REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIREMINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

Université des Sciences et de la Technologie d'Oran- Mohamed Boudiaf-

USTO-MB

Faculté des sciences

Département d'informatique

Spécialité : Informatique Option : Système, Réseaux et Bases de Données

MEMOIRE

Présenté par

Mr ALI HASSOUNE Mustafa

Pour l'obtention du

DIPLOME DE MAGISTER

L'énergie dans les réseaux sans fil

(Etude de cas)

Soutenu le 01/07/2010

Devant le jury:

Dr MESSABIH. B	MCA	Président
Dr DJEBBAR. B	PROF	Examinateur
Dr BENDELLA. F	MCA	Examinatrice
Dr SELKA. S	MCB	Invitée
Dr SAADOUNI. K	MCA	Encadreur
Dr MEKKAKIA. M. Z	MCB	Co-Encadreur

REMERCIEMENTS

Le travail présenté dans ce mémoire a été effectué sous la direction de Mme MEKKAKIA Zoulikha. Je tiens à la remercier d'avoir accepté d'encadrer ce travail.

Je tiens particulièrement à remercier :

Dr. MESSABIH, Professeur au département d'informatique, d'avoir accepté de présider ce jury,

Dr SADOUNI, Maitre de conférences au département d'informatique, d'avoir accepté d'être un membre de Jury.

Dr BENDELLA, Maitre de conférences au département d'informatique,

Dr SELKA, Maitre de conférences au département d'informatique,

Pour l'intérêt qu'ils ont porté à ce travail en m'honorant par leur présence dans le jury.

DEDICACE

Je dédie ce modeste travail, après remerciement de Dieu, tout puissant pour ça grâce,

A mon très cher père qui m'a soutenu pendant toute la durée de ces longues études,

A mes très chers beaux parents,

A ma très chère femme pour son encouragement,

A mes très chers grands parents qui m'ont énormément poussé avec leurs précieuses prières,

A mes très chers frères et sœur,

A tout mes amis d'informatique, et mes amis de travail,

A mon futur bébé (Meriem) inch ALLAH .

Table des matières

Introduction Générale…… 1

Chapitre1 : Les réseaux Adhoc et les réseaux de senseurs

1.1. Introduction…… 3
1.2. Les réseaux Ad Hoc…… 3
1.2.1. Définition…… 3
1.2.2. Caractéristiques des réseaux Ad hoc…… 4
1.2.3. Les inconvénients des réseaux Ad hoc…… 5
1.3. Les réseaux de senseurs (capteurs) …… 6
1.3.1. Introduction…… 6
1.3.2. Description et caractéristiques…… 7
1.3.3. Applications des réseaux de senseurs…… 7
1.3.4. Plates-forme…… 11
1.3.5. Matérielles…… 12
1.3.6. Logiciels…… 12
1.3.7. Architecture d'un micro-capteur…… 12
1.3.8. Architectures des réseaux de capteurs…… 14
1.4. Conclusion…… 15

Chapitre2 : Les protocoles de minimisation de la consommation d'énergie

2.1. Introduction…… 17
2.2. La couche physique…… 18
2.2.1. COMPOW…… 18
2.2.2. CLUSTERPOW…… 19
2.2.3. IMPROVING CLUSTERPOW…… 23
2.2.4. TUNNELLED CLUSTERPOW…… 24
2.2.5. MINPOW…… 24
2.2.6. Synthèse…… 25

2.3. La couche liaisons de données…… 26
2.3.1. PSM…… 27
2.3.2. Slotted PSM…… 29
2.3.3. TA-PSM…… 30
2.3.4. S-MAC…… 30

2.3.5. T-MAC…….. 31
2.3.6. SPAN…….. 34
2.3.7. Synthèse…….. 35

2.4. La couche réseau…….. 36
2.4.1. MTPR…….. 37
2.4.2. MBCR…….. 37
2.4.3. PARO…….. 38
2.4.4. LMST…….. 39
2.4.5. Synthèse…….. 40

2.5. Conclusion…….. 41

Chapitre 3 : Le protocole SMAC

3.1. Définition…….. 43
3.2. Présentation des fonctions et dispositifs mis en application dans SMAC…….. 44
3.3. Le sommeil et l'écoute périodiques…….. 46
3.4. Synchronisation…….. 49
3.4.1. Composants relatifs de synchronisation dans SMAC…….. 49
3.4.2. Le choix du premier ordonnanceur…….. 53
3.4.3. La mise à jour et le maintien des ordonnanceurs…….. 54
3.4.4. La découverte périodique des voisins…….. 57
3.4.5. Mise à jour périodique de la liste des voisins…….. 57
3.5. CSMA/CA dans SMAC…….. 60
3.5.1. Carrier sense…….. 60
3.5.2. Eviter les collisions…….. 63
3.6. Overhearing avoidance…….. 65
3.7. Message Passing…….. 67
3.8. L'écoute adaptative…….. 70
3.9. Conclusion…….. 76

Chapitre 4 : Simulation & Evaluation

4.1. Introduction…….. 78

4.2. Simulations et évaluation des performances du SMAC…….. 79
4.2.1. Simulation n° 1 : L'énergie totale restante…….. 81
4.2.2. Simulation n° 2 : L'énergie « Sleep »…….. 82
4.2.3. Simulation n° 3 : L'énergie « transmit »…….. 83
4.2.4. Simulation n° 4 : L'énergie « receive »…….. 84
4.2.5. Simulation n° 5 : L'énergie « idle »…….. 85
4.2.6. Simulation n° 6 : Les collisions…….. 88

4.2.7. Simulation n° 7 : L'overhead (Les paquets de contrôle)…….. 89
4.2.8. Simulation n° 8 : La perte des paquets…….. 90
4.2.9. Simulation n° 9 : La latence…….. 90
4.2.10. Synthèse…….. 90

4.3. SMAC avec variation de paramètre « duty cycle »…….. 91
4.3.1. Simulation n° 1 : L'énergie totale restante…….. 92
4.3.2. Simulation n° 2 : L'énergie « Sleep »…….. 93
4.3.3. Simulation n° 3 : L'énergie « transmit »…….. 94
4.3.4. Simulation n° 4 : L'énergie « receive »…….. 95
4.3.5. Simulation n° 5 : L'énergie « idle »…….. 96
4.3.6. Simulation n° 6 : Les collisions…….. 97
4.3.7. Simulation n° 7 : L'overhead (Les paquets de contrôle)…….. 98
4.3.8. Simulation n° 8 : La perte des paquets…….. 98
4.3.9. Simulation n° 9 : La latence…….. 99
4.3.10. Synthèse…….. 100

4.4. SMAC avec variation de paramètre « Rate » et « duty cycle »…….. 100
4.4.1. Scénario utilisé…….. 100
4.4.2. Résultats de simulation …….. 102
4.4.3. La latence et le trafic …….. 104
4.4.4. Les collisions et le trafic …….. 105
4.4.5. L'énergie et le trafic …….. 106

4.5. Nouveau protocole L-MAC (Latency MAC) …….. 107
4.5.1. Son objectif …….. 107
4.5.2. Fonctionnement …….. 107
4.5.3. L'algorithme LATMAC 108
…… 109
4.5.4. Comparaison entre LATMAC et SMAC
…… 111

4.6. Conclusion……..

Conclusion Générale & Perspectives…….. 113
Références

Table des figures

Figure 1.1. Topologie d'un réseau Ad Hoc 4
Figure 1.2. Le changement de la topologie des réseaux Ad Hoc 4
Figure 1.3. Les nœuds cachés 5
Figure 2.1. Problème de l'endormissement précoce 33
Figure 2.2. Mécanisme RTS/CTS 34
Figure 2.3. Un scénario avec 1000 nœuds, 18 coordinateurs 35
Figure 2.4. SPAN dans les couches OSI 35
Figure 3.1. L'écoute et sommeil périodique 45
Figure 3.2. Le format d'un frame SMAC 47
Figure 3.3. Trois points de contrôle dans un frame 47
Figure 3.4. Un nœud de frontière avec deux programmes 51
Figure 3.5. L'hiérarchie des périodes dans SMAC 57
Figure 3.6. La période de mise à jour de la liste des voisins plus petite que la période de synchronisation 60
Figure 3.7. Une transmission de donnée entre deux nœuds 65
Figure 3.8. C et D entendent les transmissions entre A et B 66
Figure 3.9. RTS/CTS/DATA/ACK et les configurations de NAV dans Message Passing 69
Figure 3.10. Un réseau à 3 sauts avec une source et une transmission destination 71
Figure 3.11. Transmission d'un paquet de données à travers un réseau à 3 sauts sans l'écoute adaptative 71
Figure 3.12. La transmission d'un paquet de données à travers un réseau à 3 sauts, avec l'écoute adaptative 72
Figure 3.13. L'écoute adaptative s'étend au prochain frame 75
Figure 3.14. L'écoute adaptative déclenchée deux fois dans un seul frame 76
Figure 4.1. Topologie du réseau 80
Figure 4.2. Taux d'énergie dans les réseaux 802.11 et SMAC en fonction de temps 81
Figure 4.3. L'énergie « Sleep » dans 802.11 et SMAC en fonction de temps 82
Figure 4.4. L'énergie de transmission dans 802.11 et SMAC en fonction de temps 83
Figure 4.5. L'énergie de réception dans 802.11 et SMAC en fonction de temps 84
Figure 4.6. L'énergie « idle » dans 802.11 et SMAC en fonction de temps 85
Figure 4.7. Les différentes énergies consommées par un nœud 802.11 86
Figure 4.8. Les différentes énergies consommées par un nœud SMAC 86
Figure 4.9. Le gain d'énergie entre 802.11 et SMAC 87
Figure 4.10. Les collisions dans 802.11 et SMAC 88
Figure 4.11. Nombre de paquets de contrôle dans 802.11 et SMAC 89
Figure 4.12. Taux d'énergie dans SMAC en changeant Duty Cycle 92
Figure 4.13. L'énergie « Sleep » en fonction de temps 93
Figure 4.14. L'énergie de transmission en fonction de temps 94
Figure 4.15. L'énergie de réception en fonction de temps 95
Figure 4.16. L'énergie « idle » en fonction de temps 96
Figure 4.17. Les collisions dans SMAC en modifiant Duty Cycle 97
Figure 4.18. Les paquets de contrôle en fonction de temps 98
Figure 4.19. La latence en fonction de « duty cycle » 99
Figure 4.20. Topologie du réseau 101
Figure 4.21. La latence en fonction du trafic du réseau 104
Figure 4.22. La latence moyenne des différents trafics du réseau 105

Figure 4.23. Taux de collisions des différents trafics du réseau 105
Figure 4.24. L'énergie consommée en fonction du trafic du réseau 106
Figure 4.25. L'énergie de réception en fonction de trafic du réseau (700s) 106
Figure 4.26. Comparaison entre latence LATMAC (0,100) et SMAC 10 109
Figure 4.27. Latence sous LATMAC (0,100) et LATMAC (100,150) 110
Figure 4.28. Comparaison de la latence moyenne 110
Figure 4.29. L'énergie consommée sous SMAC 10 et LATMAC (100,150) 111

Liste des tables

Table 1.1. Comparaison réseaux Ad-hoc / Senseurs 15
Table 2.1. Table de routage de la source S 20
Table 2.2. Table de routage de N1 21
Table 2.3. Table de routage de N2 22
Table 2.4. Table de routage de N3 22
Table 2.5. Comparaison entre les protocoles de la couche physique 26
Table 2.6. Comparaison entre les protocoles de la couche liaison de données 36
Table 2.7. Comparaison entre les protocoles de la couche réseau 41
Table 2.8. Les protocoles par catégorie 41
Table 3.1. Le frame de synchronisation SYNC 49
Table 3.2. Les champs de la table d'ordonnancement 50
Table 3.3. Définition de champs 53
Table 3.4. Algorithme général pour traiter les paquets SYNC 56
Table 3.5. Comparaison entre NAC et NAV Voisin 62
Table 4.1. Les paramètres de simulation 81
Table 4.2. Les collisions par type de paquet dans 802.11 88
Table 4.3. Les collisions par type de paquets dans SMAC 88
Table 4.4. Overhead dans 802.11 et SMAC 89
Table 4.5. Taux de perte dans les réseaux 802.11 et SMAC 90
Table 4.6. Latence entre SMAC et 802.11 90
Table 4.7. Les collisions en chiffre 97
Table 4.8. Les paquets de contrôle en chiffre 98
Table 4.9. Taux de perte de paquets dans SMAC 98
Table 4.10. Latence sous SMAC 99
Table 4.11. Les paramètres de simulation 102
Table 4.12. Résultats de simulation : Rate = 10 103
Table 4.13. Résultats de simulation : Rate = 5 103
Table 4.14. Résultats de simulation : Rate = 1 104
Table 4.15. Résultats de simulation : Rate = 0.5 104
Table 4.16. Résultats de simulation : Rate = 0.1 105
Table 4.17. Le paquet de vérification de la latence 108
Table 4.18. Le frame de synchronisation SYNC 109

Acronymes

ACK	Acknowledgment
AM	Active Mode
AODV	Ad hoc On-Demand Distance Vector
ARP	Address Resolution Protocol
ARQ	Automatic Repeat reQuest
ATIM	Announcement Traffic Indication Message
CBR	Constant Bit Rate
CLUSTERPOW	Cluster Power
COMPOW	Common power
CSMA-CA	Carrier Sense Multiple Access with Collision Avoidance
DIFS	Distributed Index for Features in Sensor Networks
DSR	Dynamic Source Routing
FRTS	Future-Request-to-Send
GPS	Global Positioning System
GSM	Global System for Mobile communications
IEEE	Institute of Electrical and Electronics Engineers
LATMAC	Latency MAC
LMST	Local Minimum Spanning Tree
MAC	Medium Access Control
MANET	Mobile Ad-hoc NETworks
MBCR	Battery Cost Routing
MINPOW	Minimum Power
MST	Minimum Spanning Tree
MTPR	Minimum (Total) Transmission Power Routing
NAV	Network Allocation Vector

NS	Network Simulator
OSI	Open Systems Interconnection
PAN	Personal Area Network
PARO	Power-Aware Routing Optimization Protocol
PS	Power Saving
PSM	Power Save Mode
PSM	Mower Saving Mode
RAM	Random Access Memory
RTS/CTS	Request-To-Send, Clear-To-Send
SCADDS	Scalable Coordination Architectures for Deeply Distributed Systems
SMAC	Sensor MAC
SPAN	Save Power for Ad Hoc Networks
SYNC	Synchronization
TA-PSM	Traffic Aware Power Saving Mode
TCP	Transmission Control Protocol
TMAC	Timeout MAC
TORA	Temporally Ordered Routing Algorithm
UDP	User Datagram Protocol
USC/ISI	University of Southern California / Information Sciences Institute
UWB	Ultra Wide Band
WSN	Wireless Sensor Network

INTRODUCTION GENERALE

INTRODUCTION GENERALE

Les nœuds mobiles dans les réseaux Ad hoc doivent se déplacer librement, mais leurs batteries ont une énergie limitée. Dans l'absence d'une entité de contrôle centrale comme une station de base, chaque nœud doit participer avec un protocole distribué.

En fait, chaque nœud se comporte comme un routeur, et toutes les machines peuvent communiquer ou échanger des informations directement ou indirectement par l'intermédiaire d'autres.

Dans les années passées, plusieurs protocoles ont été proposés, appelés les protocoles de l'énergie consciente (Energy-aware protocols).

Certains protocoles essaient de réduire le trafic, et donc l'énergie dépensée. BIP [21], LMST [8], et SPAN [14] sont basées sur la diffusion avec le minimum d'énergie en utilisant "Minimum Spanning Tree development".

Une autre série de protocoles est basée sur le contrôle de la topologie et les problèmes de routage avec énergie minimale. Les nœuds peuvent modifier leur niveau d'énergie de transmission pour maintenir le niveau de connectivité de la topologie du réseau, pour accroître la capacité du réseau, et de réduire la consommation des interférences et de l'énergie.

Dans ce contexte vient de s'inscrire notre projet de magister, notre objectif consiste en l'étude des différentes mécanismes de gestion et de conservation d'énergie selon les 3 couches du modèle OSI.

Enfin, notre choix s'est porté sur le modèle SMAC destiné pour la conservation d'énergie dans les réseaux adhoc et plus précisément les réseaux de capteurs.

A cet effet, notre mémoire est organisé comme suit :

Le chapitre 1 présente une Introduction aux réseaux ad hoc suivi d'une description détaillée des réseaux de capteurs.

Le chapitre 2 résume et classifie les différents protocoles de minimisation de la consommation d'énergie. Cette recherche a eu pour objectif de définir pour chaque couche (Physique, Liaison de données et Réseau) les protocoles qui lui sont appropriés.

Le chapitre 3 est une recherche bibliographique détaillée du protocole SMAC.

Enfin, le chapitre 4 représente l'implantation et l'étude des propriétés de SMAC comparé avec le MAC 802.11 puis l'étude de SMAC avec la variante « Duty Cycle ». Et nous proposons un nouveau protocole LATMAC, qui prend en considération l'énergie et la latence. Nous finissons notre proposition par une comparaison entre LATMAC et SMAC.

Une conclusion clore notre mémoire.

Chapitre 1
Les réseaux Ad Hoc et les réseaux de capteurs

Chapitre 1

Les réseaux Ad Hoc et les réseaux de capteurs

1.1. Introduction

Les réseaux ad hoc (en latin : « qui va vers ce vers quoi il doit aller », c'est-à-dire « formé dans un but précis », telle qu'une commission ad-hoc, formée pour régler un problème particulier) sont des réseaux sans fil capables de s'organiser sans infrastructure définie préalablement.

Les réseaux ad hoc, dans leur configuration mobile, sont connus sous le nom de MANet (pour Mobile Ad-hoc NETworks).

Les réseaux de senseurs est un type particulier des réseaux Ad Hoc. Dans ce chapitre, nous allons voir en détails la définition des réseaux Ad Hoc ainsi que les réseaux de senseurs, et les différences majeures entre eux.

1.2. Les réseaux Ad Hoc

1.2.1. Définition

Chaque nœud communique directement avec son voisin. Pour communiquer avec d'autres nœuds, il lui est nécessaire de faire passer ses données par d'autres nœuds qui se chargeront de les acheminer. Pour cela, il est d'abord primordial que les nœuds se situent les uns par rapport aux autres, et soient capables de construire des routes entre eux : c'est le rôle du protocole de routage.

Ainsi, le fonctionnement d'un réseau ad-hoc le différencie notablement d'un réseau comme le réseau GSM, les réseaux Wi-Fi avec des points d'accès : là où une ou plusieurs stations de base sont nécessaires à la plupart des communications entre les différents nœuds du réseau

(mode Infrastructure), les réseaux ad-hoc s'organisent d'eux-même et chaque nœud peut jouer différents rôles.

L'utilisation la plus simple et la plus courante des réseaux ad-hoc est faite par les réseaux sans fil Wifi en permettant une mise en place rapide d'une connexion réseau entre deux ordinateurs.

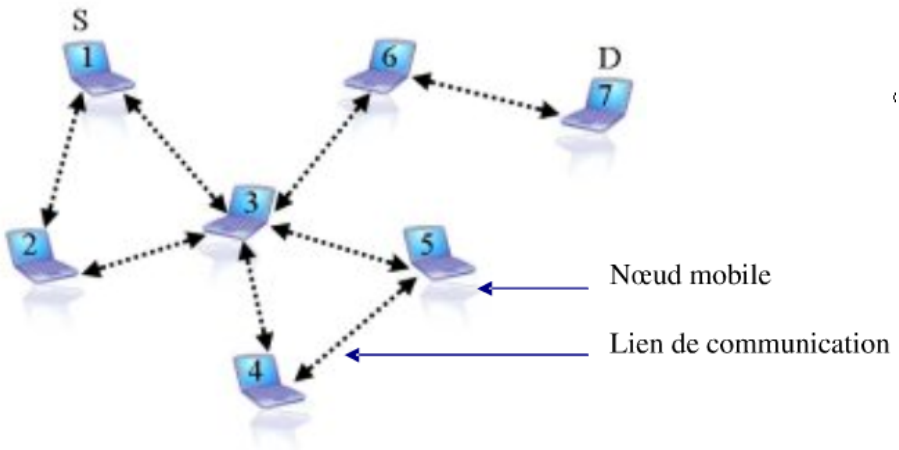


Figure 1.1. Topologie d'un réseau Ad Hoc

1.2.2. Caractéristiques des réseaux ad hoc

Les réseaux mobiles ad hoc sont caractérisés par:

1. Une topologie dynamique

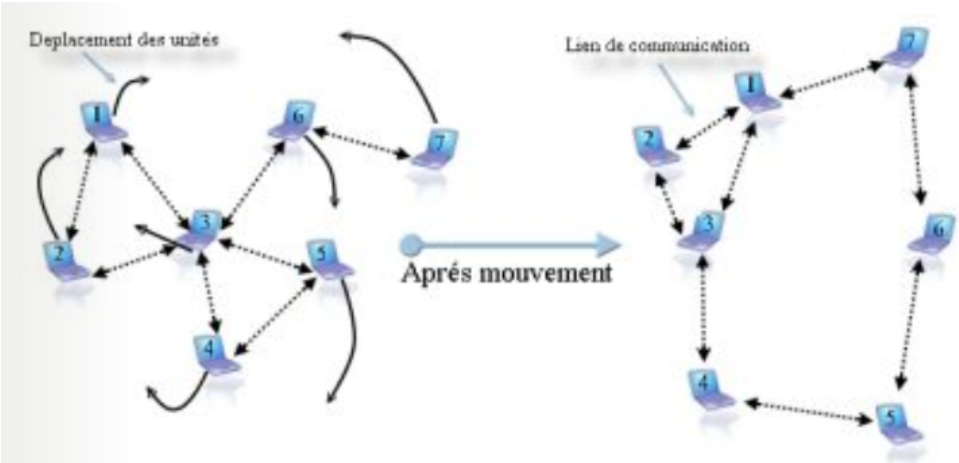


Figure 1.2. Le changement de la topologie des réseaux Ad Hoc

2. Une bande passante limitée
3. Une sécurité physique limitée
4. L'absence d'infrastructure
5. Erreurs fréquents de transmission
6. Interférences
7. Nœuds cachés

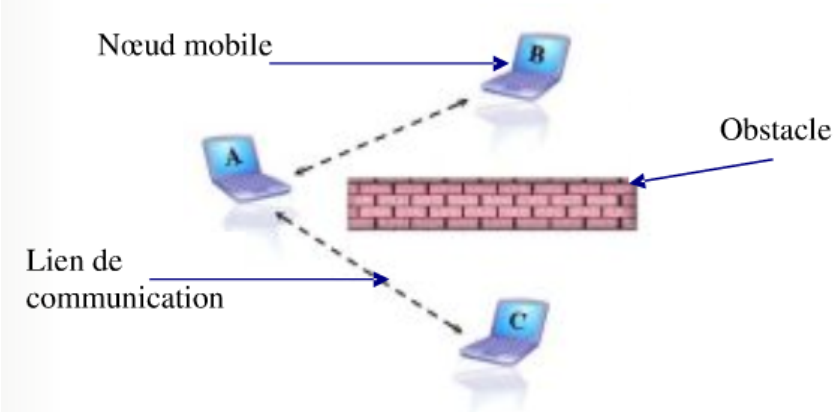


Figure 1.3. Les nœuds cachés

8. Autonomie de décision : Aucune autorité centrale n'est présente pour gérer les opérations des différents hôtes, le réseau et donc entièrement décentralisé.
9. Des **contraintes d'énergie**. Ce dernier est le sujet de notre recherche.

1.2.3. Les inconvénients des réseaux Ad hoc

a. Liens asymétriques

En pratique, la communication dans les réseaux ad hoc est à sens unique entre l'émetteur et le récepteur.

b. Interférences

Ce qui accroit le nombre d'erreurs sur la transmission et amoindrit les performances du lien radio.

Puisque l'interface radio est partagée, où chaque donnée est réceptionnée par tous les nœuds, à des puissances variables, les interférences s'ajoutent au bruit et détériorent les communications, ce qui augmente le taux d'erreur et diminue le débit de la liaison.

c. Mobilité des nœuds

La mobilité des nœuds, influe sur le réseau en modifiant la topologie du réseau, ce qui peut donner une transformation du tracé des routes lors des échanges de paquets. Puisque la topologie du réseau est dynamique, les routes sont modifiées fréquemment. Cela demande d'énormes ressources pour véhiculer la signalisation indispensable aux mouvements des nœuds.

Jusqu'à présent, la plupart des routages ad-hoc se sont montrés plutôt inaptes à faire face à une forte mobilité.

1.3. Les réseaux de capteurs

1.3.1. Introduction

Des nouveaux réseaux de capteurs (Sensor Network) viennent au secours de l'environnement et de l'industrie grâce aux récents développements réalisés dans le domaine des techniques sans fil (wireless). Depuis quelques décennies, le besoin d'observer et de contrôler des phénomènes physiques tels que la température, la pression ou encore la luminosité est essentiel pour de nombreuses applications industrielles et scientifiques. Dans le domaine de l'écologie, la surveillance de polluants comme par exemple l'ozone, le NO_2 ou encore le CO_2 pourrait considérablement augmenter la qualité de vie dans les villes. Il n'y a pas si longtemps, la seule solution pour acheminer les données du capteur jusqu'au contrôleur central était le câblage qui avait comme principaux défauts d'être coûteux et encombrant. Aujourd'hui, grâce aux récents progrès des techniques sans-fils, de nouveaux produits exploitant des réseaux de capteurs sans-fil (Wireless Sensor Network) sont employés pour récupérer ces données environnementales. En plus de ces applications civiles, il existe évidemment des applications militaires des réseaux de capteurs notamment dans la détection d'intrusion, la localisation des combattants sur un champ de bataille, etc.

Les progrès réalisés ces dernières décennies dans les domaines de la microélectronique, de la microtechnique, et des technologies de communication sans fil, ont permis de produire avec un coût raisonnable des composants de quelques millimètres cubes de volume. Ces derniers, appelés micro-capteurs, intègrent : une unité de captage chargée de capter des grandeurs physiques (chaleur, humidité, vibrations) et de les transformer en grandeurs numériques, une unité de traitement informatique et de stockage de données et un module de transmission sans fil(wireless).

De ce fait, les micro-capteurs sont de véritables systèmes embarqués. Le déploiement de plusieurs d'entre eux, en vue de collecter et transmettre des données environnementales vers un ou plusieurs points de collecte, d'une manière autonome, forme un réseau de capteurs sans fil.

Selon le magazine Technology Review du MIT, il s'agit de l'une des dix nouvelles technologies qui bouleverseront le monde et notre manière de vivre et de travailler.

1.3.2. Description et caractéristiques

Les réseaux de capteurs sans fil - Wireless Sensor Networks (WSN) - sont considérés comme un type spécial de réseaux ad hoc. Les nœuds de ce type de réseaux consistent en un grand nombre de micro-capteurs capables de récolter et de transmettre des données environnementales d'une manière autonome. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils sont dispersés aléatoirement à travers une zone géographique, appelée champ de captage, qui définit le terrain d'intérêt pour le phénomène capté. Les données captées sont acheminées grâce à un routage multi-saut à un nœud considéré comme un "point de collecte", appelé nœud puit (ou sink). Ce dernier peut être connecté à l'utilisateur du réseau via Internet ou un satellite. Ainsi, l'usager peut adresser des requêtes aux autres nœuds du réseau, précisant le type de données requises et récolter les données environnementales captées par le biais du nœud puit.

1.3.3. Applications des réseaux de capteurs

La taille de plus en plus réduite des micro-capteurs, le coût de plus en plus faible, la large gamme des types de capteurs disponibles (thermique, optique, vibrations,...) ainsi que le

support de communication sans fil utilisé, permettent aux réseaux de capteurs d'envahir plusieurs domaines d'applications. Ils permettent aussi d'étendre les applications existantes et de faciliter la conception d'autres systèmes tels que le contrôle et l'automatisation des chaînes de montage. Les réseaux de capteurs ont le potentiel de révolutionner la manière même de comprendre et de construire les systèmes physiques complexes. Les réseaux de capteurs peuvent se révéler très utiles dans de nombreuses applications lorsqu'il s'agit de collecter et de traiter des informations provenant de l'environnement. Parmi les domaines où ces réseaux peuvent offrir les meilleures contributions, nous citons les domaines : militaire, environnemental, domestique, santé, sécurité, etc. Des exemples d'applications potentielles dans ces différents domaines sont exposés ci-dessous.

a. Applications militaires

Comme dans le cas de plusieurs technologies, le domaine militaire a été un moteur initial pour le développement des réseaux de capteurs. Le déploiement rapide, le coût réduit, l'auto-organisation et la tolérance aux pannes des réseaux de capteurs sont des caractéristiques qui rendent ce type de réseaux un outil appréciable dans un tel domaine. Comme exemple d'application dans ce domaine, on peut penser à un réseau de capteurs déployé sur un endroit stratégique ou difficile d'accès, afin de surveiller toutes les activités des forces ennemies, ou d'analyser le terrain avant d'y envoyer des troupes (détection d'agents chimiques, biologiques ou de radiations). Des tests concluants ont déjà été réalisés dans ce domaine par l'armée américaine dans le désert de Californie.

b. Applications à la sécurité

Les altérations dans la structure d'un bâtiment, suite à un séisme ou au vieillissement, pourraient être détectées par des capteurs intégrés dans les murs ou dans le béton, sans alimentation électrique ou autres connexions filaires. Les capteurs doivent s'activer périodiquement et peuvent ainsi fonctionner durant des années, voire des décennies. Un réseau de capteurs de mouvements peut constituer un système d'alarme distribué qui servira à détecter les intrusions sur un large secteur. Déconnecter le système ne serait plus

aussi simple, puisqu'il n'existe pas de point critique. La surveillance de voies ferrées pour prévenir des accidents avec des animaux et des êtres humains peut être une application intéressante des réseaux de capteurs. La protection des barrages pourrait être accomplie en y introduisant des capteurs. La détection prompte de fuites d'eau permettrait d'éviter des dégâts. Les êtres humains sont conscients des risques et attaques qui les menacent. Du coup, ils mettent à disposition toutes les ressources humaines et financières nécessaires pour leur sécurité. Cependant, des failles sont toujours présentes dans les mécanismes de sécurisation appliqués aujourd'hui, sans oublier leur coût très élevé. L'application des réseaux de capteurs dans le domaine de la sécurité pourrait diminuer considérablement les dépenses financières consacrées à la sécurisation des lieux et à la protection des êtres humains tout en garantissant de meilleurs résultats.

c. Applications environnementales

Des thermo-capteurs dispersés à partir d'un avion sur une forêt peuvent signaler un éventuel début d'incendie dans le champ de captage ; ce qui permettra une meilleure efficacité pour la lutte contre les feux de forêt. Dans les champs agricoles, les capteurs peuvent être semés avec les graines. Ainsi, les zones sèches seront facilement identifiées et l'irrigation sera donc plus efficace. Sur les sites industriels, les centrales nucléaires ou dans les pétroliers, des capteurs peuvent être déployés pour détecter des fuites de produits toxiques (gaz, produits chimiques, éléments radioactifs, pétrole, etc.) et alerter les utilisateurs dans un délai suffisamment court pour permettre une intervention efficace. Une grande quantité de capteurs peut être déployée en forêt ou dans un environnement de conservation de la faune afin de recueillir des informations diverses sur l'état du milieu naturel et sur les comportements de déplacement. Par exemple, l'université de Pise en Italie a réalisé des réseaux de capteurs pour le contrôle des parcs naturels (feux, animaux,..). Il est ainsi possible "d'observer", sans déranger, des espèces animales difficiles à étudier dans leur environnement naturel et de proposer des solutions plus efficaces pour la conservation de la faune. Les éventuelles conséquences de la dispersion en masse des micro-capteurs dans l'environnement ont soulevé plusieurs inquiétudes. En effet, chaque micro-capteur est doté d'une batterie qui contient des métaux nocifs. Néanmoins, le déploiement d'un million de

capteurs de 1 millimètre cube chacun ne représente qu'un volume total d'un litre. Même si tout ce volume était constitué de batteries, cela n'aurait pas des répercussions désastreuses sur l'environnement.

d. Applications médicales

On pourrait imaginer que dans le futur, la surveillance des fonctions vitales de l'être humain serait possible grâce à des micro-capteurs qui pourront être avalés ou implantés sous la peau. Actuellement, des micro-caméras qui peuvent être avalées existent. Elles sont capables, sans avoir recours à la chirurgie, de transmettre des images de l'intérieur d'un corps humain avec une autonomie de 24 heures. Les auteurs d'une récente étude, présentent des capteurs qui fonctionnent à l'intérieur du corps humain pour traiter certains types de maladies. Leur projet actuel est de créer une rétine artificielle composée de 100 micro-capteurs pour corriger la vue. D'autres ambitieuses applications biomédicales sont aussi présentées, tel que : la surveillance du niveau de glucose, le monitoring des organes vitaux ou la détection de cancers. L'utilisation des réseaux de capteurs dans le domaine de la médecine pourrait apporter une surveillance permanente des patients et une possibilité de collecter des informations physiologiques de meilleure qualité, facilitant ainsi le diagnostic de quelques maladies.

e. Applications commerciales

Il est possible d'intégrer des nœuds capteurs au processus de stockage et de livraison. Le réseau ainsi formé, pourra être utilisé pour connaître la position, l'état et la direction d'un paquet ou d'une cargaison. Il devient alors possible pour un client qui attend la réception d'un paquet, d'avoir un avis de livraison en temps réel et de connaître la position actuelle du paquet. Pour les entreprises manufacturières, les réseaux de capteurs permettront de suivre le procédé de production à partir des matières premières jusqu'au produit final livré. Grâce aux réseaux de capteurs, les entreprises pourraient offrir une meilleure qualité de service tout en réduisant leurs coûts. Dans les immeubles, le système de climatisation peut être conçu en intégrant plusieurs micro-capteurs dans les tuiles du plancher et les meubles. Ainsi,

La climatisation pourra être déclenchée seulement aux endroits où il y a des personnes présentes et seulement si c'est nécessaire. Le système distribué pourra aussi maintenir une température homogène dans les pièces. Utilisée à grande échelle, une telle application permettrait de réduire la demande mondiale en énergie réduisant du même coup les gaz à effet de serre. Rien que pour les États-Unis, on estime cette économie à 55 milliards de dollars par an avec une diminution de 35 millions de tonnes des émissions de carbone dans l'air. Ainsi, dans un contexte mondial où le réchauffement de la planète devient une préoccupation grandissante, une telle conséquence environnementale serait un pas dans la bonne direction.

