الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire
وزارة التعليم العالي والبحث العلمي
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

**Université Abbes Laghrour KHENCHELA**
Faculté des Sciences et Technologies
Département d'Informatique et Mathématiques

**Ecole Doctorale des Sciences et Technologies d'Information et de Communication (STIC)**
**Pôle : Khenchela**

**Mémoire pour l'obtention du diplôme de magistère en informatique**
**Option : Informatique Repartie et Mobile**

**Thème**

# Interaction Robuste et Efficace des Systèmes Autonomes et Hétérogènes

**Réalisé par**

MESSIKH Chouaib

**Membres de jury**

Dr. MERAH El-Kamel – Université de Khenchela : **Président**
Dr. MELKEMI Kamel Eddine – Université de Biskra : **Examinateur**
Dr. SEGHIR Rachid – Université de Batna : **Examinateur**
Dr. CHALLAL Yacine – Ecole Nationale Supérieure de l'informatique : **Rapporteur**
Dr. BACHIR Abdelmalik – Université de Biskra : **Rapporteur**

Promotion : 2012 / 2013

# Dédicaces

*A ma famille*

*A mes amis*

*A toute personne ayant un impact sur ma vie*

*Je dédie ce travail.*

# Remerciements

Louange à Allah, que le salut et la bénédiction soient sur son Prophète.

Je remercie avant tout Allah le tout puissant qui m'a donné le courage et la force pour continuer ce travail.

Je présente mes remerciements et gratitudes à ma famille, qui m'a largement soutenu durant tout mon cursus.

Mes sincères remerciements vont à mes encadrants, Mr Challal Yacine et Mr Bachir Abdelmalik, qui m'ont accompagné durant ces années de mémoire, avec leurs conseils, leur aide et leurs encouragements précieux.

Merci à vous, mes collègues de STIC, les membres d'équipe de recherche SURES de l'ESI, mes camarades du lycée, mes amis de la cité universitaire pour m'avoir aidé et encouragé durant la réalisation du travail.

Je remercie très vivement les membres de jury pour me faire l'honneur de bien vouloir juger mon travail.

Je tiens à remercier tous ceux qui ont contribué du près ou du loin à la réalisation de ce travail.

# Résumé

A travers les développements récents des technologies sans fil, nous constatons une évolution rapide vers un écosystème technologique complexe où l'hétérogénéité devient une caractéristique récurrente à tous les niveaux. En particulier, l'hétérogénéité des besoins applicatifs (temps réel, bande passante, QoS, autonomie, portée radio, etc.) et les contraintes technologiques (énergie, spectre, …) ont engendré divers standards de communication (WiFi, ZigBee, Bluetooth, WiMax, 3G/LTE, etc.). Ces standards hétérogènes couplés à l'hétérogénéité des systèmes communiquant, rendent leur interaction complexe, coûteuse (implémentation matérielle d'interfaces multiples) et parfois inefficace (connectivité faible, interférences, affaiblissement du signal, etc.).

Ce projet a pour objectif de développer une architecture d'interaction robuste et efficace des systèmes autonomes hétérogènes. Cette architecture sera basée sur une radio intelligente adaptative prenant en compte le contexte des interactions et son évolution dans le temps : besoins applicatifs (temps réel, QoS, sécurité, …), état d'utilisation de spectre, paramètres externes ayant un impact avéré sur la qualité des transmissions radio (présence de champ électromagnétique, température, humidité, etc.)

# خلاصة

من خلال التطورات الأخيرة في تقنيات الاتصال اللاسلكي، نلاحظ التطور السريع نحو أنظمة تكنولوجية معقدة. حيث يصبح عدم التجانس سمة متكررة على جميع المستويات. على وجه الخصوص، تنوع المتطلبات التطبيقية (الآنية، النطاق الترددي، جودة الخدمة، اتساع نطاق الراديو، الخ) والقيود التكنولوجية (الطاقة، مجال الترددات المتاح، ...) سببان رئيسيان في انتاج معايير الاتصال المختلفة الحالية (واي فاي، زيجبي، بلوتوث، واي ماكس، الخ). هذه المعايير غير المتجانسة إلى جانب عدم تجانس أنظمة الاتصال، جعلوا التواصل معقدا، مكلفا (تصنيع الأجهزة متعددة معايير الاتصال) وفي بعض الأحيان غير فعالا (رداءة الاتصال ، تداخل التوترات، فقدان الإشارة، وما إلى ذلك).

يهدف هذا المشروع إلى تطوير بنية تواصل قوية وفعالة بين الأنظمة ذاتية القرار وغير المتجانسة. تستند هذه البنية على تكنولوجيا الراديو الذكية، مع مراعاة سياق التواصل وتطوره عبر الزمن: من متطلبات تطبيقية (الآنية، وجودة الخدمة، والأمن، الخ)، وحالة استخدام مجال الترددات، والعوامل الخارجية التي أثبت تأثيرها على جودة الإرسال بالراديو (وجود المجال الكهرومغناطيسي، ودرجة الحرارة، والرطوبة، الخ)

# Abstract

Through recent developments in wireless technologies, we see a rapid evolution towards a complex technological ecosystem where heterogeneity becomes a recurring feature at all levels. In particular, the diversity of application requirements (real time, bandwidth, QoS, range, radio range, etc.) and the technological constraints (energy, spectrum, ...) have produced various communication standards (WiFi, ZigBee, Bluetooth, WiMax, 3G / LTE, etc.). These heterogeneous standards coupled with the heterogeneity of communicating systems, make their complex interaction, expensive (hardware implementation of multiple interfaces) and sometimes inefficient (low connectivity, interference, signal loss, etc.).

This project aims to develop a robust and efficient interaction architecture for heterogeneous autonomous systems. This architecture will be based on an adaptive intelligent radio taking into account the context of interactions and its evolution over time: application requirements (real time, QoS, security, ...), spectrum usage status, external parameters that have a proven impact on the quality of radio transmissions (presence of electromagnetic field, temperature, humidity, etc.)

# Liste des figures

**Figure 1 :** Une architecture typique d'un système de communication sans fil dans la pratique. ... 1
**Figure 2 :** hétérogénéité basée sur la combinaison de l'architecture Ad-hoc et l'architecture hiérarchique des réseaux cellulaires. [1] ... 2
**Figure 3 :** L'impact de la température sur la réception [3] ... 3
**Figure 4 :** Architecture qui tient en compte les besoins applicatifs des nœuds. ... 4
**Figure 5 :** L'architecture générale d'une radio logicielle. [4] ... 7
**Figure 6 :** L'émetteur-récepteur de la radio logicielle. [11] ... 8
**Figure 7 :** Une superhétérodyne radio logicielle – côté réception. [10] ... 8
**Figure 8 :** Une radio logicielle avec conversion directe – côté transmission. [10] ... 9
**Figure 9 :** Un récepteur d'une radio logicielle avec une démodulation en quadrature. [11] ... 10
**Figure 10 :** Emplacement de la radio cognitive par rapport à l'ensemble des radios logicielles. 16
**Figure 11 :** Schéma général de l'architecture de la radio cognitive. ... 17
**Figure 12 :** Cycle de la cognition [17]. ... 18
**Figure 13 :** Conceptions entre-couches. [47] ... 27
**Figure 14 :** Plateforme cognitive. [47] ... 28
**Figure 15 :** Common Control Channel. [20] ... 31
**Figure 16 :** Split Phase. [20] ... 31
**Figure 17 :** Frequency Hopping Sequence. [20] ... 32
**Figure 18 :** Schéma général du réseau logiciel. [51] ... 38
**Figure 19 :** Architecture du réseau logiciel. [51] ... 39
**Figure 20 :** Exemple du réseau de deux nœuds avec un conflit de besoins. ... 58
**Figure 21 :** Exemple de calcul de distance. ... 60
**Figure 22 :** Architecture générale du modèle centralisé. ... 61
**Figure 23 :** Les tâches du contrôleur. ... 62
**Figure 24 :** Une représentation géométrique des gains. ... 70
**Figure 25 :** Schema general du noeud. ... 72
**Figure 26 :** Representation du quantum. ... 73
**Figure 27 :** Structure du paquet CCC. ... 73
**Figure 28 :** Structure du paquet de données. ... 74
**Figure 29 :** Organigramme de transmission de données. ... 77
**Figure 30 :** Organigramme de réception de données ... 78

**Figure 31 :** Topologie réseau utilisée dans la simulation. [53] ..... 81
**Figure 32 :** Evolution du débit dans le temps. ..... 83
**Figure 33 :** L'énergie moyenne consommé par les nœuds.. ..... 84
**Figure 34 :** Niveau de sécurité moyen assuré pour les paquets envoyés. ..... 85
**Figure 35 :** Nombre moyen d'échecs de transmission par second. ..... 85
**Figure 36 :** Les couts de performance pour le test 2. ..... 86
**Figure 37 :** Evolution du nombre d'échecs dans le temps. ..... 88
**Figure 38 :** Evolution du débit, énergie et taux des échecs dans le temps pour le modèle centralisé. ..... 89
**Figure 39 :** Evolution du débit, énergie et taux des échecs dans le temps pour le modèle distribué. ..... 89

# Liste des tableaux

**Table 1 :** Applications réalisées pour les réseaux logiciels. [51] ..... 42
**Table 2 :** Les vecteurs de danger trouvés dans les réseaux logiciels. [51] ..... 45
**Table 3 :** Les types d'attaques sur l'utilisation d'OpenFlow. [51]..... 46
**Table 4 :** Deroulement de la methode de classement proposee..... 70
**Table 5 :** Liste des canaux utilisés..... 81
**Table 6 :** Résultats de la simulation lorsque des noeuds demandent plus de débit. ..... 83
**Table 7 :** Résultats de la simulation lorsque les noeuds demandent moins de consommation d'énergie. ..... 86
**Table 8 :** Résultats de la simulation lorsque les nœuds demandent une communication sécurisée. ..... 87
**Table 9 :** Résultats de la simulation lorsque les nœuds demandent une transmission sans erreurs. ..... 87
**Table 10 :** Résultats de la simulation lorsque les nœuds ont un besoin équitable entre les critères. ..... 88
**Table 11 :** Résultats des deux modèles lorsque les besoins sont aléatoirement attribués aux nœuds. ..... 89
**Table 12 :** Les moyennes des performances des modeles dans les 6 tests..... 90

# Table des matières

**Dédicaces** ..... **i**
**Remerciements** ..... **ii**
**Résumé** ..... **iii**
**خلاصة** ..... **iv**
**Abstract** ..... **v**
**Liste des figures** ..... **vi**
**Liste des tableaux** ..... **viii**
**Table des matières** ..... **ix**
**Introduction générale** ..... **1**
**Chapitre 1 : Généralités sur les réseaux cognitifs** ..... **5**
Introduction ..... 5
1. Radio logicielle ..... 5
1.1. Historique ..... 5
1.2. Importance de la radio logicielle ..... 6
1.3. Architecture de la radio logicielle ..... 7
1.3.1. Fréquence Radio ..... 8
1.3.2. Conversion analogique / numérique ..... 10
1.3.3. Traitement de bande de base ..... 11
1.3.4. Traitement de données ..... 11
1.3.5. Programmation des radios logicielles ..... 11
1.4. Défis de la radio logicielle ..... 12
1.4.1. Défis liés à la performance ..... 12
1.4.2. Défis liés à la sécurité ..... 12
1.4.3. Défis liés à la certification ..... 13
1.4.4. Défis liés à la programmation ..... 13
2. Radio cognitive ..... 14
2.1. Critères et caractéristiques de la radio cognitive ..... 14
2.1.1. Sensibilité au contexte (Awareness) ..... 14
2.1.2. Adaptabilité ..... 15
2.1.3. Cognition ..... 15
2.1.4. Définition de la radio cognitive ..... 15

