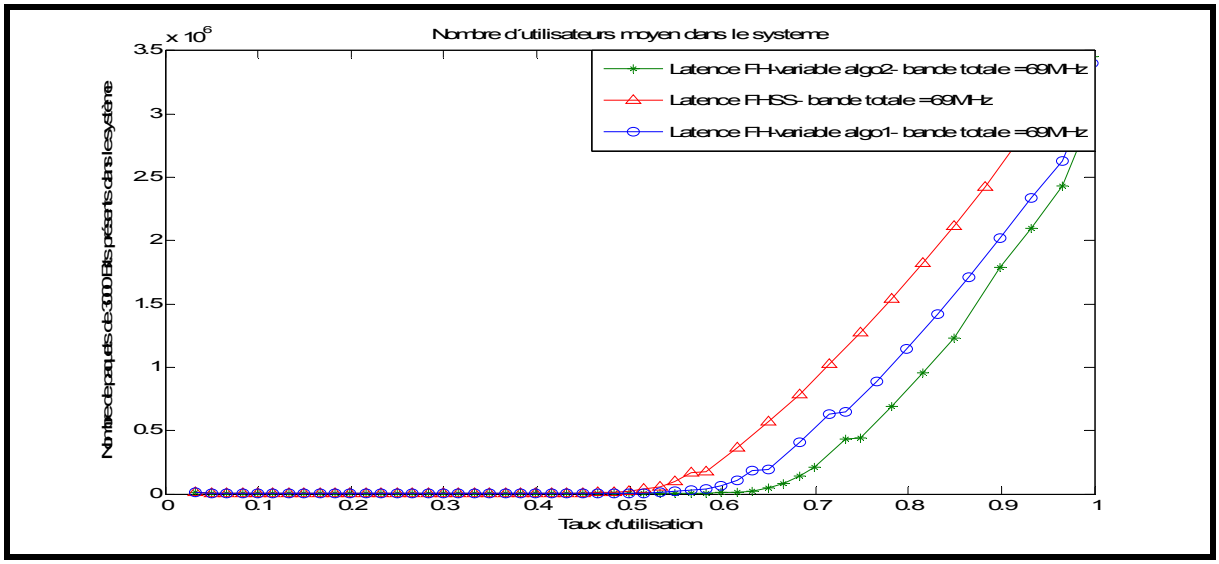

Figure 4.8 Latence algorithme 1 vs algorithme 2 vs FHSS- Cas où la bande totale de la grappe est égale à 69MHz

Dans la figure 4.8, la différence entre la latence du FHSS et celle de nos approches est due à la perte de ressource subite par le FHSS. En effet, vu que les sauts de fréquence dans le FHSS ont une taille fixe de 10 MHz, les 69 MHz donneraient 6 canaux. Dans nos deux algorithmes, l'algorithme 2 offre la plus faible latence, car elle offre un ordonnancement plus équitable que l'algorithme 1. Dans cette simulation nous avons supposé que la bande totale de la grappe garde la valeur de 69 MHz pendant tout le service des files en attente. Mais, dans une grappe où nous avons plusieurs types de flux allant des flux prioritaires (rtPS) aux flux sans priorités, la bande allouée à chaque type de flux peut être très dynamique.

4.4.5 Cas pour une bande passante dynamique

Pour simuler la bande passante dynamique, nous avons mis sur pied un système de file d'attente dans lequel les paquets viennent suivant la loi de Poisson. Ces files sont regroupées pas groupe de service. Nous avons des files rtPS, nrtPS et BE. La qualité de service est garantie pour les flux rtPS et nrtPS car, la ressource leur est allouée en premier suivant leur besoin minimal. La classe de service qui tirera le plus de profit de notre solution est alors le BE. Cette classe devra se contenter de la bande résiduelle.