1.3.4. Plates-formes

Parmi les standards les plus aptes à être exploités dans les réseaux de capteurs sans-fil se retrouvent la double pile protocolaire Bluetooth / Zigbee. La technique Bluetooth, dont Ericsson a initié le projet en 1994, a été standardisée sous la norme IEEE 802.4.15 et a comme but la création et le maintien de réseaux à portée personnelle, PAN (Personal Area Network). Un tel réseau est utilisé pour le transfert de données à bas débit à faible distance entre appareils compatibles. Malheureusement, le grand défaut de cette technique est sa trop grande consommation d'énergie et ne peut donc pas être utilisée par des capteurs qui sont alimentés par une batterie et qui idéalement devraient fonctionner durant plusieurs années. Le standard ZigBee combiné avec IEEE 802.15.4 offre des caractéristiques qui répondent encore mieux aux besoins des réseaux de capteurs en termes d'économies d'énergie. ZigBee offre des débits de données moindres, mais il consomme également nettement moins que Bluetooth. Un petit débit de données n'est pas handicapant pour un réseau de capteurs où les fréquences de transmission ne sont pas importantes. Malgré tout, la tendance actuelle des constructeurs est d'employer des techniques propriétaires qui ont pour avantage d'être spécifiquement optimisées pour une utilisation précise, mais qui ont comme gros inconvénient de ne pas être compatibles entre elles.

1.3.5. Matérielles

De nouvelles techniques vont influencer considérablement l'avenir des réseaux de capteurs. UWB (Ultra Wide Band) en est un très bon exemple. Cette technique de transmission permettra d'atteindre des niveaux de consommation extrêmement bas grâce à sa simplicité au niveau matériel. De plus, l'atténuation du signal engendré par des obstacles est moindre qu'avec les systèmes radio à bande étroite conventionnels.

1.3.6. Logiciels

Le domaine des capteurs sans fil se dirige donc vers un grand essor et de très nombreux nouveaux produits risquent d'inonder les marchés ces prochaines années. D'autant plus que des techniques "open-source" s'associent à ce succès avéré, comme par exemple TinyOS qui a été développé à l'Université de Berkeley. TinyOS est un système d'exploitation "open source" conçu pour les capteurs embarqués sans-fil qui est actuellement utilisé par plus de 500 universités et centres de recherche dans le monde. La réalisation de programmes sur cette plateforme s'effectue exclusivement en NesC (dialecte du C). La particularité principale de cet OS est sa taille extrêmement réduite en termes de mémoire (quelques kilooctets).

1.3.7. Architecture d'un micro-capteur

Un nœud capteur contient quatre unités de base : l'unité de captage, l'unité de traitement, l'unité de transmission, et l'unité de contrôle d'énergie. Il peut contenir également, suivant son domaine d'application, des modules supplémentaires tels qu'un système de localisation (GPS), ou bien un système générateur d'énergie (cellule solaire). On peut même trouver des micro-capteurs, un peu plus volumineux, dotés d'un système mobilisateur chargé de déplacer le micro-capteur en cas de nécessité.

a. L'unité de captage

Le capteur est généralement composée de deux sous-unités : le récepteur (reconnaissant l'analyte) et le transducteur (convertissant le signal du récepteur en signal électrique). Le capteur est responsable de fournir des signaux analogiques, basés sur le phénomène

observé, au convertisseur Analogique/Numérique. Ce dernier transforme ces signaux en un signal numérique compréhensible par l'unité de traitement.

b. L'unité de traitement

L'unité de traitement comprend un processeur associé généralement à une petite unité de stockage et fonctionne à l'aide d'un système d'exploitation spécialement conçu pour les micro-capteurs (TinyOS par exemple). Cette unité est chargée d'exécuter les protocoles de communications qui permettent de faire collaborer le nœud avec les autres nœuds du réseau. Elle peut aussi analyser les données captées pour alléger la tâche du nœud puits.

c. L'unité de transmission

Cette unité est responsable d'effectuer toutes les émissions et réceptions des données sur un medium sans fil. Elle peut être de type optique (comme dans les nœuds Smart Dust), ou de type radio-fréquence. Les communications de type optique sont robustes vis-à-vis des interférences électriques. Néanmoins, elles présentent l'inconvénient d'exiger une ligne de vue permanente entre les entités communicantes. Par conséquent, elles ne peuvent pas établir de liaisons à travers des obstacles. Les unités de transmission de type radio-fréquence comprennent des circuits de modulation, démodulation, filtrage et multiplexage ; ce qui implique une augmentation de la complexité et du coût de production du micro-capteur. Concevoir des unités de transmission de type radio-fréquence avec une faible consommation d'énergie est un véritable défi. En effet, pour qu'un nœud ait une portée de communication suffisamment grande, il est nécessaire d'utiliser un signal assez puissant. Cependant, l'énergie consommée serait importante. L'autre alternative serait d'utiliser de longues antennes, mais ceci n'est pas possible à cause de la taille réduite des micro-capteurs.

d. L'unité de contrôle d'énergie

Un micro-capteur est muni d'une ressource énergétique (généralement une batterie) pour alimenter tous ses composants. Cependant, en conséquence de sa taille réduite, la ressource énergétique dont il dispose est limitée et généralement irremplaçable. Dès lors, l'énergie est la ressource la plus précieuse dans un réseau de capteurs, puisqu'elle influe directement sur la durée de vie des micro-capteurs et du réseau en entier. L'unité de contrôle d'énergie

constitue donc l'un des systèmes les plus importants. Elle est responsable de répartir l'énergie disponible aux autres modules et de réduire les dépenses en mettant en veille les composants inactifs par exemple. Cette unité peut aussi gérer des systèmes de rechargement d'énergie à partir de l'environnement observé telles que les cellules solaires, afin d'étendre la durée de vie totale du réseau.

1.3.8. Architectures Réseau de capteurs

Il existe plusieurs topologies pour les réseaux à communication radio. Nous discutons dans ce qui suit les topologies applicables sur les réseaux de capteurs.

a. La Topologie en étoile

Dans cette topologie une station de base peut envoyer ou recevoir un message à certaine nombre de nœuds .Ces nœuds peuvent seulement envoyer ou recevoir un message de l'unique station de base, il ne leur est pas permis de s'échanger des messages. L'avantage de cette topologie est sa simplicité, sa capacité à minimiser la consommation d'énergie des nœuds et la minimisation de latence de la communication entre les nœuds et la station de base .Son inconvénient est que la station de base n'est pas robuste puisque tout le réseau est géré par un seul nœud.

b. La topologie en grille (Mesh Network)

Dans ce type de topologie, n'importe quel nœud peut envoyer à n'importe quel autre nœud dans le réseau qui est à sa portée de transmission. Ceci est appelé la communication multi-sauts, dans laquelle, si un nœud veut transmettre un message à un autre nœud qui est en dehors de sa portée de transmission, il peut utiliser un nœud intermédiaire pour envoyer son message au nœud destinataire. L'avantage de cette topologie est la possibilité du passage à l'échelle, la redondance et la tolérance aux fautes, L'inconvénient de cette topologie est la consommation d'énergie dans la communication multi-sauts et la latence qui sont créés par le passage des messages des nœuds par plusieurs autres avant d'arriver à la station de base.

c. La topologie hybride

Une topologie hybride entre celle en étoile et en grille fournit des communications réseau robustes et diverses, en assurant la minimisation de la consommation d'énergie dans les réseaux de capteurs. Dans ce type de topologie, les nœuds capteur à faible puissance ne routent pas les messages, mais il y a d'autres nœuds qui ont la possibilité de faire le routage des messages. En général, ces nœuds ont une puissance élevée.

1.4. Conclusion :

A partir de ce qui précède, nous déduisons les différences majeures entre les réseaux Ad hoc et les réseaux de capteurs et leurs particularités. Nous les résumons dans le tableau suivant :

Réseau Ad hoc	Réseau de capteurs
Générique / Communication	Objectif ciblé
Chaque nœud a son propre objectif	Nœuds collaborent pour remplir un objectif
Flots de données « Any-to-Any »	Flots de données « Many-to-one »
Notion d'ID	Très grand nombre de nœuds n'ayant pas tous un ID
Débit est majeur	Energie est un facteur déterminant
Communication point à point	Communication broadcast

Table 1.1. Comparaison réseaux Ad-hoc / Capteurs

Chapitre 2
Les protocoles de minimisation de la consommation d'énergie

Chapitre 2

Les protocoles de minimisation de la consommation d'énergie

2.1. Introduction

Les nœuds mobiles dans les réseaux Ad hoc se déplacent librement avec une batterie à énergie limitée. Dans l'absence d'une entité de contrôle centrale comme une station de base, chaque nœud participe avec un protocole distribué pour gérer et contrôler le réseau.

Chaque nœud se comporte comme un routeur, et peut communiquer ou échanger des informations directement ou indirectement par l'intermédiaire d'autres nœuds. Il faut donc économiser l'énergie.

Dans les années passées, plusieurs protocoles ont été proposés, appelés les protocoles de l'énergie consciente (Energy-aware protocols).

Certains protocoles essaient de réduire le trafic, et donc l'énergie dépensée. LMST [8], et SPAN [14] sont basées sur la diffusion avec le minimum d'énergie en utilisant "Minimum Spanning Tree development".

Une autre série de protocoles est basée sur le contrôle de la topologie et le routage à énergie minimale. Les nœuds peuvent modifier leur niveau d'énergie de transmission dans le but de :

- maintenir le niveau de connectivité de la topologie du réseau ;
- accroître la capacité du réseau ;
- réduire la consommation des interférences et de l'énergie.

Dans ce chapitre, nous présentons la classification des protocoles qui aident à minimiser l'énergie selon le modèle des couches réseaux OSI, à savoir la couche physique, la couche liaison de données et la couche réseaux.

2.2. La couche physique :

Actuellement, les études se font sur l'optimisation de l'énergie des équipements et du niveau de circuit et leur management dans les périphériques sans fil (exemple : Affichage avec énergie basse, CPU avec moins d'énergie, …).

L'énergie de transmission peut être ajustée dans la couche physique. L'utilisation excessive d'une énergie de transmission peut augmenter les interférences ainsi que l'énergie de transmission des autres nœuds. Pour éviter cela, la couche physique doit contenir les données de transmission avec un niveau d'énergie minimum pour maintenir les liens et l'adapter aux changements. Un contrôle d'énergie peut maintenir un lien avec le niveau d'énergie minimum, mais peut aussi protéger un lien existant contre l'interférence en augmentant l'énergie de transmission. Nous pouvons citer les protocoles suivants:

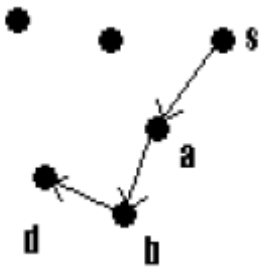
1. COMPOW
2. CLUSTERPOW
3. IMPROVING CLUSTERPOW
4. TUNNELLED CLUSTERPOW
5. MINPOW

2.2.1. COMPOW (COMmon POWer):

Ce protocole sert à choisir un niveau de puissance commun qui permettra à n'importe quelle paire de nœuds de communiquer via d'autres nœuds.

Le protocole COMPOW augmente la capacité de réseau sous la contrainte d'un niveau de puissance commun dans le réseau.

Exemple :



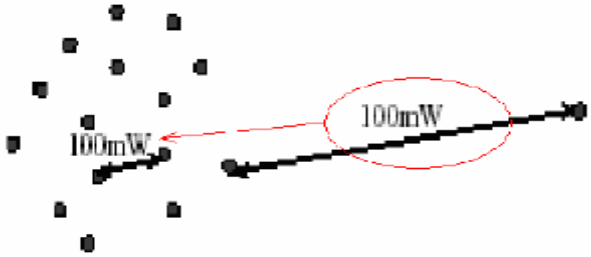
L'énergie requise entre la source (s) et (a) : 15mW

L'énergie requise entre (a) et (b) : 10mW

L'énergie requise entre (b) et la destination (d) : 5mW

Dans ce réseau, l'énergie maximale requise pour connecter tous les noeuds est de 15mW, c'est la puissance commune de réseau.

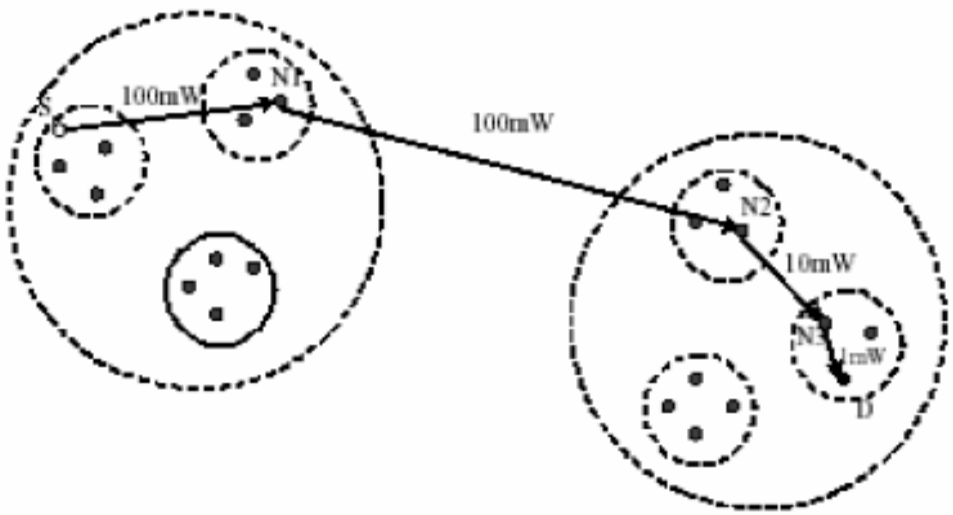
Néanmoins ce protocole est limité puisqu'il est nécessaire d'avoir une topologie homogène du réseau. En effet on voit sur le schéma ci-dessous que si un nœud est isolé, il porte alors la puissance commune à un niveau élevé. Ce qui augmente les interférences. [10]



2.2.2. CLUSTERPOW :

Le but de CLUSTERPOW est d'optimiser la capacité du réseau. Quand la distribution spatiale des noeuds n'est pas homogène, ce protocole n'est pas optimal pour utiliser un niveau de puissance commun dans tout le réseau.

Ce protocole est basé sur les clusters.



Lorsque S veut atteindre D, il passe par les clusters intermédiaires. Dans chacun de ces clusters un seul noeud est utilisé pour cette communication. Ce protocole utilise la plus petite puissance nécessaire entre S et N1, N1 et N2, N2 et N3, et N3 et D (Voir les tables de routage de chaque nœud ci-dessous)

Nous avons trois tables qui correspondent à chaque niveau de puissance que l'on retrouve dans l'exemple précédent, et un tableau de routage Kernel.

Les tables sont comme suit :

1 mW

Destination	**Saut suivant**	**Métrique**
D		Inf

10 mW

Destination	**Saut suivant**	**Métrique**
D		Inf

100 mW

Destination	Saut suivant	Métrique
D	N1	3

Table de routage Kernel

Destination	Saut suivant	Métrique	Energie commune
D	N1	3	100 mW

Table 2.1. Table de routage de la source S

Dans ce cas, seule la table qui concerne 100mW contient des informations, car de S à N1, il est nécessaire d'émettre à 100mW. La table Kernel indique le saut suivant, la puissance nécessaire, le nombre de saut ainsi que la destination finale.

1 mW

Destination	Saut suivant	Métrique
D		Inf

10 mW

Destination	Saut suivant	Métrique
D		Inf

100 mW

Destination	Saut suivant	Métrique
D	N2	3

Table de routage Kernel

Destination	Saut suivant	Métrique	Energie commune
D	N2	3	100 mW

Table 2.2. Table de routage de N1

De la même manière, pour que N1 atteigne N2, il doit émettre à 100mW.

1 mW

Destination	Saut suivant	Métrique
D		Inf

10 mW

Destination	Saut suivant	Métrique
D	N3	2

100 mW

Destination	Saut suivant	Métrique
D	D	1

Table de routage Kernel

Destination	Saut suivant	Métrique	Energie
D	N3	2	10 mW

Table 2.3. Table de routage de N2

Le passage de N2 à N3 se fait avec une puissance de 10mW.

1 mW

Destination	Saut suivant	Métrique
D	D	1

10 mW

Destination	Saut suivant	Métrique
D	D	1

100 mW

Destination	Saut suivant	Métrique
D	D	1

Table de routage Kernel

Destination	Saut suivant	Métrique	Energie commune

D	D	1	1 mW

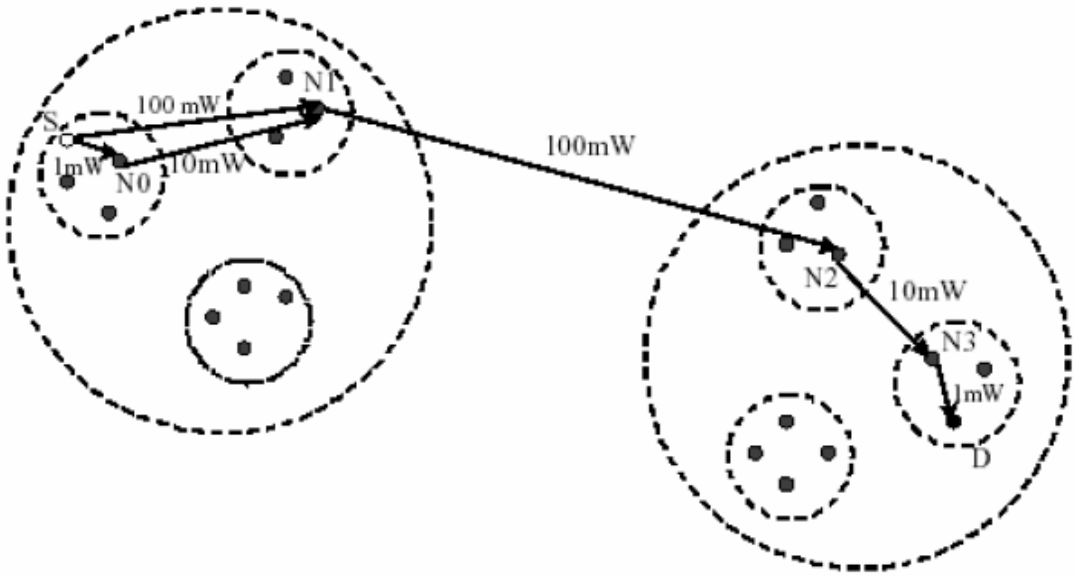
Table 2.4. Table de routage de N3

Les propriétés du Clusterpow se résument comme suit :

- Le choix des routes est fait en n'ayant pas d'augmentation de la puissance
- Le niveau de puissance d'un paquet est dépendant de la destination
- Clusterpow peut être utilisé avec tous protocoles de routage

2.2.3. IMPROVING CLUSTERPOW

C'est une amélioration de Clusterpow, il est basé sur la recherche récursive du noeud suivant menant à la destination avec le plus bas niveau de puissance à chaque étape.

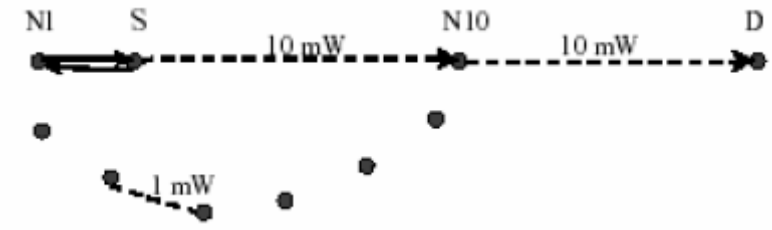


En effet, pour atteindre N1, S passe par N0, ce qui permet une économie de puissance, qui peut réduire le taux de collision dû aux interférences.

Par contre si le trafic n'est pas dense, il sera préférable d'utiliser la méthode précédente (ClusterPow). La meilleure solution est de trouver un compromis entre les deux systèmes.

Toute fois cette technique peut engendrer l'entrée des paquets dans une boucle infinie.

Exemple :

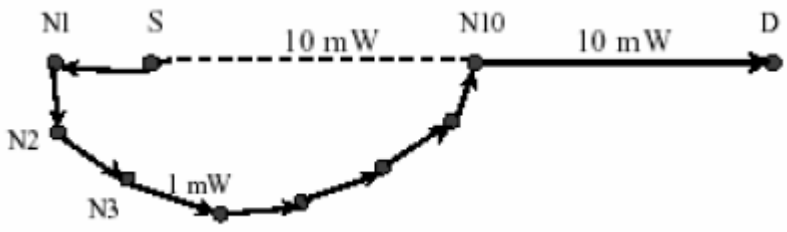


Dans cet exemple, S veut envoyer à D, son prochain nœud est N10. Par contre, le protocole Improving Clusterpow indique (par soucis d'économie de puissance) qu'il faut passer par N1 qui a pour noeud suivant S. Donc le paquet entre dans une boucle infinie entre S et N1. [11]

Pour éviter cette situation, Tunnelled Clusterpow est proposé.

2.2.4. TUNNELLED CLUSTERPOW

Même principe que Improving clusterpow, sauf que dans ce cas, lors de la consultation récursive, l'adresse du nœud suivant est mis dans le paquet, pour éviter les boucles infinies. [11]



S envoi à D

S envoi à N1 à 1mW et la destination est N10 maintenant

S envoi à N10 via N2, N3 …

N10 décapsule le paquet et envoi à D

2.2.5. MINPOW

MINPOW est un protocole qui minimise la consommation de l'énergie totale (d'une communication) sur une route. La métrique de coût utilisée pour un saut est l'algorithme de Bellman-Ford distribué et la consommation d'énergie totale.

L'idée fondamentale de MINPOW n'est pas nouvelle, elle a été proposée avec différentes formes [2], [3], [4], [5].

Plusieurs métriques ont été proposées comme le signal, le coût de l'énergie consommée de la transmission et la durée de vie de batterie restante. Cette méthode marche aussi bien sur les protocoles proactifs que les réactifs.

Le coût d'une route ou la consommation de l'énergie de communication, a trois composants :

- PRxelec : énergie consommée dans les électroniques du récepteur,
- PTxelec : énergie consommée dans les électroniques de l'émetteur,
- PTxRod(p) : énergie consommée par l'amplifier pour transmettre un paquet avec le niveau d'énergie p qui est mis dans l'aire.

PTxelec et PRxelec sont connues localement par l'émetteur et le récepteur respectivement, selon les spécifications de leurs équipements. Le but est d'estimer la plus petite énergie de transmission nécessaire pour traverser la route.

L'énergie de transmission minimum nécessaire est calculée en mesurant la distance entre les deux nœuds. L'énergie de réception avec une distance *p* d'un émetteur en utilisant un niveau d'énergie Ptrans est $\frac{cP_{\text{trans}}}{\rho^{\alpha}}$ où c est un constant.

Pour recevoir un paquet, l'énergie de réception doit être supérieure ou égale à γ $\frac{cP_{\text{trans}}}{\rho^{\alpha}} \geq \gamma$.

L'énergie de transmission minimum nécessaire est au moins égale à $\frac{\gamma \rho^{\alpha}}{c}$. (α : la perte d'énergie α >= 2)

Donc, Ptrans peut-être estimé, sachant que l'énergie de réception et la distance de transmission sont connues. [11]

2.2.6. Synthèse

Nous résumons les différentes techniques de minimisation d'énergie utilisées au niveau de la couche physique comme suit :

	COMPOW	**CLUSTERPOW**	**IMPROVING CLUSTERPOW**	**TUNNELLED CLUSTERPOW**	**MINPOW**
Type de réseau	Adhoc	Adhoc	Adhoc	Adhoc	Adhoc
But	Minimiser la consommation de l'énergie	Minimiser la consommation de l'énergie	Minimiser la consommation de l'énergie	Minimiser la consommation de l'énergie	Minimiser la consommation de l'énergie
Principe	La puissance commune	Clusters	Amélioration ClusterPow	Même principe que Improving clusterpow avec attribution de l'adresse du nœud suivant dans le paquet	Estimer la plus petite énergie de transmission nécessaire pour traverser la route
Interférence	Oui (si un noeud est isolé)	Non	Non	Non	Non
Avantages	Bien pour les	Bien pour les	Bien pour les	Evite les	-

	topologies homogènes	topologies non homogènes	topologies non homogènes	boucles infinies	
Inconvénients	Mauvais si un noeud est isolé	Un seul nœud dans le cluster est le coordinateur	Boucle infinie possible	-	-
Boucle infinie	Non	Non	Oui	Non	Non
Trafic préféré	-	Non dense	Dense	-	-
Topologie préférée	Homogène	Non Homogène	Non homogène	-	-

Table 2.5. Comparaison entre les protocoles de la couche physique

2.3. La couche liaisons de données :

Dans la couche liaisons de données, la conservation de l'énergie peut être faite en utilisant des requêtes de retransmission effective et l'opération de mode "Sleep". Elle détecte les erreurs de transmissions, puis elle retransmet le paquet correct en utilisant une requête de répétition automatique (ARQ).

Dans les réseaux Ad Hoc, à cause de la présence de la mobilité et des interférences, les erreurs de transmission peuvent se produire fréquemment, ce qui conduit à des requêtes fréquentes de retransmission. Tant que les retransmissions augmentent la consommation de l'énergie, un nouveau schéma de requête de retransmission efficace est nécessaire pour les réseaux ad hoc.

Un schéma possible est le suivant:

Quand un émetteur ne reçoit pas des ACK après les transmissions des paquets, le schéma de requête de retransmissions sait que le canal de transmission est mal et cesse de transmettre

pour ne pas perdre l'énergie quand les chances de succès de réception sont basses [8]. Ce schéma réduit l'énergie inutilement perdue.

Une autre solution pour les schémas de requête de retransmissions pour augmenter l'énergie de retransmission, consiste à réduire la possibilité des erreurs de transmission mais à augmenter le ratio du signal d'interférence (SIR) du réseau.

Par conséquent, il est important de déterminer quand et à quel niveau d'énergie un noeud mobile doit tenter de retransmettre, pour minimiser l'énergie dépensée dans la couche lisions de données. [9]

Ceci peut se faire grâce aux protocoles suivants :

1. PSM
2. Slotted PSM
3. TA-PSM
4. S-MAC et T-MAC
5. SPAN

2.3.1. PSM

Un des mécanismes de référence pour la conservation d'énergie dans les réseaux sans fil est PSM (Power Saving Mode) proposé par la norme IEEE 802.11 [20].

Du point de vue consommation en énergie, un noeud peut avoir trois états. Il peut être:

1. **en réception**, dissipant une certaine énergie,
2. **en émission**, dissipant une autre énergie
3. ou **en mode veille** (écoute passive du canal) et dans ce cas il dissipe aussi de l'énergie.

Le standard 802.11 définit le PS (Power Save) pour conserver l'énergie des noeuds dans un réseau sans fil avec ou sans infrastructure.

Une station utilisant PSM peut voir un des deux états suivants à un instant donné:

- Awake : la station utilise toute sa puissance pour envoyer et recevoir des paquets à tout moment.

- Doze : la station est incapable de transmettre ou de recevoir, elle utilise le minimum de son énergie. Si elle a des messages à envoyer, elle les sauvegarde localement.

Dans 802.11, on définit deux modes de gestion d'énergie :

- Active mode (AM) : la station peut recevoir des paquets à n'importe quel instant. Elle est dans l'état Awake.
- Power Save (PS) : la station peut être dans l'état Doze mais doit se réveiller et revenir à l'état Awake pour recevoir certains messages de contrôle, de synchronisation et recevoir les paquets sauvegardés.

Tous les noeuds du réseau doivent être réveillés en même temps pour échanger les messages qu'ils auront sauvegardés durant la période doze. De ce fait, 802.11 définit une fonction de synchronisation qui se base sur l'échange d'un paquet appelé Beacon au début de chaque période BeaconInterval.

Cette période commence par une sous période appelée ATIMWindow durant laquelle uniquement les messages de contrôle et des annonces (messages ATIM : Announcement Trafic Indication Message) sur des messages sauvegardés doivent être échangés entre les noeuds.

Au terme du ATIMWindow, si un noeud n'a annoncé aucun paquet à envoyer et n'a reçu aucune annonce alors il peut entrer en état Doze pour le reste de BeaconInterval. Dans le cas contraire, il doit rester Awake pour réaliser les transferts nécessaires.

La synchronisation des noeuds du réseau au niveau du mécanisme PSM fait que tous les noeuds utilisant le mode PS doivent, si leurs activités le permettent, entrer en mode Doze durant la même période puis se réveiller aux mêmes instants. Cette caractéristique fait que durant une communication entre une source et une destination utilisant une route à plusieurs sauts, plusieurs noeuds en mode Doze peuvent se trouver sur la route. Un paquet traversant le réseau peut rencontrer tout un îlot de noeuds en mode Doze empêchant le paquet d'être routé à destination. Ceci représente une faille au niveau de PSM qui peut induire un risque de partitionnement du réseau au moment du routage.