2.2. Architecture de la radio cognitive ....17
2.2.1. Architecture physique ....17
2.2.2. Architecture logique ....18
2.3. Applications de la radio cognitive ....19
2.3.1. Gestion des ressources de la radio ....19
2.3.2. Gestion du réseau ....19
2.3.3. Livraison des services ....20
2.3.4. Certification ....20
3. Réseau cognitif ....21
3.1. Définition du réseau cognitif ....21
3.2. Caractéristiques des réseaux cognitifs ....22
3.2.1. Cognition ....23
3.2.2. Coopération ....23
3.2.3. Individualité des nœuds ....23
3.2.4. Accès à une bande spectrale plus large ....23
3.3. Architecture globale des réseaux cognitifs ....23
3.3.1. Réseau primaire ....24
3.3.2. Réseau secondaire ....24
3.4. Architecture conceptuelle des réseaux cognitifs ....25
3.4.1. Conception entre couche ....25
3.4.2. Plateforme cognitive ....27
3.5. Protocoles MAC et NET pour les réseaux cognitifs ....28
3.5.1. Protocoles MAC pour le réseau cognitive ....28
3.5.2. Protocoles de Routage Pour La Radio Cognitive ....35
4. Réseau logiciel ....36
4.1. Définition du réseau logiciel ....36
4.1.1. Niveau de données ....36
4.1.2. Niveau de control ....37
4.1.3. Niveau de gestion ....37
4.2. Architecture du réseau logiciel ....38
4.2.1. Couche infrastructure ....39
4.2.2. Couche interface Southbound ....39
4.2.3. Couche hyperviseur du réseau ....40
4.2.4. Couche système d'exploitation du réseau ....40
4.2.5. Couche interface Northbound ....41
4.2.6. Couche virtualisation par langage ....41
4.2.7. Couche langage de programmation ....42

4.2.8. Couche application réseau ..... 42
4.3. Défis des réseaux logiciels ..... 44
Conclusion ..... 46
**Chapitre 2 : Optimisation multicritère des réseaux cognitifs ..... 48**
Introduction ..... 48
1. Travaux basés sur les méthodes non-Pareto ..... 48
2. Travaux basés sur les méthodes Pareto ..... 50
Conclusion ..... 52
**Chapitre 3 : Interaction robuste et efficace des systèmes autonomes et hétérogènes à base d'infrastructure réseau cognitive ..... 53**
Introduction ..... 53
1. Spécifications générales ..... 54
2. Modèle centralisé ..... 55
2.1. Modélisation du problème ..... 55
2.1.1. Distribution des canaux ..... 55
2.1.2. Calcul des table de routage ..... 59
2.2. Intégration de la solution ..... 61
3. Modèle distribué ..... 62
3.1. Spécifications ..... 63
3.2. Calcul des actions permises $A_n$ ..... 64
3.3. Choix de l'action ..... 64
3.3.1. Exploration ..... 65
3.3.2. Exploitation ..... 68
3.4. Schéma du nœud ..... 71
3.5. Algorithmes des évènements ..... 73
3.5.1. Transmission du paquet CTRL ..... 74
3.5.2. Réception du paquet CTRL ..... 75
3.5.3. Traitement des paquets CTRL ..... 75
3.5.4. Choix de l'action ..... 75
3.5.5. Transmission des paquets de données ..... 76
3.5.6. Réception des paquets de données ..... 77
Conclusion ..... 78
**Chapitre 4 : Simulation et résultats ..... 80**
Introduction ..... 80
1. Outils utilisés ..... 80
2. Conditions de l'expérimentation ..... 80
3. Résultats obtenus ..... 82

3.1. Test 1 : Importance totale au débit ..... 82
3.2. Test 2 : Importance totale à la consommation d'énergie ..... 85
3.3. Test 3 : Importance totale à la sécurité ..... 87
3.4. Test 4 : Importance totale aux interférences ..... 87
3.5. Test 5 : Importance équitable entre les critères ..... 88
3.6. Test 6 : Besoins aléatoires des nœuds ..... 89
Conclusion ..... 90
**Conclusion générale ..... 92**
1. Conclusion ..... 92
2. Perspectives ..... 93
3. Orientation ..... 93
**Acronymes ..... 94**
**Bibliographie ..... 97**

# Introduction générale

L'utilisation de la communication sans fil ne cesse d'augmenter. A cause de la facilité de la connexion et la liberté du mouvement donnée à son utilisateur, la communication sans fil devient de plus en plus un facteur majeur pour le développement technologique.

Aujourd'hui, L'interaction à travers la technologie sans fil joue un rôle nécessaire dans la création des espaces intelligents (Smart Home, Smart City, Smart Health, …). La diversité des besoins applicatifs (Bande passante, énergie, QoS, …) pour ce type de systèmes autonomes a engendré le développement de plusieurs standards de communication sans fil comme Wifi, Bluetooth, ou Wi Max.

Cette diversité de solutions a permis de développer des systèmes hétérogènes beaucoup plus complexes, où les nœuds doivent être compatibles avec les différents standards. La figure 1 illustre l'implémentation courante des différents standards hétérogènes sur les dispositifs multifonctionnels.


**Figure 1 :** *Une architecture typique d'un système de communication sans fil dans la pratique.*

De plus, le besoin d'optimisation de l'interaction dans les réseaux mobiles a causé d'autres faces d'hétérogénéité, comme la combinaison de l'ad-hoc avec les topologies structurelles [1], ou comme le remplacement d'une technologie par une autre pour améliorer un même service [2].

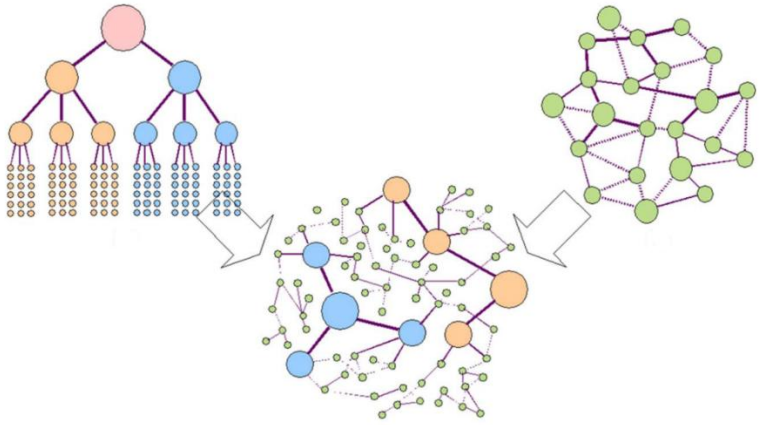


**Figure 2 :** *hétérogénéité basée sur la combinaison de l'architecture Ad-hoc et l'architecture hiérarchique des réseaux cellulaires. [1]*

Toute cette cohabitation et combinaison des solutions dans un même réseau, le rend moins performant et plus complexe. De plus, cette complexité est doublée, spécialement pour les réseaux sans fil mobiles. D'une part, un nœud mobile peut changer son environnement à chaque instant, donc il est fort probable qu'il y aurait une incompatibilité entre le standard de la communication utilisé par le nœud et le standard utilisé par une partie de son environnement (à cause de l'hétérogénéité du réseau). Le réseau peut être plus performant si le nœud peut communiquer par n'importe quel standard.

Par ailleurs, les interférences radio causent une dégradation de la qualité des transmissions. Ces interférences sont souvent dues à la saturation des bandes de fréquence qui sont aujourd'hui une ressource rare. L'usage de ces bandes de fréquence et leur degré de saturation est également hétérogène dans le temps et l'espace : au moment où certaines bandes sont saturées à cause du nombre de communications sur cette même bande, d'autres bandes de fréquences se voient sous exploitées. L'idéal serait qu'une radio puisse détecter ces bandes sous-exploitées pour émettre dessus et décharger ainsi les bandes les plus sollicitées et minimiser les interférences pour une meilleure qualité des transmissions.

La figure 3 montre aussi l'impact de la température sur la communication. Il est très utile donc d'avoir une radio qui ajuste sa force de transmission selon la température (ou d'autres perturbateurs naturels).

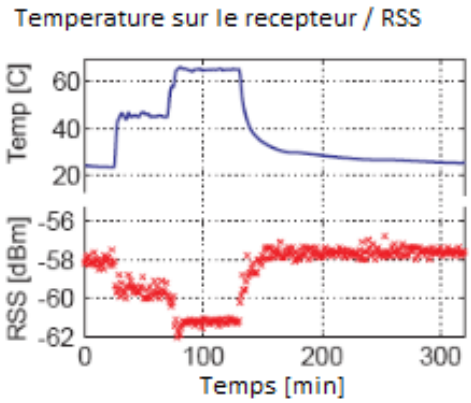



**Figure 3 :** *L'impact de la température sur la réception [3]*

Pour pallier cette complexité des réseaux sans fils, les nœuds doivent être flexibles et qu'ils peuvent contrôler l'utilisation des différents standards, la force utilisée pour la transmission, la consommation de leur énergie, et le partage des bandes de fréquences. D'où l'invention de la radio logicielle et plus particulièrement la radio cognitive ou la radio sensible au contexte.

Cette invention a ouvert les yeux des chercheurs vers d'autres possibilités. Au lieu de se contenter à un seul nœud, pourquoi ne pas rendre tout le réseau cognitif ? Un réseau qui peut prédire les changements qui vont se produire dans son environnement et agir en changeant sa structuration pour répondre à ces changements, est le futur de la communication sans fil.

Malgré tous les efforts faits dans cette filière de recherche et d'autres, il reste encore beaucoup de challenges à surmonter. Un de ces challenges, et qui concerne l'objectif fondamental de la création de la radio cognitive, est la recherche de la meilleure architecture pour les réseaux cognitifs *en tenant compte de tous les besoins applicatifs des nœuds* (Figure 4). Une telle architecture va permettre le passage de la technologie d'aujourd'hui vers la technologie de future basée sur la radio cognitive.

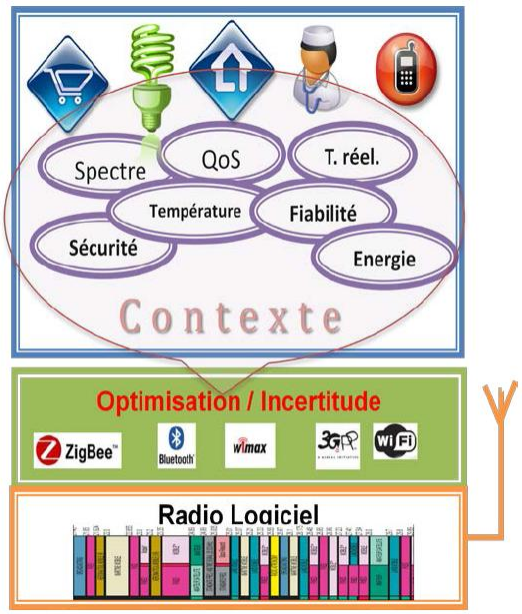



**Figure 4 :** *Architecture qui tient en compte les besoins applicatifs des nœuds.*

Dans ce projet, nous nous intéressons plus au contexte multicritère qui représente les besoins applicatifs des nœuds dans un réseau. Nous essayons donc de proposer une architecture qui permet une interaction efficace entre les nœuds d'un réseau de radio cognitive, en tenant compte de leurs différents besoins applicatifs. Nous proposons deux modèles : Une solution centralisée basée sur une technologie appelée « réseau logiciel », et une autre distribuée et basée sur la théorie des jeux, plus spécifiquement sur l'apprentissage par renforcement.

Le rapport est divisé en quatre chapitres. Le premier chapitre introduit des généralités sur les technologies qui entourent ce projet. Notamment, la radio cognitive, le réseau cognitif, et les concepts de solutions utilisés dans la modélisation de ces réseaux. Le deuxième chapitre propose une classification des travaux faits concernant les architectures des réseaux à radio cognitive qui prennent en compte le contexte multicritère. Dans le troisième chapitre, on introduit les solutions. Dans le dernier chapitre, on présente notre expérimentation sur les deux solutions proposées et on discute les résultats. Et on finit par des conclusions et de ce qui reste à faire dans le futur.

# Chapitre 1 :
# Généralités sur les réseaux cognitifs

## Introduction

La base des réseaux cognitifs est la radio intelligente qui, quant à elle, est basée sur la radio logicielle. Ce chapitre a pour objectif d'introduire ces trois concepts. De plus, on introduit le concept des réseaux logiciels. Il est à noter que bien que la radio intelligente est dérivée de la radio logicielle, les concepts du réseau cognitif et réseau logiciel n'ont pas la même relation, et ces deux concepts sont totalement séparés.

## 1. Radio logicielle

Le besoin de la communication entre les différents équipements radio est uniquement résolu par l'utilisation d'une radio logicielle programmable qui aura une architecture plus flexible. La radio logicielle (Software-Defined Radio) se réfère à une communication sans fil dans laquelle la modulation de l'information transmise et la démodulation de l'information reçue sont définies par un programme ou un logiciel. En d'autres termes, c'est une radio qui peut utiliser n'importe quel standard pour communiquer. L'objectif principal de la radio logicielle est d'essayer de remplacer la majorité des composants analogiques et des circuits intégrés à très grande échelle d'un émetteur-récepteur par des composants programmables [4].