Étant donné que les paquets rtPS viennent à intervalle régulier, ils prendront alors un certain pourcentage de la bande. Ce pourcentage de bande sera calculé après application de l'algorithme de courtoisie à ce flux. Grâce à l'algorithme de courtoisie, et aux paquets rtPS qui sont de tailles variables, le besoin en bande du flux rtPS est très dynamique et peut varier d'un intervalle de temps à un autre. Nous supposerons dans notre simulation que les paquets rtPS peuvent utiliser entre 0% et 60% de la bande disponible. Ce pourcentage peut varier en fonction du nombre de paquets courtois. Alors avec une QoS garantie pour le flux rtPS, nous ne nous sommes pas intéressés à simuler la latence des files rtPS, mais plutôt à la bande résiduelle obtenue après application de l'algorithme de courtoisie. Notons, en nous basant sur nos simulations précédentes, que le service au sein du flux rtPS peut se fait suivant notre algorithme 2 à sauts de fréquence variable. Ainsi, nous avons plus de chance de récupérer le maximum de bande, en diminuant la latence de service des flux rtPS, par rapport au FHSS.

Pour les flux nrtPS, nous supposons que la garantie d'une bonne QoS exige 40% de la bande. Tout comme pour le cas des flux rtPS, proposons que notre algorithme 2 soit utilisé pour servir les flux nrtPS. Nous garantissons ainsi la QoS de l'algorithme de courtoisie tout en économisant de la bande. Ainsi, nous irons chercher le maximum de bande résiduelle possible.

La bande résiduelle collectée sera utilisée pour le service des flux sans exigence tels que les flux BE. Les paquets qui viennent dans la file ont un délai après lequel ils seront perdus s'ils ne sont pas traités par le serveur. La garantie de la QoS des classes rtPS et nrtPS est équivalente à énoncer, dans notre cas, qu'aucun paquet de ces files (rtPS et nrtPS) n'est perdu suite à un délai d'attente excessif dans la file. Par contre, une amélioration du service diminuerait le taux de perte des paquets des flux sans exigence. Tel sera le cas pour le trafic BE qui verrait sa bande disponible varier dans le temps. Dans la simulation nous symboliserons cette variation par un choix aléatoire de la valeur de la bande disponible entre 0 et 100% de toute la bande allouée à la grappe. Soit :

$$Bande\_disponible_{BE} = rand\,\mathrm{int}(1,1,[0,100]) \tag{4.9}$$

Cette valeur change à chaque intervalle de temps. Les mêmes valeurs dynamiques de la bande disponible seront utilisées pour le service FHSS et le service FH-variable. Pour mieux représenter le comportement de l'algorithme de courtoisie et le léger gain de bande que nous donne les sauts de fréquence variable (FH-variable) dans le service des flux rtPS et nrtPS, la bande disponible pour le FH-variable pour le service des flux BE sera légèrement supérieure à celle du FHSS pour le service des flux BE. Alors, nous aurons :

$$Bande\_disponible_{BE\_FH-\mathrm{variable}} = Bande\_disponible_{BE\_FHSS} + B_{\zeta} \tag{4.10}$$

Avec B_{ζ} le gain de bande issu de l'utilisation de nos approches à sauts variables dans le service des flux rtPS et nrtPS.

Les paquets arrivent dans la file selon la loi de Poisson. Après une seconde de service, de nouveaux paquets arrivent et font grossir la file. Ainsi, nous avons obtenu les résultats de simulations présentés dans la figure 4.9.