Par ailleurs, un noeud utilisant le mode PSM restera actif durant tout le reste de la période BeaconInterval s'il a reçu des annonces de messages durant la période ATIM ou s'il a des messages à envoyer. Cependant, le noeud ne sera pas réellement en communication durant toute la période puisqu'il n'aura pas accès au canal à tout instant et peut terminer la réception ou la transmission de ses données avant la fin de la période.

2.3.2. Slotted PSM

Une amélioration de PSM consiste à limiter la durée d'activité d'un nœud durant le reste d'un BeaconInterval afin de limiter la consommation d'énergie inutile et ceci en lui allouant des slots prédéfinis pour l'échange de ses données. En dehors de ses slots, le noeud peut entrer en mode doze.

Slotted PSM [21] consiste donc à diviser la période allant de la fin de la période ATIM jusqu'au reste de la période BeaconInterval en un certain nombre de slots de temps, chacun sera alloué pour un noeud donné pour effectuer sa communication. Chaque noeud restera actif uniquement durant les slots qui lui seront alloués et pourra ainsi être en mode doze plus longtemps.

La réservation des slots pour l'échange de données s'effectue dynamiquement durant la période ATIMWindow. La source inclut les informations de réservation de slots dans les messages ATIM. Chaque noeud maintient une table de réservation de slots qui sera actualisée à chaque réception de trames ATIM ou ACK-ATIM. Un noeud source choisit les slots qu'il va inclure dans la trame ATIM en se basant sur ses informations locales à partir de la table des états des slots et la quantité du trafic qui sera échangé.

A la réception du message ATIM, le noeud destination réplique par un paquet ATIMRE (ATIM Response) dans lequel il inclut la liste de ses slots libres. La source rectifie alors sa réservation selon le nouvel état des slots et envoie un nouveau paquet ATIM.

2.3.3. TA-PSM

Une amélioration récente de PSM, adoptant le même principe que Slotted PSM, est le mécanisme TA-PSM (Trafic Aware Power Saving Mode) [22]. Cette nouvelle approche est

partie de la constatation du fait que deux noeuds en mode PS restent en état Awake pour le reste de BeaconInterval afin d'échanger des paquets de données, mais peuvent terminer leur communication avant la fin de cette période. TA-PSM consiste à réduire la consommation d'énergie des noeuds du réseau activant PSM en les rendant plus sensibles à la charge du trafic. Chaque source doit à cet effet, indiquer à sa destination l'éventuelle fin de leur communication et ceci en activant un champ particulier dans le dernier paquet qu'elle lui adresse.

Ce mécanisme permettra ainsi à une source et une destination d'entrer immédiatement en mode Doze dès qu'elles ne sont plus impliquées dans aucun trafic, sans attendre la fin d'un BeaconInterval et ceci jusqu'à la fin de cette période. Ceci est réalisé par l'ajout d'une indication dans le dernier paquet transmis de la source à la destination indiquant ainsi la fin du transfert de données entre eux.

2.3.4. SMAC

a/ Bases

SMAC (Sensor MAC) [23] est un protocole de couche de liaison relativement similaire au protocole 802.11 et qui utilise une méthode d'accès au canal de type CSMA-CA RTS/CTS (Request-To-Send, Clear-To-Send) qui permet d'éviter les collisions et le problème de station cachée. La principale innovation, apportée par ce protocole, est d'avoir développé un mécanisme de mise en veille distribué à chaque nœud du réseau dans le but de réduire la consommation énergétique des équipements réseau. Avec le protocole S-MAC chaque nœud peut périodiquement se mettre en veille, la principale difficulté est alors de synchroniser les nœuds entre eux pour que la communication soit toujours possible. S-MAC définit donc un frame composé d'une période de sommeil et d'une période d'éveil. Si un nœud est en train de transmettre ou recevoir des données, il ne suit pas son ordonnanceur de sommeil avant la fin de la transmission.

b/ Synchronisation

Pour synchroniser les nœuds entre eux, le protocole S-MAC permet à chacun des nœuds d'émettre des paquets SYNC qui permettent aux autres nœuds de se synchroniser. Le

paquet SYNC est broadcasté à tous les voisins immédiats. La période active est divisée en deux phases, la première étant réservé à la synchronisation.

Quand un nœud se réveille, il commence par écouter. Si il n'entend rien pendant un certain temps, il choisit un ordonnancement de frame et le transmet par un paquet SYNC qui contient le temps jusqu'à son endormissement.

Si le nœud reçoit un paquet SYNC d'un autre nœud, il suit l'ordonnancement défini dans ce paquet et transmet son propre SYNC en conséquence. Les nœuds retransmettent leur SYNC de temps en temps. Si un nœud a déjà adopté un ordonnancement et qu'il reçoit un paquet SYNC d'un autre nœud avec un autre ordonnancement, alors il doit adopter les 2 ordonnancements. Il doit aussi transmettre un paquet SYNC à l'autre nœud pour que celui-ci apprenne la présence d'un autre ordonnancement. Le fait d'adopter les deux ordonnancements signifie que le nœud va se réveiller au début des deux frames. Il faut donc éviter qu'un nœud adopte plusieurs ordonnancements.

c/ Overhearing Avoidance

Chaque nœud maintient un vecteur d'allocation du réseau (network allocation vector NAV) dans lequel il indique l'activité de ses voisins. Chaque paquet de données contient un champ indiquant le temps restant. Lorsqu'un nœud reçoit un paquet de données qui ne lui est pas destiné, il met à jour son NAV. A chaque intervalle de temps, la valeur du NAV est décrémentée (si elle est non nulle). Ainsi une valeur non nulle du NAV signifie qu'il y a une transmission. Le nœud peut donc s'endormir et ne se réveiller ensuite que lorsque le NAV devient nul.

2.3.5. TMAC

a/ Bases

Dans T-MAC (Timeout MAC) [24], chaque nœud se réveille périodiquement pour communiquer avec ses voisins. Il existe, comme dans S-MAC, des périodes actives et inactives. Les nœuds communiquent entre eux en utilisant un mécanisme de RTS/CTS (Request-To-Send, Clear-To-Send) qui permet d'éviter les collisions et le problème de stations cachées. Un nœud écoute le canal radio et peut potentiellement transmettre tant

qu'il est dans la période active. Une période active se termine quand aucun évènement d'activation ne se produit pendant un temps TA. Un événement d'activation est :

– le déclenchement d'un temporisateur de frame

– la réception de données sur la radio

– la détection (grâce à un indicateur de puissance du signal sur la radio) d'une communication sur le canal radio

– l'acquittement de ses propres paquets

– la connaissance grâce à l'écoute qui précède l'émission de RTS et CTS que la transmission de données d'un voisin s'est terminée.

TA est donc le temps minimum d'écoute à vide par frame.

b/ Synchronisation

Quand un nœud se réveille, il commence par écouter. Si il n'entend rien pendant un certain temps, il choisit un ordonnancement de frame et le transmet par un paquet SYNC qui contient le temps jusqu'au début du frame suivant. Si au démarrage, le nœud reçoit un paquet SYNC d'un autre nœud, il suit l'ordonnancement défini dans ce paquet et transmet son propre SYNC en conséquence. Les nœuds retransmettent leur SYNC de temps en temps. Si un nœud a déjà adopté un ordonnancement et qu'il reçoit un paquet SYNC d'un autre nœud avec un autre ordonnancement, alors il doit adopter les deux ordonnancements. Il doit aussi transmettre un paquet SYNC à l'autre nœud pour que celui-ci apprenne la présence d'un autre ordonnancement.

Le fait d'adopter les deux ordonnancements signifie que le nœud va avoir un évènement d'activation au début de chacune des deux frames. Les nœuds ne doivent débuter une transmission de données qu'au début de leur propre période active. A ce moment, à la fois les voisins ayant le même ordonnancement et ceux l'ayant adopté en plus sont réveillés.

TA doit être suffisamment long pour prendre en compte l'attente avant l'émission du paquet SYNC et le temps de la réception éventuelle d'un paquet RTS.

c/ Overhearing Avoidance

Le mécanisme utilisé dans S-MAC peut-être utilisé dans T-MAC en option. C'est juste une option. Cela peut entrainer dans certaines conditions qu'un nœud manque des RTS et des CTS.

d/ FRTS :

Avec T-MAC, il peut y avoir un problème quand le trafic est essentiellement unidirectionnel (ce qui est le cas quand les capteurs doivent surtout remonter les informations vers une station collectante). Par exemple, supposons qu'un nœud A doit envoyer des données à un nœud D en passant par les nœuds B et C, A ayant pour voisin B, B ayant pour voisins A et C, C ayant pour voisins B et D et D ayant pour voisin C. Considérons le nœud C avec des paquets en attente pour le nœud. Le nœud C peut perdre la contention à cause de B (en recevant un paquet RTS de B) ou à cause de A (en recevant un paquet CTS de B en réponse à un paquet RTS que B a reçu de A). Considérons le deuxième cas. C ayant reçu le CTS de B, il doit rester silencieux et donc D ne reçoit rien. Par conséquent il retourne en sommeil. Quand A a fini de transmettre les données, B émet un paquet d'acquittement (ACK) que tous ses voisins reçoivent. C peut donc espérer obtenir la contention pour transmettre ses paquets à D mais D étant en sommeil, il ne peut pas recevoir le paquet RTS de C.

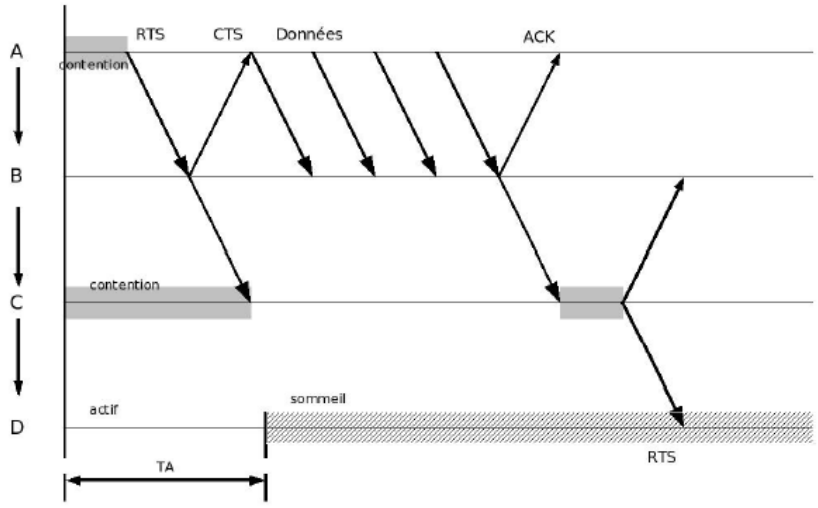


Figure 2.1. Problème de l'endormissement précoce

Une des solutions est l'introduction du future request-to-send (FRTS). Quand C reçoit le CTS de B, C envoie un paquet FRTS à D (B entend ce paquet) qui contient la longueur de la communication de données bloquante (cette information est contenue dans le CTS que C a reçu). Ainsi le nœud peut déterminer qu'il sera ensuite le destinataire d'un paquet RTS et ne pas être endormi quand ce paquet arrivera. (Il faut tenir compte du temps de réception d'un éventuel paquet FRTS pour le calcul de TA). L'ajout du mécanisme de FRTS entraine qu'il faut rajouter un paquet Data-Send (DS) avant l'émission de paquets de données. En effet puisque C émet le paquet FRTS après avoir reçu le paquet CTS de B, A envoie un paquet DS (de même taille qu'un paquet FRTS) après avoir reçu le paquet CTS. Ainsi il ne risque pas de perdre pas la contention et les données qu'il doit envoyer à B n'entrent pas en collision avec le paquet FRTS que B reçoit.

Le mécanisme de FRTS ne doit être utilisé que sous certaines conditions de trafic car sinon cela rajoute une consommation d'énergie qui peut être évitée.

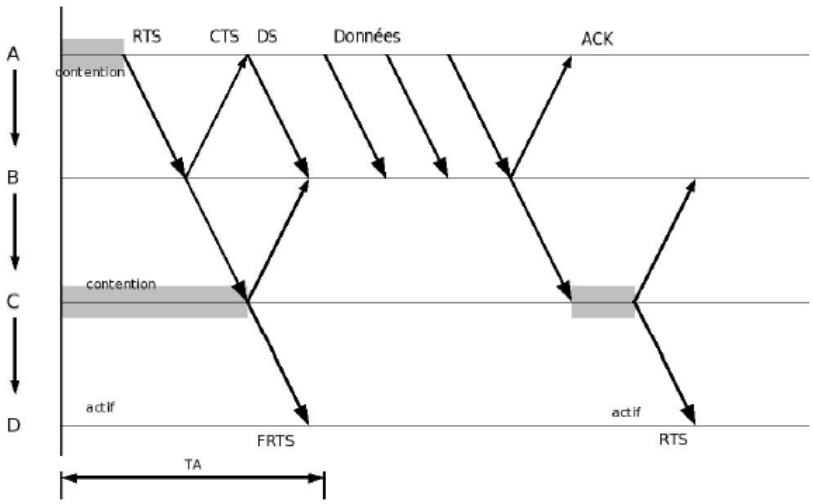


Figure 2.2. Mécanisme de FRTS

2.3.6. SPAN

SPAN, est une technique pour économiser l'énergie qui réduit la consommation sans diminuer la capacité ou la connectivité du réseau.

Span est un algorithme distribué où les nœuds mobiles prennent des décisions locales périodiques s'il va passer à l'état "Sleeping" ou rester "awake" comme un coordinateur et participe dans la topologie de Forwarding BackBone. Chaque nœud base ses décisions sur

une estimation du nombre de ses voisins qui bénéficieront de ses services et de l'énergie disponible pour lui.

Pour préserver la capacité, un nœud décide volontairement d'être un coordinateur s'il découvre que deux de ses voisins ne peuvent pas communiquer entre eux directement ou en passant par un autre coordinateur. Pour minimiser le nombre de coordinateurs redondants, et passer ce rôle à tous les nœuds, chaque nœud reporte d'annoncer sa volonté d'un délai aléatoire en prenant deux facteurs en compte: l'énergie de batterie restante et le nombre de paires des voisins qui peuvent se connecter ensemble.

Span fait cette procédure en utilisant uniquement les informations locales. Span est un protocole proactif où chaque nœud envoie périodiquement un message "Hello" qui contient l'état du nœud (càd s'il est coordinateur ou non), ses coordinateurs actuels, et ses voisins actuels.

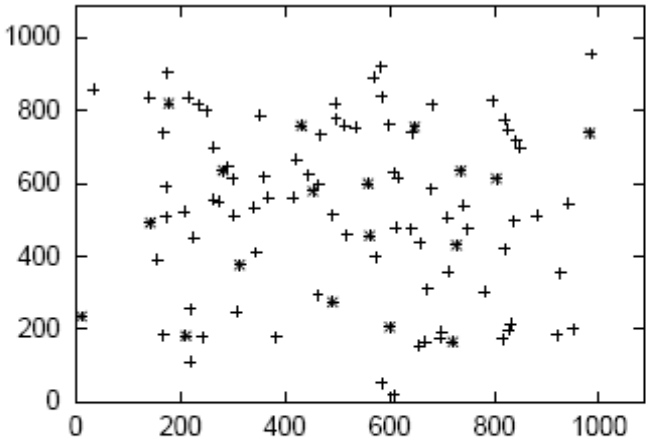


Figure 2.3. Un scénario avec 100 nœuds, 18 coordinateurs

Span se situe au dessus de la couche physique et la couche liaisons de données, et il travaille avec la couche réseau. [14]

Figure 2.4. SPAN dans les couches OSI

2.3.7. Synthèse

Nous résumons les différentes techniques de contrôle d'énergie utilisées par la couche liaison de données comme suit :

	PSM	**Slotted PSM**	**TA-PSM**	**S-MAC**	**T-MAC**	**SPAN**
Type de réseau	Adhoc	Adhoc	Adhoc	Adhoc et sensor network	Adhoc et sensor network	Adhoc
But	Contrôler l'énergie	Contrôler l'énergie	Contrôler l'énergie	Contrôler l'énergie	Contrôler l'énergie	Contrôler l'énergie
Idée	Mode veille	Mode veille	Mode veille	Mode veille	Mode veille	Mode veille
Principe		Allouer des slots prédéfinis pour l'échange des données	Prendre en compte la charge du trafic du réseau	Un nœud entre en mode veille quand un voisin est en cours de transmission	un nœud entre en mode veille quand un voisin est en cours de transmission avec temps ajusté	un nœud décide d'être un coordinateur s'il découvert que deux de ses voisins ne peuvent pas se communiquer
Boucle infinie	Non	Non	Oui	Non	Non	

Table 2.6. Comparaison entre les protocoles de la couche liaison de données

2.4. La couche réseau

Pour maximiser la durée de vie d'un nœud mobile, les algorithmes de routage doivent sélectionner la meilleure route du point de vue des contraintes d'énergie comme une partie de la stabilité d'une route. Les routes qui nécessitent des niveaux plus bas d'énergie de transmission sont préférées. La transmission avec une grande énergie augmente la probabilité de succès de transmission. Mais, cette transmission rend plus haut les interférences aux autres nœuds mobiles, qui peut détruire une bande passante existante et peut obliger le réseau à bloquer les appels et peut diminuer la capacité du réseau. Donc, La transmission avec une basse énergie n'a pas toujours un impacte négatif sur le réseau, au contraire elle peut augmenter le débit entre les terminaux.

Quand l'efficacité de l'énergie est considérée, les réseaux ad hoc ont besoin d'un algorithme de routage qui peut distribuer les paquets à chaque nœud pour éviter aux nœuds d'être sur-utilisés ou maltraités.

Nous pouvons citer les protocoles suivants :

1. MTPR
2. MBCR
3. LMST
4. PARO

Dans la couche réseau, les algorithmes de routage doivent sélectionner la route pour minimiser l'énergie totale nécessaire pour router les paquets dans le réseau et maximiser la durée de vie pour tous les noeuds.

2.4.1. MTPR: (Minimum Total Transmission Power Routing) [30]

Permet de choisir la route minimisant la consommation totale d'énergie en considérant que le meilleur chemin est celui assurant la consommation minimale des capacités des batteries. L'inconvénient est que ce protocole ne reflète pas directement la durée de vie d'un noeud donné et qu'il ne tient pas compte des noeuds qui se réveillent régulièrement et redécouvrent leurs voisins.

2.4.2. MBCR: (Minimum Battery Cost Routing)

L'énergie de transmission totale est une métrique importante, elle concerne la durée de vie des nœuds, mais elle a un inconvénient critique. Cette métrique peut réduire la consommation d'énergie totale du réseau, mais elle ne reflète pas directement sur la durée de vie de chaque nœud. Si les routes d'énergie totale de transmission minimum sont via un nœud spécifique, la batterie de ce nœud va être épuisée rapidement, et ce nœud va mourir. Pour cela, La capacité de batterie restante de chaque nœud est une métrique aussi importante pour décrire la durée de vie de chaque nœud. [1]

La fonction de l'énergie consommée d'un nœud i est:

$$f_i(c_i^t) = \frac{1}{c_i^t}.$$

Tel que c_i^t est la capacité de batterie d'un nœud i dans le temps t.

2.4.3. PARO

PARO, une optimisation de routage d'une énergie consciente qui aide à la minimisation de l'énergie de la transmission nécessaire pour envoyer les paquets entre les unités sans fil dans les réseaux Ad hoc.

En utilisant PARO, un ou plusieurs nœuds intermédiaires appelés "redirecteurs" élit pour envoyer les paquets entre une paire Source-Destination ce qui réduit l'énergie consommée par les équipements sans fil.

PARO peut être appliqué sur différents environnements de réseau comme les réseaux de capteurs, les réseaux domiciles et les réseaux ad hoc mobile. PARO essaye de maximiser le nombre de "redirecteurs" entre la source et la destination pour minimiser l'énergie. Cela est en contradiction directe avec les protocoles de routage comme AODV, DSR, et TORA qui essayent de minimiser le nombre de sauts entre la source et la destination.

Il assume que l'énergie de transmission nécessaire pour transmettre un paquet entre un nœud A et B et presque similaire à l'énergie de transmission entre B et A.

Un noeud laisse son transmiter "ON" pour transmettre un paquet de données à un autre nœud pour L/C secondes où L est la taille frame envoyé en bits, et C est la vitesse brute d'une chaîne sans fil en bits/seconde.

Considérant un réseau composé de plusieurs nœuds statiques. Sachant qu'il y a plusieurs routes alternatives entre une paire Source-Destination dans le réseau et que chaque route implique un ensemble différent et un nombre de nœuds d'envoi. Alors, l'énergie de transmission globale pour envoyer un paquet pour une route « k » est comme suit:

$$P_k = \sum_{i=0}^{N_k} (T_{i,i+1}L + T_{i+1,i}l)/C$$

Le facteur Ti,j dans cette équation est l'énergie minimum de transmission pour un nœud i sachant que le nœud récepteur est le nœud j.

Nk est le nombre de fois qu'un paquet a été envoyé sur la route k, en incluant le nœud source. [15]

2.4.4. LMST

Un algorithme de contrôle de topologie MST (Minimum Spanning Tree), appelé LMST (Local Minimum Spanning Tree) pour les réseaux sans fil multi-hop.

Dans cet algorithme, chaque nœud construit son arbre de spanning minimum (MST) local indépendamment.

Nous allons citer quelques directives qui sont essentielles pour un algorithme de contrôle de topologie effectif:

1- La connectivité de réseau doit être préservée avec l'utilisation minimale de l'énergie possible. C'est l'objectif le plus important des algorithmes de contrôle de topologie.
2- L'algorithme doit être distribué. Ce point est dû à l'absence d'une autorité centrale, donc chaque nœud doit prendre une décision, basée sur les informations collectées.

3- Pour être moins sensible à l'impacte de la mobilité, l'algorithme doit dépendre uniquement des informations collectées localement.

L'algorithme LMST est composé de 3 phases:

- **Phase1: La collecte des informations**

Les informations nécessaires dans le processus de construction de topologie sont les informations de tous les nœuds voisins visibles.

Ces informations peuvent être obtenues en envoyant périodiquement des messages "Hello" avec son énergie de transmission maximale. Les informations contenues dans un message "Hello" doivent contenir au moins l'identificateur de nœud et sa position.

- **Phase2: La construction de la topologie**

Après l'obtention de ces informations, chaque nœud applique l'algorithme Prim [6] indépendamment pour obtenir son arbre LMST.

- **Phase3: La détermination de l'énergie de transmission**

On assume que l'énergie de transmission maximale est connue et elle est la même pour tous les nœuds. En mesurant l'énergie de réception des messages "Hello", chaque nœud peut déterminer les niveaux d'énergie spécifiques dont il a besoin pour atteindre chaque voisin. Cette approche peut être appliquée pour n'importe quel modèle de propagation.

Par exemple dans le modèle de propagation de l'espace libre (Free Space) la relation entre l'énergie utilisée pour la transmission des paquets Pt, et l'énergie reçue Pr , est:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi d)^2 L},$$

Où Gt est le gain d'antenne de l'émetteur, Gr est le gain d'antenne du récepteur, λ est la longueur d'onde, d est la distance entre l'antenne de l'émetteur et l'autre du récepteur et L est la perte du système. [13]

2.4.5. Synthèse:

Pour le contrôle d'énergie, nous résumons les techniques appliqués au niveau de la couche réseau comme suit :

	MTPR	**MBCR**	**LMST**	**PARO**
Type de réseau	Adhoc	Adhoc	Adhoc	Adhoc, Sensors, Paw
But	Contrôler l'énergie	Contrôler l'énergie	Contrôler l'énergie	Contrôler l'énergie
Principe	L'utilisation de minimum d'énergie totale de transmission	Allouer des slots prédéfinis pour l'échange des données	Indiquer à sa destination l'éventuelle fin de leur communication	un noeud entre en mode veille quand un voisin est en cours de transmission
Boucle infinie	Non	Non	Oui	Non

Table 2.7. Comparaison entre les protocoles de la couche réseau

2.5. Conclusion

Nous avons exposé une classification des différents protocoles destinés à la gestion d'énergie dans les réseaux ad hoc et de capteurs.

Réseau	**Liaisons de données**	**Physique**
MTPR	SPAN	COMPOW
MBCR	PSM	CLUSTERPOW
PARO	Slotted PSM	IMPROVING CLUSTERPOW

LMST	TA-PSM	TUNNELLED CLUSTERPOW
	SMAC	MINPOW
	TMAC	

Table 2.4. Les protocoles par catégorie

Au niveau de l a couche physique, on opte pour la minimisation de la consommation d'énergie. Au niveau de la couche liaison de données et réseau c'est plutôt le contrôle et la gestion d'énergie qui dominent avec toujours un minimum de consommation d'énergie possible.

Chapitre 3
Le protocole SMAC

Chapitre 3

Le protocole SMAC

3.1. Définition

SMAC, l'abréviation de Sensor-MAC, un protocole de la couche MAC conçu pour les réseaux sans fil de capteurs, proposé par le groupe de projet SCADDS à USC/ISI.

Afin de concevoir un bon protocole de MAC pour les réseaux de capteurs sans fil, Il faut répondre aux problèmes cités ci-dessous :

1. L'efficacité énergétique

Les nœuds capteurs sont censés être équipés d'une batterie. En raison de leur principe de fonctionnement, la recharge ou le remplacement des batteries de chaque nœud est difficile et non économique voire parfois impossible.

Pour résoudre ce problème, on doit identifier les sources de perte d'énergie. Comme vu dans la couche physique, la communication radio consomme la plus grande partie d'énergie. Toutefois, l'utilisation de la radio a une grande relation avec les protocoles MAC. De ce fait, nous avons identifié les sources principales de la perte d'énergie sont :

- L'écoute à vide.
- La collision ou corruption.
- L'overhearing (audition abusive), qui se produit quand un nœud reçoit quelques paquets qui sont destinés à d'autres nœuds.
- L'overhead des paquets de contrôle.

2. Scalabilité et capacité d'auto-configuration

Un réseau sans fil de capteurs, la topologie et la taille peut changer avec le temps. Donc un bon protocole MAC devrait s'adapter à de tels changements.

3. Latence, utilisation de largeur de bande, etc…

Il y a des attributs communs pour la plupart des protocoles de MAC. Prenons la latence comme exemple, son importance dépend des applications réelles. Pour la majorité des applications du réseau des capteurs, la vitesse de changements sur les objets physiques captés par les nœuds de capteurs est beaucoup plus lente que la vitesse du réseau. Donc, la latence est moins importante et pourrait être tolérée dans certains cas comme celui-ci.

A partir de cette définition, nous pouvons déduire que SMAC est différent des autres protocoles sans fil de MAC traditionnels. L'efficacité énergétique et la capacité de l'auto configuration sont les buts fondamentaux.

Nous présentons dans ce qui suit les fonctions et les dispositifs mis en application dans SMAC.

3.2. Présentation des dispositifs mis en application dans SMAC

Le premier dispositif que SMAC présente est le sommeil et l'écoute périodique. L'idée fondamentale est de laisser chaque nœud suivre un programme d'ordonnancement de sommeil et d'écoute périodique, comme le montre la figure 3.1.

Dans la période d'écoute, le noeud se réveille pour communiquer avec les autres noeuds. Quand la période de sommeil arrive, les nœuds essayent de dormir en désactivant leurs radios. De cette façon, le temps passé sur l'écoute peut être sensiblement réduit, ce qui économise, par conséquent, beaucoup d'énergie, particulièrement quand la charge de la circulation est basse. Le coefficient d'utilisation « DutyCylce » est défini comme rapport de la période d'écoute par rapport à un cycle complet (sommeil et écoute). Dans SMAC, le mode Low-duty-cycle est l'opération par défaut pour tous les nœuds.

Figure 3.1. L'écoute et sommeil périodique

Concernant les protocoles MAC à base d'ordonnancement, la synchronisation des nœuds constitut un sérieux problème. Pour arriver à une économie maximum d'énergie, SMAC définit un mécanisme complet de synchronisation, y compris l'envoie des paquets périodiques de synchronisation SYNC, table d'ordonnancement, la découverte périodique des voisins, la mise à jour périodique de la liste des voisins, etc…

L'idée fondamentale est la suivante. Chaque nœud SMAC met l'information de son propre programme d'ordonnancement dans un paquet de synchronisation SYNC et le diffuse à ses voisins périodiquement. Quand un nouveau nœud joint le réseau, il essaie de suivre un programme existant sinon il crée son propre programme. Quand un nœud reçoit différents programmes de ses voisins, il les enregistre dans une table et les suit.

Dans les réseaux à medium partagé, le problème de collision constitue un sujet commun pour tous les protocoles MAC. Les dispositifs que SMAC a adoptés incluent le carrier sense physique et virtuel et la séquence RTS/CTS/DATA/ACK pour le problème de station cachée.

L'Overhearing va devenir une autre source principale de perte d'énergie, surtout quand la densité des nœuds est haute et la charge de trafic de réseau est lourde. SMAC essaie d'éviter l'overhearing en laissant tous les nœuds d'intervention (les voisins immédiats de l'expéditeur et du récepteur) dormir après qu'ils entendent un paquet RTS ou CTS. Nous parlerons de ce problème dans les paragraphes qui suivent.

Pour transmettre des longs messages en respectant la notion d'énergie et de la latence avec efficacité, SMAC supporte le MESSAGE PASSING. Cette fonction fractionne un long message en un certain nombre de petits fragments et les transmet dans un éclat. SMAC utilise seulement une paire RTS/CTS pour un message passing, mais requière un paquet ACK pour chaque fragment. Les paquets RTS/CTS réservent le medium pour transmettre tous les fragments. C'est-à-dire, plus la longueur du message est importante plus le medium va être réservé.

L'opération de Low-duty-cycle réduit la consommation d'énergie mais augmente la latence. Quand un nœud reçoit un paquet de données du noeud précédent, il ne peut pas transmettre à son tour le paquet au nœud suivant au même moment. Il doit attendre l'arrivée de la prochaine période d'écoute du nœud voisin. Donc, il existe un potentiel retard sur chaque saut, quand un paquet de données est transmis sur un réseau multi-sauts. Pour réduire une telle latence, SMAC propose un mécanisme appelé l'écoute adaptative. Le principe fondamental est de donner à tous les noeuds, impliqués dans une transmission, une chance additionnelle pour transmettre leurs paquets au destinataire. Ces nœuds incluent l'expéditeur, le récepteur, et tous leurs voisins immédiats qui participent dans la transmission. De cette façon, un paquet de données peut être retransmis juste après sa dernière transmission.

3.3. Le sommeil et l'écoute périodiques

Dans SMAC, la technique principale utilisée pour réduire la consommation d'énergie est l'utilisation des cycles d'écoute et de sommeil pour chaque nœud. Un cycle complet de période d'écoute et de sommeil s'appelle un frame [26]. Pendant la période de sommeil, le nœud désactivera sa radio. Pendant la période d'écoute, le nœud peut commencer à envoyer ou recevoir des paquets selon le besoin. SMAC fournit un paramètre contrôlable duty cycle, dont la valeur est le rapport de la période d'écoute à la longueur du frame. En fait, la période d'écoute est

normalement fixée selon quelques paramètres de la couche physique et de la couche MAC. L'utilisateur peut ajuster la valeur de duty cycle de 1% à 100% pour contrôler la longueur de la période du sommeil. Normalement, la longueur d'un frame est la même pour tous les nœuds du réseau [26].