### 1.1. Historique

Le terme « Radio logicielle » (Software Radio en anglais) n'est pas nouveau dans le domaine informatique. Il est apparu en 1984, par un groupe de recherche dans la division Garland Texas du Raytheon, pour nommer leur travail sur un récepteur digital [5].

En 1992, Joseph Mitola, connu comme le père de la radio logicielle, a publié un article titré « Software Radio: Survey, Critical Analysis and Future Directions » où il a défini plus proprement la radio logicielle comme un méta-modèle qui combine les primitives du traitement numérique du signal en des fonctions systèmes de communication et un groupe de processeurs qui vont être la hôte de la radio logicielle et qui peuvent manipuler ces fonctions [6]. Contrairement au groupe du Garland, le modèle décrit est conçu pour la réception et la transmission, alors que l'ancien n'était qu'un récepteur.

Entre cette année et 1995, un projet militaire américain a été développé pour rendre l'interaction possible entre les différents standards utilisés par les divisions de l'armée, et qui utilise comme principe l'idée du Mitola. Ce projet qui porte le nom « SPEAKeasy », dans cette période, devrait être capable de transmettre et recevoir les signaux entre 2 et 2000 MHz. En Aout 1994, la première phase du projet était soutenue devant le gouvernement américain, et il a été montré qu'un tel projet est réalisable, en testant le modèle avec deux réseaux qui utilisent deux standards différents. La radio logicielle créée a servi comme un pont entre un réseau HAVE QUICK et un réseau SINCGARS [7]. La deuxième phase n'était qu'un essai réussi d'augmenter la performance du SPEAKeasy.

En 1996, le forum scientifique qui s'intéresse à cette innovation a été créé. Il a été nommé « The Modular Multifunction Information Transfer System (MMITS) Forum ». En 1998 le nom est changé en « SDR Forum » (SDR pour Software-Defined Radio). Aujourd'hui, et depuis 2010, ce forum porte le nom « Wireless Innovation Forum » [8].

En 2004, la commission fédérale des communications (FCC) a donné son premier accord concernant la commercialisation de la première radio logicielle développée par Vanu Inc. et qui s'appelle Anywave [9]. Cette radio peut communiquer en CDMA (le standard utilisé à l'Asie et l'Amérique du Nord) et en GSM (le standard utilisé en Europe et le reste du monde).

## 1.2. Importance de la radio logicielle

Dans le secteur militaire, où il est nécessaire de prolonger la durée de vie des services des systèmes de communication, l'utilisation des radios logicielles devient très importante ; car elles permettent de mettre à jours les services existants ou ajouter de nouveaux services au même équipement. Dans le domaine commercial, et avec le développement rapide de la technologie de la communication, remplacer toute une station de base par une autre devient une opération couteuse pour les opérateurs de la communication cellulaire. La radio logicielle dans ce cas aussi devient très intéressante par sa flexibilité. L'importance de la radio logicielle se montre aussi dans sa capacité de reproduire des applications déjà existantes et d'exploiter de nouvelles applications qui n'ont pas pu exister avec les radios traditionnelles. Un exemple de ces applications est l'utilisation des radios sensibles au contexte et qui peuvent s'adapter à leurs environnements [14].

La recherche sur la radio logicielle a été principalement motivée par les problèmes d'interopérabilité entre les radios installées sur des supports physiques dédiés [15]. Pour le secteur militaire, l'anticipation des attaques des terroristes et des ennemies lors d'un conflit est une étape très importante pour mettre des plans effectifs. Avoir un équipement capable

d'intercepter toutes les fréquences et tous les standards de communication offre une forte motivation de recherche sur la radio logicielle [10].

## 1.3. Architecture de la radio logicielle

Bien qu'on parle de la radio logicielle depuis 1984, les chercheurs n'ont pas fixé une définition complète à cette technologie. Cela est principalement dû à la flexibilité qu'elle nous offre, et les différentes formes qu'elle peut prendre à cause de cette flexibilité. Mais il existe des caractéristiques qui peuvent être propres à la radio logicielle. Comme son nom l'indique, la radio logicielle est une radio qui peut changer sa forme en utilisant un logiciel [10]. L'architecture générale d'une telle radio est montrée dans la figure ci-dessous (Figure.5). On a besoin d'une antenne, des convertisseurs analogiques/numériques (CAN) et numériques/analogiques (CNA), et un processeur DSP ou FPGA (ou les deux au même temps) [4].

L'information à transmettre doit passer par le processeur qui est déjà configuré à un certain standard de communication (la configuration peut être attachée à l'information). Après le traitement selon le standard utilisé, l'information est convertie et envoyée après le réglage des paramètres de conversion selon le même standard.

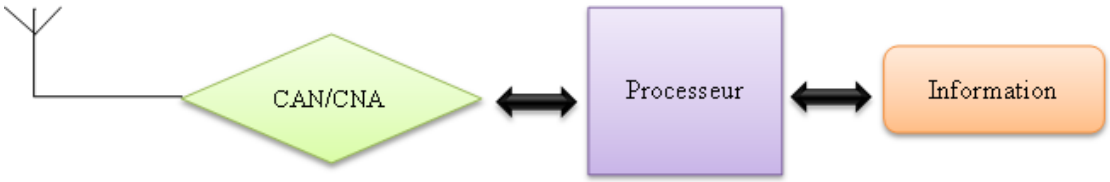



**Figure 5 :** *L'architecture générale d'une radio logicielle. [4]*

Dans une architecture plus détaillée (Figure 6), on remarque la différence entre une radio traditionnelle et une radio logicielle. Cette différence réside dans le bus de control ajouté à la radio, et qui nous permet de contrôler les paramètres de chaque unité dans la radio [11].

Dans ce qui suit, nous allons détailler chaque unité dans cette architecture. Où nous allons montrer les composants les plus importants dans la réalisation de ces unités. Et les méthodes utilisées pour la programmation des radios logicielles.

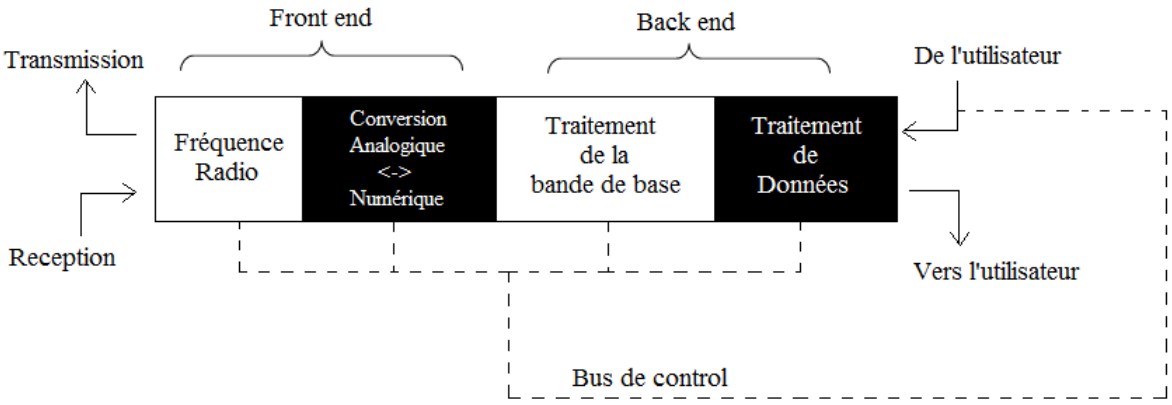



**Figure 6 :** *L'émetteur-récepteur de la radio logicielle. [11]*

### 1.3.1. Fréquence Radio

Idéalement, l'unité responsable à la conversion analogique / numérique est reliée directement à l'antenne. Mais à cause des problèmes de sélectivité et de sensibilité, on doit implémenter une architecture qui a pour objectif le traitement analogique des signaux entrants et sortants. Il existe plusieurs architectures, les plus communes sont la conversion directe, et l'architecture superhétérodyne. La première est généralement utilisée dans les applications moins exigeantes. Plus récemment, on peut trouver cette architecture implémentée sur les hôtes pour les communications cellulaires. Tandis que la deuxième est plus ancienne et plus utilisée dans les architectures des radios logicielles, parce qu'elle offre un intervalle de fréquences plus large en gardant une bonne sensibilité et sélectivité [10].

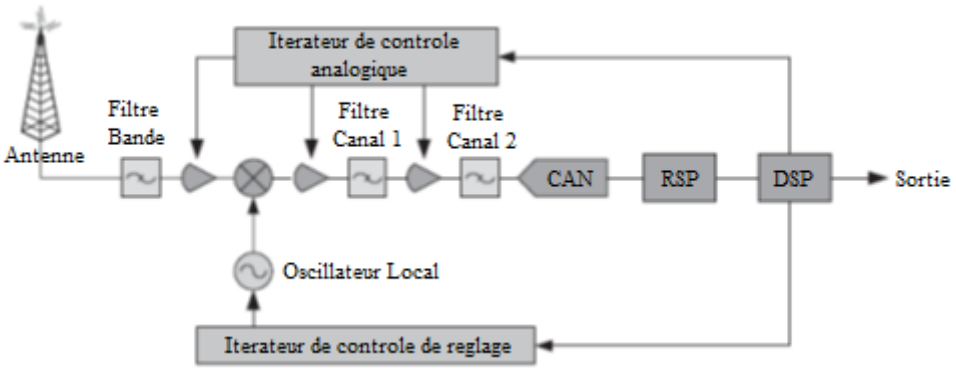



**Figure 7 :** *Une superhétérodyne radio logicielle – côté réception. [10]*

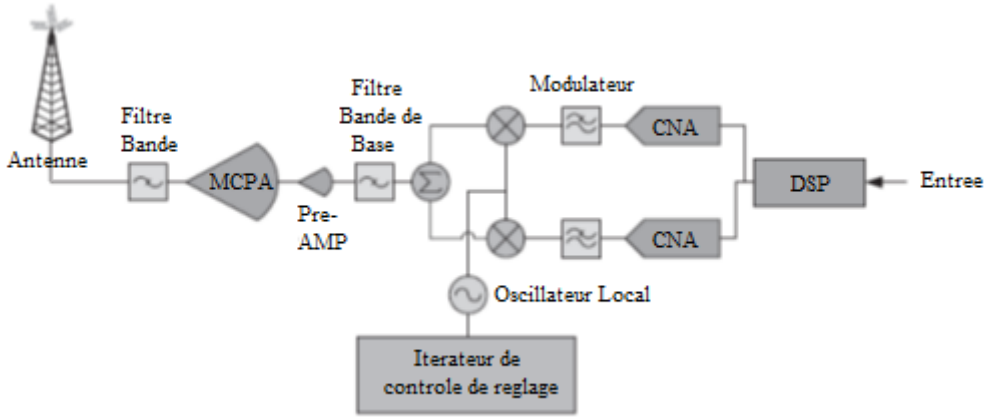



**Figure 8 :** *Une radio logicielle avec conversion directe – côté transmission. [10]*

Une radio logicielle n'est pas totalement développée sauf si tous ces composants sont programmables y compris l'antenne. Ce composant qui est le plus important est le premier composant dans la chaine du signal reçu. Tandis qu'il marche bien pour la plupart des applications où on utilise une seule bande, il devient un point faible pour les applications qui doivent recevoir les informations sur plusieurs bandes (comme SpeakEasy). Dans ce cas, on exige une antenne un peu plus particulière [10].

L'antenne est reliée à un filtre de bande qui limite l'intervalle des fréquences dans lequel on transmit ou on reçoit le signal. Ceci est fait d'abord pour minimiser les effets de la distorsion au niveau de l'intermodulation. Et de plus, pour assurer la bonne sensibilité, qui peut être dérangée par les signaux hors les intervalles souhaités. Aussi, la radio logicielle doit idéalement incorporer un amplificateur faible bruit ou un préamplificateur (LNA pour Low-noise amplifier) qui peut être opérationnel pour l'intervalle des fréquences souhaités. De plus, elle doit aussi incorporer un amplificateur de tension pour ajuster la force du signal transmis au seuil voulu [10].

On a aussi les mixeurs qui traduisent le spectre des fréquences radio (RF) vers une fréquence IF (Intermediate frequency) qui va être exploitable et prête à être filtrée (ou traduire l'IF vers RF en cas de transmission). Comme le préamplificateur, il est préférable d'avoir des mixeurs configurables pour maximiser la performance des filtres et avoir un signal plus clair [10]. Il existe des architectures de réceptions où on utilise aussi un démodulateur en quadrature à la place ou avec un mixeur. Ceci va séparer le composant en phase (I) et le composant en quadrature (Q). Après que les routes des deux signaux sont séparées, et parce qu'on fait un traitement analogique, il se peut que les deux signaux deviennent déséquilibrées. Donc une correction doit être faite sur les deux signaux [10, 11]. Lors de transmission, ces mixeurs peuvent

aussi être remplacés où utilisés avec des modulateurs qui servent principalement à la traduction d'IF vers RF.