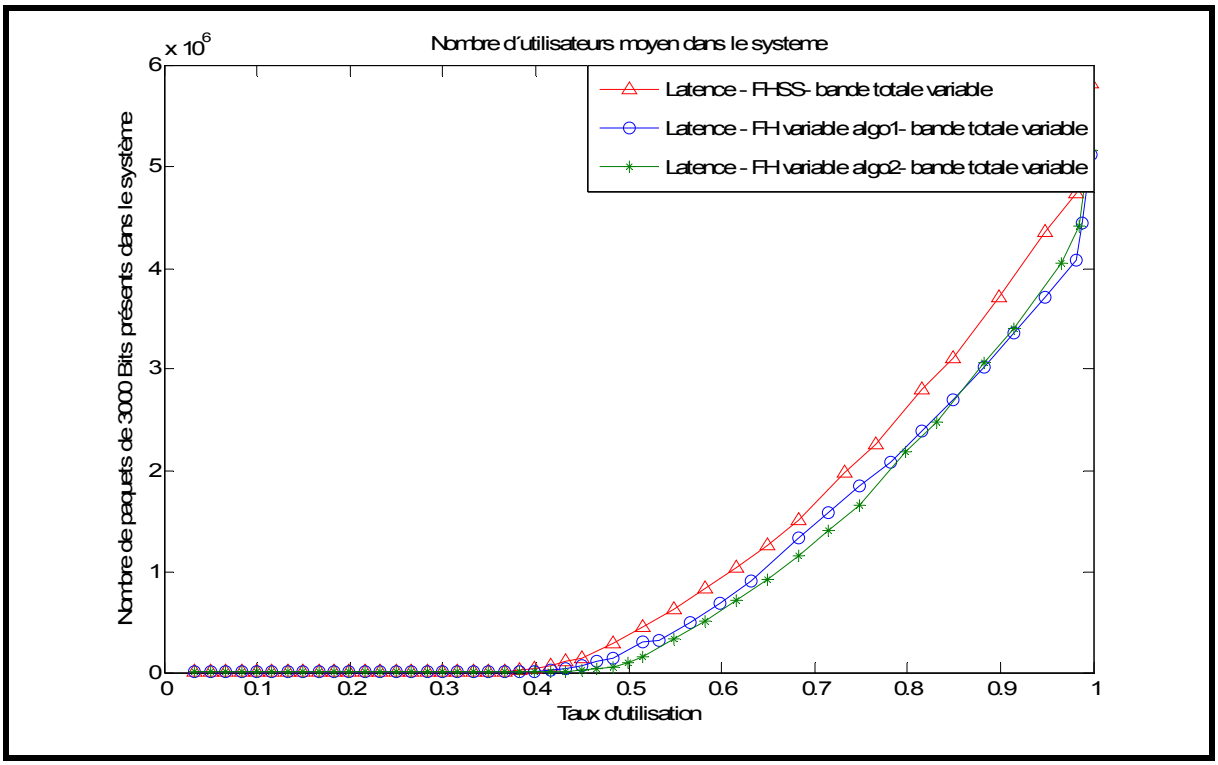

Figure 4.9 Latence algorithme 1 vs latence algorithme 2 vs FHSS dans le cas où la bande totale varie dans le temps

Dans cette courbe nous remarquons un avantage de nos solutions par rapport au FHSS. Il y a moins de paquets BE en attente dans la file. Cela implique un taux de perte des paquets BE faible par rapport au FHSS. Cet avantage s'explique par le fait que dans le FHSS, la largeur de la bande d'un saut de fréquence est fixe soit, 10MHz. Alors si après le calcul de courtoisie, la bande résiduelle n'est pas multiple de 10MHz, une partie de la ressource est perdue dans le cas du FHSS. Cette ressource perdue ne pouvant être utilisée pour le BE, est retournée à la classe de service prioritaire. D'après les travaux de (Tata, 2009), le temps de courtoisie est bénéfique seulement lorsqu'il est supérieur au seuil d'un intervalle de temps. Vu que nous utilisons ce même algorithme de courtoisie pour déterminer la bande de

courtoisie, alors dans un ordonnancement à subdivision fixe, le seuil de décision sera égal à la taille d'un saut de fréquence FHSS. Dans notre solution, étant donné que l'intervalle de temps et l'intervalle fréquentiel sont variables, alors nous pouvons mieux nous ajuster au seuil de décision de l'algorithme de courtoisie. Dans nos simulations, nous avons remarqué qu'il n'y a pas une grande différence entre la latence de l'algorithme 1 et celle de l'algorithme 2. Le choix de l'un par rapport à l'autre est donc sans importance. Toutes les simulations faites jusqu'ici ne concernent que des cas où l'intervalle de temps est très petit, compris entre 0.5ms et 2 ms. Nous nous intéressons à augmenter la plage de variation de l'intervalle temporel pour voir le comportement des différentes approches.