La période d'écoute est encore divisée en deux parties. La première appelée la période SYNC, conçue pour les paquets de synchronisation SYNC, qui sont des paquets d'émission et sont utilisés pour résoudre les problèmes de synchronisation entre les nœuds voisins. La seconde appelée la période DATA, conçu pour les paquets de transmission des DONNÉES. Le format d'un frame de SMAC est montré en fig. 3.2.

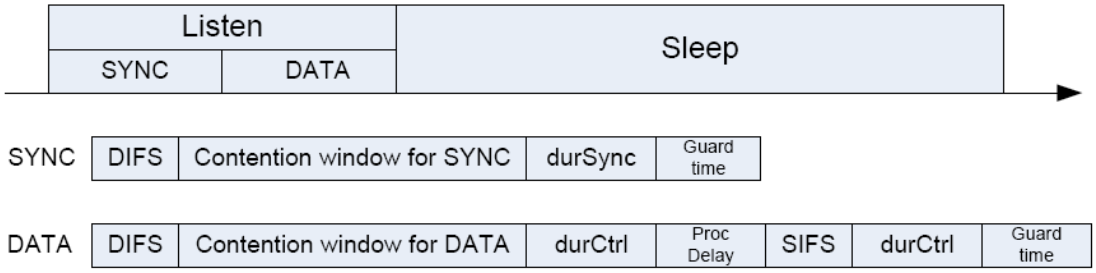


Figure 3.2. Le format d'un frame SMAC

Chaque nœud de SMAC doit avoir au moins un programme d'ordonnancement à suivre. Chaque programme est contrôlé par un timer de programme. Dans la figure 3.3 ci-dessous, nous pouvons voir que chaque frame aura trois points d'expiration, qui s'appelle *Checking Points* (Points de contrôle).

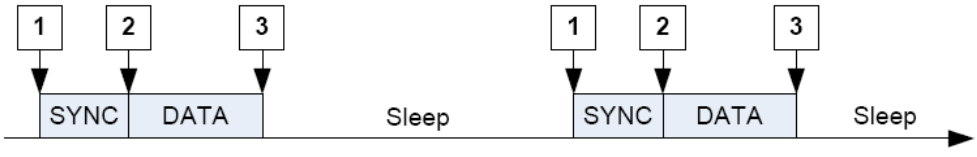


Figure 3.3. Trois points de contrôle dans un frame

À n'importe quel point de contrôle, SMAC décidera quoi faire dans la prochaine période. Par exemple, au point de contrôle 2, SMAC vérifie s'il a un paquet de données dans le buffer à envoyer. Si oui, il commencera le Carrier Sensing. SMAC n'enverra jamais de paquets de données dans ce frame, si l'écoute adaptative n'est pas appliquée ; En d'autres termes, même si SMAC reçoit un nouveau paquet de données de sa couche supérieure au milieu de la période des DONNÉES, il va la mettre dans le buffer et attendra le prochain point de contrôle 2.

Le fait de laisser le nœud suivre un programme de sommeil et d'écoute ne signifie pas que le nœud doit rester en écoute dans la période d'écoute ou doit dormir quand la période programmée de sommeil arrive. Les actions de SMAC à un certain temps ne dépendent pas seulement de la période programmée courante, mais également de quelques autres facteurs. Ces facteurs contiennent l'état courant de MAC, l'état de radio, l'état de canal, l'état des voisins, etc…

Par exemple, au point de contrôle 3, SMAC va dormir en désactivant sa radio, seulement quand toutes les conditions suivantes sont accomplies

- la radio n'est pas en état de recevoir ou d'envoyer
- S'il n'a pas un seul programme dans sa table d'ordonnancement (un nœud de frontière), ce point de contrôle doit appartenir au programme primaire.
- Il est en état « idle », ni en émission ni en réception. Par exemple, il ne peut pas dormir, s'il a envoyé un paquet CTS à son voisin dans la période passée de DONNÉES et il attend maintenant ce paquet de données.
- Il n'est pas dans une période de découverte des voisins.
- Il n'est pas en cours d'exécution d'une écoute adaptative.

Dans cette section, nous avons parlé de l'idée fondamentale du mécanisme de programme d'ordonnancement dans SMAC. Dans les deux prochaines sections, nous aborderons le programme de sommeil et d'écoute ainsi que d'autres dispositifs de SMAC, tels que l'Overhearing Avoidance et l'écoute adaptative.

3.4. Synchronisation

3.4.1. Composants relatifs de synchronisation dans SMAC

Le paquet SYNC

Comme il est sus-mentionné, chaque nœud SMAC doit échanger son programme en annonçant périodiquement un paquet de synchronisation SYNC à ses voisins. La période d'envoi un paquet de synchronisation s'appelle la période de synchronisation. La valeur par défaut dans ns-2 est 10 frames (un frame = un cycle de sommeil et d'écoute). L'envoi ou la réception des paquets SYNC aura lieu dans la période SYNC. La définition de frame de paquet de synchro dans ns-2 est présentée dans la table 3.1 [26].

Champs	**Commentaires**
type	Le drapeau indiquant ceci est un paquet de synchro
longueur	Taille fixe avec 9 bytes
srcAddr	Identification de l'expéditeur
syncNode	Identification du nœud de synchronisation de l'expéditeur
sleepTime	Le prochain sommeil de l'expéditeur dès maintenant
State	Indiquant si l'expéditeur a changé récemment son programme d'ordonnancement
crc	Contrôle par redondance Cyclique

Table 3.1. Le frame de synchronisation SYNC

A partir de cette table, nous pouvons constater que l'information la plus valable dans un paquet SYNC est le sleepTime, qui informe tous les nœuds, qui reçoivent ce paquet, quand la prochaine période de sommeil (de l'expéditeur) arrive. Pour éviter

des erreurs de synchronisation provoquées par l'horloge dérivée sur chaque noeud, le sleepTime utilise une valeur relative plutôt qu'un temps absolu.

La table d'ordonnancement

Chaque nœud SMAC maintient une table d'ordonnancement, qui stocke son propre programme et les programmes de ses voisins connus. L'établissement d'une table d'ordonnancement sera présenté dans le prochain paragraphe. Il y a deux classes de programmes dans la table d'ordonnancement. On appelle le programme primaire le programme que le nœud suit, et les autres programmes sont des programmes secondaires. Un nœud peut ne pas avoir de programme secondaire, par exemple, quand tous ses voisins suivent le même programme. Mais chaque nœud doit avoir un programme primaire. Le nombre maximum d'enregistrements dans la table est limité par un paramètre de SMAC ajustable par l'utilisateur, qui définit le nombre maximum de différents programmes. La table 3.2 montre chaque champ dans la table d'ordonnancement (comme défini dans ns2) [26].

Champ	**Commentaire**
txSync	Drapeau indiquant le besoin d'envoyer SYNC
txData	Drapeau indiquant le besoin d'envoyer des données
numPeriods	Compteur pour envoyer SYNC périodiquement
numNodes	Nombre de nœuds sur ce programme
syncNode	Le nœud qui a initialisé ce programme
chkSched	Drapeau indiquant le besoin de vérifier des numNodes

Table 3.2. Les champs de la table d'ordonnancement

Pourquoi SMAC présente une telle table. On considère intuitivement qu'un seul programme existe dans le réseau entier, parce que les durées de sommeil et l'économie d'énergie seront maximisées quand tous les nœuds seront sur le même

programme. Et soient quelques nœuds voisins pouvants se synchroniser en échangeant le paquet SYNC entre eux. Mais dans la pratique, ce mécanisme n'agit que localement. L'existence des programmes multiples dans un réseau est tout à fait commune, particulièrement dans les grands réseaux multi-sauts où les nœuds ne commencent pas l'initialisation en même temps et se joignent d'une manière aléatoire. On appelle ces nœuds qui partagent le même programme un *cluster virtuel*, et les nœuds avec des voisins dans deux (ou plus) clusters, *les nœuds de frontières*. Évidemment, si le nœud de frontière veut parler à ses voisins durant leur temps d'écoute, il doit savoir quand ses voisins se réveillent et quand ils vont dormir.

Un programme est réellement un temporisateur (timer). Le programme primaire et d'autres programmes secondaires ont leurs propres timers. Ces derniers s'exécutent indépendamment et leur fonctionnement est tout à fait semblable. L'exemple suivant pour illustre comment un nœud de frontière fonctionne sous des programmes multiples.

On suppose que le nœud B a deux voisins le nœud A et C (Figure. 3.4). B a deux programmes dans sa table d'ordonnancement. Le premier est son propre programme (programme primaire), qui est identique à celui de A, et l'autre est pris par C. B assignera un timer d'ordonnancement pour l'un ou l'autre des deux programmes. Les deux timers s'exécutent indépendamment.

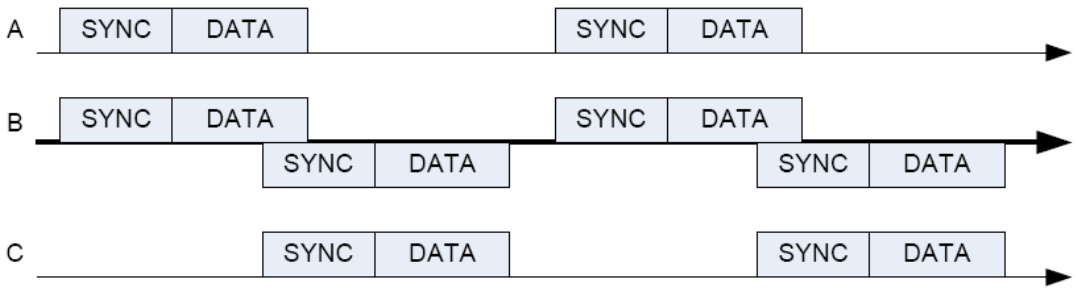


Figure 3.4. Un nœud de frontière avec deux programmes

Pour s'assurer que A et C peuvent recevoir les paquets SYNC dans leur propre période SYNC, B doit diffuser ses paquets SYNC périodiquement sur les deux ordonnanceurs.

On note que, dans un paquet SYNC, un nœud de frontière indique toujours à ses voisins le temps, dès maintenant, de sa prochaine période de sommeil selon son programme primaire, mais pas à n'importe quelle partie de son programme secondaire.

La même méthode est appliquée aux paquets de données. Quand B a un paquet de données à diffuser, il doit le diffuser deux fois. La première sera sur la période de DONNÉES « DATA » du nœud A et la seconde sur la période DATA du C.

Supposons que B reçoit un paquet de données destiné à C de la couche supérieure, il recherche d'abord dans sa table d'ordonnancement pour découvrir le programme, dont C le suit, et il met le drapeau txData sur ce programme à 1. Quand la prochaine période DATA du C arrive, le timer correspondant à B informera B que son Buffer a un paquet de données destiné à C et qu'il est temps de l'envoyer.

A partir de là, on peut constater qu'un nœud de frontière doit suivre plusieurs programmes pour pouvoir se synchroniser avec ses voisins qui sont sur différents programmes. Bien sûr, les nœuds de frontière consommeront beaucoup plus d'énergie que des nœuds normaux (non-frontière).

Liste des voisins

Un autre composant important dans SMAC est : la liste des voisins. Chaque nœud de SMAC doit créer une telle table pour sauvegarder les informations de tous ses voisins connus. Le nombre d'enregistrements dans la liste est aussi limité par un paramètre de SMAC ajustable par l'utilisateur, qui définit le nombre maximum des voisins pour chaque nœud. Comme la table d'ordonnancement, la liste des voisins est établie également en échangeant les paquets SYNC entre les nœuds voisins. La table 3.3. ci-dessous montre la définition de chaque champ dans la liste des voisins.

Champs	Commentaire
nodeId	ID de ce nœud
schedId	ID de programme d'ordonnancement que ce nœud suit
actif	Drapeau indique que ce nœud est actif récemment
état	Drapeau indique que le nœud a changé le programme

Table 3.3. Définition de champ de liste des voisins

La liste des voisins joue un rôle important dans SMAC. Quand SMAC traite une demande de données unicast reçue de la couche supérieure, il va d'abord vérifier sa liste des voisins pour voir si le nœud destinataire est sur la liste. Sinon, la demande sera refusée. Si oui, le txData du programme de destinataire est mis à 1. Puis, quand la période DATA suivante arrive, le nœud essayera d'envoyer ce paquet.

3.4.2. Le choix du premier ordonnanceur

Quand un nouveau nœud joint le réseau, il écoute d'abord pendant une période fixe (normalement une période de synchronisation). On appelle cette période *l'écoute initiale*. Un des cas suivants peut se produire [26]:

a. Aucun paquet SYNC reçu pendant l'écoute initiale.

À la fin de cette période, le nœud crée et choisit son propre programme et lui assigne un timer. Normalement, le premier cycle commence par la période SYNC. Le programme est ajouté au premier enregistrement de sa table d'ordonnancement. Pour annoncer ce nouveau programme, le txSync de drapeau de ce programme est créé, et indique que le nœud essayera de diffuser un paquet SYNC dans la période suivante de synchronisation SYNC.

b. Recevoir un paquet SYNC avant la fin de la période d'écoute initiale.

C'est le premier paquet SYNC reçu. Il suit immédiatement le programme qui vient avec le paquet SYNC. La valeur du sleepTime dans le paquet SYNC détermine le point de départ du premier cycle. Alors, le programme est ajouté au premier enregistrement de la table d'ordonnancement, et l'expéditeur du paquet SYNC est ajouté à la liste des voisins.

3.4.3. La mise à jour et le maintien des ordonnanceurs

Après l'écoute initiale, le nœud peut obtenir son premier programme d'ordonnancement. Comment SMAC maintient sa table et sa liste des voisins chaque fois qu'il reçoit un paquet SYNC d'un de ses voisins après qu'il ait choisi son premier programme. L'algorithme, pour manipuler les paquets SYNC reçus, est écrit comme suit [26]:

a. C'est mon premier paquet SYNC.

Ceci peut se produire seulement après que j'ai choisi mon premier programme par moi-même (option 1). Dans ce cas, je jetterai mon premier programme en l'enlevant de ma table d'ordonnancement et suivrai le nouveau dans le paquet SYNC en l'ajoutant au premier enregistrement de la table d'ordonnancement. Et le timer associé au programme jeté sera reconfiguré selon la valeur du sleepTime dans le paquet SYNC. Naturellement, le nœud qui a envoyé ce paquet SYNC sera ajouté à ma liste des voisins (mon premier voisin).

b. Ce n'est pas mon premier paquet SYNC après que j'aie choisi mon premier programme.

Nous devons considérer les cinq situations possibles suivantes. Pour la simplification, nous utilisons N qui représente l'expéditeur de ces paquets de SYNC et S représente le programme d'ordonnancement dans le paquet SYNC. L'algorithme est mis dans la table 3.4 ci-dessous.

Situation	**Condition**	**Action**
1	N est un voisin connu dans ma liste des voisins et il n'a pas changé son programme primaire depuis que j'ai obtenu son dernier paquet SYNC.	Mettre à jour S dans ma table de programme en remettant son temporisateur de programme avec la valeur du sleepTime dans le paquet SYNC. Ceci peut éliminer la dérive d'horloge entre deux nœuds.
2	N est un voisin connu dans ma liste des voisins, mais il a commuté son programme primaire à S, qui est nouveau à moi, puisque j'ai obtenu son dernier paquet SYNC.	Étape 1: Traiter le programme dont N a quitté. Le nombre de nœuds sur ce programme doit être diminué par un. Si le nombre va à 0 après décrémentation ce programme doit être enlevé de ma liste de programme. Si maintenant je veux avoir une requête d'envoi de données, je dois reporter le changement jusqu'à ce que l'envoi se termine. Etape2: Traiter le nouveau programme S au lequel N a commuté. Si ma table de programme n'est pas pleine, S est ajouté et assigné à un nouveau temporisateur d'ordonnancement. Autrement N doit être supprimé de ma liste des voisins. Etape3: Si le programme dont N a quitté est mon programme primaire, je dois exécuter le check_my_schedule, qui vérifie si je deviens le seul sur mon programme primaire. Si oui, je dois choisir le prochain programme disponible dans ma table de programme et le placer en tant que mon nouveau programme primaire (supprimer l'ancien). Mais si maintenant je m'avère avoir une requête d'envoi de données, je dois reporter le check_my_schedule jusqu'à ce que l'envoi

		courant se termine.
3	N est un voisin connu dans ma liste des voisins, mais il a commuté son programme primaire à S, existant dans ma table de programme, puisque j'ai obtenu son dernier paquet de synchro.	Étape 1 : Traiter le programme dont N a commuté. Le nombre de nœuds sur le programme que N utilisait doit être décrémenté par 1. Si le nombre va à 0 après décrémentation, ce programme doit être enlevé de ma liste de programme. Si maintenant je m'avère avoir une requête d'envoi des DONNÉES, je dois reporter de changer jusqu'à ce que l'envoi se termine. Etape2 : Traiter le programme S au lequel N a commuté. Mettre à jour S dans ma table de programme en remettant son temporisateur avec la valeur du sleepTime dans le paquet SYNC. Et le nombre de nœuds sur S est augmenté par 1.
4	N est un voisin inconnu à moi et S est également nouveau à moi.	Si, ni ma table de programme ni ma liste de voisins n'est pleine, ajouter S à ma table de programme et assigner un nouveau temporisateur à S. Puis ajoute N à ma liste des voisins.
5	N est un voisin inconnu à moi, mais S est connu dans ma table de programme.	Mettre à jour S dans ma table de programme en remettant son temporisateur à la valeur du sleepTime dans le paquet SYNC. Si ma liste des voisins n'est pas pleine, N est ajouté. Et le nombre de nœuds sur S est augmenté par 1.

Table 3.4. Algorithme général pour traiter les paquets SYNC

3.4.4. La découverte périodique des voisins

Dans SMAC, les nœuds voisins se découvrent en échangeant des paquets SYNC. Parfois, deux nœuds voisins peuvent ne pas se voir pour toujours, par exemple, quand ils suivent des programmes différents, dont les périodes de SYNC ne se recouvrent pas. La découverte périodique des voisins peut empêcher ceci de se produire.

L'idée fondamentale de la découverte des voisins dans SMAC est que chaque nœud exécute périodiquement la découverte des voisins pendant tout un temps de synchronisation. Pendant le temps de la découverte des voisins, SMAC n'ira jamais dormir quand sa période de sommeil intervient, de sorte que le nœud puisse écouter plus de temps qu'habituellement et ait plus de chances de trouver un nouveau voisin. Pour un nœud de frontière, la découverte des voisins est seulement exécutée sur son programme primaire. La raison est que sur le programme secondaire, le nœud n'ira jamais dormir quand la période programmée de sommeil arrive.

La figure 3.5 montre comment la période de découverte des voisins est définie et son rapport avec d'autres périodes. La période de découverte des voisins peut changer, selon le nombre courant de ses voisins connus. Dans ns-2, la période de la découverte des voisins est égale à 2 périodes SYNC si le nœud n'a aucun voisin. Autrement, la période est beaucoup plus longue et peut atteindre 33 périodes SYNC.

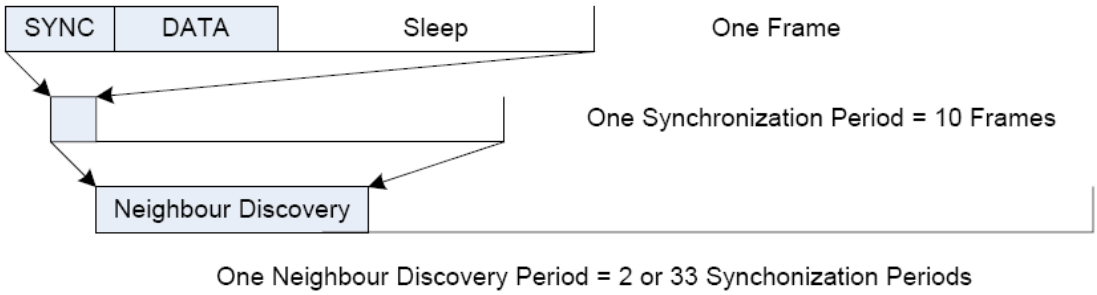


Figure 3.5. L'hiérarchie des périodes dans SMAC

3.4.5. Mise à jour périodique de la liste des voisins

La liste des voisins et la table d'ordonnancement sera mise à jour chaque fois que le nœud reçoit un paquet SYNC d'un de ses voisins. Chaque nœud de SMAC doit

vérifier sa liste des voisins périodiquement, pour voir si des voisins qui ont été inactifs pendant un certain temps (écarté ou mort pour quelque raison), doivent être enlevés de la liste des voisins. Cette tâche est nécessaire et importante, parce qu'il ne peut pas uniquement vider de l'espace pour la liste des voisins, mais éviter également la consommation d'énergie inutile provoquée en essayant de parler à un voisin inactif, particulièrement dans un réseau de capteurs multi-sauts.

Un timer est assigné pour commander la mise à jour périodique de la liste des voisins. Sa période d'expiration doit être plus grande que la période de synchronisation SYNC (La raison sera expliquée plus tard). Quand le timer expire, les étapes suivantes seront exécutées.

Étape 1 : Vérifier si j'ai une demande d'envoi de DONNÉES maintenant. Si oui, sauter les étapes suivantes et reporter la mise à jour jusqu'à la fin de l'envoi courant. Sinon, dès maintenant, je serai temporairement désactivé pour recevoir de nouvelles demandes de la couche supérieure pour éviter les erreurs pendant la mise à jour. (Dans ns-2, ceci est fait en mettant le drapeau txRequest à 1, qui empêche SMAC de recevoir une nouvelle demande).

Étape 2 : Mettre à jour le nombre de nœuds sur un certain programme d'ordonnancement si son drapeau chkSched est activé (mis à 1). Ceci peut se produire lors de l'obtention d'un paquet SYNC d'un de mes voisins connus, qui avaient commuté à un autre programme depuis que j'ai obtenu son dernier paquet de synchronisation, et je devrais diminuer le nombre de nœuds sur le programme que ce voisin avait commuté. Mais à ce moment-là j'ai eu une demande d'envoi de DONNÉES, donc j'ai dû reporter la diminution.

Étape 3 : Mise à jour de la la liste des voisins. Vérifier chaque enregistrement dans ma liste des voisins pour voir si son drapeau *actif* est mis à 1 ou non.

- Si oui, le drapeau est remis à zéro (Le drapeau sera mis à 1

probablement quand je recevrais un paquet SYNC de ce voisin, avant que ma prochaine mise à jour de la liste des voisins arrive).

- Sinon, cela signifie que je n'ai pas reçu pendant longtemps un paquet SYNC de ce nœud voisin (au moins une période de mise à jour d'une liste des voisins), ce noeud voisin doit être supprimé de ma liste des voisins. Par conséquent, le nombre de nœuds sur le programme que le voisin supprimé utilise devrait être diminué. Si après la diminution le nombre devient 0, ce programme doit être enlevé de ma table de programme.

Étape 4 : Si les nœuds voisins inactifs, qui ont été supprimé dans l'étape 3, sont sur mon programme primaire, je dois exécuter le check_my_schedule.

Étape 5 : Remettre à zéro le timer pour programmer la prochaine mise à jour de la liste des voisins et pour permettre de recevoir encore de nouvelles demandes de la couche supérieure.

Maintenant, on explique la raison pour laquelle la période de la mise à jour de la liste des voisins doit être plus longue que la période de synchronisation [26].

La figure 3.6 montre un exemple de deux nœuds voisins A et B. On suppose que la période pour mettre à jour la liste des voisins est plus petite que la période de synchronisation SYNC, et A peut toujours recevoir des paquets de SYNC de B avec succès. On peut voir de la figure que A exécute la mise à jour de la liste des voisins deux fois dans une seule période de synchronisation de nœud B. Il est facile d'imaginer cela, à la deuxième mise à jour de A (flèche rouge), A pense que B n'a pas été activé récemment et enlève B de sa liste des voisins, parce que A n'a pas reçu un paquet SYNC de B depuis la dernière mise à jour de la liste des voisins. B deviendra inconnu à A jusqu'à ce que la période d'annonce prochaine du B vienne. Dans cette période blanche, A ne peut pas parler à B, parce que pour lui B n'est pas en activité.

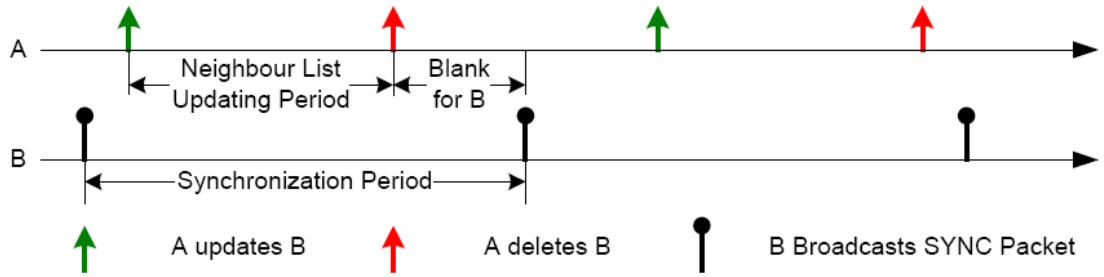


Figure 3.6. La période de mise à jour de la liste des voisins plus petite que la période de synchronisation

3.5. CSMA/CA dans SMAC

Le protocole SMAC procure principalement l'économie d'énergie en utilisant le mécanisme d'ordonnanceur. L'ordonnanceur contrôle le sommeil et l'écoute périodiques des nœuds. De ce point de vue, SMAC ressemble à un protocole de la couche MAC, par exemple TDMA. Cependant, en ce qui concerne le contrôle de la transmission et l'action d'éviter les collisions, SMAC est semblable au DCF (Distributed Coordination Function) dans les protocoles IEEE 802.11. Dans cette section, nous présenterons le mécanisme de CSMA/CA dans SMAC, y compris Carrier sense (ainsi que les mécanismes physiques et virtuels) et le mécanisme de RTS/CTS pour éviter les collisions et le problème de station cachée.

3.5.1. Carrier Sense :

Comme dans 802.11, Carrier Sense dans SMAC est exécuté par des mécanismes physiques et virtuels. Le medium est déterminé en tant que libre seulement quand les deux (physique et virtuel) indiquent qu'il est libre.

Le Carrier sense physique est exécuté à la couche physique en vérifiant l'état actuel de la radio. Chaque fois que la radio commence à recevoir ou transmettre, la couche PHY informera le MAC. Elle se produit également quand la réception ou la transmission est terminée.

Évidemment, le medium sera déterminé en tant qu'occupé quand la radio est dans l'état de réception ou de transmission.

Le Carrier Sense virtuel est exécuté à la couche MAC. Nous savons que dans 802.11, chaque station maintient un NAV (Network Allocation Vector, Vecteur d'allocation de réseau), qui est réellement un compteur. Il comptera de zéro à un taux uniforme après qu'il soit assigné avec une certaine valeur. Le mécanisme de NAV demande que chaque paquet envoyé par MAC contienne un champ de durée, indiquant la durée de cette transmission. Quand un nœud reçoit un paquet destiné à un autre nœud, la valeur de durée dans le paquet indiquera au nœud combien de temps il doit rester silencieux. Le NAV du nœud sera mis à jour avec la valeur de durée, seulement si la valeur de durée est plus grande que la valeur actuelle de NAV. Le Carrier sense virtuel indique que le medium est libre, quand la valeur de NAV est zéro. Si la valeur est différente de zéro, c'est-à-dire qu'il est occupé.

Le mécanisme de carrier sense virtuel est appliqué également au SMAC. Cependant, différemment à celui de 802.11, SMAC définit deux NAVs. Le premier est simplement appelé NAV et utilisé pour indiquer si le medium est occupé ou non, comme 802.11. L'autre s'appelle Neighbor NAV (NAV Voisin), qui prend le NAV de ses voisins. Seulement quand les deux NAVs arrivent à zéro, le carrier sense virtuel indique que le medium est libre. La structure et la manière de travailler des deux NAVs sont tout à fait semblables. La table 3.5 cite tous les événements qui permettent la mise à jour des valeurs des deux NAVs. Sinon, la valeur de durée utilisée pour mettre à jour la valeur de NAV est toujours la valeur de durée dans le paquet qui est reçu ou envoyé dans l'événement correspondant.

NAV	**NAV Voisin**
Recevoir RTS destiné à un autre nœud	Recevoir RTS destiné à moi
Recevoir CTS destiné à un autre nœud	Après l'envoi de CTS (pour timeout de DATA)
Recevoir DATA destinées à un autre nœud	Recevoir DATA destinée à moi

Recevoir ACK destiné à un autre nœud	Après que DATA (unicast) est envoyée
Erreur trouvée dans le paquet reçu, NAV est mise à jour par une valeur d'EIFS	Après qu'ACK est envoyé
La collision détectée quand la radio reçoit un paquet et un autre paquet arrive, NAV sera mise à jour par une valeur d'EIFS juste après que la radio finisse la réception	Timeout de ACK, mais nombre maximum de fois non atteint pour prolonger le temps de Tx

Table 3.5. Comparaison entre NAV et NAV Voisin [26]

A partir de la table précédente, nous pouvons déduire qu'un nœud utilise son NAV pour Carrier sense virtuel quand il interfère dans une des transmissions de son voisin, et utilise le NAV de son voisin quand il est expéditeur ou récepteur. En dehors des fonctions de carrier sense virtuel, les NAVs jouent également d'autres rôles dans SMAC cités ci-dessous.

1. Le voisin NAV agira en tant qu'un timer pour l'arrêt de DONNÉES (DATA) sur le nœud, qui a envoyé le paquet de CTS et attend l'arrivée du paquet de DONNÉES.

2. L'écoute adaptative sera déclenchée quand l'un des deux NAV devient zéro.

Nous constatons qu'il y a deux fenêtres de contention définies dans un frame. L'une est pour envoyer des paquets SYNC dans la période synchronisation SYNC et l'autre pour envoyer des paquets DATA dans la période de données DATA. Les deux fenêtres de contention ont une taille fixe (nombre fixe de slots), qui doit être 2n -1, par exemple 31 slots pour SYNC et 63 pour DATA. En fait, SMAC peut avoir une troisième contention window dans la période de sommeil, si l'écoute adaptative est appliquée. Ici nous considérons SMAC sans écoute adaptative. Maintenant, prenons un exemple simple et voyons comment le Carrier sense est lancé avant d'envoyer un paquet de synchronisation dans la période SYNC. Les étapes suivantes

sont également applicables à envoyer un paquet de DONNÉES dans la période DATA.

Quand le timer du programme d'ordonnancement expire à la fin de la période de sommeil indiquant l'arrivée d'une nouvelle période SYNC, il examinera les conditions pour envoyer un paquet SYNC. Si le syncFlag sur ce programme est mis à 1 tous les deux NAVs ont la valeur zéro (le Carrier sense virtuel indique que le medium est libre), le noeud sera prêt à envoyer un paquet de synchronisation SYNC. Si la radio est dans l'état de sommeil, elle sera d'abord réveillée. Alors le nœud commence le carrier sense physique. La durée du carrier sense est choisie uniformément dans la « contention window » et se compose d'un DIFS. Une des deux possibilités suivantes peut se produire pendant cette période :

1. Si rien n'est entendu tout au long de toute la période de carrier sense, le nœud supposera que le medium est libre et commence à transmettre le paquet tout de suite.
2. Une fois que le medium est sensé occupé pendant la période de carrier sense, le nœud cessera de sentir (détecter) et reportera l'envoie du paquet. Quand la même période arrive dans le prochain frame, le nœud réessayera l'envoi. Tout en réessayant, il suivra les mêmes procédures décrites ci-dessus.