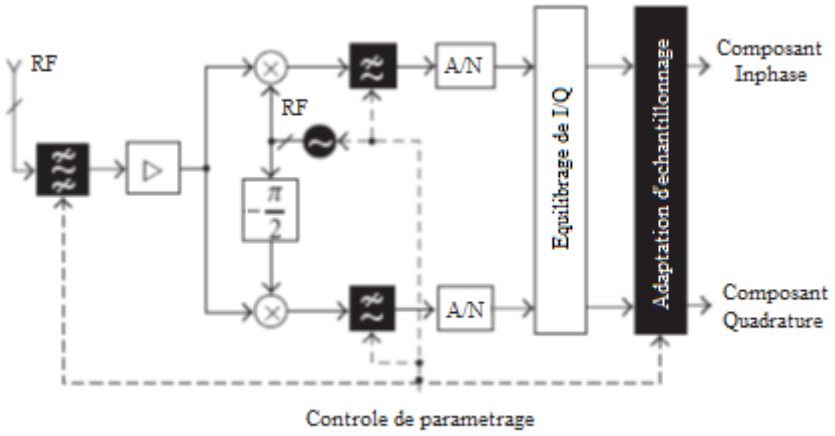



**Figure 9 :** *Un récepteur d'une radio logicielle avec une démodulation en quadrature. [11]*

L'oscillateur local (LO) est utilisé comme une entrée dans les mixeurs pour transformer la fréquence radio (RF) en fréquence IF si on reçoit le signal, et de IF vers RF si on émet. Il génère une fréquence fixe ou variable, selon le besoin pour la transformation. Il est relativement facile à programmer [10].

Aussi, pour ajuster la fréquence IF, et rendre le signal plus lisible lors de la réception, on doit utiliser des amplificateurs généralement sous forme d'AGC (Automatic gain control) qui vont être contrôlés par un contrôleur ; soit analogique qui va être suffisant pour les applications qui utilisent une seule porteuse, soit numérique pour bien contrôler le signal sortant de l'amplificateur dans les applications où on a besoin d'une communication multi-porteuse [10].

### 1.3.2. Conversion analogique / numérique

L'unité de conversion analogique-numérique et numérique-analogique est l'unité qui décide principalement l'architecture de la radio logicielle. Comme pour toute radio, le choix des convertisseurs est une étape élémentaire pour définir les caractéristiques de la radio. Les CANs sont utilisés pour convertir le signal IF à un signal numérique traitable par un FPGA ou un DSP. Et les CNAs sont utilisés pour construire le signal IF traitable par l'unité « Fréquence radio » pour l'envoyer.

Comme on l'a déjà mentionnée, la radio logicielle doit être compatible avec les applications qui utilisent une communication multi-porteuse. Avec les technologies avancées des CANs, la radio logicielle devient très utile pour ces applications. Principalement, la radio logicielle peut implémenter les CANs à suréchantillonnage avec une large bande passante. Un

CAN à suréchantillonnage est un CAN qui échantillonne le signale à une fréquence qui peut satisfaire le critère de Nyquist (la fréquence d'échantillonnage doit être double la bande passante). Pour la radio logicielle la bande passante est variable, donc le choix de la fréquence d'échantillonnage doit être double la plus large bande passante que la radio logicielle peut utiliser. Pour les CANs, il est préférable de pouvoir changer la fréquence d'échantillonnage, la plage d'entrée et la bande passante active. Pour les CNAs, il existe beaucoup de convertisseurs qui donnent la possibilité de contrôler les paramètres de conversion, comme le gain et la correction du décalage pour minimiser le déséquilibrage de I et Q, et comme l'interpolation de débit de données et la correction de phase entre les signaux I et Q [10].

### 1.3.3. Traitement de bande de base

L'unité de traitement de la bande de base est l'unité chargée de la traduction du signal numérique reçue depuis l'unité de conversion A/N en chaine binaire, et du remplissage des porteuses utilisées par les informations appropriées. Cette unité doit avoir quelques paramètres qui lui aident à traiter soit le signal, soit la chaine binaire, comme la fréquence d'échantillonnage et la taille de la porteuse.

Cette unité est généralement implémentée sur un circuit FPGA qui offre une grande flexibilité lors de la programmation. Il peut être programmé pour faire la démodulation en quadrature, le filtrage des canaux, et même pour traiter les données. Mais il reste un peu couteux. Pour un choix moins couteux, il existe aussi les circuits ASIC, mais en revanche on perd la flexibilité offert par les FPGA [10].

### 1.3.4. Traitement de données

Depuis son nom, L'unité de traitement de données est l'unité chargée principalement à traiter l'information reçue ou à transmettre. Elle peut contenir toutes les couches de communication. Elle est aussi l'unité centrale qui reçoit les informations de control depuis l'utilisateur et qui envoie les paramètres appropriés aux autres unités. Pour cela, on utilise généralement des DSPs à usage général qui sont programmables pour couvrir particulièrement tout type de tâches [12].

### 1.3.5. Programmation des radios logicielles

Il existe plusieurs plateformes pour la programmation de la radio logicielle. La plus utilisée est appelée GNU Radio. Cette plateforme est bonne pour développer des programmes de traitement de signal du récepteur. La programmation se fait uniquement pour les deux dernières unités (le Back end dans la figure 5), les deux premières sont construites physiquement. A cause de l'utilisation de l'ordinateur pour traiter le signal, et parce que le traitement de signal se fait en

série, la performance de la radio diminue et elle devient inutilisable dans la pratique. Une autre plateforme qui est complète est plus utilisable dans la pratique, est la plateforme SCA (Software Communication Architecture). Une plateforme riche par les services basés sur CORBA pour la configuration de la radio [13].

## 1.4. Défis de la radio logicielle

La radio logicielle avec son architecture flexible et reconfigurable offre une grande opportunité pour améliorer la technologie des réseaux sans fil. Bien qu'on parle de la radio logicielle depuis 1992, il reste encore d'autres issues à surmonter. Principalement au niveau de la conception des applications, et au niveau de la sécurité. Tous ces problèmes sont les obstacles qu'on doit les éliminer afin de construire la radio logicielle complète.

Dans cette section, nous allons résumer les plus grands défis qui sont ciblés par la recherche sur la radio logicielle. Pour une étude approfondie, l'article [14] cite ces défis en détail.

### 1.4.1. Défis liés à la performance

- *Exigences de calcul :* comment augmenter la capacité de calcul de la radio logicielle pour qu'elle puisse traiter les algorithmes qui utilisent un haut débit à large bande tout en minimisant sa taille, son poids et son coût, est un défi fondamental pour ce type de technologie.
- *Unités de traitement :* Il existe beaucoup d'éléments de traitement, on site : ASIC, FPGA, GPP, DSP, CCM, etc… Chacun de ces éléments a son propre avantage. Par exemple : FPGA donne une flexibilité de programmation et de reconfiguration que ASIC mais il est lent lors de la reconfiguration qu'un DSP. Alors quel est le meilleur élément ? cette question est un autre obstacle pour la radio logicielle car la réponse dépend de plusieurs facteurs (La performance, le temps de reconfiguration, la taille, le poids, …).
- *Exigences vs capacité :* L'observation de l'évolution de la puissance des systèmes cellulaires mobiles et des DSPs, nous indiquent qu'au futur, l'utilisation des DSPs dans les radios logicielles utilisées dans le domaine de la communication cellulaire va diminuer la qualité des services. Celui-ci est un autre défi pour la radio logicielle.

### 1.4.2. Défis liés à la sécurité

- *Protection contre le chargement du logiciel non-autorisé :* Bien que la radio logicielle soit bénéfique, sa flexibilité offre aussi un énorme risque pour la sécurité. Le majeur problème est la possibilité de charger un logiciel non-autorisé sur la radio. Quels sont

les services de sécurité qu'on doit installer ? et qui va autoriser un tel ou tel logiciel ? sont les questions qui ont besoin de réponses.

- *Systèmes de haute confiance :* Ces systèmes sont le coffre-fort utilisés généralement dans le secteur militaire pour garder les données de communication importantes comme la clé du cryptage. Le défi de la radio logicielle c'est d'avoir un tel système en conjonction avec la flexibilité offerte par cette radio.
- *Portabilité des modules de sécurité :* La portabilité des programmes installés sur la radio logicielle est un défi très présent dans la recherche. Ce problème est dû au fait qu'il n'existe pas un standard commun pour les APIs de la sécurité et la documentation qui vient avec.

### 1.4.3. Défis liés à la certification

Le challenge de la radio logicielle s'étend vers la certification, un énorme travail se fait par la FCC de l'USA et la directive R&TTE pour classer la radio logicielle dans sa convenable catégorie et pour définir les règles sur les programmes utilisés par ce type d'équipement.

### 1.4.4. Défis liés à la programmation

- *Portabilité des applications des radios logicielles :* Comme on l'a déjà mentionné, la portabilité des applications de la radio logicielle reste un problème pour la programmation des radios logicielle. Cela est causé par la variété des composants utilisés pour la construction de la radio et l'absence d'un standard et un langage de programmation dédiés pour développer des logicielle pour cette radio. Il existe des travaux dans ce domaine mais ils restent insuffisants, comme les APIs du JTRS et le projet ESSOR.
- *Développement des applications basées sur SCA :* Le renforcement de l'efficacité de la conception des applications basées sur la plateforme SCA en utilisant des outils MDD, et la compilation du code pour tout support physique, restent des défis pour la radio logicielle. Aussi, en évoluant vers la radio cognitive, il est nécessaire d'avoir une plateforme mieux que la SCA actuelle qui donne la possibilité de reconfigurer une partie de la radio en temps réel.
- *Problèmes liés aux middlewares :* A l'heure actuelle, il n'existe pas un middleware moins complexe et plus performant que le middleware CORBA qui a des problèmes un peu gênants au niveau ses composants (comme l'envoi et la réception des messages de petite taille).

# 2. Radio cognitive

L'invention de la radio logicielle a offert une très grande opportunité pour la recherche informatique, car enfin, avoir une radio qui peut s'adapter à son environnement et améliorer sa transmission et réception automatiquement devient un objectif réalisable. Une telle radio qui peut déterminer l'état de son environnement et qui change son comportement en fonction de cet état, s'appelle une radio cognitive, une radio sensible au contexte, ou une radio intelligente.

Donc, on peut comprendre qu'une radio cognitive est une radio logicielle reliée à des capteurs pour obtenir de l'information depuis l'environnement, et chargée avec un logiciel ou un programme muni d'une intelligence qui peut décider de l'action à mener. On peut classer la radio cognitive en deux classes selon la définition de l'environnement et le type d'action qu'elle peut prendre [11] :

- *Une radio cognitive avec des propriétés centrées sur l'utilisateur* : c'est une radio qui peut détecter les besoins de l'utilisateur (comme trouver un emplacement, ou savoir l'état d'un autre utilisateur) et qui peut agir selon ces besoins pour les satisfaire automatiquement.
- *Une radio cognitive avec des propriétés centrées sur la technologie* : c'est une radio qui s'intéresse aux besoins de la communication elle-même (comme l'utilisation de la bande passante, l'utilisation de la batterie, l'amélioration de la qualité de service). Donc cette radio peut agir automatiquement pour changer ou modifier le standard de communication pour satisfaire ces besoins.

La première classe propose un objectif global pour le développement des réseaux sans fils où la radio peut établir automatiquement les communications nécessaires pour satisfaire les besoins de son utilisateur. Par contre, la deuxième classe offre un challenge pour les chercheurs. Ces derniers doivent définir la bonne conception de la radio logicielle et du logiciel implémenté pour détecter et satisfaire les besoins techniques d'une façon autonome [11].

## 2.1. Critères et caractéristiques de la radio cognitive

### 2.1.1. Sensibilité au contexte (Awareness)

Pour dire qu'une radio est sensible au contexte, elle doit être capable d'extraire l'information depuis ses capteurs et les transformer en information qui a du sens pour le programme qui la traite. Elle doit utiliser les capteurs de localisation et du temps, les capteurs pour détecter la force des signaux transmis ou reçus, et d'autres capteurs qui doivent être convenables aux besoins de l'utilisateur ou les besoins techniques. Et elle doit être capable de

traiter les informations envoyées par ces capteurs et enregistrer et/ou utiliser les données importantes extraites depuis ce traitement [16].