4.4.6 Cas de l'intervalle de temps compris entre 100 ms et 300ms

Dans les simulations précédentes, nous avons considéré des intervalles de temps assez petits, ce qui a fait que les pertes ne sont pas complètement mises en évidence au niveau du FHSS. Dans cette sous-section, nous avons repris les mêmes simulations, mais cette fois-ci avec un intervalle de temps semblable à celui rencontré dans le FHSS utilisé dans le wifi 802.11b (Tanenbaum et Wetherall, 2011). Le tableau 4.11 présente les nouvelles valeurs de l'intervalle de temps.

Tableau 4.2 Paramètres de simulation avec des intervalles de temps plus grand

Paramètre	**FH-variables**	**FHSS**
Intervalle de temps	Varie de 100ms à 300 ms	200ms
Bande totale	100MHz	100MHz

Nous avons simulé la latence des différentes approches. La figure 4.10 présente les résultats de la simulation.

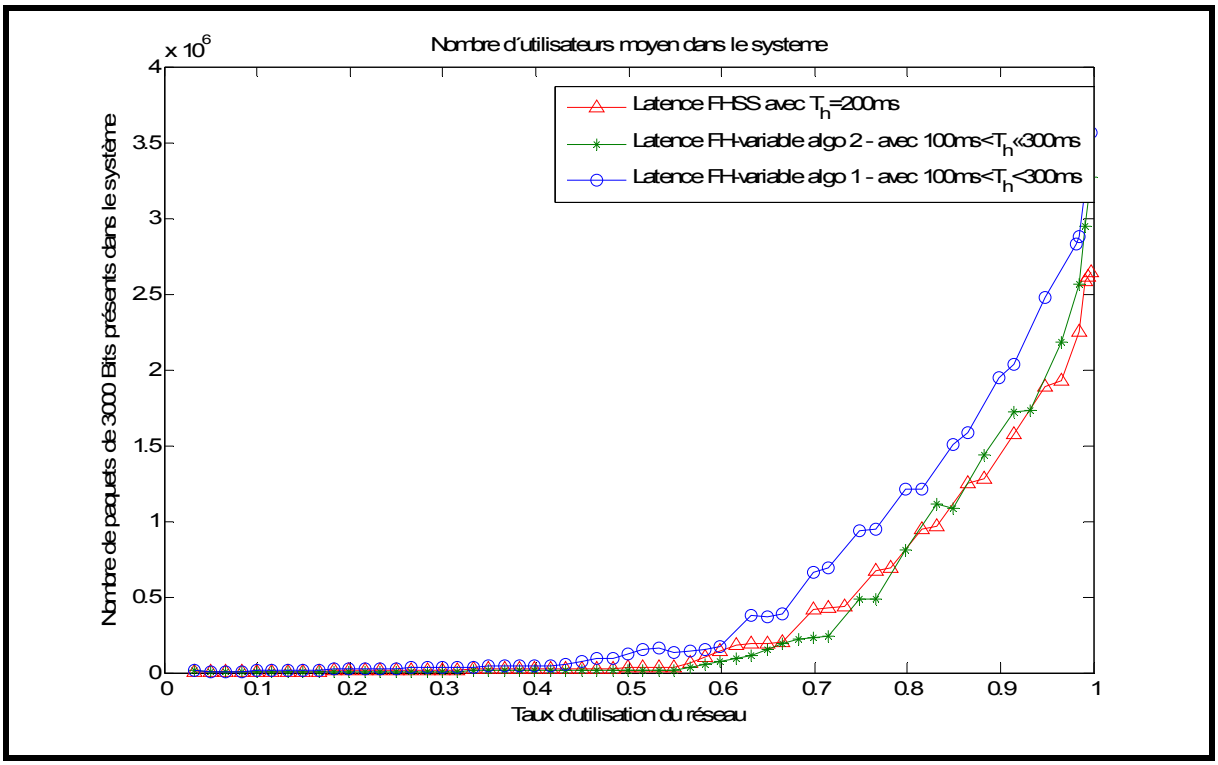

Figure 4.10 Latence algorithme 1 vs latence algorithme 2 vs latence FHSS avec des intervalles de temps de 100ms à 300 ms