3.5.2. Eviter les collisions (Collision Avoidance)

Nous savons que le mécanisme RTS/CTS défini dans le DCF peut efficacement réduire les durées des collisions et résoudre le problème prétendu de station cachée. Cette technique de réservation du medium a été présentée dans SMAC. Avant le paquet DATA actuel, l'expéditeur devrait échanger des paquets de RTS et de CTS avec le récepteur. Mais seulement les paquets unicast suivent l'ordre de RTS/CTS/DATA/ACK. Les paquets broadcast seront envoyés sans utiliser RTS/CTS. Dans cette sous-section, nous discutons uniquement le paquet unicast.

Nous avons appris que le mécanisme de carrier sense virtuel est réalisé en distribuant les informations de réservation annonçant l'utilisation imminente du medium. Et

l'information de réservation est enregistrée dans le champ de durée des paquets RTS/CTS/DATA/ACK. Les paquets RTS/CTS réserveront pour le processus entier de transmission. Tous les voisins immédiats de l'expéditeur et du récepteur apprendront la prochaine transmission des paquets RTS ou de CTS et maintiendront silencieux pendant la période réservée. Bien que la combinaison du carrier sense et des mécanismes RTS/CTS puisse réduire largement la probabilité et les durées des collisions, les collisions ne peuvent pas être complètement évitées. Si deux nœuds voisins finissent carrier sense et commencent l'envoi au même temps, la collision va se produire. L'exemple suivant montre comment SMAC envoie un paquet de données et suppose que le canal est idéal (aucun cas de corruption).

On considère deux nœuds voisins A et B, qui suivent le même programme d'ordonnancement et se sont découverts. A reçoit une demande d'envoi destinée à B de la couche supérieure. On suppose également que A n'a aucune autre demande d'envoi actuellement. Le carrier sense ne commence pas immédiatement et doit attendre jusqu'à ce que la période suivante de DONNÉES vienne. A va suivre les étapes décrites dans la dernière sous-section pour commencer le carrier sense. La figure 3.7 montre un cas possible. On peut constater à partir de la figure que les nœuds A et B échangent RTS/CTS pendant la période DATA et utilisent leur temps programmé de sommeil pour la transmission de paquet de DONNEES. En fait, la durée de la période de DONNÉES est soigneusement désignée. Elle est fixée selon quelques paramètres de la couche physique et la couche MAC, par exemple la largeur de bande de radio et la taille de « contention window ». L'échange de RTS et de CTS peut toujours être fait au cours de la période de DONNÉES si aucune collision et corruption ne se produisent. Mais la transmission du paquet de DONNÉES doit normalement être prolongée à la période de sommeil de programme. Normalement, SMAC fonctionne sous un coefficient Low Duty Cycle (par exemple 10%). La période de sommeil dans un frame est beaucoup plus longue que la période de DONNÉES. Ainsi dans des cas normaux, le processus entier de transmission peut être fait dans un frame et ne sera pas prolongé à la période suivante de synchronisation SYNC. Dans l'exemple suivant, la transmission entre A et B finit avant la

période SYNC suivante. Si l'écoute adaptative n'est pas appliquée, A et B auront toujours le temps pour dormir.

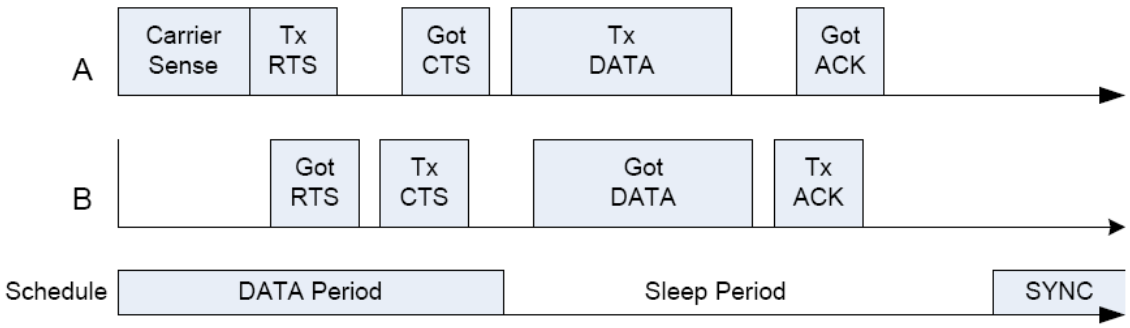


Figure 3.7. Une transmission de donnée entre deux nœuds

On peut voir ce qui va se passer lorsque la collision se produit. Si B envoie également RTS presque au même moment où A a envoyé le RTS, deux paquets de RTS se rencontrent et une collision se passe. Ni l'un ni l'autre ne recevront le paquet de RTS de l'autre ni renverront un paquet CTS. Après un petit moment, le timer du CTS de A expire. Si maintenant le programme primaire de A est dans la période de sommeil, A ira dormir en désactivant sa radio. Quand la période suivante de DONNÉES sur le même programme vient, A renverra le RTS. Le nombre maximum des nouvelles tentatives de RTS pour envoyer un paquet de DONNÉES est ajustable par l'utilisateur dans SMAC. Quand les nouvelles tentatives atteignent leur nombre maximum, A cessera d'envoyer le paquet de DONNÉES et signalera sa couche supérieure de l'échec de l'envoi. Les mêmes procédures seront suivies quand le timer de CTS de B expire.

3.6. Overhearing Avoidance

Pour des protocoles contention-based comme IEEE-802.11, Overhearing est l'une des sources principales de perte d'énergie. L'Overhearing se passe sur un nœud quand il reçoit des paquets destinés à d'autres nœuds. Dans 802.11, des mesures comme l'utilisation de latence et de largeur de bande sont considérées en premier

lieu. Pour arriver à une meilleure performance dans un réseau de medium-partagé, carrier sense, particulièrement le carrier sense virtuel devrait être exécuté plus efficacement. La meilleure manière de faire est de laisser chaque nœud écouter les transmissions de tous ses voisins. Évidemment, cette méthode mènera à de grandes quantités de consommations d'énergie sur overhearing, particulièrement quand la densité de nœud est haute et la charge de trafic dans le réseau est lourde.

Pour SMAC, la minimisation de consommation d'énergie est son but primaire. Pour éviter l'overhearing, SMAC force des nœuds interférés pour aller dormir après qu'ils reçoivent un RTS ou CTS qui ne sont pas destiné à eux. De cette façon, les nœuds qui interfèrent les transmissions de leurs voisins n'entendront pas les paquets de DONNÉES, qui prennent beaucoup plus de temps de transmission que de contrôler les paquets, et suivre des paquets d'ACK. Nous prenons un exemple dans la figure 3.8 pour illustrer cet algorithme.

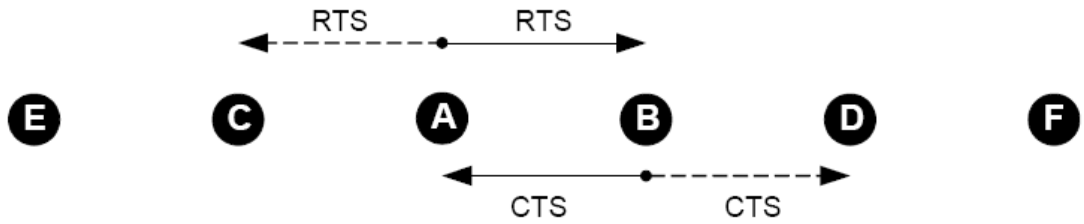


Figure 3.8. C et D entend la transmission entre A et B

La figure ci-dessus montre un réseau linéaire à cinq-saut. Chaque nœud peut seulement entendre ses voisins directs. On suppose que tous les nœuds partagent le même programme d'ordonnancement. Supposant que A communique avec B de la même manière que dans figure 3.7.

On sait que la collision se produit seulement du côté de récepteur. Évidemment, D est censé aller dormir, parce que sa transmission interfère la réception du paquet de DONNÉES du B. C est à deux sauts loin de B, alors la transmission du C n'interférera pas avec la réception du B. Mais si C parle à E en même temps que A envoie des données à B, C ne recevra aucun paquet de E parce que la collision se produit sur la transmission de C.

La transmission de C est un gaspillage d'énergie et lui également a besoins d'aller dormir. E et F sont au moins à deux sauts loin des nœuds qui transmettent, et ils ne produiront jamais une interférence. Par conséquent, E et F n'ont aucun besoin de dormir. Tout simplement, ces nœuds qui sont les voisins directs de l'expéditeur ou le récepteur devraient aller dormir.

Maintenant nous allons voir quand C et D devront aller dormir. C qui est le voisin direct de A peut entendre le paquet RTS que A envoie à B. Cependant, C ne va pas dormir après réception du RTS, parce qu'à ce moment là, la communication n'est pas vraiment réalisée et peut être annulée pour n'importe quelle raison, par exemple une collision sur les paquets RTS. Par conséquent, il doit continuer à écouter pendant une période CTS pour voir s'il peut entendre le paquet suivant de CTS. Si C reçoit le paquet CTS, il va aller dormir tout de suite. Cependant, dans cet exemple, C ne peut pas entendre le CTS que B envoie de nouveau à A. Pour cela, après la période CTS, C continuera à écouter une autre période CTS et essayera d'entendre le paquet de DONNÉES. Dès que C entend le paquet de DONNÉES, il sait, du champ de durée dans le paquet de DONNÉES, combien de temps la transmission actuelle va durer et va dormir tout de suite. Si aucun paquet de DONNÉES n'est entendu dans cette période, C va dormir seulement quand son programme primaire est dans la période de sommeil. Pour D, qui est le voisin direct du B, il peut uniquement entendre les paquets CTS envoyés par B. D ira dormir juste après qu'il obtienne le CTS et met à jour son NAV avec la valeur de durée dans le CTS et va dormir bien après la réception du CTS.

3.7. Message Passing

Si Overhearing mène à la perte d'énergie dans des protocoles comme IEEE 802.11, Overhead de paquet de contrôle est une autre raison. Overhead de contrôle signifie le coût (énergie ou temps) dépensé en échangeant les paquets de contrôle (RTS/CTS/ACK) lorsqu'un nœud envoie un paquet de données. Dans cette partie,

nous étudierons comment SMAC réduit les consommations d'énergie provoquées par l'Overhead de contrôle.

On sait que la transmission d'un long message en utilisant un seul paquet de données par un canal qui perd les données est risquée. Même lorsque quelques bits dans le paquet sont corrompus pendant la transmission, le paquet entier doit être retransmis. Ceci gaspillera beaucoup de temps et d'énergie. Par conséquent, certains protocoles MAC comme 802.11 supportent un mécanisme de fragmentation, qui fractionne un long message en quelques petits fragments. Tous les fragments sont envoyés dans un éclat en utilisant une paire RTS/CTS, si aucun d'eux n'est corrompu. Si un des fragments est corrompu, une autre paire de RTS/CTS est nécessaire. Quand le récepteur reçoit tous les fragments, son MAC prend en charge l'assemblage de tous les fragments dans un seul et de les faire passer vers le haut.

SMAC adopte un mécanisme de fragmentation modifié pour transmettre un long message, appelé Message Passing. Son idée fondamentale est de fragmenter un long message en plusieurs petits fragments et de les envoyer en fragments dans un éclat. Ceci signifie que seulement une paire RTS/CTS est utilisée pour tous les fragments, mais le récepteur devrait reconnaître chaque fragment qu'il reçoit. RTS et CTS réservent le medium pour transmettre tous les fragments. Quand un des fragments est corrompu pendant la transmission, ACK timeout se produira du côté de l'expéditeur. Ce dernier doit prolonger la durée réservée de transmission pour une paire DATA/ACK et renvoyer le fragment perdu immédiatement. SMAC fixe le nombre limité de prolongements qui peuvent être faits pour un message. Ceci empêche quelques nœuds d'occuper le medium pour longtemps dans quelques cas spéciaux, par exemple lorsque le récepteur perd sa connexion avec l'expéditeur pendant la transmission.

Quant aux nœuds voisins qui peuvent interférer la transmission, ils suivront la même méthode décrite, pour aller dormir après qu'ils entendent le RTS ou le CTS.

Cependant, quelques cas spéciaux peuvent se produire. Par exemple, les nœuds voisins perdent la chance de recevoir le RTS ou le CTS pour quelques raisons que lorsqu'ils dorment à ce moment-là, et ils se réveillent après. Autre cas, quand un nouveau nœud joint le réseau au milieu de la transmission. Alors que doivent faire les nœuds dans ces cas là ?

En fait dans SMAC, le fragment de DONNÉES et les paquets d'ACK ont un champ de durée, qui est maintenant le temps nécessaire pour transmettre tous les fragments de données restants et les paquets d'ACK.

La figure 3.9 Illustre comment placer la valeur de durée (valeur de NAV) pour chaque paquet dans la transmission d'un message de 3-fragment dans un éclat. Donc il n'y a pas de problème pour connaitre quel paquet est reçu par avance par les nœuds voisins interférés, ils sauront la durée de la transmission courante et le temps nécessaire pour dormir. C'est également la raison pour laquelle SMAC invite le récepteur à reconnaître chaque fragment de DONNÉES qu'il reçoit. L'utilisation de la paire Fragment/ACK dans Message Passing ressemble à la paire de RTS/CTS dans la manière de résoudre le problème de la station cachée. L'envoi fréquent du ACK fréquemment va laisser les nœuds voisins, qui peuvent seulement entendre le récepteur, mettre à jour leur NAVs à temps. Par exemple, dans la figure. 3.8, D dort après qu'il ait reçu le CTS de B. Cependant, un seul fragment est corrompu pendant la transmission et le temps original réservé pour la transmission entière doit être prolongé. Quand le temps original réservé est fini, D peut se réveiller du sommeil. Il saura le temps prolongé de transmission du paquet d'ACK et ira dormir encore.

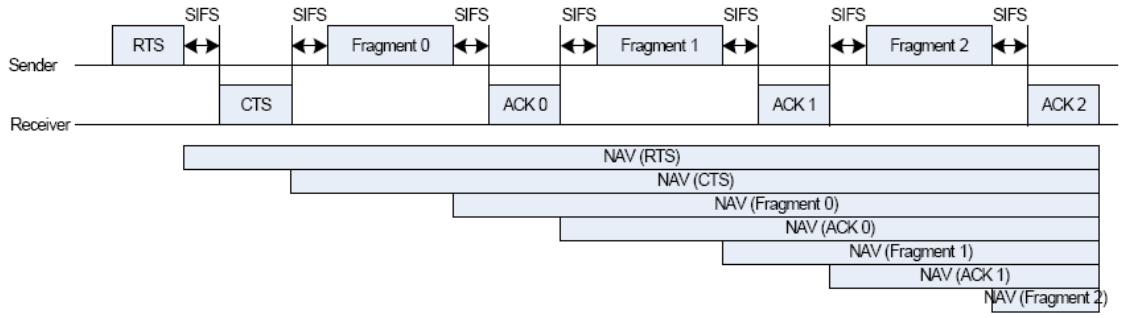


Figure 3.9. RTS/CTS/DATA/ACK et les configurations de NAV dans Message Passing

De ce fait , On constate que le mécanisme de message Passing dans SMAC peut efficacement réduire l'overhead de contrôle pour transmettre un long message, en envoyant tous les fragments dans un éclat et en utilisant seulement une paire de RTS/CTS. Quand l'expéditeur ne reçoit pas un ACK pour n'importe quel fragment, le Message passing présentera quelques moyens pour retransmettre le fragment perdu ou corrompu. Cette approche semble être injuste à ces nœuds qui ont un message court à envoyer. Cependant, de cette façon, la performance du niveau d'application pour chaque nœud peut être considérablement améliorée, si on le compare avec les protocoles MAC traditionnels.

3.8. L'Écoute adaptative

Dans les parties précédentes, nous avons principalement discuté de la façon dont SMAC réduit les consommations d'énergie. D'autre part, le mécanisme d'ordonnancement augmentera la latence pour envoyer des paquets dans un réseau de multi-saut. Bien que le mécanisme de synchronisation et le message passing aient diminué la latence à un certain degré, l'écoute adaptative est la technique dominante que SMAC propose pour réduire efficacement et largement la latence dans une transmission multi-sauts.

Examinons un exemple pour voir comment SMAC transmet un paquet dans un réseau linéaire de trois-sauts. Supposant qu'il y a quatre nœuds A, B, C, et D, qui sont mis dans une ligne (Figure 3.10). Chaque nœud peut entendre seulement ses voisins directs. A est la source et D est le destinataire. Maintenant un paquet de données est produit au nœud source et destiné au nœud destinataire. Nous supposons que tous les nœuds partagent le même programme et aucune technique de Message passing n'est appliquée.

Figure 3.10. Un réseau à 3 sauts avec une source et une destination

On Commence d'abord par le cas illustré par la figure 3.11. Quand l'écoute adaptative n'est pas utilisée, chaque nœud a au plus une chance d'envoyer le paquet de données dans chaque frame, car la vérification pour envoyer le paquet de données a lieu seulement au début de chaque période de DONNÉES. Si l'arrivée du paquet de données à un certain nœud vient juste après ce point de contrôle de ce frame, elle doit attendre jusqu'à ce que la période suivante de DONNÉES vienne. C'est-à-dire que quand chaque nœud suit strictement son programme de sommeil, il y a un potentiel retardé sur chaque saut. A partir de la figure 3.11, on constate qu'on a besoin de trois frames pour transmettre le paquet de données de la source A à D. Le paquet de données fait juste un seul saut dans chaque frame.

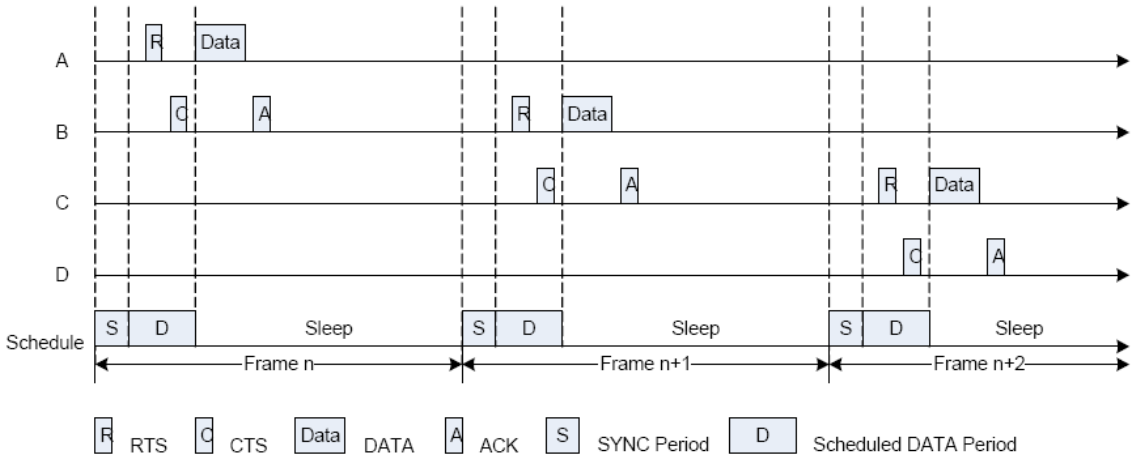


Figure 3.11. Transmission d'un paquet de données à travers un réseau à 3 sauts sans l'écoute adaptative

Maintenant on considère le second cas où l'écoute adaptative est appliquée. L'idée fondamentale est qu'à la fin d'une transmission, SMAC donnera à ces nœuds (qui sont impliqués par cette transmission comprenant l'expéditeur, le récepteur, et les voisins de tous les deux qui ont participé dans cette transmission) une autre chance pour transmettre les paquets de données. De cette façon, un nœud peut avoir deux périodes de DONNÉES pour envoyer ou recevoir des paquets de données dans un seul frame. La figure ci-dessous illustre comment l'écoute adaptative fonctionne dans SMAC.

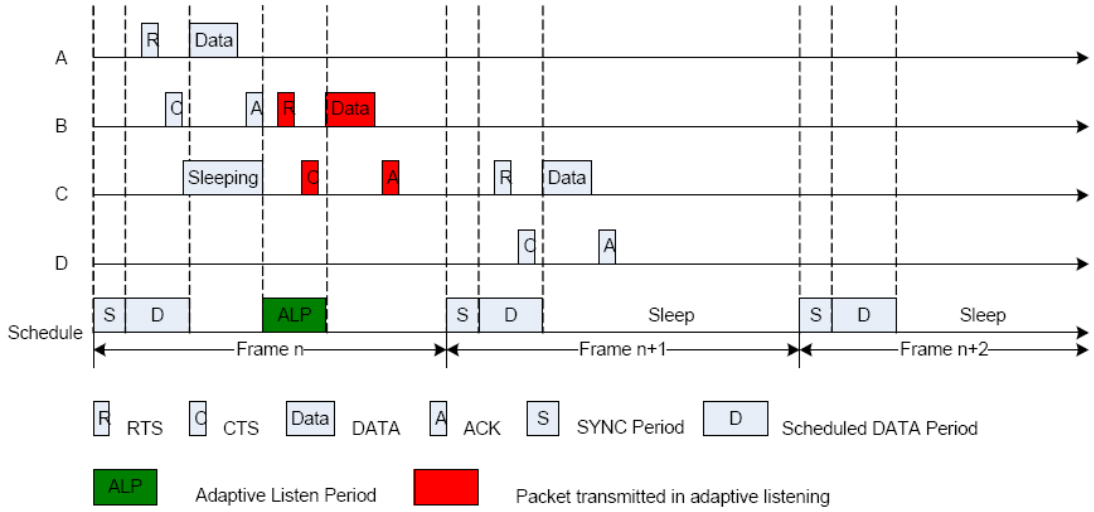


Figure 3.12. La transmission d'un paquet de données à travers un réseau à 3 sauts, avec l'écoute adaptative

On commence par la transmission entre A et B. Comme montré dans la figure 3.12, l'envoi d'un paquet de données de A à B (y compris carrier sense et l'échange de RTS/CTS) commence à partir de la période programmée de DONNÉES et se prolonge à la période suivante de sommeil. L'expéditeur A et le récepteur B utilisent les NAVs de leurs voisins pour exécuter le carrier sense virtuel, alors que C entend cette transmission et utilise son NAV pour carrier sense virtuel. NAV et NAV voisin contiennent la même valeur, qui est le temps réservé pour la transmission en cours. L'expiration de ces NAVs (allant à zéro, indiquant la fin de la transmission courante)

déclenchera l'exécution de l'écoute adaptative sur leurs nœuds. Les étapes universelles suivantes seront exécutées quand l'écoute adaptative est déclenchée sur un certain nœud.

1). Vérifier si le temps restant dans le frame courant (dès maintenant jusqu'à le prochain temps d'écoute) est plus court qu'une période de DONNÉES. Si oui, cesser d'exécuter les étapes suivantes et sortir. Sinon, aller à l'étape 2.

La raison d'exécuter cette étape sera expliquée plus tard dans cette partie et on suppose que l'exemple donné dans la figure 3.12 satisfait cette condition.

2). Créer un Timer (appelé le temporisateur d'écoute adaptative) pour m'apporter de nouveau au sommeil (si les conditions me permettent de dormir à ce moment-là). Le temps d'expiration du temporisateur est égal à une période de DONNÉES (ici appelée période d'écoute adaptative, voir la boîte verte dans figure 3.12).

3). Si je suis toujours endormi, je dois me réveiller maintenant. Ceci peut arriver au nœud, tel que C dans la figure 3.12, qui dort parce qu'il a entendu la transmission précédente de son voisin.

4). Vérifier si le drapeau txData sur mon programme primaire est placé (pour voir si j'ai un paquet de données unicast dans le buffer à envoyer dans mon programme primaire). Si oui, j'essayerai d'envoyer le paquet de données en suivant la même manière que je fais dans la période de DONNÉES de programme. Autrement je dois rester éveillé, parce que mes voisins, qui sont également en écoute adaptative maintenant, peuvent vouloir me parler.

On note que le paquet de données de diffusion ne sera jamais envoyé quand je suis en écoute adaptative, parce que je ne suis pas sûr si tous mes voisins sont également en écoute adaptative maintenant.

Pour l'exemple montré dans la figure 3.12, seulement B satisfait la condition d'envoi, parce qu'il vient de recevoir le paquet de données de A qui doit être retransmis à C. Après l'échange de RTS/CTS avec C, B transmet le paquet de données à C, comme montré dans la boîte rouge dans la figure 3.12. Pour A, il n'a rien à envoyer et doit rester éveillé. Mais il va entendre le RTS que B envoie à C. Alors, A doit dormir pour éviter d'entendre la transmission entre B et C.

5). Le timer d'écoute adaptative créé à l'étape 2 expire. Si je suis encore éveillé seulement parce que je n'ai rien à envoyer et entendu pendant cette période d'écoute adaptative, maintenant j'irai de nouveau au sommeil.

Pour l'exemple de la figure 3.12, aucun nœud ne va dormir pour les raisons indiquées ci-dessus. B et C sont impossibles, parce que B transmet au C. et A a entendu cette transmission et a pu dormir avant que son temporisateur d'écoute adaptative expire.

Pourquoi SMAC vérifie le temps restant dans le frame courant avant de démarrer l'écoute adaptative. A partir de ce qui a été présenté ci-dessus, on constate que l'essentiel de l'écoute adaptative est d'ajouter une autre période de DONNÉES (période d'écoute adaptative) dans la période programmée de sommeil, de telle sorte que les nœuds puissent obtenir une chance additionnelle pour envoyer ou recevoir des paquets de données. Aussi, afin d'éviter que la période d'écoute adaptative chevauche la période suivante de synchronisation, car ceci peut interférer la transmission des paquets SYNC. Mais on doit réaliser un fait celui de la transmission qui peut se prolonger à la période suivante de synchronisation même si la période d'écoute adaptative est placée exactement à l'intérieur de la période de sommeil (voir la figure 3.13). On peut ainsi dire que le fait de ne pas laisser la période d'écoute adaptative chevaucher la période suivante de SYNC est d'accorder la priorité aux paquets SYNC, mais l'écoute adaptative ne peut pas s'assurer de ne jamais se heurter avec la période suivante de SYNC.

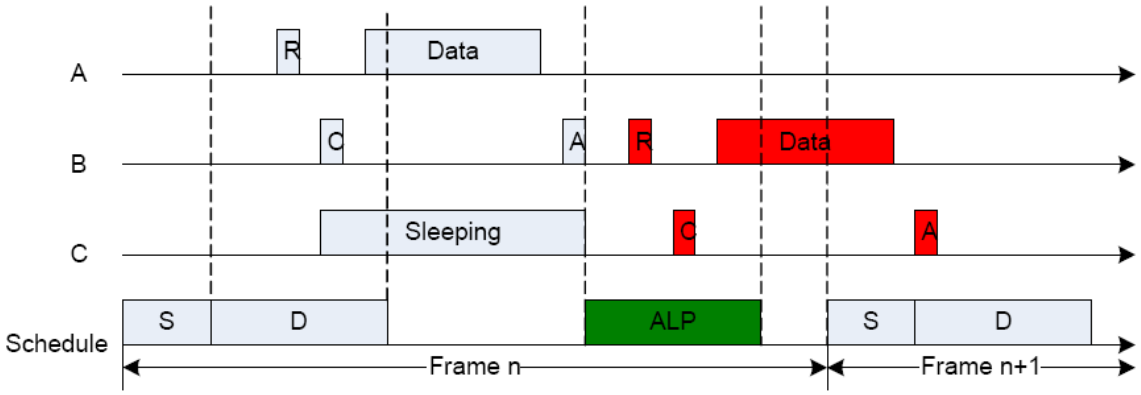


Figure 3.13. L'écoute adaptative s'étend au prochain frame

A partir de là, on peut déduire que l'écoute adaptative peut considérablement réduire la latence provoquée par le sommeil périodique de chaque nœud dans les réseaux multi-sauts. Considérer un paquet de données, qui doit être transmis à travers un réseau multi-saut. Lorsque l'écoute adaptative n'est pas appliquée, le paquet de données peut sauter seulement un seul saut dans un temps d'un frame. Dans le cas contraire (quand l'écoute adaptative est appliquée) le paquet de données peut être retransmis au nœud suivant juste après que sa dernière transmission soit terminée.

Il est à noter que l'écoute adaptative est limitée théoriquement à un saut, car les voisins qui sont loin de deux sauts sont supposés incapables de s'entendre. A partir de cette supposition, on peut déduire qu'un paquet de données est capable de sauter au plus deux sauts dans un temps d'un frame (figure 3.14).

À la fin de la transmission entre A à B, l'écoute adaptative est déclenchée et B commence à transmettre le paquet de données à C et entre B et C, une deuxième écoute adaptative sera déclenchée. On suppose que le temps restant dès maintenant à la prochaine période écoute est assez long pour adapter à une période d'écoute adaptative. Maintenant C essaye de retransmettre le paquet de données à D et envoie un paquet de RTS après la Carrier sense. Évidemment D dort, parce qu'il

n'est pas au courant de l'écoute adaptative qui est arrivée à ses voisins. C doit attendre jusqu'à ce que la période suivante de DONNÉES vienne.

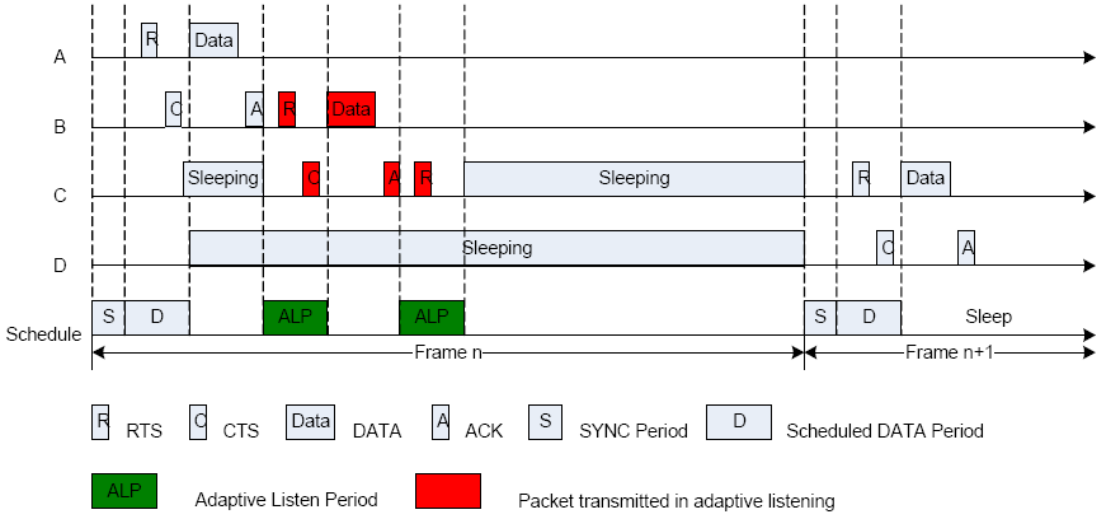


Figure 3.14. L'écoute adaptative déclenchée deux fois dans un seul frame [26]

3.9. Conclusion

Nous avons présenté les différents fonctionnalités et dispositifs de SMAC, tel que le sommeil et l'écoute périodiques, la synchronisation, en définissant des notions comme la liste des voisins et la table d'ordonnancement, et des fonctionnalités comme la découverte périodique des voisins, overhearing avoidance , message passing et l'écoute adaptative.

Nous concluons que l'efficacité énergétique et la capacité de l'auto configuration sont les buts fondamentaux de SMAC, alors que d'autres critères, tel que la latence et l'équité, sont secondaires.

Chapitre 4
Simulation et évaluation

Chapitre 4

Simulation et évaluation

4.1 Introduction

La réduction de la consommation d'énergie dans les réseaux mobiles ad hoc et de capteurs est un important défi où beaucoup de progrès ont été réalisés. Comme l'interface réseau peut devenir libre à plusieurs reprises, l'énergie doit être conservée en éteignant les transmetteurs radios lorsque ces derniers ne sont pas utilisés.