### 2.1.2. Adaptabilité

Le deuxième critère pour la radio cognitive est qu'elle doit être adaptable. L'adaptabilité veut dire trouver une décision correcte pour gérer une nouvelle situation. Pour que la radio cognitive soit adaptable, elle doit avoir recours aux données extraites depuis son environnement, elle doit avoir les outils nécessaires pour performer ces actions, et elle doit installer un algorithme de recherche ou d'apprentissage qui permet de trouver la ou les bonnes actions lors de la rencontre d'un nouvel état d'environnement [16].

### 2.1.3. Cognition

La cognition est le critère principal qui rend la radio intelligente ou cognitive. Plus que la sensibilité au contexte et l'adaptabilité, pour que la radio soit cognitive elle doit être capable d'anticiper le changement de la situation et prendre les mesures correctes pour conserver la continuité de service. Donc, la radio doit implémenter un mécanisme efficace qui permet non seulement d'analyser les états, mais en plus, d'analyser le chemin que le changement de ces états va prendre. Lors de la détection d'un tel chemin, une prédiction de futur état devient possible ce qui permet à la radio de prendre ses cautions pour un éventuel changement [16].

### 2.1.4. Définition de la radio cognitive

Depuis ces critères, on peut donner une définition de la radio cognitive comme une radio logicielle sensible à son contexte, adaptative aux changements, et qui a une capacité d'analyse, d'apprentissage, et d'anticipation sur son environnement. On peut schématiser l'appartenance de la radio cognitive dans l'ensemble des radios logicielles dans la figure suivante.

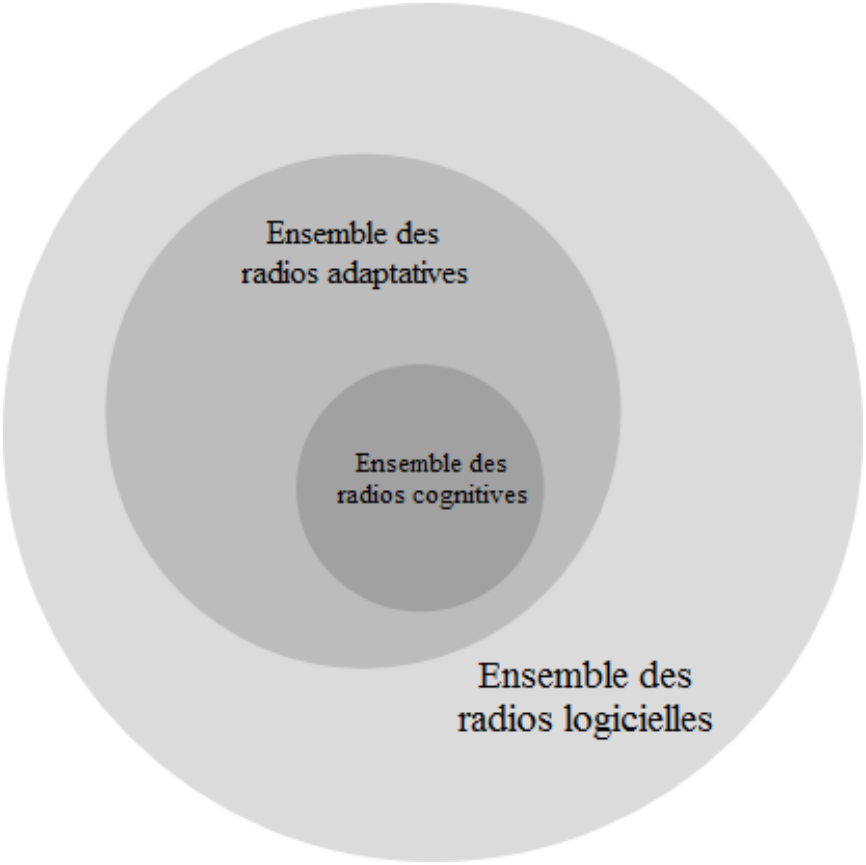



**Figure 10 :** *Emplacement de la radio cognitive par rapport à l'ensemble des radios logicielles.*

La radio cognitive se caractérise par [16]:

- Sa faculté d'observer son environnement. (comme la location et la bande passante)
- Sa capacité d'interpréter les informations en données utiles. (comme la relation localisation - bande passante)
- Sa faculté de faire la relation entre les besoins et ces données (comme le besoin d'être connecté dans la maison)
- Sa faculté de définir les obstacles et les contraintes pour satisfaire un besoin. (comme le temps de passage d'un canal vers un autre)
- Sa faculté de prendre les bonnes décisions pour arriver à ses objectifs. (comme changer le canal pour permettre une continuité de connexion)
- Son apprentissage. (elle n'est pas programmée pour un cas d'utilisation particulier mais elle est programmée pour apprendre les divers cas possibles)

## 2.2. Architecture de la radio cognitive

Pour la radio cognitive, on se focalise principalement sur le côté logique. Il est par contre reconnu que l'architecture physique de la radio doit être modifiée pour qu'elle puisse capter les informations nécessaires depuis son environnement.

### 2.2.1. Architecture physique

La figure 11 montre un schéma général de l'architecture physique de la radio cognitive. Dans ce qui suit, on va citer quelques composants de perception qui sont utilisés dans les architectures de la radio cognitive.

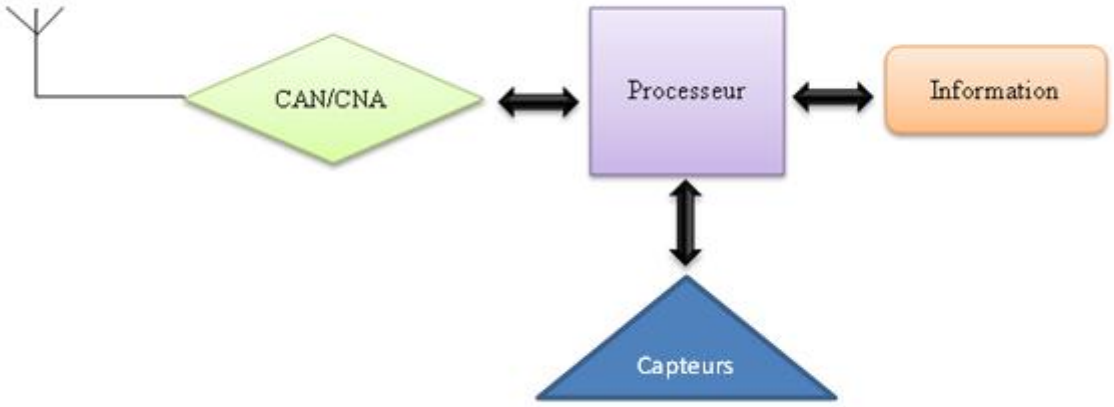



**Figure 11 :** *Schéma général de l'architecture de la radio cognitive.*

- *Capteurs de localisation :* Le capteur le plus connu et le plus utilisé pour détecter la localisation est le GPS, qui offre une grande fiabilité aux cordonnées de l'emplacement actuel de la radio avec les 28 satellites qui tournent autour de la terre. Les capteurs de mouvement sont aussi des outils intéressants pour détecter la vitesse de mouvement de la radio.
- *Capteurs d'environnement :* La perception de la température et de l'humidité permet à la radio de connaitre la cause de la faiblesse du signal. Ce type de capteurs existe dans le marché. Il y a aussi les capteurs de lumière qui permettent de faire la différence entre l'intérieur et l'extérieur ou la différence entre le jour et la nuit.
- *Capteurs des fréquences radio :* Qui permettent à la radio de détecter un signal et sa force. Ils permettent aussi de détecter s'il y a des interférences qui perturbent la transmission ou la réception.
- *Capteurs de l'état de la batterie :* Pour percevoir la consommation de l'énergie et agir selon sa valeur.

- *Capteurs intégrés :* Ceux sont les capteurs qui existent déjà dans le matériel, comme l'horloge pour capter le temps, le capteur des interruptions qui permet de détecter le clavier de l'utilisateur ou quel autre type d'entrées.

Tous ces capteurs et d'autres qui existent dans le marché sont utilisables pour concevoir la radio cognitive qui peut répondre à nos besoins.

### 2.2.2. Architecture logique

L'architecture logique de la radio cognitive est toujours représentée par un modèle d'agent. Cette modélisation est due au fait que la radio agisse selon une boucle perception-action. Joseph Mitola a présenté cette boucle et il l'a nommé le cycle de la cognition (Figure 12) dans sa thèse de doctorat [17].

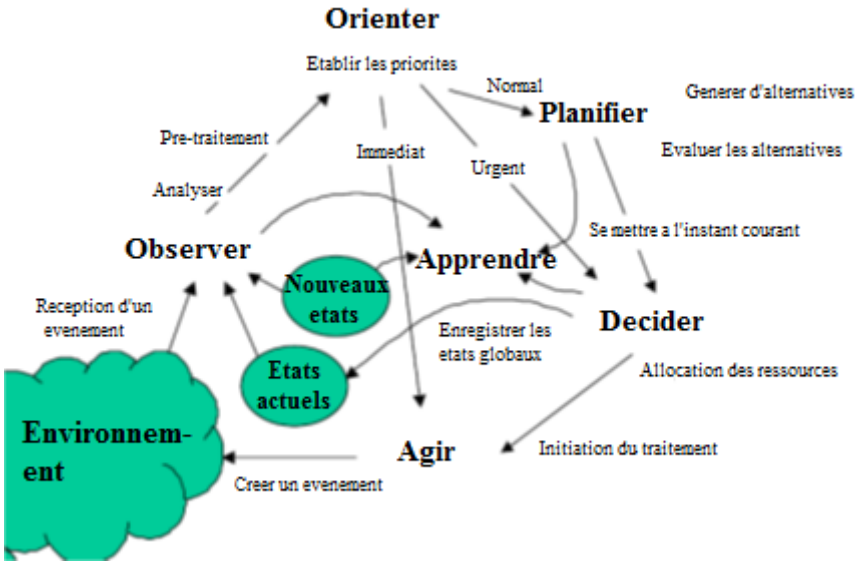



**Figure 12 :** *Cycle de la cognition [17].*

La radio cognitive doit continuellement observer son environnement, s'orienter vers l'action appropriée, planifier, décider à propos de l'allocation des ressources selon le plan à exécuter, et finalement agir. De plus, l'apprentissage est une tâche principale pour la radio cognitive. Le calcul fait lors de l'apprentissage est relativement complexe. Pour cela, la radio cognitive doit avoir un temps libre pour exécuter cette tâche sans oublier les besoins de l'utilisateur. Ce temps libre est référé comme la période de mise en veille (Sleep Time). Dans cette période, la radio n'utilise l'énergie que pour l'apprentissage. Si les opportunités d'apprentissage n'ont pas été traitées dans cette période, elles sont soit mentionnées à l'utilisateur, soit partagées (et calculées d'une façon collaborative) avec le réseau, ou bien envoyées au concepteur, lors de la période du secours (Prayer Time) [17].

Aussi, il y a la période du réveil usuelle (Wake Time) de la radio cognitive. Dans cette période, la radio reçoit les informations du réseau sous forme de messages, les informations depuis le clavier d'utilisateur, et les informations d'environnement depuis ses capteurs. Elle fait une observation sur ces informations et elle les compare avec l'état actuel. Elle fait son traitement de connaissance du contexte, puis elle s'oriente vers l'étape suivante. L'orientation est faite selon trois niveaux de priorités :

- Traitement immédiat (par exemple dans le cas d'absence d'énergie)
- Traitement urgent (Dans le cas d'absence de signal depuis un canal RF)
- Traitement normal (Si tout va bien)

La planification est l'étape reliée à l'anticipation. On doit implémenter une méta-heuristique pour la génération des plans. Après, il y a l'étape de la décision. Depuis les plans déjà générés, on doit choisir le plan le plus bénéfique à exécuter. On doit implémenter un algorithme d'allocation de ressources (Comme les machines virtuelles parallèles) pour pouvoir allouer les ressources nécessaires aux tâches du plan choisi. L'étape d'action est l'étape la plus primitive qui fait le traitement des tâches [17].

## 2.3. Applications de la radio cognitive

La flexibilité de la radio cognitive ouvre plusieurs opportunités dans le développement des réseaux sans fils. En effet, avec cette flexibilité la création des nœuds qui peuvent établir des communications automatiquement dans n'importe quel standard devient une réalité. Plusieurs systèmes complexes vont être simplifiés.