Dans cette figure, nous remarquons que pour un intervalle de temps assez grand, l'algorithme 1 à sauts de fréquence variable, tel que proposé dans ce travail, offre une latence plus grande que la latence du FHSS. Vu que la ressource est subdivisée de façon aléatoire dans l'algorithme 1, certaines files peuvent recevoir une grande bande pendant un intervalle de temps élevé tandis que d'autres files reçoivent une faible bande. Dans de tels cas, la règle de priorité utilisée pour allouer la ressource aux usagers n'est pas bénéfique car la taille des paquets est très grande. Les files sont donc servies en très peu de paquets. Il n'y a donc pas eu assez de tours de service pour que l'algorithme puisse s'ajuster aux priorités changeantes des files. Ceci augmente inutilement la latence du système. Dans le FHSS, tous les flux sont considérés à priorité égale donc certaines files ne sont pas trop affectées par le service des autres files. Nous pouvons alors dire que l'algorithme 1 n'est pas adapté pour des systèmes où la taille de l'intervalle de temps est grande.

Par contre, l'algorithme 2 offre une latence presque identique à celle du FHSS. Ceci s'explique par le fait que l'algorithme 2 subdivise la ressource selon les files en attente. Étant donné que les flux ont des priorités proches les unes des autres, alors la subdivision donne des valeurs d'intervalles fréquentiels proches de la taille d'un saut FHSS. Le seul paramètre qui fait la différence est la taille de l'intervalle de temps. Mais le choix aléatoire de cette dernière peut être la cause des fluctuations de la courbe de l'algorithme 2 autour de la courbe du FHSS. Ces fluctuations sont proches des valeurs de la latence du FHSS car la valeur moyenne de l'intervalle de temps dans l'algorithme 2 est égale à la valeur de l'intervalle de temps FHSS, soit 200ms. Toutefois, nous avons remarqué que, pour un faible taux d'utilisation du réseau l'algorithme 2 offre une latence légèrement plus faible que celle du FHSS. Ceci prouve que l'algorithme 2 offre une meilleure équité de service, mais lorsque le système est saturé et que les pertes de ressource dues au remplissage deviennent plus importantes, cette équité peut être insuffisante pour offrir une faible latence. De cette simulation, nous pouvons dire que les ordonnancements à sauts de fréquence à débit variable ne sont pas adaptés aux sauts de fréquence lents.

4.5 Conclusion

Dans les simulations présentées dans ce chapitre, nous remarquons qu'avec de faibles intervalles de temps, la variation du débit permet d'avoir une latence plus faible que l'approche à débit fixe. Cela s'explique par la flexibilité qu'offre l'ordonnancement à débit variable. L'algorithme 1 est peu complexe en termes de calcul et est aussi bénéfique dans la plupart des cas, mais peu s'avérer très désavantageuse lorsque la taille de l'intervalle de temps devient importante.

Nos approches garantissent une bonne QoS aux paquets prioritaires tout en offrant une meilleure latence aux paquets moins prioritaires. Lorsque l'intervalle de temps est très petit, les latences dans nos approches sont plus faibles que celle du FHSS, car nos approches perdent moins de ressources.

Nous pouvons alors dire qu'avec des débits variables dans des systèmes à sauts de fréquence rapide, la latence ne risque pas d'augmenter, mais plutôt de diminuer. Ces résultats sur la latence ne tiennent pas compte des cas de retransmissions dues aux erreurs binaires. Avant de négliger les retransmissions, nous faisons les simulations dans les mêmes conditions, c'est-à-dire le même taux d'erreur binaire. Pour avoir ces mêmes conditions de simulation, nous avons supposé que le BER est le même pour le FHSS et pour nos approches à sauts variables. Pour cela, il a fallu rendre variable la puissance du signal en fonction du débit variable dans nos approches. En déterminant à chaque fois l'énergie nécessaire pour l'envoi des paquets formés, nous remarquons que les sauts de fréquence variables jumelés à des puissances variables permettent d'avoir une économie en énergie. Cette économie peut être bénéfique dans les réseaux ad hoc.