Afin de concevoir un bon protocole MAC pour les réseaux de capteurs sans fil, il faut prendre en considération plusieurs paramètres.

Les nœuds de capteurs sont censés être équipés d'une batterie. En raison de leur principe de fonctionnement, la recharge ou le remplacement des batteries de chaque nœud est difficile et non économique voire parfois impossible.

Pour résoudre ce problème, nous devons identifier les sources de perte d'énergie. La communication radio consomme la plus grande partie d'énergie. Toutefois, l'utilisation de la radio a une relation étroite avec les protocoles MAC. De ce fait, nous avons identifié les différentes sources de perte d'énergie :

- L'écoute à vide (L'énergie « idle »)
- La collision ou corruption.
- L'overhead des paquets de contrôle.
- L'énergie de transmission.
- L'énergie de réception.
- L'énergie de sommeil « sleep »

Un autre paramètre important est la latence. C'est un attribut commun pour la plupart des protocoles MAC. Son importance dépend des applications réelles. Pour la majorité des applications du réseau de capteurs, la vitesse de changements sur les objets physiques captés par les nœuds de capteurs est beaucoup plus lente que la vitesse d'un réseau. Donc, la latence est moins importante et pourrait être tolérée dans certains cas.

A partir de là, nous allons procéder à des simulations pour prouver l'efficacité de SMAC d'un point de vue énergétique et la préservation de capacité de réseau. Puis nous allons procéder à des simulations en changeant la durée de la période active, pour vérifier si SMAC garde son efficacité.

4.2. Simulations et évaluation des performances du SMAC

Dans cette partie, les différentes simulations réalisées ont pour but l'évaluation des performances du SMAC.

Pour avoir une bonne vue sur les performances du SMAC, des simulations ont été réalisées sur deux réseaux dont le scénario est le même (celui décrit ci-dessous) mais la politique de conservation d'énergie diffère. Le premier réseau utilise la sous couche MAC IEEE 802.11 et le deuxième utilise SMAC avec les paramètres par défaut ce qui dit le duty cycle à 10 %.

Cette première phase de simulation a pour but de confirmer l'efficacité de protocole SMAC par rapport 802.11. Nous allons faire une comparaison entre le protocole 802.11 et SMAC sur les critères suivants :

L'énergie totale consommée, l'énergie « idle », l'énergie de transmission, l'énergie de réception, l'énergie « sleep », l'overhead (les paquets de contrôle), les collisions,…

- **Environnement de simulation**

Les simulations présentées dans la section suivante ont été réalisées avec le ns-allinone-2.28. Le choix s'est porté sur cette version en raison des aides trouvées pour implémenter SMAC sur cette version. Les simulations ont été effectuées sur un ordinateur portable Core 2 Duo d'une capacité de 1.4 GHZ pour chaque core, doté d'une mémoire RAM de 1 Go.

- **Scénario utilisé:**

Le scénario des simulations effectuées est décrit dans les points suivants :

-La durée de la simulation est de 700 secondes. Durant cette période, la moyenne des quantités d'énergie des nœuds sera mesurée toutes les 20 secondes.

- Le réseau est constitué de 21 nœuds en tout. Il existe 15 nœuds chargés de relayer les paquets et assurer ainsi le trafic qui est généré par 3 nœuds sources à destination de 3 autres nœuds. La quantité d'énergie contenue dans les nœuds sources et destinations n'est pas prise en compte dans les résultats des simulations.

- Chaque nœud source génère un trafic de type CBR (Constant Bit Rate). Le flux CBR est composé de paquets de 512 octets.

- Pour s'assurer du fait que les paquets de chaque flux CBR passent par plusieurs sauts avant d'atteindre le nœud destination, les nœuds sources sont placés aléatoirement et

uniformément dans la partie la plus à gauche du plan de la simulation tandis que les nœuds destinations sont placés dans la partie la plus à droite du plan de la simulation. Les positions initiales des autres 15 nœuds sont choisies aléatoirement et uniformément dans tout le plan de la simulation. La figure suivante montre la topologie du scénario.

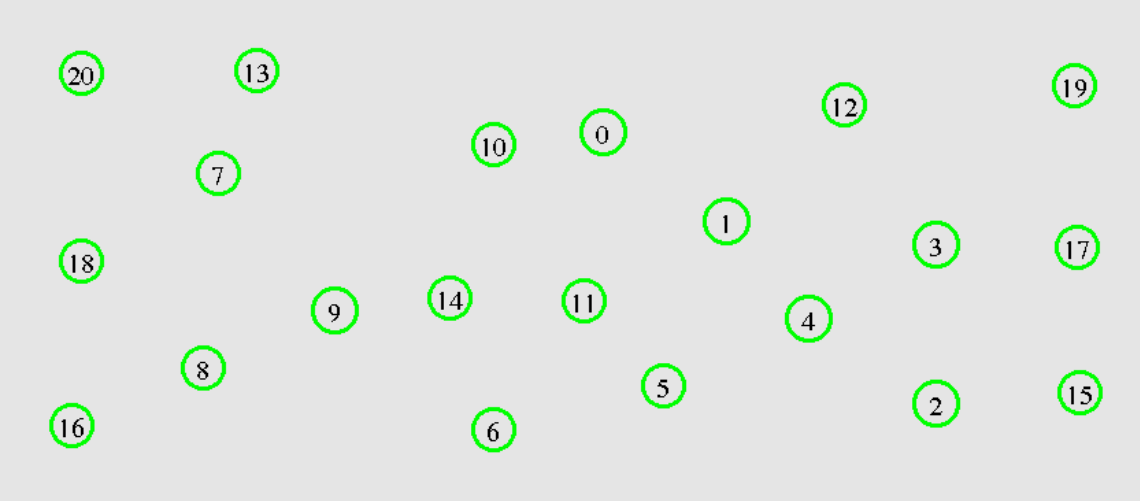


Figure. 4.1. Topologie du réseau

- La surface du plan de simulation est sur 800x800 mètres, l'énergie initiale des nœuds est de 1000 Joules. Ce modèle a été choisi afin de permettre l'étude des performances et l'adaptation aux moyens utilisés pour la simulation.

Dans la partie qui suit nous allons présenter les différentes simulations réalisées et la synthèse de chacune d'elles.

Le tableau suivant illustre les paramètres utilisés dans toutes les simulations :

Les paramètres de simulation	**Valeur**
Version NS	NS2.28
Script de protocole de routage	DumbAgent
Taille de paquet de contrôle	512 bytes
La puissance de transmission « transmit »	10 mW

La puissance de réception « receive »	4 mW
La puissance « idle »	1 mW
La puissance « Sleep »	0.001 mW
La puissance de transition	0.2 mW
Le temps de transition	5 ms
Nombre de nœuds	21
Nombre d'application CBR (UDP)	3
Nombre de source	3
Nombre de Sink (Destination)	3
Energie initiale	1000 Joules

Table. 4.1. Les paramètres de simulation

4.2.1. Simulation n°1 : L'énergie totale

Le premier graphe, résultant des simulations effectuées, donne l'énergie totale d'un nœud dans le réseau en fonction du temps.

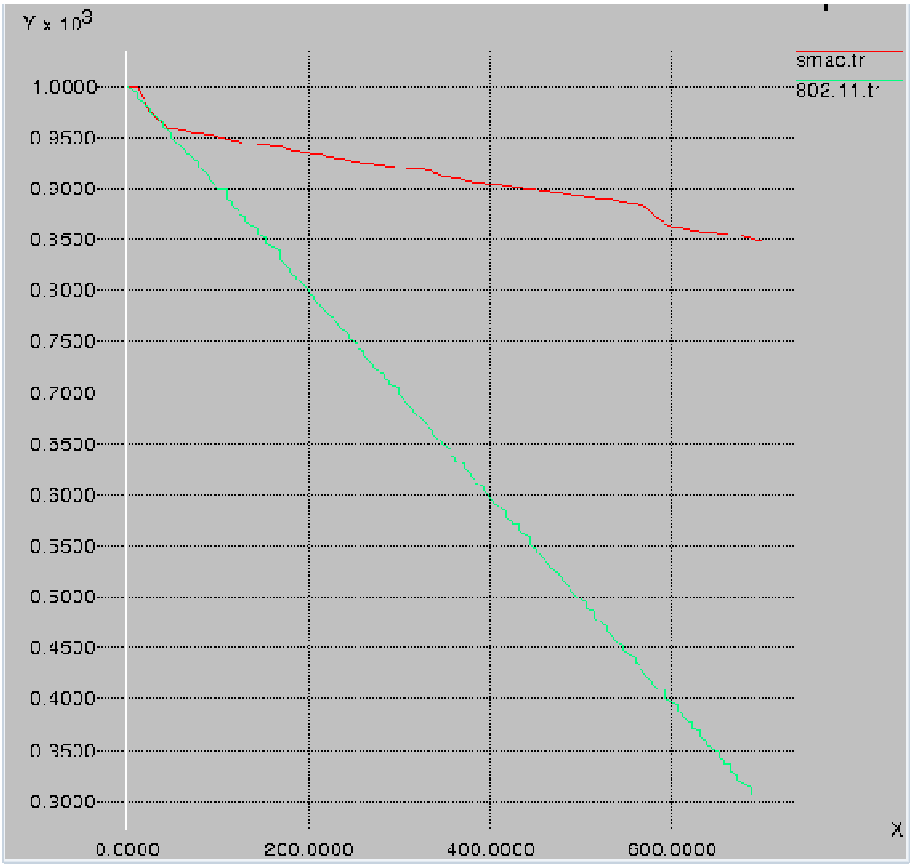


Figure 4.2. Taux d'énergie dans les réseaux 802.11 et SMAC en fonction de temps

On observe que SMAC conserve très bien l'énergie par rapport 802.11. On voit bien que le réseau 802.11 meurt très vite par rapport SMAC. L'énergie du réseau 802.11, dans un délai de 700 secondes, arrive à un niveau très bas « 300 Joules », ce qui dit la consommation de « 700 Joules » dans une durée très rapide. Cependant, le nœud de réseau SMAC consomme dans la même période (700 secondes) que « 150 Joules ». Ce qui donne un gain de « 550 Joules » par rapport à 802.11.

Pour savoir la consommation de chaque source de consommation d'énergie, nous avons fait les simulations qui suivent.

4.2.2. Simulation n°2 : L'énergie « Sleep »

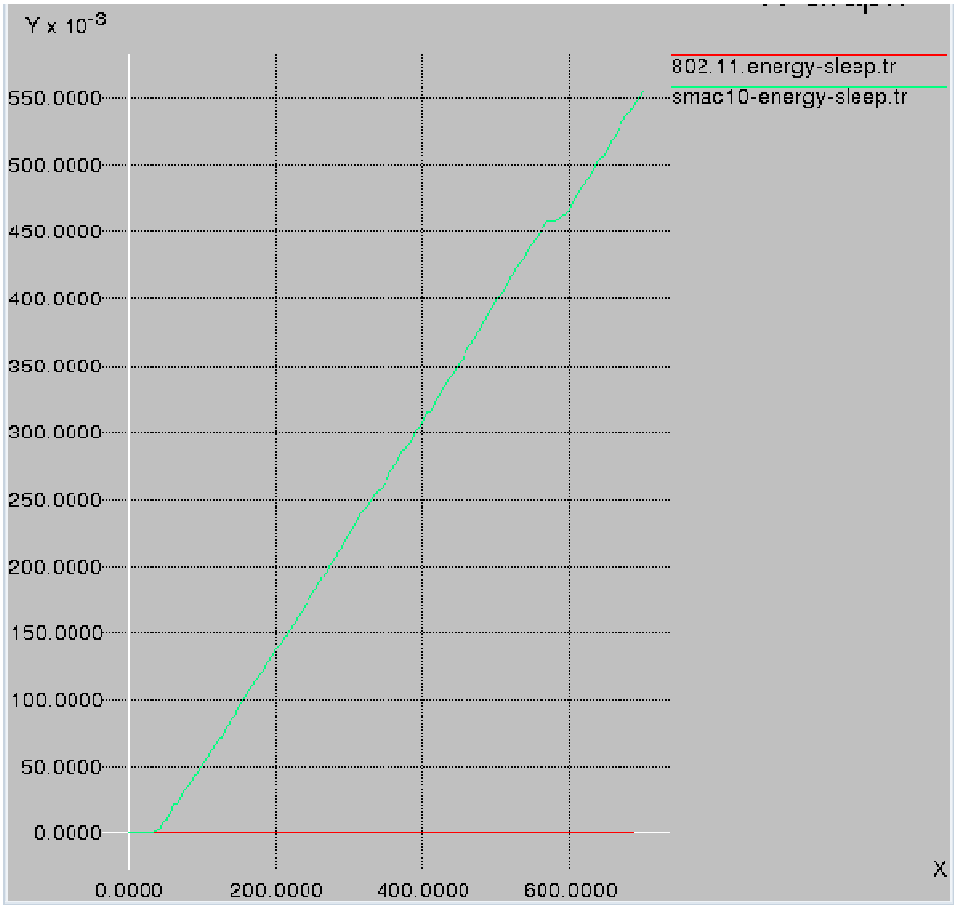


Figure 4.3. L'énergie « Sleep » dans 802.11 et SMAC en fonction de temps

De ce résultat, nous observons qu'un nœud SMAC consomme presque « 0,555 Joules » dans la période de simulation (700 secondes). Nous pouvons confirmer que SMAC consomme plus d'énergie dans l'état « sleep » par rapport 802.11 qui ne consomme rien puisque il n'utilise jamais l'état « sleep » ce qui dit que les nœuds restent toujours actifs. Mais, l'énergie consommée par le nœud SMAC est négligeable par rapport à l'énergie initiale du nœud.

4.2.3. Simulation n°3 : L'énergie « transmit »

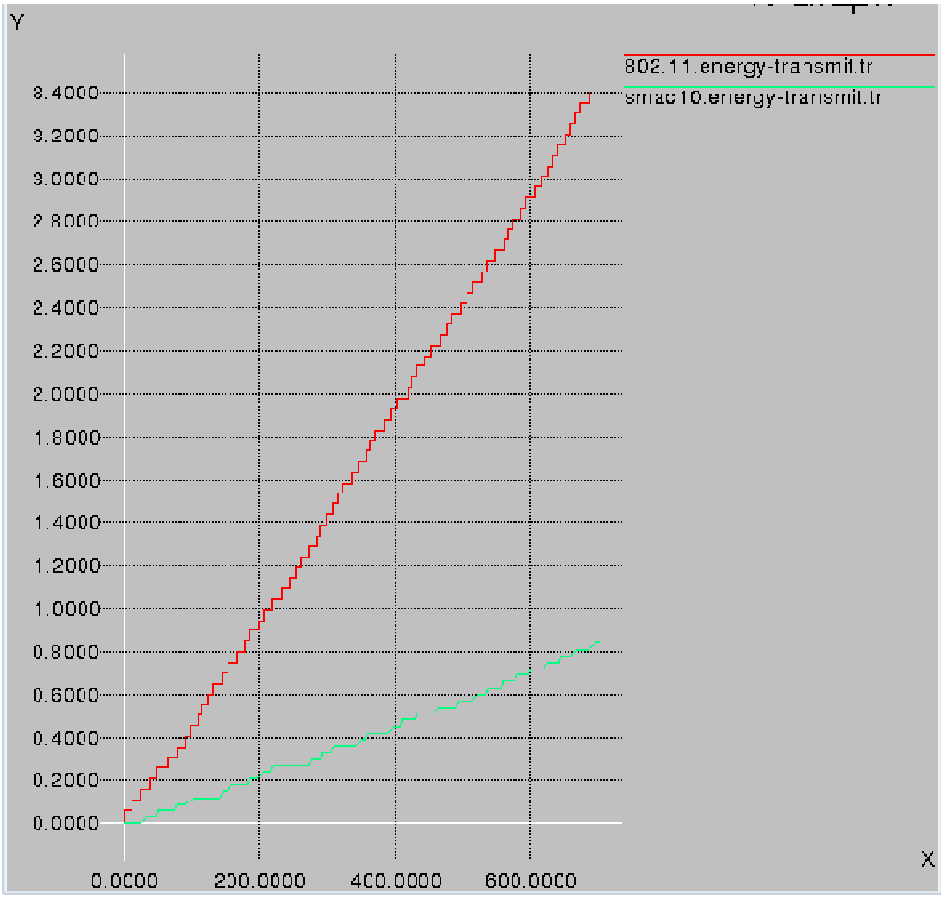


Figure 4.4. L'énergie de transmission dans 802.11 et SMAC en fonction de temps

A partir de la figure 4.4, nous observons que le nœud 802.11 consomme plus d'énergie de transmission des paquets par rapport à SMAC. Nous observons qu'un nœud 802.11 consomme, dans la période de simulation, « 3,402 joules », et SMAC consomme « 0,840 joules ».

4.2.4. Simulation n°4 : L'énergie « receive »

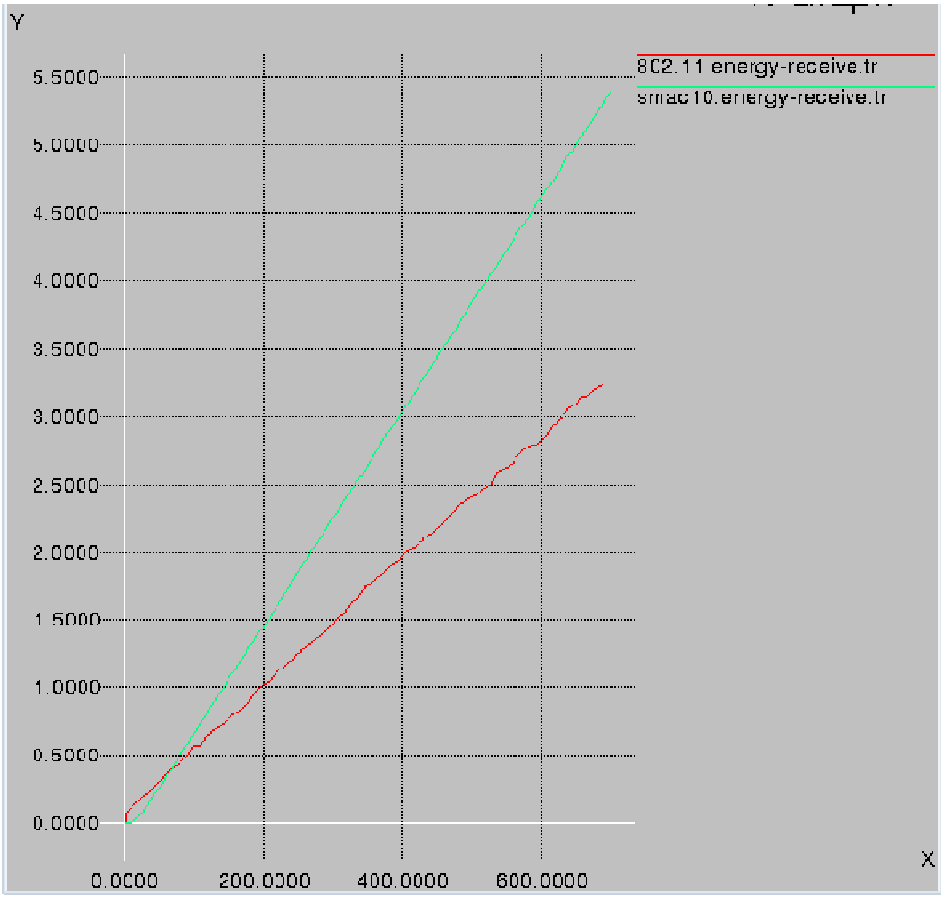


Figure 4.5. L'énergie de réception dans 802.11 et SMAC en fonction de temps

De ce résultat, nous observons que le nœud 802.11 consomme moins d'énergie de réception des paquets par rapport à SMAC. Nous observons qu'un nœud 802.11 consomme, dans la période de simulation, « 3,238 joules », et SMAC consomme « 5,388 joules ».

4.2.5. Simulation n°5 : L'énergie « idle »

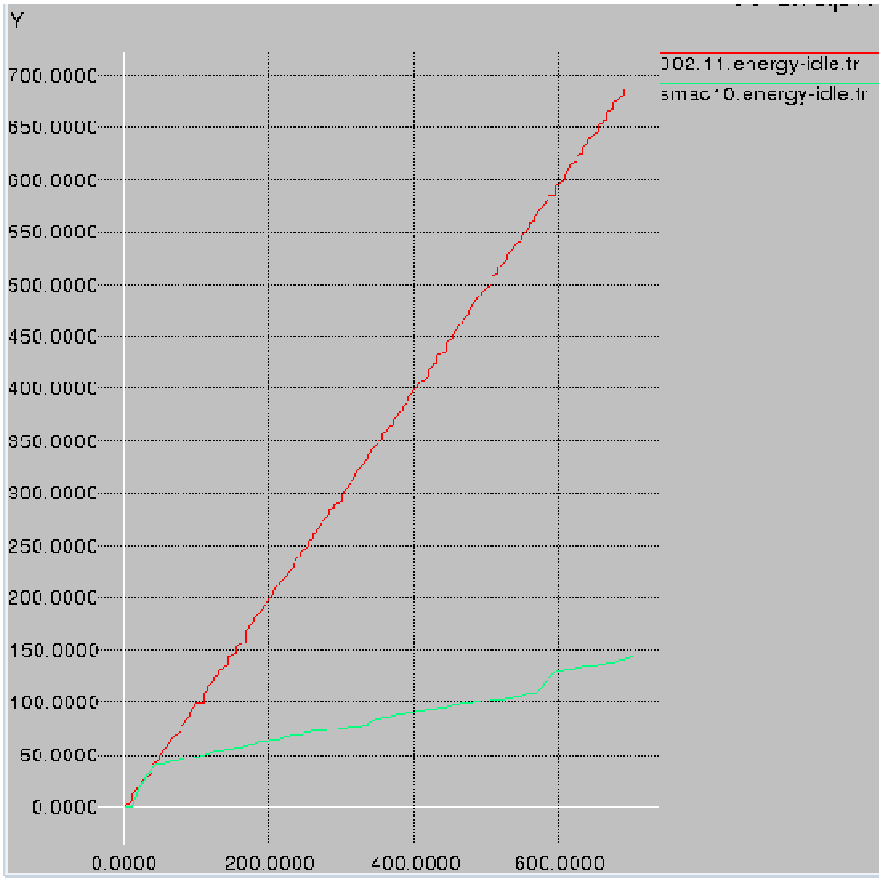


Figure 4.6. L'énergie « idle » dans 802.11 et SMAC en fonction de temps

Nous observons que le nœud 802.11 consomme beaucoup plus d'énergie à l'écoute à vide (en état « idle ») par rapport au SMAC. Nous observons qu'un nœud 802.11 consomme, dans la période de simulation, « 686,848 joules », et SMAC consomme « 143,402 joules ». Ce qui dit une différence de « 543,446 joules ».

Pour avoir une idée plus claire sur la consommation d'énergie, nous devons faire une comparaison entre les différentes énergies consommées par un nœud SMAC et 802.11.

La figure suivante illustre une comparaison entre les énergies consommées « transmit », « receive », « sleep » et « idle » par un nœud 802.11.

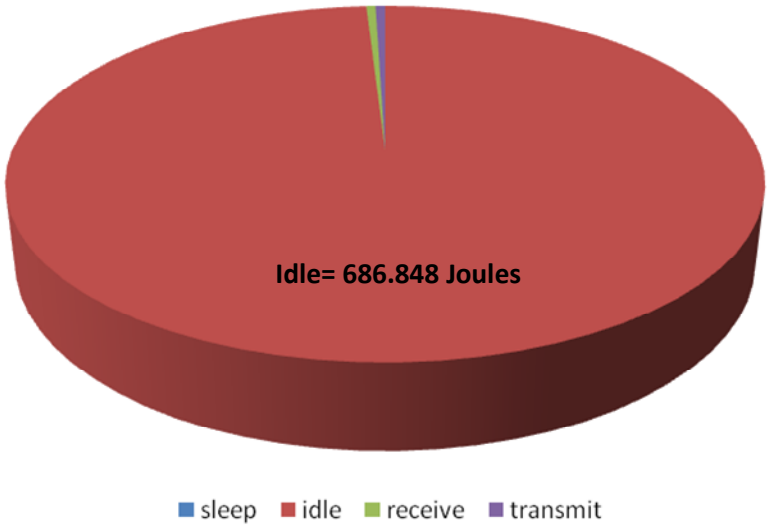


Figure 4.7. Les différentes énergies consommées par un neoud 802.11

Ainsi, la figure ci-dessous illustre une comparaison entre les énergies consommées « transmit », « receive », « sleep » et « idle » par un nœud SMAC.

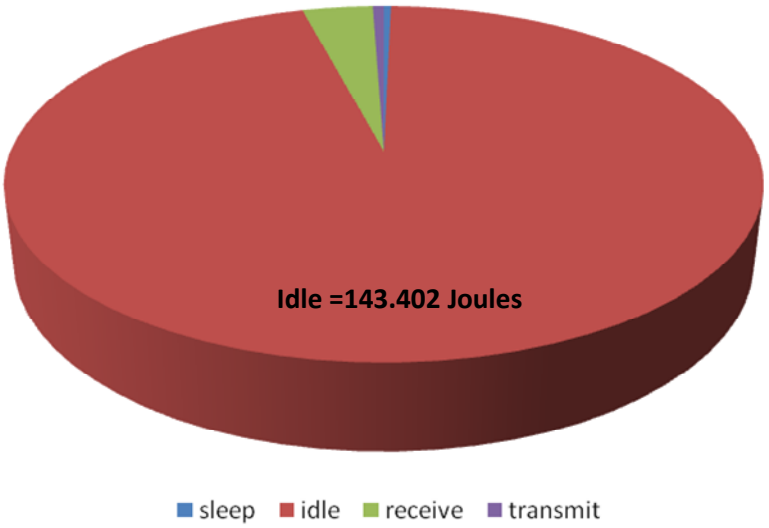


Figure 4.8. Les différentes énergies consommées par un noeud SMAC

A partir de la figure 4.7 et 4.8, nous constatons que l'écoute à vide consomme beaucoup plus d'énergie ,que ce soit dans un réseau 802.11 ou un réseau SMAC, par rapport aux transmissions ou récetpions ou l'état de « sleep ».

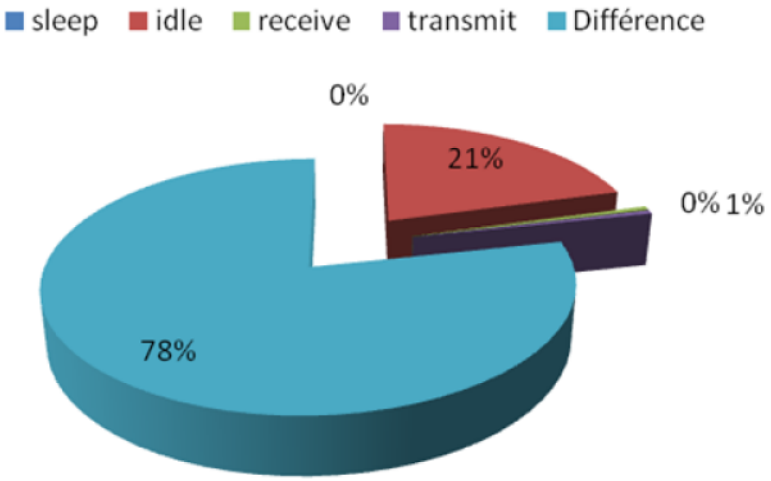


Figure 4.9. Le gain d'énergie entre 802.11 et SMAC

Malgré que l'énergie consommé de l'écoute à vide est grande en utilisant les deux protocoles, nous constatons que SMAC consomme beaucoup moins d'énergie ce qui donne un gain de l'ordre de 78% d'énergie de la consommation totale d'énergie.

- **Synthèse1 :**

A partir des dernières simulations, nous pouvons confirmer que l'un des grands consommateurs d'énergie dans un réseau adhoc ou de capteurs est l'écoute à vide « idle ».

Ainsi, nous confirmons que SMAC est très efficace en minimisant l'énergie consommée totale, tout en minimisant l'énergie de l'écoute à vide en utilisant le cycle de sommeil et d'écoute périodique.

4.2.6. Simulation n°6 : Les collisions

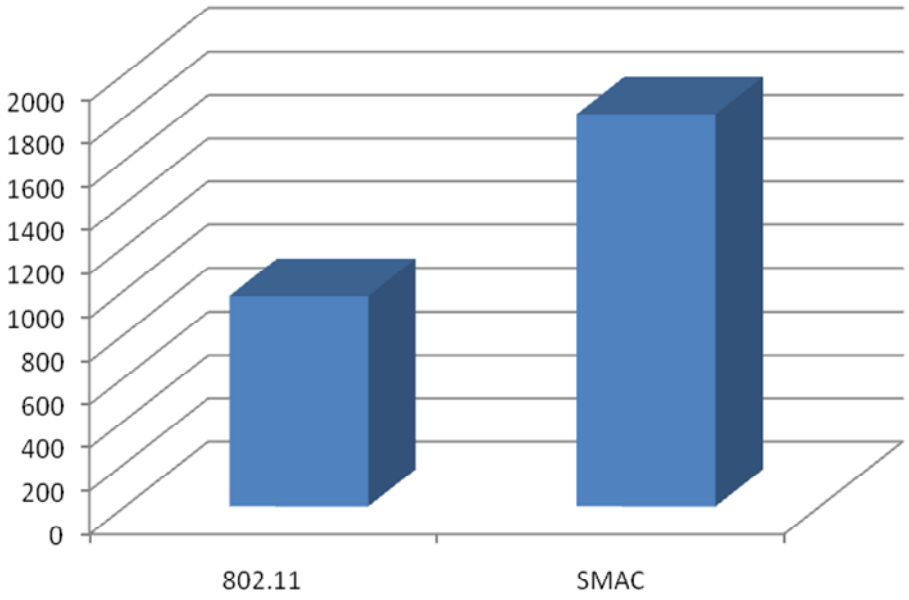


Figure 4.10. Les collisions dans 802.11 et SMAC

A partir de cette figure, nous observons que dans SMAC beaucoup plus de collisions se passent par rapport à 802.11 et ceci dans un délai de 700 secondes. Ces résultats peuvent être expliqués par

La table suivante illustre une classification par type de paquets des collisions :

CBR	ACK	ARP	CTS	RTS
0	0	0	0	970

Table 4.2. Les collisions par type de paquet dans 802.11

CBR	ARP	SYNC

0	0	1802

Table 4.3. Les collisions par type de paquets dans SMAC

Sur 802.11, nous observons que toutes les collisions ont été détectées sur les paquets RTS. Ainsi sur SMAC, toutes les collisions se passent sur les paquets SYNC. De ces résultats, nous pouvons déduire que les collisions sont passées uniquement sur les paquets de contrôle. Pour cela, nous allons voir dans la prochaine partie, une estimation de nombre de paquets de contrôle dans 802.11 et SMAC.