La radio cognitive ne s'arrête pas à cette possibilité. L'utilisation de cette technologie s'étend vers la création des réseaux qui peuvent satisfaire tout type de besoins. On peut classer les applications de la radio cognitive en quatre catégories : La gestion des ressources de la radio, La gestion du réseau, la livraison des services, et la certification [17].

### 2.3.1. Gestion des ressources de la radio

Cette catégorie d'applications s'intéresse à l'optimisation de l'utilisation des composants de la radio. Généralement, on peut concevoir des applications pour la radio cognitive qui s'intéressent à l'allocation du spectre, l'optimisation de l'énergie utilisée pour la communication, l'optimisation du débit de transfert, etc. [18].

### 2.3.2. Gestion du réseau

C'est la catégorie d'applications qui s'intéresse au développement des outils de gestion de réseaux, comme les protocoles de routage, la sécurité du réseau, l'amélioration de la qualité de

service, l'allocation du canal (comme pour les réseaux cellulaire), etc. Aussi, elle s'intéresse au développement des applications qui peuvent détecter et se connecter sur un réseau, et qui rendent le déplacement entre les différents réseaux possible sans perdre la connexion. D'une manière générale, la gestion du réseau englobe les applications qui relient plus efficacement les nœuds aux réseaux permis selon le besoin [17].

### 2.3.3. Livraison des services

Comme son nom l'indique, la catégorie d'applications de livraison des services englobe les applications qui interagissent avec l'utilisateur. Ces applications peuvent connaitre les besoins de l'utilisateur par la perception visuelle, audio, ou textuelle, ou par la détection de son emplacement, puis elles agissent pour satisfaire le mieux possible ces besoins. De telles applications sont connues en pratique, et elles sont utilisées déjà dans les PDAs et les Smartphones (même avec une petite portion, car la perception visuelle et orale reste toujours un grand défi de recherche). Mais les intégrer dans la radio cognitive est un autre challenge [17].

### 2.3.4. Certification

Cette catégorie d'applications englobe les outils qui permettent à la radio cognitive de s'adapter aux nouveaux standards de communication. Principalement, la radio cognitive doit avoir une possibilité de [17] :

- Télécharger de nouvelles applications de communication et du traitement.
- Télécharger les interfaces d'utilisateur et les pilotes.
- S'adapter à l'interface radio pour implémenter les nouveaux standards qui sont utilisés dans une région avec leurs fonctionnalités.
- Télécharger les mises à jour concernant les applications installées.
- Télécharger les références des services et des opérateurs locaux, tout en donnant la possibilité d'acheter les licences d'utilisation d'un réseau particulier.

La radio cognitive est la technologie de rêve pour le développement des réseaux sans fils. Avec sa flexibilité avec tout type de communication sans fil et avec son adaptation avec son environnement, la radio cognitive simplifie beaucoup de problèmes complexes dans les réseaux sans fil.

La recherche sur la radio cognitive implique la recherche sur les deux mots qui composent ce nom. D'une part (le mot : radio), on doit répondre aux questions posées par la recherche sur la radio logicielle, car elle est l'appui de la radio cognitive. D'une autre part (le mot : cognitive), on

doit trouver des solutions plus efficaces pour les problèmes posés par la cognition, comme le problème d'allocation des ressources, ou la compréhension du langage naturel.

L'objectif ultime de la radio cognitive est d'avoir un composant complet qui permet la communication dans n'importe quelle interface radio, et qui peut satisfaire tous les besoins de son utilisateur. Pour y arriver, la capacité d'analyse, de compréhension, et d'action doit être suffisamment grande.

# 3. Réseau cognitif

Comme nous l'avons indiqué dans la section précédente, le mot « cognition » a été introduit la première fois dans le domaine de communication par Mitola. Son travail a été focalisé sur les deux premières couches du modèle OSI (PHY et MAC). Dans ces deux couches, la radio cognitive utilise un cycle de cognition pour interagir avec l'environnement. Depuis, le sens original de la radio cognitive a été changé. Plus précisément, c'était quand la FCC l'avait définit comme *« une radio qui peut changer les paramètres de son transmetteur selon son interaction avec l'environnement dans lequel elle opère… cette interaction peut impliquer des négociations actives avec d'autres utilisateurs du spectre et/ou la détection et la prise de décision passive (radio intelligente) dans la radio. La majorité des radios cognitives vont probablement être des radios logicielles. Mais la radio cognitive n'utilise pas nécessairement un logiciel, ni elle doit être reprogrammable.* ». Cette définition a mis les chercheurs hors du chemin initial vers un autre chemin où on s'intéresse beaucoup plus à l'optimisation de l'utilisation du spectre [47].

Quand l'objectif de la recherche a dépassé l'individu (La radio cognitive) vers le groupe (Le réseau cognitif), on sous-entend souvent que le chemin basé sur la définition de la FCC s'oppose à celui qui se base sur la définition originale de Mitola. Mais en réalité, on peut dire que le premier est inclut dans le deuxième. Car le spectre est aussi un des caractéristiques, si ce n'est pas le plus important, de l'environnement que les nœuds avec une radio logicielle interagissent avec.

## 3.1. Définition du réseau cognitif

A cause du sens large du mot « cognition ». Plusieurs définitions ont été données pour le réseau cognitif [48] :

Selon ***Sifalakis et Al***. Un réseau cognitif est un réseau capable à s'adapter aux conditions et évènements en se basant sur le raisonnement et les connaissances acquises.

***Boscovic*** propose une définition plus technique. Selon lui, le réseau cognitif est un réseau qui peut changer dynamiquement sa topologie et/ou les paramètres opérationnelles pour répondre aux besoins d'un utilisateur particulier toute en optimisant la performance globale du réseau.

***Thomas et Al***. définit le réseau cognitif comme un réseau qui peut percevoir son état actuel, planifier, décider, agir selon cet état et apprendre les conséquences de ses actions, tout en suivant un objectif précis.

Selon ***Fortuna***, un réseau cognitif est un réseau qui, non seulement a un cycle de cognition, mais il a aussi un mécanisme qui lui permet à acquérir une connaissance sur son environnement.

Il est intéressant de voir que toutes ces définitions parlent du réseau qui doit suivre, plus ou moins, le cycle de cognition, et pas des individus (nœuds) qui le constituent. Il est claire aussi, que la définition du Fortuna est la plus générale. Car elle parle aussi des mécanismes permettant au réseau de connaitre son environnement. Et celle-là est la définition que nous optons le plus.

Donc pour nous, un réseau cognitif est un réseau qui peut percevoir l'environnement dans lequel il existe, ainsi que ses états actuels, qui peut planifier les changements de ses états selon le changement de son environnement, qui peut apprendre de ces actions pour améliorer sa performance, et qui peut s'adapter et mieux prédire les changements de son état ou de son environnement.

Comme on peut remarquer, la notion du réseau cognitif est vaste et plus générale. Elle peut inclure les réseaux sans fil comme elle peut inclure les réseaux à fil. Pour ce qui nous concerne, on appelle un réseau sans fil où les nœuds qui le composent sont cognitifs, un réseau cognitif. Car les nœuds eux même suivent le cycle de cognition, et chaque nœud peut percevoir son environnement, ce choix donc, ne contredit pas la définition optée.

Dans ce cas, les nœuds opèrent sur une radio cognitive. Ils vont utiliser leur intelligence pour améliorer le comportement de tout le réseau ; en essayant de satisfaire leurs propres besoins.

### 3.2. Caractéristiques des réseaux cognitifs

Pour répondre plus aux objectifs de ce rapport, nous nous intéressons plus aux réseaux de radios cognitives ad hoc, qui représentent mieux l'hétérogénéité et l'autonomie des systèmes. Car chaque nœud a ses propres besoins, il va, par conséquence, communiquer avec les autres

nœuds en utilisant un standard qui lui convient, et qu'il peut être diffèrent à ce que les autres nœuds utilisent.

Dans ce cadre, on peut résumer les caractéristiques du notre réseau cognitif comme suit :

### 3.2.1. Cognition

Il est clair que la première et la plus importante caractéristique des réseaux cognitifs est la cognition. Comme il est précisé précédemment, ce type de réseaux doit être adaptable aux changements de son environnement. De plus, il doit avoir une certaine perception sur les futurs états que peut prendre cet environnement.

### 3.2.2. Coopération

Dans un réseau, un nœud ne peut pas exister tout seul. Et car les nœuds de ce type de réseaux sont cognitifs. Avoir plus de connaissance sur l'environnement, et les actions des autres nœuds ne peuvent qu'à améliorer l'adaptabilité et la prévoyance des nœuds. Ce qui augmente sûrement les performances du réseau.

### 3.2.3. Individualité des nœuds

Bien qu'il y ait une coopération entre les nœuds, il reste clair qu'une des caractéristiques des réseaux cognitifs est que les nœuds agissent pour satisfaire leurs propres besoins. Leurs actions et leur niveau de coopération sont principalement guidés par leurs besoins.

### 3.2.4. Accès à une bande spectrale plus large

Comme on parle d'un réseau des radios cognitives, la particularité de ce réseau réside donc dans la capabilité des nœuds d'avoir des mécanismes pour détecter le spectre non utilisé, choisir la ou les fréquences qui puissent répondre mieux aux exigences du réseau, reconfigurer leur modulation de ces fréquences et régler la force du signal transmis. Le spectre autorisé aux nœuds constitue donc la plus grande partie de l'environnement où le réseau cognitif fonctionne.

## 3.3. Architecture globale des réseaux cognitifs

L'architecture est un ensemble complet et cohérent de règles de conception par les-quelles un ensemble de composants spécifiés réalise un ensemble déterminé de fonctions et de services qui évoluent à travers divers points de conception au cours du temps [49].

Nous pouvons décrire l'architecture des réseaux cognitifs en deux groupes : réseau primaire et réseau secondaire.

### 3.3.1. Réseau primaire

Le réseau primaire est doté d'une licence pour utiliser certaines bandes spectrales. Cette licence est donnée par des organismes gouvernementaux comme l'ANF (Agence Nationale des Fréquences) à l'Algérie. Les réseaux cellulaires sont un bon exemple de réseaux primaires dans lesquels les bandes spectrales sont propriétaires. Un réseau primaire est composé de :

- *Utilisateur primaire (UP)* : L'utilisateur primaire dispose d'une licence qui lui permet d'accéder à certaines bandes spectrales. Cette licence lui offre la possibilité d'opérer à tout moment sur des stations de base propres au propriétaire de la bande spectrale et de ne pas être dérangé par des utilisateurs étrangers. Les UP ne doivent subir aucune modification pour permettre la coexistence avec les nœuds cognitifs ou leurs stations de base.
- *Station de base des UP* : C'est une structure fixe pour le réseau primaire. Les stations de bases sont conçues de telle sorte à ne pas partager le spectre avec aucune entité extérieure au système, même avec les utilisateurs à radio cognitive (Utilisateurs secondaires). Cependant, il peut exister des stations de bases licenciées qui reconnaissent les protocoles des US (Utilisateurs Secondaires).

### 3.3.2. Réseau secondaire

Le réseau secondaire, appelé aussi réseau de radios cognitives ou réseau non-licencié, ne possède pas de licence pour opérer sur la bande spectrale. Aussi, on peut avoir plusieurs réseaux secondaires partageants la même bande spectrale. D'où le besoin d'un ensemble de fonctionnalités et outils additionnels pour pouvoir partager les bandes spectrales licenciées. On cite :

- *Utilisateur secondaire (US)* : C'est un utilisateur qui n'a pas de licence pour transmettre sur la bande spectrale. Cependant, grâce aux fonctionnalités additionnelles dont ils disposent, ils pourront partager la bande spectrale avec les utilisateurs primaires ou bien profiter de l'absence de ces utilisateurs pour transmettre.
- *Station de base des US* : La station de base des US, nommée aussi station de base non licenciée, est une infrastructure fixe avec des capacités cognitives. L'US se connecte à la station de base pour accéder à d'autre réseaux ou services.
- *Serveur spectral* : Le serveur spectral est une entité du réseau cognitif qui sert à partager les ressources spectrales entre les différents US dans le même réseau. Ce serveur est connecté à chaque réseau secondaire et agit comme un gestionnaire d'information spectrale. Dans un réseau ad hoc, un ou plusieurs nœuds peuvent jouer le rôle des serveurs spectraux.

- *Courtier spectral* : Le courtier spectral est une entité du réseau cognitif qui partage les ressources spectrales entre différents réseaux cognitifs. Ce serveur est connecté à plusieurs réseaux et agit comme un gestionnaire d'information spectrale.