CONCLUSION

L'apparition des réseaux ad hoc dans le domaine militaire donne lieu à de nouveaux défis. Parmi ces défis, nous avons l'ordonnancement des paquets pour une communication efficace. Dans les réseaux militaires, pour permettre une bonne sécurisation de l'information, les méthodes d'ordonnancements sont souvent basées sur les sauts de fréquence. La technique de saut la plus utilisée est l'AFH, *Adaptive Frequency Hopping*. Elle fait des sauts de façon intelligente sur les canaux non bruités par d'autres systèmes utilisant la même bande. L'ordonnanceur alloue les sauts aux usagers suivant un principe prédéfini. Dans la plupart des cas, la QoS est garantie pour les paquets de haute priorité comme le rtPS. Mais les paquets de faibles priorités peuvent subir des pertes de paquets dues à une attente excessive. Pour diminuer ce taux de perte, (Tata, 2009) propose un algorithme de courtoisie qui sert les paquets de faible priorité, tant que les paquets de haute priorité peuvent attendre encore dans leur file. Ainsi, elle diminue la latence de service des files de faibles priorités tout en garantissant la QoS aux files prioritaires. L'application de la courtoisie dans les systèmes à sauts de fréquence traditionnels demande que le temps de courtoisie soit supérieur au seuil d'un intervalle de temps. Dans les cas où le temps de courtoisie est inférieur ou n'est pas multiple d'un intervalle de temps du FHSS, les files de faible priorité ne tirent pas tout le profit de la courtoisie des paquets prioritaires. Une meilleure utilisation de l'algorithme de courtoisie peut diminuer la latence des files non prioritaires et ainsi diminuer le nombre de pertes de paquets.

Dans ce mémoire, nous proposons une nouvelle approche basée sur un ordonnancement à sauts de fréquence mais avec des débits variables pour améliorer la latence du système. Pour varier le débit, nous offrons aux usagers des longueurs de bandes différentes à chaque intervalle de temps. Dans une première approche, nous proposons une variation des longueurs de bande indépendamment de la taille des files. Par contre, l'allocation de ces bandes se fera en priorisant les files les plus grandes. Cette approche n'est pas trop complexe à implémenter car la subdivision peut se baser sur un choix de valeurs aléatoires au niveau de nœud maître. Cette approche offre une latence et une consommation d'énergie plus faible que

celle du FHSS dans un environnement où la variation du débit est jumelée à une variation de la puissance et à un petit intervalle de temps, soit 0.5 ms à 2 ms. Dans le cas où cet intervalle de temps devient très grand, de l'ordre de 200 ms à 300 ms, la latence de l'algorithme 1 est plus grande que celle du FHSS.

Notre seconde approche à sauts de fréquence variable fait une subdivision de la bande en fonction des paramètres de priorités des files en attente. Ainsi, l'équité du service est améliorée. Cette approche offre une latence et une dépense énergétique plus faible que celles de l'algorithme 1. L'algorithme 2 permet une meilleure utilisation du temps de courtoisie, car elle peut ajuster la subdivision de la ressource au seuil calculé par l'algorithme de courtoisie. Ce qui n'est pas le cas dans les sauts traditionnels car, les ressources sont pré subdivisées. Dans un environnement où l'intervalle de temps est grand, l'algorithme 2 offre une latence presque identique aux sauts de fréquence traditionnels.

Le débit étant variable dans nos approches, alors pour avoir un BER identique à celui du FHSS, nous proposons de faire varier la puissance du signal en fonction du débit offert sur les liens. Grâce à cette variation de la puissance et à la réduction de la latence observée dans nos solutions, nous dépensons moins d'énergie que le FHSS pour servir les mêmes files. Ceci est bénéfique pour les réseaux ad hoc dans lesquels la ressource énergétique reste une limite.