4.2.7. Simulation n°7 : Overhead (les paquets de contrôle)

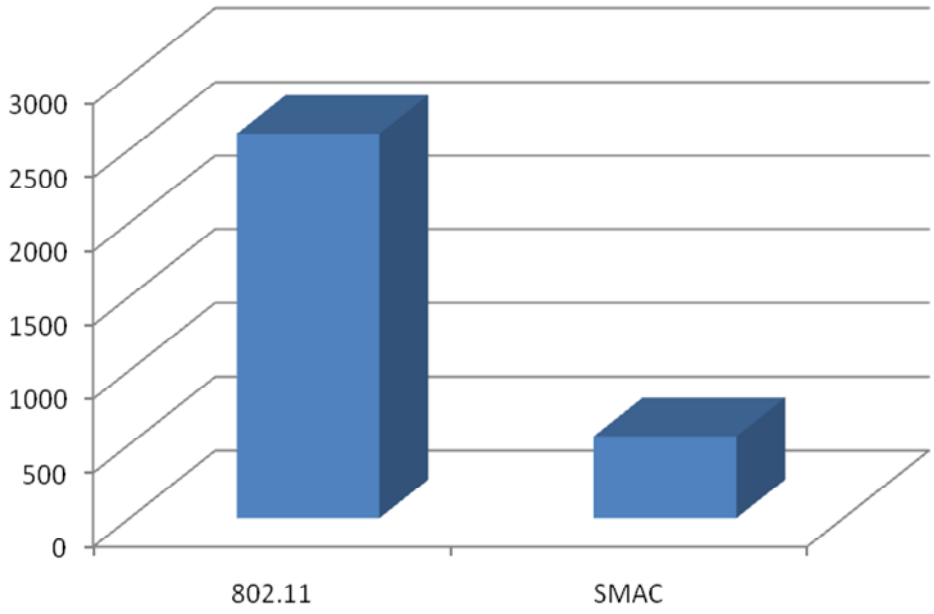


Figure 4.11. Nombre de paquets de contrôle dans 802.11 et SMAC

802.11	SMAC
2607	553

Table 4.4. Overhead 802.11 et SMAC

Nous observons que 802.11 utilise beaucoup plus de paquets de contrôles par rapport à SMAC. Cela peut expliquer l'utilisation de la technique « Message Passing » en évitant l'utilisation de plusieurs paquets de contrôle pour un ensemble de paquets de données.

4.2.8. Simulation n°8 : Taux de perte des paquets

802.11	SMAC
0	37,76

Table 4.5. Taux de perte dans les réseaux 802.11 et SMAC

Nous observons que le taux de perte de paquet sous SMAC est beaucoup plus grand que 802.11. On peut expliquer ça en comprenant le système de fonctionnement de SMAC qui utilise les modes veille et active.

4.2.9. Simulation n°9 : Latence

	Latence moyenne	Max Latence	Min Latence
802.11	0	0	0
SMAC	224,97	360	35

Table 4.6. Latence(ms) entre SMAC et 802.11

On remarque bien que la latence sous 802.11 est négligeable par rapport SMAC et va vers « 0 ». En termes de latence, le protocole 802.11 est meilleur que celui de SMAC, ce qui est logique puisque SMAC utilise le mode veille.

4.2.10. Synthèse

De ce qui précède, il est possible d'affirmer que SMAC prolonge la durée de vie du réseau, conserve l'énergie des nœuds en préservant les capacités du réseau.

Les performances du SMAC sont nettement meilleures que celles du protocole de 802.11.

SMAC répond donc très bien aux principaux critères d'un bon protocole de conservation d'énergie, à savoir prolongement de la durée de vie du réseau et préservation des capacités du réseau.

4.3. SMAC avec variation de paramètre « Duty cycle »

Le paramètre « Duty_cycle » est un paramètre auto-configurable par l'utilisateur, son rôle est de définir la durée active par rapport à la durée d'un frame.

Les simulations ont été réalisées avec six réseaux et le scénario est le même que celui qui vient d'être décrit. Ces simulations utilisent, tous les six, le protocole SMAC. Il faut noter que :

Le 1er réseau : Le paramètre « Duty_cycle » est mis par défaut à 10 (ce qui signifie que les nœuds restent active 10% de la durée d'un frame)

Le 2ème réseau : Le paramètre « Duty_cycle » est mis par défaut à 30.

Le 3ème réseau : Le paramètre « Duty_cycle » est mis par défaut à 50.

Le 4ème réseau : Le paramètre « Duty_cycle » est mis par défaut à 70.

Le 5ème réseau : Le paramètre « Duty_cycle » est mis par défaut à 90.

On évaluera les performances de chaque réseau en se basant sur les mêmes critères que lors des précédentes simulations c'est-à-dire la conservation d'énergie, le prolongement de la durée de vie du réseau et la préservation des capacités du réseau, l'overhead, les collisions,….

Pour réaliser cette simulation, on utilise le même scénario précédent.

4.3.1. Simulation n°1 :L'énergie totale restante (La consommation d'énergie) :

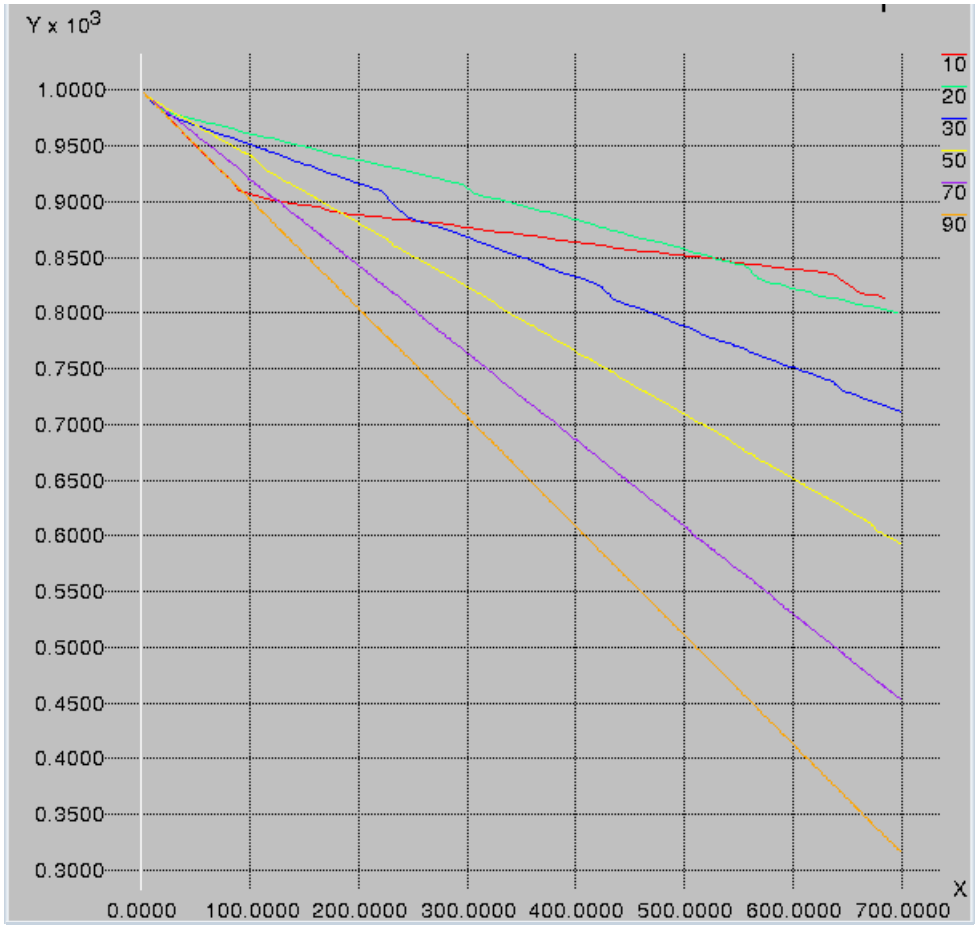


Figure 4.12. Taux d'énergie dans SMAC en changeant Duty Cycle

On observe que sur SMAC, plus le paramètre « Duty cycle » est grand plus l'énergie consommée est grande. On voit bien que le réseau avec SMAC avec duty cycle à 90% meurt très vite par rapport SMAC avec duty cycle à 10%. L'énergie du réseau SMAC avec duty cycle à 90%, dans un délai de 700 secondes, arrive à un niveau très bas « 320 Joules », ce qui dit la consommation de presque « 700 Joules » dans une durée très rapide. Cependant, le nœud de réseau SMAC avec duty cycle à 10% consomme dans la même période « 700 secondes » que « 200 Joules ».

Pour savoir la consommation de chaque source de consommation d'énergie, nous avons fait les simulations qui suivent.

4.3.2. Simulation n°2 :L'énergie « Sleep »

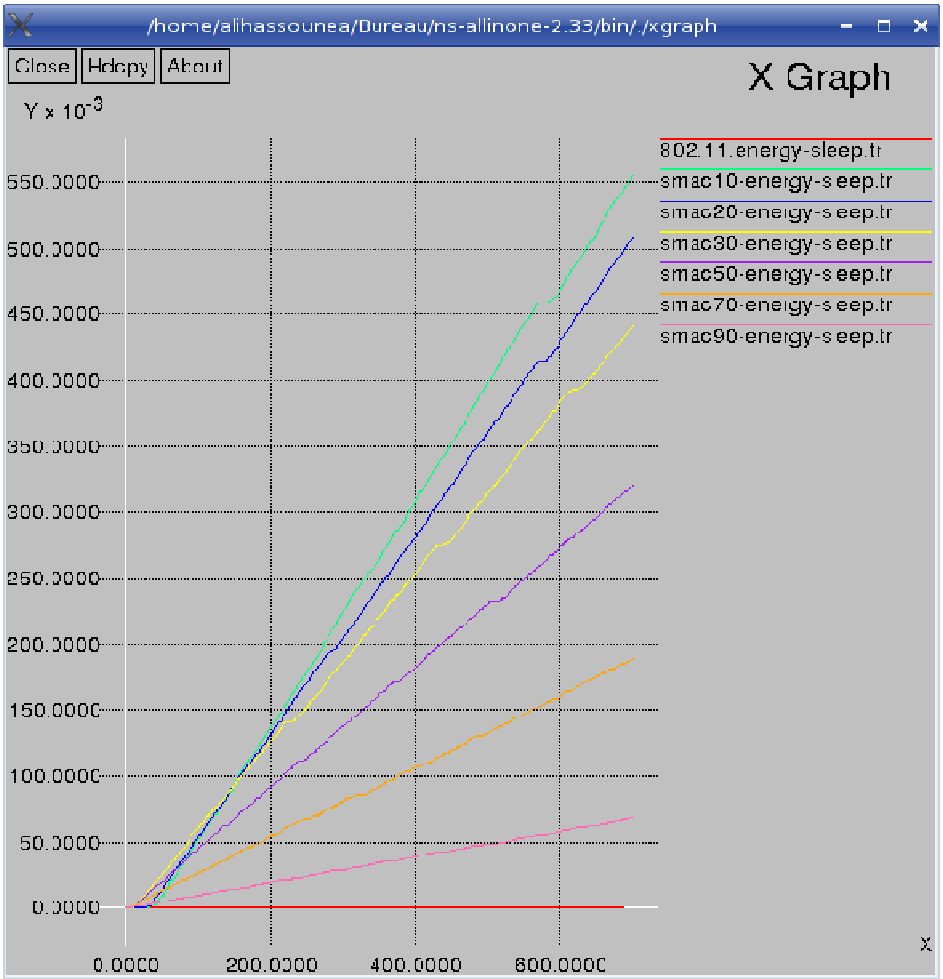


Figure 4.13. L'énergie « Sleep » en fonction de temps

De ce résultat, nous observons qu'un nœud SMAC avec duty cycle à 10% consomme presque « 0,555 Joules » dans la période de simulation (700 secondes) et avec duty cycle à 90% consomme «0,068 Joules». Nous pouvons confirmer que SMAC avec duty cycle à 10 % consomme plus d'énergie dans l'état « sleep » par rapport SMAC avec duty cycle à 90% qui ne consomme presque rien puisque il utilise l'état « sleep » juste à 10% de cycle complet.

4.3.3. Simulation n°3 : L'énergie « transmit »

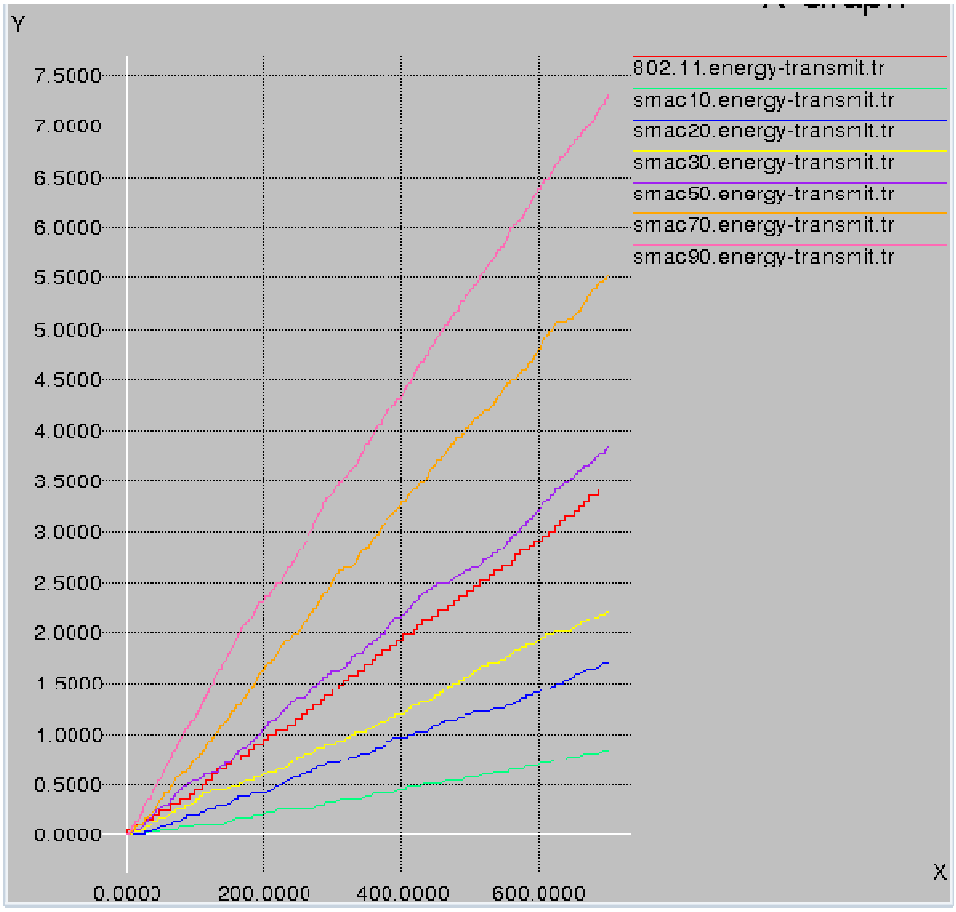


Figure 4.14. L'énergie de transmission en fonction de temps

A partir de la figure 4.14, nous observons qu'un nœud SMAC avec duty cycle à 10% consomme moins d'énergie (0,840 joules) qu'avec duty cycle à 90% consomme

4.3.4. Simulation n°4 :L'énergie « receive »

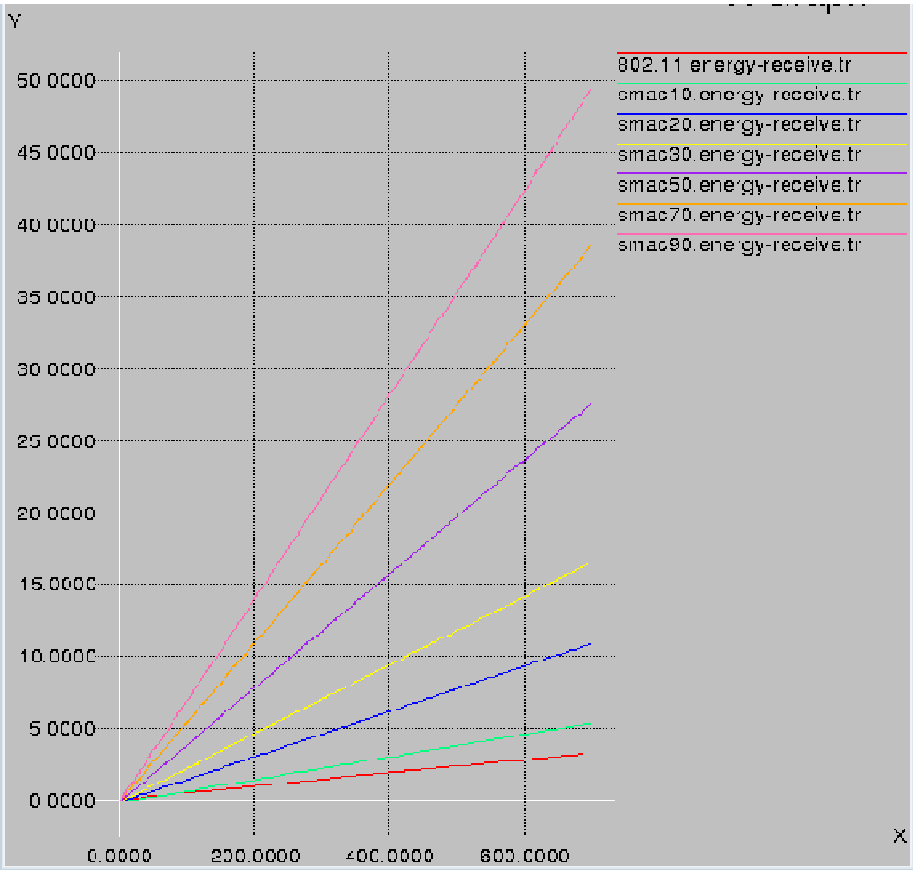


Figure 4.15. L'énergie de réception en fonction de temps

De ce résultat, nous observons que SMAC avec duty cycle à 10% consomme moins d'énergie de réception des paquets par rapport à SMAC avec duty cycle à 90%. Nous observons que SMAC avec duty cycle à 10% consomme, dans la période de simulation, « 5,388 joules », et duty cycle 90% consomme « 49,393 joules ».

4.3.5. Simulation n°5 : L'énergie « idle »

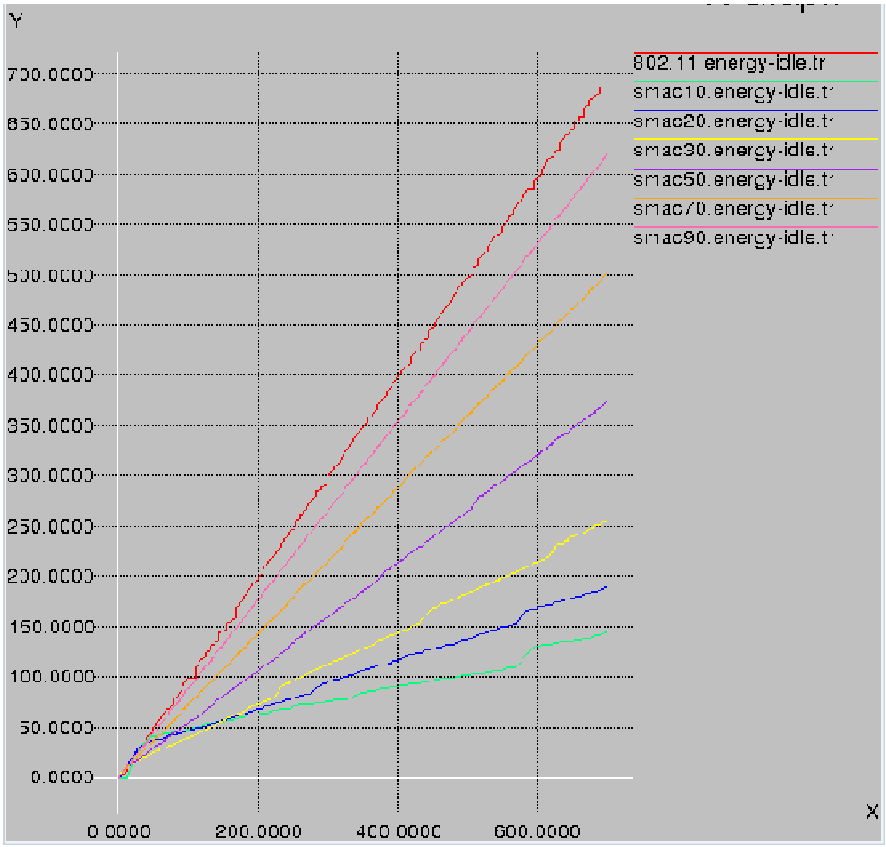


Figure 4.16. L'énergie « idle » en fonction de temps

Puisque l'écoute à vide est la plus importante source de consommation d'énergie, ce facteur est très important à étudier. Nous observons que plus la période active est plus grande (duty cycle est grand), plus l'énergie consommée dans l'état « idle » est grande.

4.3.6. Simulation n°6 : Collisions

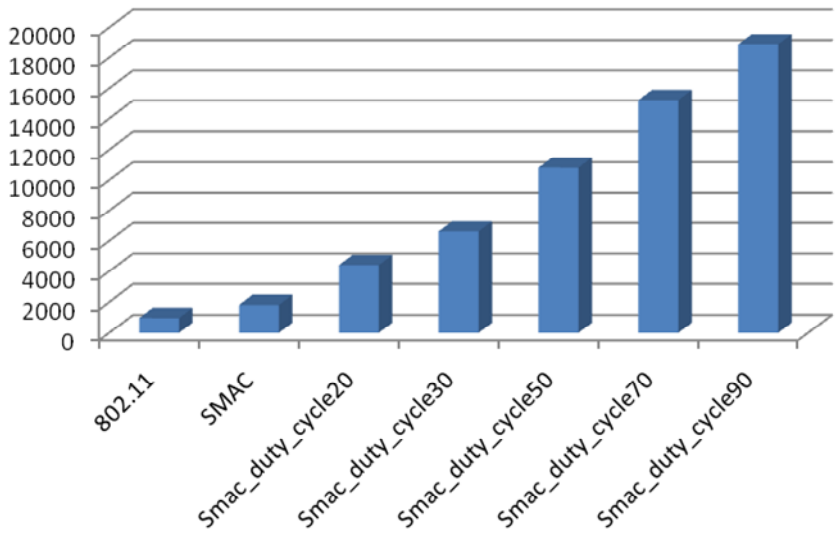


Figure 4.17. Les collisions dans SMAC en modifiant Duty Cycle

Smac duty cycle 10	Smac duty cycle 30	Smac duty cycle 50	Smac duty cycle 70	Smac duty cycle 90
1812	6617	10815	15218	18898

Figure 4.7. Les collisions en chiffre

Nous observons que : plus le paramètre « duty cycle » est grand, plus le nombre de collisions est grand.

4.3.7. Simulation n°7 : Les paquets de contrôle (L'overhead)

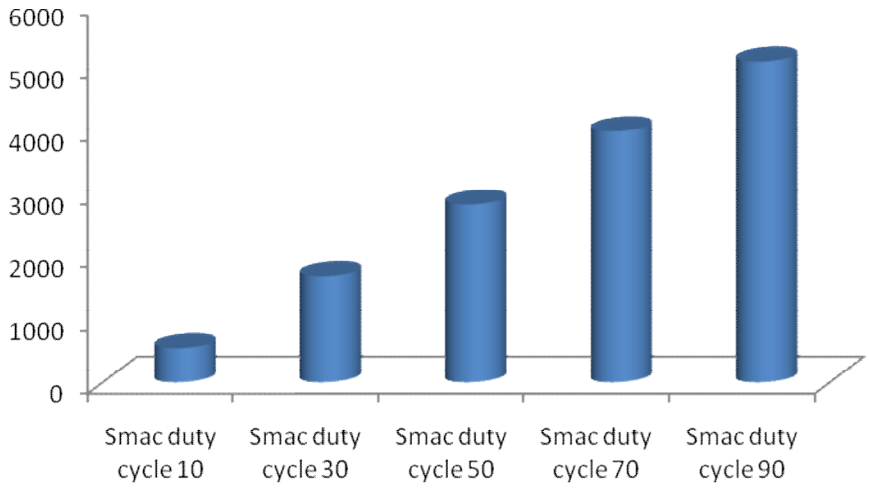


Figure 4.18. Les paquets de contrôle sous SMAC

Smac duty cycle 10	Smac duty cycle 30	Smac duty cycle 50	Smac duty cycle 70	Smac duty cycle 90
553	1698	2840	3993	5108

Figure 4.8. Les paquets de contrôle en chiffre

Nous observons que le nombre de paquets de contrôle varie entre 553 et 5108 paquets en changeant le paramètre « duty cycle » de 10 à 90. Nous déduisons que plus le paramètre « duty cycle » est grand plus le nombre des paquets de contrôle utilisés est grand.

4.3.8. Simulation n°8 : Taux de perte des paquets

Smac duty cycle 10	Smac duty cycle 30	Smac duty cycle 50	Smac duty cycle 70	Smac duty cycle 90
89,68%	54,68%	37,5%	29,75%	51,62%

Figure 4.9. Taux de perte de paquets dans SMAC

Nous observons que le taux de perte de paquets varie entre 29% et 89% en changeant le paramètre « duty cycle » de 10 à 90 (selon le scénario choisi). Nous déduisons que plus le paramètre « duty cycle » est grand plus le taux de perte des paquets est grand.

4.3.9. Simulation n°9 : La latence

	Max Latence	Min Latence	Latence moyenne
Smac10	360	35	224,9697
Smac30	201	22	140,3172
Smac50	114	28	73,7600
Smac70	22	1	5,0113
Smac90	12	1	3,0636

Table 4.10. Latence (ms) sous SMAC

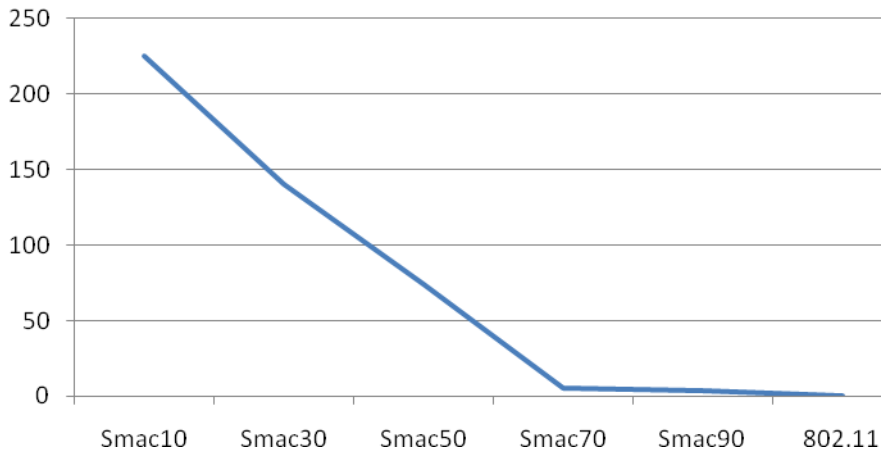


Figure 4.19. Latence(ms) en fonction de « duty cycle »

Nous observons très clairement la différence de la latence entre les résultats de Smac en changeant le paramètre « duty cycle ». Plus le paramètre « duty cycle » devient grand (ce qui dit la durée de veille est petite) plus la latence est meilleure.

4.3.10. Synthèse

SMAC avec le paramètre « duty cycle » plus faible offre une meilleure conservation d'énergie car les nœuds restent plus de temps en sommeil qu'en état active. Et pour la même raison, SMAC avec le paramètre « duty cycle » plus faible consomme moins d'énergie dans l'état de « sleep ». Cette énergie peut être négligeable par rapport à l'énergie totale consommée. Ainsi, SMAC consomme moins d'énergie que ce soit en transmission ou en réception si le paramètre « duty cycle » est faible.

En ce qui concerne l'écoute à vide, plus le paramètre « duty cycle » est faible plus l'énergie consommée est faible.

Plus le paramètre « duty cycle » est grand plus le nombre de collisions est grand sachant que toutes les collisions se produisent dans les paquets de contrôle.

En ce qui concerne ces derniers, le nombre d'utilisation des paquets de contrôle devient plus grand en augmentant la valeur de duty cycle.

Et c'est la même remarque pour le taux de perte de paquets, plus le paramètre duty cycle est grand plus le taux de perte est grand.

4.4. SMAC avec variation de paramètre « Rate » et « duty cycle »

Cette phase a pour objectif l'étude de l'impact du taux de trafic (rate) sur SMAC avec variation du duty-cycle.

4.4.1. Scénario utilisé: Etude de cas de 10 nœuds

Le scénario des simulations effectuées est décrit dans les points suivants :

-La durée de la simulation est de 700 secondes.

- Le réseau est constitué de 10 nœuds en tout. Il existe 8 nœuds chargés de relayer les paquets et assurer ainsi le trafic qui est généré par 1 nœud source à destination de 1 autre nœud. La quantité d'énergie contenue dans les nœuds sources et destinations n'est pas prise en compte dans les résultats des simulations.

- Chaque nœud source génère un trafic de type CBR (Constant Bit Rate). Le flux CBR est composé de paquets de 512 octets.

- La figure suivante montre la topologie du scénario.

Figure. 4.20. Topologie du réseau

- La surface du plan de simulation est sur 800x800 mètres. Ce modèle a été choisi afin de permettre l'étude des performances et l'adaptation aux moyens utilisés pour la simulation.