## 3.4. Architecture conceptuelle des réseaux cognitifs

Comme le réseau primaire ne doit pas être modifié, la conception des réseaux cognitifs touche principalement le réseau secondaire. Car on parle d'un réseau où les nœuds sont cognitifs, la conception d'un tel réseau revient donc à la conception des protocoles permettant à ces nœuds d'interagir et d'auto-organiser pour profiter le mieux possible des opportunités dans les bandes spectrales. A ce propos, la grande partie de la recherche dans les réseaux cognitifs essaye de définir des modèles pour les nœuds à radios cognitives. Deux majeures conceptions sont utilisées pour modéliser les nœuds : la conception dite entre-couches (cross-layered design) et la plateforme cognitive [47].

### 3.4.1. Conception entre couche

La conception entre-couche est une nouvelle technique pour modéliser les nœuds d'un réseau. Et qui se base sur la conception classique basée sur le modèle OSI. On peut identifier deux approches pour cette conception. Une approche implicite, et une approche explicite.

Dans l'approche implicite, Il n'y a aucune modification forcée sur le modèle OSI. Mais les couches sont programmées en tenant compte des autres couches. Bien qu'elle optimise un ou plusieurs objectifs, cette approche limite la flexibilité et la capacité d'amélioration pour les nœuds. Car la modification (amélioration) d'un protocole dans une couche risque d'engendrer des modifications sur des protocoles dans des autres couches.

Dans l'approche explicite, le modèle OSI lui-même est modifié. Soit en ajoutant des couches supplémentaires ou bien on divise/combine des couches. On peut trouver trois types de conception explicite :

- *Division/Combinaison des couches* : Dans la technologie des réseaux cognitifs, conduite par la demande d'une grande bande passante ou une bonne qualité de service, la division ou bien la combinaison des couches touche principalement les trois premières couches du modèle OSI. Une des plus connue, est la division de la deuxième couche (liaison des données) en deux sous-couches : une couche MAC (contrôle d'accès au media) qui, principalement, interagit avec la couche physique, et une couche LLC (contrôle des liaisons logiques) qui interagit avec la couche réseau.

- *Interaction entre couches* : Ce type de conception s'intéresse plus aux interactions entre les protocoles des différentes couches. Elle propose des modèles de communication et de partage d'information entre les différents protocoles qui résident dans les couches. Ce qui va permettre une amélioration de la performance du réseau. Comme pour l'internet, dont son succès revient plus à ce type de communication entre ses protocoles.
- *Calibration verticale* : Cette conception est plus organisée par rapport à la conception précédente. Dans la calibration verticale, on ajoute une couche supplémentaire qui peut communiquer avec toutes les couches.

Comme on peut remarquer que toutes ces approches sont centrées sur le modèle des couches. Elles ne sont pas des concepts qui peuvent améliorer la performance des réseaux cognitifs. Car il n'y a pas la notion de la cognition, même pas la notion de l'adaptation. Pour cette raison, une amélioration sur ces conceptions doit être faite pour avoir un modèle spécifique aux réseaux cognitifs.

**Clark et al.** sont les premiers qui ont proposé un nouveau genre de réseau qui est sensible à son contexte, et qui peut apprendre, décider et agir selon ces décisions pour répondre aux objectifs voulus. L'architecture conçue est aussi basée sur le modèle des couches. C'est un modèle qui prend la calibration vertical comme base et ajoute une autre dimension dans laquelle les nœuds évoluent leurs décisions (choix de protocoles pour toutes les couches). Ce modèle est appelé le Knowledge Plan (KP).

Dans ce modèle, chaque couche fait ses tâches propres à elle, et elle communique ces actions à la couche verticale appelée la KP. Cette couche garde trace de toutes les actions prises par les couches et le résultat obtenu, Puis elle calcule, apprendre, et planifier les ensembles des actions que les couches doivent prendre dans le futur si on se trouve dans la même position.

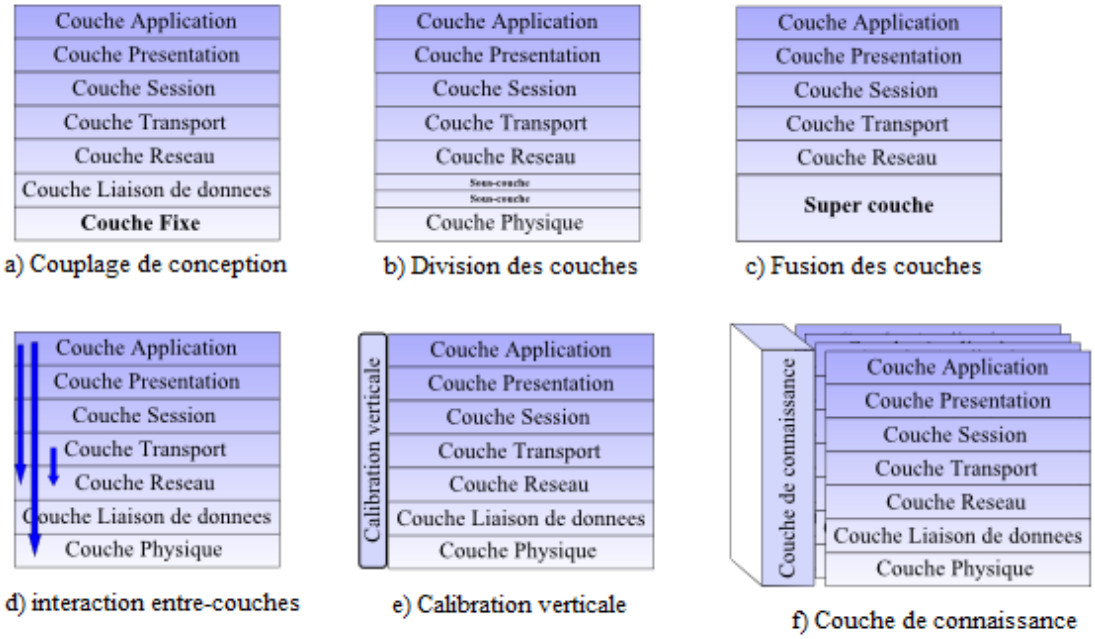



**Figure 13 :** *Conceptions entre-couches. [47]*

### 3.4.2. Plateforme cognitive

Le point de départ vers le développement d'un réseau cognitif est l'agent intelligent. Un agent est une entité qui perçoit l'environnement grâce à des capteurs et agit sur cet environnement grâce à des actionneurs.

À cet égard, les réseaux cognitifs sont attendus pour améliorer le niveau d'intelligence des systèmes de communication actuels en incorporant des agents intelligents dans le KP. Leurs agents peuvent effectuer des actions pour le compte de l'utilisateur. Plus spécifiquement, ils peuvent interagir avec les données, les applications ou les services. Ils peuvent aussi percevoir, raisonner, planifier et apprendre. Ces actions sont exactement les mêmes que celles souhaitées par les nœuds des réseaux cognitifs. Et lesquelles on peut trouver dans les états de la boucle de la cognition.

Une conception proposée dans [50] pour les agents du réseau cognitif est la plateforme cognitive. Dans cette plateforme (Figure 14), La couche la plus haute permet de spécifier les objectifs du réseau. Puis on traduit ces objectifs en besoins, et contraintes pour permettre une description plus modélisée pour la couche de cognition. Cette dernière utilise les informations reçues par le matériel physique des nœuds dans la boucle de la cognition, pour essayer de répondre aux besoins de ces nœuds.

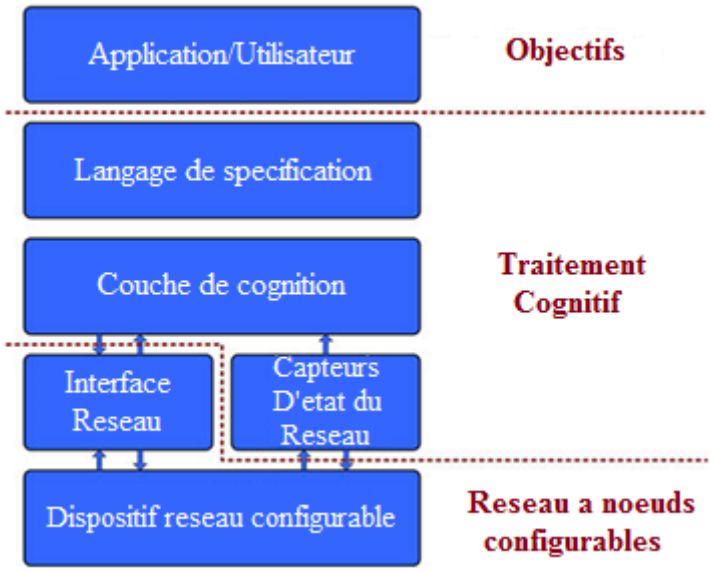



**Figure 14 :** *Plateforme cognitive. [47]*

## 3.5. Protocoles MAC et NET pour les réseaux cognitifs

Naturellement, la recherche sur la radio cognitive prend plusieurs branches à cause de la flexibilité offerte. Elle couvre toute les couches réseaux : du plus bas niveau où on s'intéresse au traitement de signal et aux composants physiques comme les DSPs et les FPGAs, au plus haut niveau où on s'intéresse aux applications offertes à l'utilisateur comme l'accès aux données à distance et le traitement des fichiers. Elle s'étend aussi à la 8$^{eme}$ couche qui est la perception du langage naturel et la compréhension du comportement humain [16].

Pour l'aspect architectural des réseaux cognitifs, nous nous focalisons plus particulièrement sur les deux premières catégories d'applications mentionnées dans la section 2.3, la gestion des ressources de la radio, et la gestion des réseaux. En particulier, nous nous intéressons aux objectifs liés à la couche accès au medium (MAC) et la couche réseau (NET). Il est à noter que nous parlons des objectifs et non pas des couches eux-mêmes.

### 3.5.1. Protocoles MAC pour le réseau cognitive

Une des plus intenses branches de recherche sur la radio cognitive est l'optimisation de l'utilisation du spectre. Quand on parle de la gestion des ressources pour la radio cognitive, on réfère souvent à la gestion du spectre. Malgré qu'il y a aussi d'autres critères, comme la gestion de l'énergie de transmission, et l'amélioration de la qualité de service, on a constaté que les chercheurs ne prennent pas ces autres critères comme leur première priorité, mais ils les intègrent partiellement dans leurs solutions qui essayent de résoudre les différents problèmes de la sensibilité, l'allocation, le partage et la mobilité du spectre.

Au niveau des instances de règlementation, presque toutes les bandes de fréquences sont déjà prises [20]. Mais en pratique, on découvre que seulement 6% du spectre alloué (licencié) est utilisé [21]. On appelle les utilisateurs du spectre licenciés *des utilisateurs primaires*. On peut donc exploiter les 94% du spectre restant. Les utilisateurs de ce spectre libre sont appelés *des utilisateurs secondaires* [20, 21].

Car on voit les nœuds cognitifs (les utilisateurs secondaires) comme des visiteurs dans le spectre qu'ils occupent [19], les utilisateurs secondaires ne doivent en aucun cas interférer avec les utilisateurs primaires. Dès qu'on détecte un utilisateur primaire, les utilisateurs secondaires doivent changer leurs paramètres de communication (la force du RF, le débit, le canal, etc.) pour ne pas dégrader la qualité de service de cet utilisateur primaire. De plus, les utilisateurs secondaires doivent coordonner leur accès au spectre disponible pour éviter les collisions entre eux [20].

On peut diviser les protocoles MAC qui gèrent le spectre en deux classes [19,20]:

- *Protocoles MAC basés sur l'accès direct (DAB – Direct Access Based Protocols).*
- *Protocoles MAC d'allocation dynamique du spectre (DSA – Dynamic Spectrum Allocation driven Protocols).*

### *Les Protocoles DAB*

Dans cette catégorie, le nœud cognitif adopte une stratégie directe pour accéder à un canal ou le libérer. Aussi, l'objectif du nœud est direct. Il essaye d'arriver à cet objectif sans prendre en compte les objectifs globaux du réseau. Les protocoles DAB, qui sont dérivés de l'IEEE 802.11 et améliorés pour qu'ils peuvent être adaptés aux environnements multicanaux, sont classés comme (1) des protocoles MAC centralisés pour les réseaux à infrastructure, ou comme (2) des protocoles MAC distribués pour les réseaux qui n'ont pas une infrastructure. Ces derniers sont divisés en deux catégories : les protocoles basés sur la contention, et les protocoles basés sur la coordination [19].