Le FHSS et l'AFH permettent d'avoir une bonne sécurité de l'information. La sécurité des FH-variable n'a pas fait l'objet de ce mémoire. Étudier le niveau de sécurité offert dans les FH-variables pourrait être une suite à ce travail. Si au terme de cette étude, la sécurité est meilleure que celle dans les sauts traditionnels, alors un des brouilleurs les plus redoutés, tel que le brouilleur suiveur, aura moins d'impact sur le signal. Ceci diminuerait le BER causé par ce brouilleur donc la latence peut en tirer profit.

RECOMMANDATIONS

Le but de ce mémoire est d'offrir un algorithme d'ordonnancement à sauts de fréquence pour les réseaux militaires ad hoc. Cet ordonnancement intègre des débits variables. Pour comparer les performances, les modèles mathématiques ont été implémentés dans Matlab. Les résultats obtenus restent théoriques.

Pour une validation, ces simulations doivent être reprises dans un simulateur réseau comme OPNET ou NS3. Ces simulations permettront de voir toute l'influence des sauts de fréquence variable sur les réseaux ad hoc sans fil.

Cette simulation est impossible sous simulink parce que simulink prend ses valeurs de simulation en début de simulation et ne tient pas compte des valeurs subséquentes prises par les débits.

LISTE DE RÉFÉRENCES BIBLIOGRAPHIQUES

Baumgartner, Karl, et Jérôme Racloz. 2002. *Coexistence entre WLAN 802.11 et Bluetooth*. Vaud: Haute Ecole d'Ingénierie et de Gestion du Canton, 221 p.

Ekelin, C. 2006. « Clairvoyant non-preemptive EDF scheduling ». In *Real-Time Systems, 2006. 18th Euromicro Conference on*. p. 7 pp.-32. IEEE.

Elsner, J., R. Tanbourgi et F. K. Jondral. 2011. « Multiple access interference mitigation through multi-level locally orthogonal FH-CDMA ». In *MILCOM 2011 - 2011 IEEE Military Communications Conference, 7-10 Nov. 2011*. (Piscataway, NJ, USA), p. 378-83. Coll. « MILCOM 2011 - 2011 IEEE Military Communications Conference »: IEEE. < http://dx.doi.org/10.1109/MILCOM.2011.6127690 >.

Felstead, E.B. 2009. « The vulnerability region of frequency hoppers against follower jammers ». In., p. 1-7. IEEE.

Golmie, N., O. Rebala et N. Chevrollier. 2003. « Bluetooth adaptive frequency hopping and scheduling ». In *MILCOM 2003 - 2003 IEEE Military Communications Conference, October 13, 2003 - October 16, 2003*. (Monterey, CA, United states) Vol. 2, p. 1138-1142. Coll. « Proceedings - IEEE Military Communications Conference MILCOM »: Institute of Electrical and Electronics Engineers Inc. < http://dx.doi.org/10.1109/MILCOM.2003.1290352 >.

Holmes, Jack K. 2007. *Spread spectrum systems for GNSS and wireless communications*. Boston ; London: Artech House, 855 p.

Implustit. 2012. « Fréquence Shift Keying – FSK ». < http://www.wireless-techbook.com/techniques-de-modulation/15-frequence-shift-keying--fsk.html >. Consulté le 1 avril 2012.

Kadoch, M., et C. Tata. 2010. « Courteous algorithm: performance optimization in WiMAX networks ». In *Proceedings of the 4th international conference on Communications and information technology*. p. 23-32. World Scientific and Engineering Academy and Society (WSEAS).

Lagoutte, Pierre. 2000. « Réseaux de télécommunication militaires ». Dans *Techniques de l'ingénieur*. Numéro de l'article : TE7490. Paris: Techniques de l'ingénieur. < http://www.techniques-ingenieur.fr/base-documentaire/technologies-de-l-information-th9/administration-de-reseaux-applications-et-mise-en-oeuvre-42481210/reseaux-de-telecommunication-militaires-te7490/ >. Consulté le 5 janvier 2012.