Le tableau suivant illustre les paramètres utilisés dans les toutes les simulations :

Les paramètres de simulation	**Valeur**
Version NS	NS2.28
Script de protocole de routage	DumbAgent
Taille de paquet de contrôle	512 bytes
La puissance de transmission « transmit »	10 mW
La puissance de réception « receive »	4 mW
La puissance « idle »	1 mW
La puissance « Sleep »	0.001 mW
La puissance de transition	0.2 mW
Le temps de transition	5 ms
Nombre de nœuds	10
Nombre d'application CBR (UDP)	1
Nombre de source	1
Nombre de Sink (Destination)	1
Energie initiale	2000 Joules

Table. 4.11. Les paramètres de simulation

4.4.2. Résultats de simulation :

	Paquets envoyés	Paquets reçus	Paquets détruits	Collisions	Latence moyenne (ms)	Energie restante	Idle	Sleep	Transmit	Receive
802.11	64	64	0	0	0	1003,092	688,872	0	3,896	1 ,745
Smac10	64	56	8	161	15,75	1491,879	409,700	0,272	55,090	42,636
Smac30	64	62	2	314	4,8710	1594,580	299,506	0,379	55,9	47,372
Smac50	64	64	0	588	2,5312	1489,912	384,044	0,296	70	51,648
Smac70	64	64	0	1112	1,4531	1370,534	492,89	0,186	75,93	54,50
Smac90	64	64	0	494	2,375	1236,545	607,795	0,068	88,26	59,728

Table 4.12. Résultats de simulation : Rate = 10 et Duty cycle = variable

	Paquets envoyés	Paquets reçus	Paquets détruits	Collisions	Latence moyenne (ms)	Energie restante	Idle	Sleep	Transmit	Receive
802.11	128	128	0	0	0	1293,653	692,566	0	7,536	6,045
Smac10	12	87	41	160	104,2184	1350,03	508,43	0,16	79,750	61,34

	8					6	7	8		5
Smac30	128	124	4	326	5,0480	1450,429	361,035	0,305	102,620	83,788
Smac50	128	128	0	594	2,5078	1386,005	402,773	0,263	118,530	88,780
Smac70	128	128	0	1051	1,4062	1284,756	492,909	0,172	124,570	92,092
Smac90	128	128	0	523	2,3516	1161,061	593,911	0,068	141,720	96,236

Table 4.13. Résultats de simulation : Rate = 5 et Duty cycle = variable

	Paquets envoyés	**Paquets reçus**	**Paquets détruits**	**Collisions**	**Latence moyenne** (ms)	**Energie restante**	**Idle**	**Sleep**	**Transmit**	**Receive**
802.11	640	640	0	0	0	1245,728	687,991	0	36,834	29,447
Smac10	640	60	574	228	224,9697	1428,009	443,770	0,235	72,930	54,692
Smac30	640	290	350	316	140,3172	1090,178	489,128	0,141	235,470	184,241
Smac50	640	506	134	601	82,804	782,959	480,090	0,099	418,480	317,002
Smac70	640	639	1	308	3,7856	591,023	490,489	0,058	519,150	397,436
Smac90	640	283	357	811	3,063	979,251	561,736	0,065	269	184,055

Table 4.14. Résultats de simulation : Rate = 1 et Duty cycle = variable

	Paquets envoyés	Paquets reçus	Paquets détruits	Collisions	Latence moyenne (ms)	Energie restante	Idle	Sleep	Transmit	Receive
802.11	1280	1280	0	0	0	1190,642	677,574	0	73,224	58,559
Smac10	1280	66	1214	228	231,4697	1428,009	443,77	0,235	72,930	54,692
Smac30	1280	290	990	316	144,3552	1090,178	489,128	0,141	235,470	184,241
Smac50	1280	505	775	600	116,5307	781,192	483,657	0,095	418,210	315,524
Smac70	1280	708	572	360	94,1794	496,895	473,259	0,058	586,67	441,266
Smac90	1280	353	927	563	57,2153	887,848	550,915	0,057	319,740	235,625

Table 4.15. Résultats de simulation : Rate = 0.5 et Duty cycle = variable

	Paquets envoyés	Paquets reçus	Paquets détruits	Collisions	Latence moyenne (ms)	Energie restante	Idle	Sleep	Transmit	Receive
802.11	6400	6400	0	3	0	753,957	590,717	0	364,082	291,243
Smac10	6400	64	6336	201	267,25	1434,424	443,083	0,235	68,310	53,584

Smac30	6400	264	6136	339	149,8447	1143,374	463,055	0,172	218,380	173,995
Smac50	6400	495	5905	600	115,1535	793,664	480,294	0,101	412,920	311,627
Smac70	6400	707	5693	441	97,0863	493,227	474,534	0,056	586,340	444,064
Smac90	6400	353	6047	714	67,5496	887,940	550,979	0,057	319,740	235,471

Table 4.16. Résultats de simulation : Rate = 0.1 et Duty cycle = variable

4.4.3. La Latence et le trafic

A partir de ces résultats et pour extraire la relation entre la latence et la densité du trafic du réseau, nous déduisons la figure suivante :

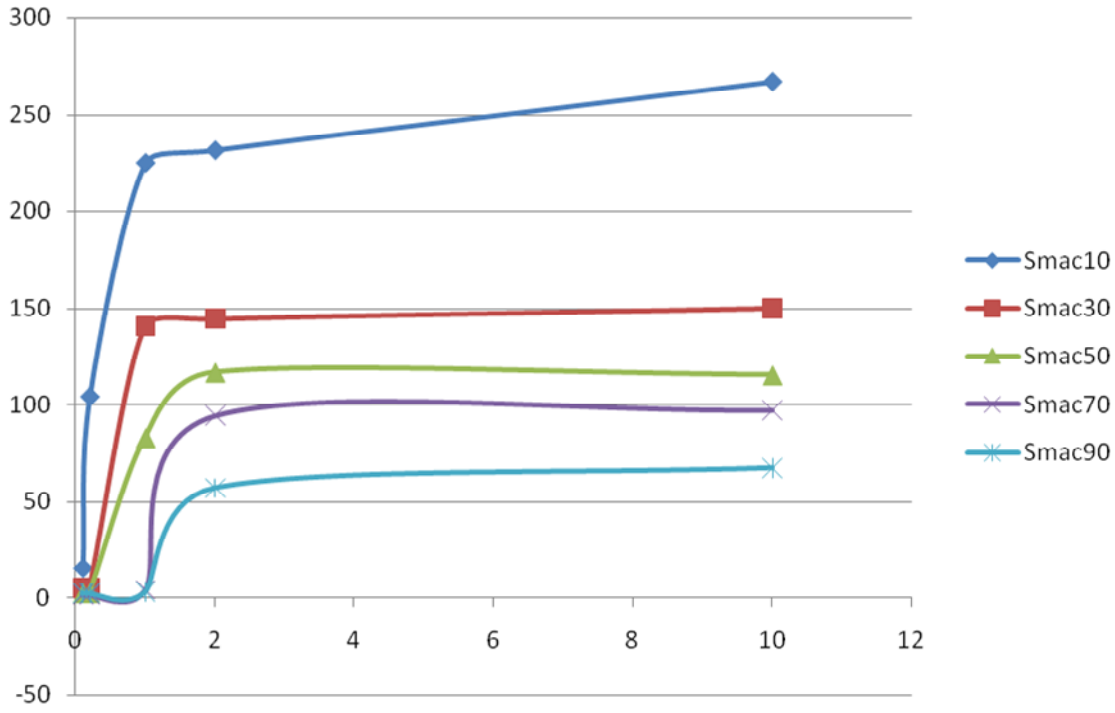


Figure 4.21. La latence(ms) en fonction du trafic du réseau

Nous observons que la latence devient plus grande en minimisant le paramètre « duty cycle ». Ainsi, plus le trafic devient plus dense plus la latence devient importante (Figure 4.22)

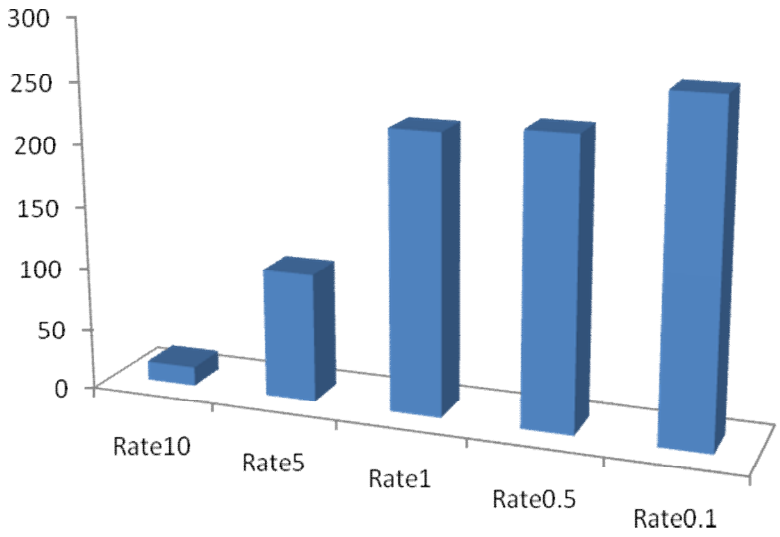


Figure 4.22. La latence moyenne (ms) des différents trafics du réseau

4.4.4. Les collisions et le trafic

En ce qui concerne les collisions, elles varient selon la densité du trafic du réseau, mais on ne peut pas déduire une relation entre les collisions et le trafic du réseau (voir la figure suivante).

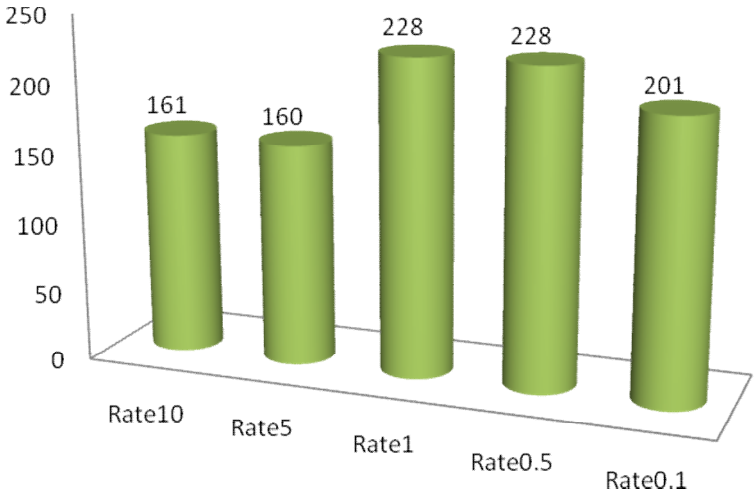


Figure 4.23. Taux de collisions des différents trafics du réseau

4.4.5. L'énergie et le trafic

En discutant de l'énergie, nous observons que plus le trafic devient plus dense, plus l'énergie consommée par les nœuds devient plus grande (Figure 4.24)

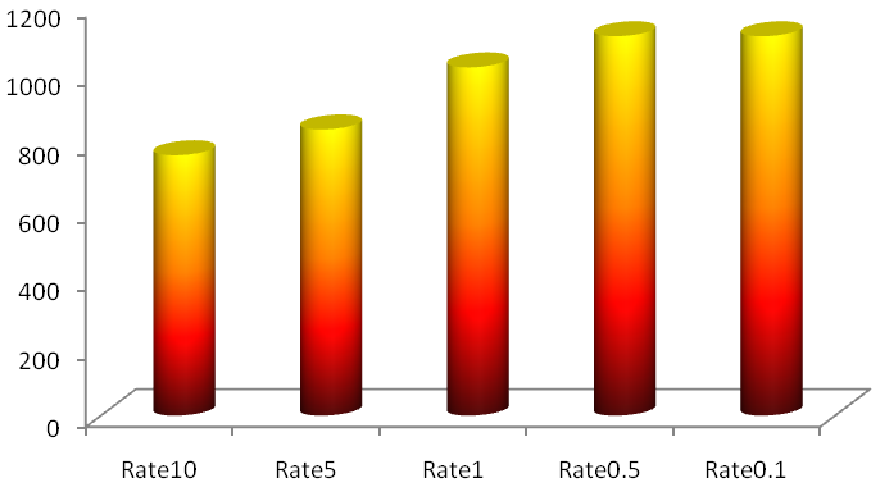


Figure 4.24. L'énergie consommée en fonction de trafic du réseau

Celui la peut être expliqué par la réception et l'émission de plus nombre de paquet, ce qui peut consommer plus d'énergie en émission et en réception (Figure 4.25)

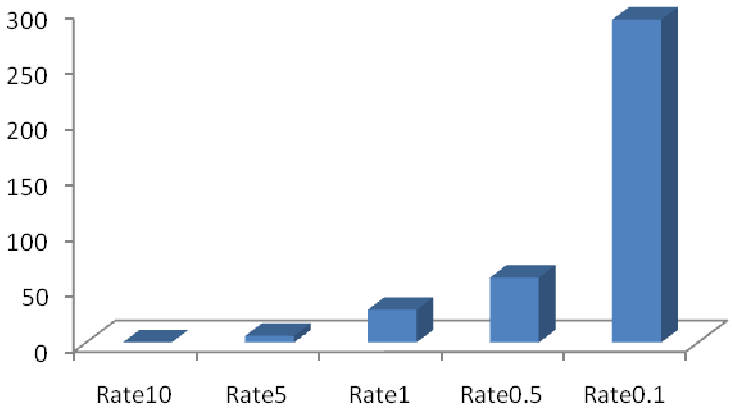


Figure 4.25. L'énergie de réception en fonction de trafic du réseau

4.5. Protocole LATMAC (LATency MAC)

4.5.1. Son objectif

Suite à l'étude et aux résultats obtenus, nous proposons un nouveau protocole LATMAC qui est similaire à SMAC, il prend en charge la minimisation de la consommation d'énergie tout en tenant en considération la latence.

4.5.2. Fonctionnement

Nous résumons le fonctionnement de notre proposition (LATMAC) en sept étapes comme suit:

1- L'utilisateur doit définir :

a) un seuil de latence maximale LATMAX à ne pas dépasser.

b) la latence minimale LATMIN.

c) un nœud source « S » et un nœud destination « D » pour le calcul de la latence.

d) l'intervalle de temps entre deux envoies de paquets LAT (LATTime).

Le paquet de vérification de la latence LAT contient les paramètres suivants :

Champs	**Commentaires**
type	Le drapeau indiquant ceci est un paquet de vérification de la latence
srcAddr	Identification du nœud source
DesAddr	Identification du nœud destination
SendTime	Le temps de transmission de paquet LAT
ReceiveTime	Le temps de réception de paquet LAT par le nœud destination

Table 4.17. Le paquet de vérification de la latence

2- LATMAC commence par DutyCycle =10, ce qui donne un minimum d'énergie consommée.

3- LATMAC calcule périodiquement la latence en envoyant un paquet LAT du nœud source « S » au nœud destination « D ».

4- LATMAC vérifie si la latence est entre l'intervalle fixée par l'utilisateur, LATMIN < LAT < LATMAX.

5-Un nouveau paramètre (dutyCycle) est ajouté au paquet de synchronisation SYNC. Le paquet SYNC devient :

Champs	**Commentaires**
type	Le drapeau indiquant ceci est un paquet de synchro
longueur	Taille fixe avec 9 bytes
srcAddr	Identification de l'expéditeur
syncNode	Identification du nœud de synchronisation de l'expéditeur
dutyCycle	Le nouveau duty cycle
sleepTime	Le prochain sommeil de l'expéditeur dès maintenant
State	Indiquant si l'expéditeur a changé récemment son programme d'ordonnancement
crc	Contrôle par redondance Cyclique

Table 4.18. Le frame de synchronisation SYNC

6- Si la latence mesurée dépasse la latence maximale LAT MAX, le nœud source incrémente le paramètre Dutycycle en envoyant à ses voisins un paquet SYNC avec la nouvelle valeur Dutycycle.

7- Si la latence mesurée est inférieure de la latence minimale définie par l'utilisateur le nœud source décrémente le paramètre Dutycycle en envoyant à ses voisins un paquet SYNC avec la nouvelle valeur DutyCycle.

4.5.3. L'algorithme LATMAC:

Si (LAT > LATMAX) alors

{ Si DutyCycle < 90 alors DutyCycle = DutyCycle + 10 }

Si (LAT < LATMIN) alors

{ Si DutyCycle > 10 alors DutyCycle = DutyCycle - 10 }

4.5.4. Comparaison entre LATMAC et SMAC

Afin de prouver l'efficacité de LATMAC, nous devons le comparer avec SMAC. Nous avons utilisé SMAC avec une valeur de duty cycle égale à « 10 » et LATMAC avec valeur de latence minimale LATMIN = 0 et latence maximale LATMAX = 100 (LATTime =1 seconde). La figure 4.26 illustre les résultats obtenus.

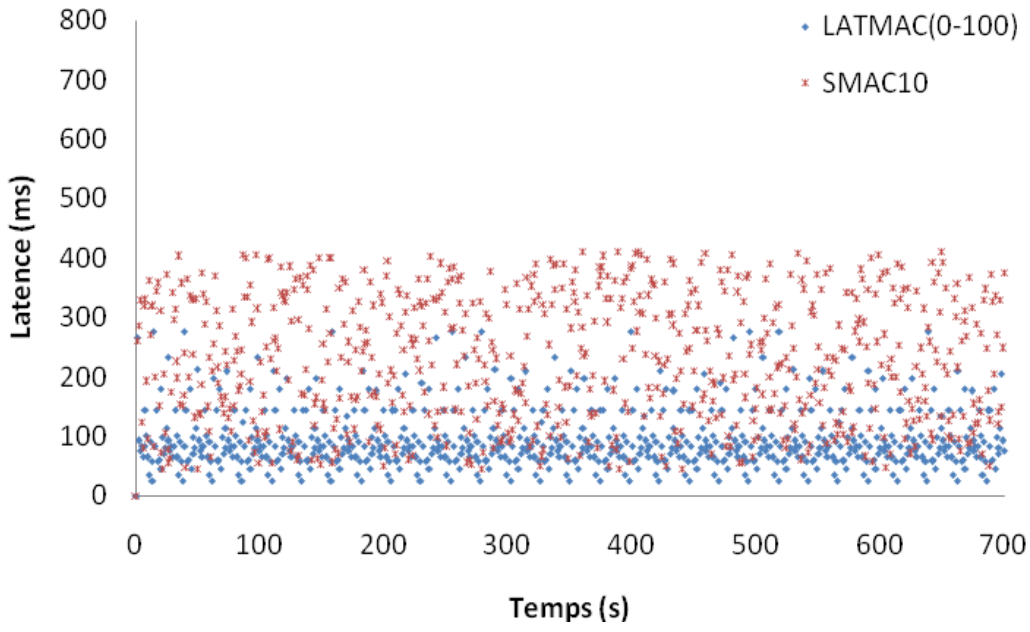


Figure 4.26. Comparaison entre latence LATMAC (0,100) et SMAC 10

Nous observons que LATMAC (0,100) est très efficace en termes de latence par rapport aux SMAC.

Ainsi, nous avons réalisé une simulation entre LATMAC avec valeur de latence minimale LATMIN = 0 et latence maximale LATMAX = 100 et entre LATMAC avec valeur de latence minimale LATMIN = 100 et latence maximale LATMAX = 150. Nous observons que LATMAC suit très bien les valeurs définies par son utilisateur (Figure 4.27).

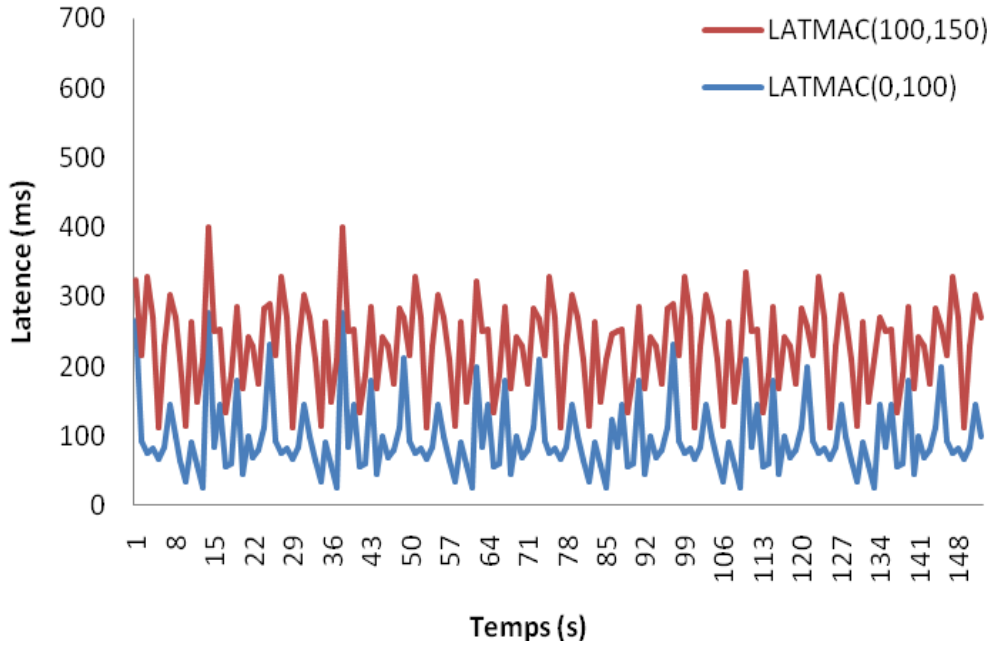


Figure 4.27. Latence sous LATMAC (0,100) et LATMAC (100,150)

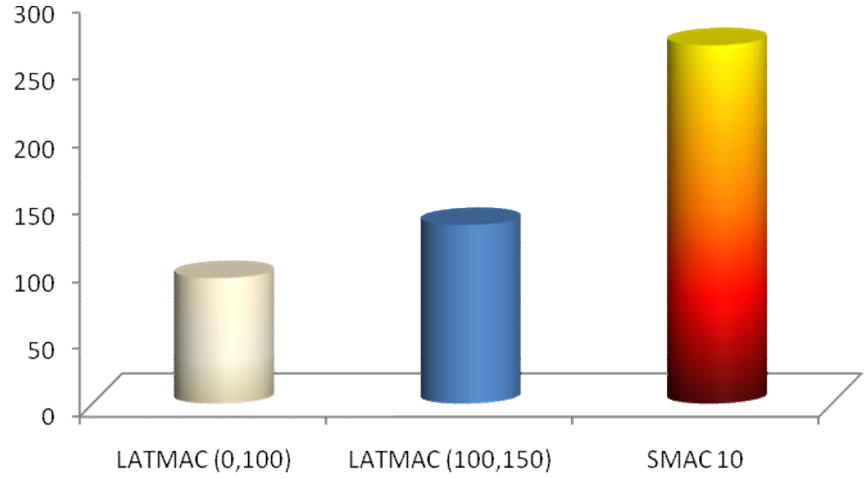


Figure 4.28. Comparaison de la latence moyenne

En discutant de l'énergie, LATMAC conserve l'énergie aussi efficacement que SMAC si c'est pas un peu mieux.(Figure 4.29)

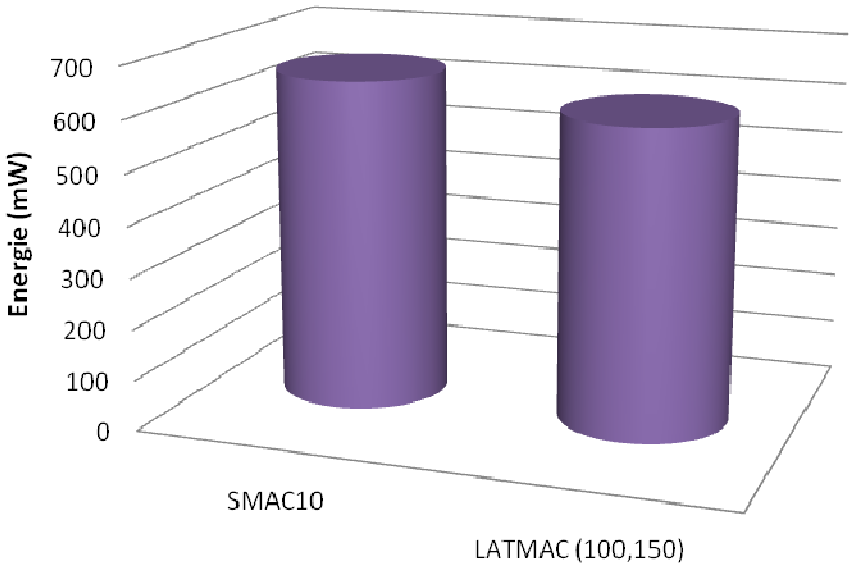


Figure 4.29. L'énergie consommée sous SMAC 10 et LATMAC (100,150)

4.6. Conclusion

Le premier volet de notre étude a eu pour objectif l'implémentation de SMAC et sa comparaison avec le MAC 802.11. Les résultats obtenus montrent que le modèle SMAC répond mieux au critère de conservation d'énergie que le modèle MAC 802.11. En plus SMAC offre une durée de vie de réseau plus longue et préserve les capacités du réseau.

Le second volet a eu pour but d'étudier l'impact de facteur « duty cycle » et « rate » sur SMAC. Les résultats obtenus montrent qu'un « duty cycle » de l'ordre de 10% offre de meilleure conservation d'énergie avec n'importe quel trafic de réseau.

Pour cela, nous avons proposé un nouveau protocole LATMAC, qui conserve l'énergie de la même efficacité que SMAC et en même temps il prend en considération la latence.

CONCLUSION GENERALE & PERSPECTIVES

CONCLUSION GENERALE & PERSPECTIVES

SMAC est un protocole de conservation d'énergie pour les réseaux de capteurs qui réduit nettement la consommation d'énergie dans le réseau sans diminuer significativement la connectivité du réseau ni ses capacités.

SMAC présente la notion du sommeil et d'écoute périodique. Dans la période d'écoute, le noeud se réveille pour communiquer avec les autres nœuds. Quand la période de sommeil arrive, les nœuds essayent de dormir en désactivant leurs radios.

Quand un nœud reçoit un paquet de données du nœud précédent, il ne peut pas transmettre à son tour le paquet au nœud suivant au même moment. Il doit attendre l'arrivée de la prochaine période d'écoute du nœud voisin. Donc, il existe un potentiel retard sur chaque saut ce qui augmente la latence.

Les résultats obtenus montrent que le modèle SMAC répond mieux au critère de conservation d'énergie que le modèle MAC 802.11. En plus SMAC offre une durée de vie de réseau plus longue et préserve les capacités du réseau.

Ainsi, les résultats obtenus montrent que SMAC avec un minimum de paramètre « duty cycle » offre une meilleure conservation d'énergie avec n'importe quel trafic de réseau, mais perturbe à un certain niveau sa latence.

Pour cela, nous avons proposé un nouveau protocole LATMAC, qui conserve l'énergie de la même efficacité que SMAC et en même temps il prend en considération la latence.

Notre protocole LATMAC résout ce problème et diminue la latence en fixant seuil minimal et un seuil maximal de latence. Les nœuds gèrent localement la durée de veille en modifiant la valeur du paramètre « duty cycle » à l'aide des échanges de messages. Ce protocole minimise la consommation d'énergie de la même efficacité que SMAC et diminue la latence considérablement.

Comme perspectives, nous proposons de faire des expériences de LATMAC sur des cas réels comme la communication entre voitures dans une autoroute, et des cas médicales où la contrainte d'énergie et la latence sont des facteurs essentiels importants.

REFERENCES

Références

[1] S. Singh, M. Woo, and C. S. Raghavendra, "Power-aware Routing in Mobile Ad Hoc Networks," Proc. MobiCom '98, Dallas, TX, Oct. 1998.

[2] S. Singh, M. Woo and C. S. Raghavendra, "Power aware routing in mobile ad hoc networks" , in Proceedings of ACM MOBICOM, 1998, pp. 181-190.

[3] M. W. Subbarao, "Dynamic power-concvious routing for manets: An initial approach, " in IEEE Vehicular Technology Confernece, 1999, pp. 1232-1237.

[4] Q. Li, J. Aslam, and D. Rus, "Online power-aware routing in wireless ad-hc networks," in Proceedings of the Seventh Annual International Confernece on Mobile Computing and Networking, July 2001, pp. 97-107.

[5] R. Dube, C. D. Rais, K.-Y. Wang, and S. K. Tripathi, "Signal stability based adaptive routing (SSA) for ad hoc mobile networks," in IEEE Personal Communications, 1997.

[6] R. Prim, “Shortest connection networks and some generalizations,” The

[7] Bell System Technical Journal, vol. 36, pp. 1389–1401, 1957.

[8] P. Latteieri, C. Schurgers, and M. B. Srivastava, "Adaptive Link Layer Strategies for Energy Efficient Wireless Networking" ACM WINET.

[9] A. Alwan et al., "Adaptive Mobile Multimedia Networks," IEEE Pers. Commun., Apr. 1996.

[10] Swetha Narayanaswamy, Vikas Kawadia, R. S. Sreenivas and P. R. Kumar, "Contrôle de la puissance dans les réseaux ad hoc : La théorie, l’architecture, l’algorithme et l’implémentation du protocole COMPOW", University of Illinois, 1308 West Main Street, Urbana, IL 61801, USA

[11] Vikas Kawadia and P. R. Kumar, "Power Control and Clustering in Ad Hoc Networks", University of Illionois at Urbana-Champaign, 1308 West Main St. Urbana, IL-61801

[12] Géraud Allard, Pascale Minet, Dang-Quan Nguyen, Nirisha Shrestha,"Evaluation of the Energy Consumption in MANET", Institut National de Recherche en informatique et en automatique, Février 2006

[13] Ning Li, Jennifer C. Hou, and Lui Sha, "Design and Analysis of an MST-Based Topology Control Algorithm" IL- 61801 nli, jhou, lrs}@cs.uiuc.edu

[14] Benjie Chen, Kyle Jamieson, Hari Balakrishnan, and Robert Morris , "Span: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks" , Cambridge, Massachusetts

[15] Javier Gomez, Andrew T. Campbell, Mahmoud Naghshineh, Chatschik Bisdikian "Conserving Transmission Power in Wireless Ad Hoc Networks" Columbia University, NY 10027, USA

[16] C.-K. Toh, "Maximum Battery Life Routing to Support Ubiquitous Mobile Computing in Wireless Ad Hoc Networks", Georgia Institute of Technology

[17] ABBARA Farid, BERTAUD David ET CELIK Ramazan , "Simulation et test de réseaux Ad-Hoc à topologie contrôlée", Université d'Avignon, 26/05/2005

[18] P.-J. Wan , G. Calinescu , X. –Y. Li , et O. Frieder, “Minimum Energy Broadcast Routing in Static Ad Hoc Wireless Networks”

[19] HANEN IDOUDI, WAFA AKKARI, ABDELFATTEH BELGHITH, MIKLÓS MOLNÁR, "ALTERNANCE SYNCHRONE POUR LA CONSERVATION D’ÉNERGIE DANS LES RÉSEAUX AD HOC" , IRISA , Novembre 2006

[20] ANSI/IEEE Standard 802.11, "Wireless LAN Medium, Access Control (MAC) and Physical, Layer (PHY) Specifications", 1999

[21] Changsu Suh, Young-Bae Ko and Jai-Hoon Kim, "Enhanced Power Saving for IEEE 802.11 WLAN with Dynamic Slot Allocation," LNCS, Vol. 3794, 2005, pp. 466-477.

[22] A. Belghith et W. Akkari, "Traffic Aware Power conservation mechanism for ad hoc networks", soumis au International Journal of Computing and Information Sciences (IJCIS), Canada, 2006.

[23] Ye W., Heidemann J., et Estrin D., "An Energy-Efficient MAC Protocol for Wireless Sensor Networks", Actes de la conférence IEEE INFOCOM 2002, Juin 2002.

[24] Van Dam T. et Langendoen K., "An Adaptive Energy-Efficient MAC Protocol for Wireless Sensor Networks ", Actes de la conférence ACM SenSys 2003, November 2003.

[25] Rabaey J. M., Ammer M. J., da Silva Jr. J. L., Patel D., and Roundy S., " PicoRadio Supports for Ad Hoc Ultra-Low Power Wireless Networking ", Actes de la conférence IEEE Computer, Juillet 2000.

[26] Wei Ye, John Heidemann, Deborah Estrin, “An Energy-Efficient MAC protocol for Wireless Sensor Networks”, USC/ISI TECHNICAL REPORT ISI-TR-543, September 2001

[27] Y. Li, W. Ye and J. Heidemann. Energy and latency control in low duty cycle MAC protocols. USC/ISI Technical Report ISI-TR-595, August 2004.

[28] “Smac-users -- Discussions by users of S-MAC web page” ,http://mailman.isi.edu/mailman/listinfo/smac-users

[29] “Energy Model Update in ns-2 web page” , http://www.isi.edu/ilense/software/smac/ns2_energy.html

[30] PARK Wongil; PARK Sangjoon; JANG Yoonchul ; KIM Kwanjoong ; KIM Byunggi ; “Enhancing connectivity based on the thresholds in mobile ad-hoc networks”, First international conference, HPCC 2005, Sorrento, Italy, September 21-23, 2005

Résumé

Un réseau ad hoc est une collection de terminaux mobiles interconnectés par une technologie sans fil formant un réseau temporaire sans aucune administration ou infrastructure fixe. L'un des objectifs majeurs pour ces réseaux consiste à ce que les terminaux mobiles soient utilisés au maximum « n'importe où et n'importe quand ». Cependant, l'une des grandes limitations de cet objectif concerne le support énergétique. Cette contrainte est beaucoup plus importante dans les réseaux ad hoc, où les terminaux consomment leur propre énergie en routant des données pour d'autres terminaux. Ainsi, la consommation d'énergie devrait être une question cruciale lors de la conception de nouveaux protocoles de communication.

Dans notre étude, nous présentons les différents protocoles destinés à la minimisation de la consommation d'énergie d'un réseau ad hoc et de capteurs; Puis, nous étudions en détails le protocole SMAC, et nous concluons notre travail par des simulations sous NS, pour prouver l'efficacité de SMAC par rapport 802.11 et l'effet de la modification de paramètre « Duty Cycle » sur l'efficacité du protocole. Nous avons aussi proposé un nouveau protocole LATMAC, qui conserve l'énergie de la même efficacité que SMAC mais qui prend en considération la latence.

Mots clés : Ad Hoc, senseur, mobile, énergie, capteur, LATMAC, latence