Mo, Shaomin, John Gu, Reza Ghanadan, Matthew Sherman, Joseph Farkas, John Tranquilli, Joshua Niedzwiecki et Bruce Fette. 2010. « Distributed scheduler design for

multiuser detection enabled wireless mobile ad-hoc networks ». In *2010 IEEE Military Communications Conference, MILCOM 2010, October 31, 2010 - November 3, 2010*. (San Jose, CA, United states), p. 98-103. Coll. « Proceedings - IEEE Military Communications Conference MILCOM »: Institute of Electrical and Electronics Engineers Inc. < http://dx.doi.org/10.1109/MILCOM.2010.5679569 >.

Nguyen Vo, Doan, Michel Bellemare et Michel Forté. 2012. « CPFSK-OFDM Modulation For Fast Frequency Hopping ». In *CCECE2012 - 2012 25th IEEE Canadian Conference on Electrical and Computer Engineering, 29 May -02 April, 2012*. (Montreal, QC, Canada). IEEE.

Prasath, G.A., C.P. Fu et M. Ma. 2008. « QoS scheduling for group mobility in WiMAX ». In *Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference on*. p. 1663-1667. IEEE.

Rackspace. 2012. « Ultra Electronics TCS ». < http://www.armedforces-int.com/suppliers/ultra-electronics-tactical-communication-systems.html >. Consulté le 28 avril 2012.

Roshan, Pejman, et Jonathan Leary. 2004. *Réseaux Wi-Fi : notions fondamentales*. Paris: CampusPress, 291 p.

Smith, David R. 1993. *Digital transmission systems*, 2nd ed. New York, N.Y.: Van Nostrand Reinhold, 818 p.

Stabellini, Luca, Lei Shi, Ahmad Al Rifai, Juan Espino et Veatriki Magoula. 2009. « A new probabilistic approach for adaptive frequency hopping ». In *2009 IEEE 20th Personal, Indoor and Mobile Radio Communications Symposium, PIMRC 2009, September 13, 2009 - September 16, 2009*. (Tokyo, Japan), p. 2147-2151. Coll. « IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC »: Institute of Electrical and Electronics Engineers Inc. < http://dx.doi.org/10.1109/PIMRC.2009.5450211 >.

Tanenbaum, Andrew S., et D. Wetherall. 2011. *Réseaux*, 5e éd. Paris: Pearson Education, 958 p.

Tata, Chafika. 2009. « Algorithme de courtoisie : optimisation de la performance dans les réseaux WIMAX fixes ». Mémoire de maîtrise en génie - concentration : réseaux de télécommunications. Montréal, École de technologie supérieure, 135 p. < http://espace.etsmtl.ca/id/eprint/83 >. Consulté le 7 juin 2011.

Thaliath, J., M.M. Joy, E. Priya John et D. Das. 2008. « Service class downlink scheduling in WiMAX ». In *Communication Systems Software and Middleware and Workshops, 2008. COMSWARE 2008. 3rd International Conference on*. p. 196-199. IEEE.

Ultra Electronics TCS. 2011a. « EHCLOS – Enhanced High-Capacity LOS Radio ». Canada: Ultra Electronics TCS, 2 p. < http://www.ultra-tcs.com/files/datasheets/Ultra%20Electronics%20TCS_EHCLOS.pdf >. Consulté le 2 avril 2012.

Ultra Electronics TCS. 2011b. « AN/GRC-245A(V) HCLOS™ Radio ». Canada: Ultra Electronics TCS, 2 p. < http://www.ultra-tcs.com/files/datasheets/Ultra_TCS_ANGRC245_2011-01-19.pdf >. Consulté le 2 avril 2012.

Ultra Electronics TCS. 2011c. « HCLOS™ PMP-E ». Canada: Ultra Electronics TCS, 4 p. < http://www.ultra-tcs.com/files/datasheets/HCLOS%20PMP-E.pdf >. Consulté le 2 avril 2012.

Ungar, Jim. 1999. « QOS notes ». < http://jungar.net/network/QOS/QOSNotes.html >. Consulté le 1 avril 2012.

Xie, X., H. Chen et H. Wu. 2008. « Simulation studies of a Fair and Effective Queueing algorithm for WiMAX resource allocation ». In *Communications and Networking in China, 2008. ChinaCom 2008. Third International Conference on*. p. 281-285. IEEE.