### Les Protocoles Basés sur La Contention

Dans cette catégorie, les négociations sont faites entre l'émetteur et le récepteur. Le majeur problème pour les protocoles basés sur la contention est l'échange des messages de control (comme pour le partage et la négociation du spectre libre) qui sont pratiquement nombreux par rapport aux échanges entre les nœuds des réseaux sans fil classiques [19]. Donc la plupart des solutions sont basées sur l'ajout d'un canal hors de la bande utilisée pour l'échange des données, et qui peut être aussi utilisé pour la synchronisation et la diffusion. Il existe deux

choix pour ce canal : Il peut être un canal licencié, dont la bande passante doit être adaptative, spécialement lorsque le réseau comprend un grand nombre d'utilisateurs. Et il peut être aussi un canal partagé. Et quand on dit partagé, on dit aussi collisions et interférences. Donc ce type doit avoir un mécanisme pour qu'il ne dégrade pas la performance du réseau [20].

Pour gérer ce canal commun, il existe trois approches :

### Common Control Channel (CCC)

Dans cette approche (Figure 15), il est préférable d'avoir deux émetteurs-récepteurs. Un pour émettre et recevoir dans le canal des données, et un autre pour les messages de control. Tous les nœuds écoutent ce canal de control, ce qui permet aux nœuds voisins de l'émetteur et du récepteur d'écouter leur négociation et avoir un état précis sur l'allocation du spectre à tout moment. Il est possible aussi d'avoir un seul émetteur-récepteur. Dans ce cas, le nœud émetteur doit basculer vers le canal de control pour négocier avec le récepteur avant d'envoyer les données. Aussi, la négociation entre l'émetteur est le récepteur se fait par CFSR handshake (Channel Filtring Sender Receiver), dont l'émetteur envoie sa liste des canaux au récepteur, et ce dernier choisi un de ces canaux avec une méthode de sélection et l'envoie à l'émetteur pour lui permettre d'envoyer ses données [19].

Plusieurs protocoles utilisent ce modèle. On cite le protocole RBCS-MAC (Receiver Based Channel Selection MAC Protocol) [26], DCA-PC (Dynamic Channel Assignment with Power Control) [27] qui a amélioré le protocole DCA [28] pour qu'il optimise l'énergie de transmission, AS-MAC (Ad-hoc SEC) [29] qui est conçu pour la coexistence de l'architecture cellulaire qui utilise GSM et les utilisateurs secondaires et qu'ils n'ont pas une infrastructure (ils forment un réseau ad-hoc), et le protocole proposé par J. So et N. Vaidya dans [22] qui prend la qualité de service comme critère pour l'allocation du spectre [19].


**Figure 15 :** *Common Control Channel. [20]*

### Split Phase (SP)

Pour l'approche SP (Figure 16), on divise le temps en deux phases. La première phase est le délai de control, où les nœuds communiquent sur un canal choisi pour partager le spectre. La deuxième phase consiste à communiquer (échanger les données) sur le canal choisi entre l'émetteur et le récepteur. Le majeur problème dans cette approche est le choix d'intervalle du temps pour les deux phases. Cela est généralement déterminé par le volume de trafic. Si le trafic des données est faible, l'utilisation d'une phase longue du control dégrade la performance du réseau [19].

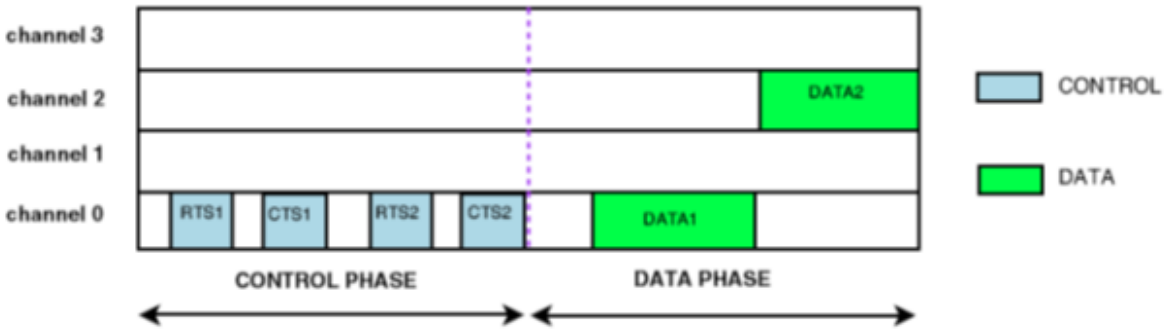



**Figure 16 :** *Split Phase. [20]*

### Frequency Hopping Sequence (FHS)

Cette approche (Figure 17) consiste à voir tous les canaux de la bande comme des canaux de control et des canaux de données au même temps. Les nœuds partage une liste des fréquences. Ils basculent séquentiellement d'un canal vers un autre jusqu'à ce qu'ils entrent en communication. L'existence de cette liste va donner une certaine fiabilité à la transmission, car on ne dépend plus d'un seul canal commun [20].

Il existe plusieurs travaux qui implémentent cette approche, et qui sont classés dans deux classes [19]: (1) Le saut habituel, où tous les nœuds sont synchronisés. Lorsque deux nœuds veulent communiquer ils s'arrêtent sur le canal courant, et quand ils terminent, ils doivent

resynchroniser avec les autres nœuds. On cite pour cette classe le protocole HRMA (Hop-Reservation Multiple Access) [30], CHMA (Channel Hopping Multiple Access) [31], et CHAT (Channel Hopping Access with Trains) [32] qui est l'amélioration du CHMA pour permettre le multicast et le broadcast. (2) La deuxième classe est la FHS dynamique, où chaque nœud fait son saut sur les canaux à sa manière (il utilise une fonction pseudo-aléatoire). La communication donc implique l'utilisation des techniques de choix du canal de rendez-vous qui permet les négociations à propos du canal d'envoi. On cite pour cette classe, le protocole SSCH (Slotted Seeded Channel Hopping) [33] qui est le protocole maquette de cette classe, et le protocole MC-MAC (Multi-Channel MAC) [34].


**Figure 17 :** *Frequency Hopping Sequence. [20]*

### Les Protocoles Basés sur La Coordination

Dans cette catégorie, tous les nœuds voisins coopèrent entre eux pour choisir le meilleur canal pour l'émetteur et le récepteur. Cette approche doit optimiser l'utilisation du spectre mieux que l'approche de la contention [19]. Ici aussi, on peut diviser les protocoles selon les trois classes décrites précédemment (c.à.d. CCC, SP et FHS).

Il existe plusieurs protocoles appartenant à cette catégorie. On cite CSCC (Common Spectrum Coordination Channel) [35] et HD-MAC (Heterogenous Distributed MAC) [36] qui sont dérivés de la classe CCC [19]. Aussi, On cite MMAC-CR (Multichannel MAC for Cognitive Radio) [22] qui est le protocole maquette de l'approche SP, et le protocole CMAC (Cognitive MAC) [37] qui est conçu pour régler les problèmes du MMAC-CR tout en prenant la qualité de service en considération [20]. Avec la stratégie DFH (Dynamic Frequency Hopping) [38] et son amélioration DH (Double Hopping) [39], l'IEEE 802.22 [40] est le protocole le plus populaire qui est basé sur la coordination et qui représente la classe FHS [20].

#### *Les Protocoles DSA*

Les protocoles DSA se distinguent des protocoles DAB dans la prise en compte des objectifs des nœuds d'un réseau. Malgré que le but des négociations est le même, c.à.d. les nœuds veulent accéder efficacement au spectre, les protocoles DAB utilisent des stratégies non-

cognitives et qui n'optimisent pas l'utilisation du spectre plus proprement. Alors que les protocoles DSA utilisent des stratégies basées sur l'intelligence artificielle pour non seulement allouer les canaux, mais aussi choisir les bons paramètres pour améliorer la performance comme la force de transmission, la méthode de modulation, la division du temps, la configuration de l'antenne, etc. [19]. Les problèmes traités par ces protocoles sont généralement définis comme NP-Complet, et la résolution de ces problèmes revient à la bonne modélisation, et la bonne utilisation des méta-heuristiques (Les algorithmes génétiques, la recherche dispersée, colonies de fourmis, etc.).

Selon la modélisation du problème, les protocoles sont classés en trois approches :

### Protocoles DSA Basés sur La Théorie Des Graphes

Dans cette classe, on représente le réseau par un graphe G = (V, E) où V est l'ensemble des sommets et E est l'ensemble des arcs. Il existe deux types de modélisation pour le réseau : (1) Le graphe de contention des nœuds (NCG – Node Contention Graph) qui représente les utilisateurs comme des sommets et leurs voisinages comme des arcs. (2) Le graphe de contention des liens (LCG – Link Contention Graph), où les sommets représentent les liens actifs entre les utilisateurs, et les arcs représentent le conflit entre les liens [19]. Le travail de H. Zheng et C. Peng [23] modélise le réseau comme un graphe bidirectionnel, et le problème d'allocation du spectre comme un problème de coloration de graphe. Le modèle est $G= (U,E_C,L_B)$, où $U$ est l'ensemble des nœuds, $E_C$ est l'ensemble des bandes de fréquences existante dans le spectre, et $L_B$ est l'ensemble des contraintes qui limitent les couleurs ($E_C$) utilisées pour un arc sortant d'un nœud. L'objectif est de minimiser le nombre des bandes occupées par les utilisateurs tout en prenant en considération les contraintes (bande passante et le champ de transmission) $L_B$.

Un défi pour les protocoles basés sur les graphes, est l'extensibilité. Le problème devient plus complexe si le réseau s'élargit, car chaque nœud doit toujours connaître l'état de tout le réseau. Pour cela, une solution suggérée est de diviser le réseau en un ensemble de sous réseaux. Il y a deux obstacles à surmonter dans ce cas : Comment partitionner le réseau ? Et comment relier les sous réseaux ? [19]

### Protocoles DSA Basés sur Les Variables Stochastiques

Dans cette classe, le processus d'accès dynamique au spectre est vu comme un processus stochastique. Par construction de matrices des probabilités d'action, le spectre va être alloué d'une façon juste et optimale. Généralement, le processus le plus utilisé est la chaîne de Markov. Le défi pour cette modélisation est de mieux répondre aux deux questions : Comment déterminer

le canal ou la probabilité de transition d'état pour l'utilisateur en se basant sur l'historique local et l'observation d'état actuel ? Et comment équilibrer entre la précision des statistiques et la complexité du modèle ? [19].

Un exemple de protocoles utilisant cette approche est DC-MAC (Decentralized Cognitive MAC) [41]. Ce protocole utilise le processus de décision markovien partiellement observable (POMDP) où il représente les caractéristiques du trafic de données d'un utilisateur primaire comme une chaine de Markov afin de savoir quels sont les canaux qui vont être libres d'une façon probabiliste [20].

### Protocoles DSA Basés sur La Théorie Des Jeux

Dans cette approche, les nœuds cognitifs sont modélisés par un modèle du jeu $\Gamma=<N,\{S_i\},\{u_i\}>$, où $N$ est l'ensemble des joueurs (Généralement les nœuds), $\{S_i\}$ est l'ensemble des stratégies $S$ adoptées par le joueur $i$ (comme la force de transmission, la modulation, etc.), et $\{u_i\}$ est l'ensemble des fonctions d'utilité locales qui doivent être optimisées par les ajustements effectués sur les stratégies. Aussi, si les nœuds font des ajustements périodiques et selon les ajustements d'autres nœuds, on doit introduire aussi $T$ qui est le temps de décision où un nœud doit choisir sa stratégie. La gestion de cette période est faite par plusieurs méthode, comme la méthode synchrone, asynchrone ou la méthode de tourniquet qui et la plus utilisée dans cette approche [19]. G-MAC est un protocole basé sur la théorie des jeux pour optimiser le rendement de tout le réseau en limitant la force de transmission. L'équilibre de Nash est achevé par un jeu récursif distribué [20].

La performance offerte par cette approche vient avec le coût des obstacles qu'on doit surmonter. Comme pour la théorie des graphes, un de ces obstacles est l'extensibilité. Il est clair que si le nombre des joueurs augmente pour le même jeu, le temps d'évolution du jeu devient très long, et on prend beaucoup de temps juste pour évaluer la situation. D'autre part, il est absurde de mettre deux nœuds qui ne s'interagissent pas souvent dans le même jeu. Pour cela, on doit diviser les nœuds dans les bons clusters pour régler ce problème. Un autre obstacle est le temps de décision. Pour les jeux itératifs, les joueurs doivent décider leurs stratégies avant que le temps s'écoule. Donc, trouver un bon ordonnancement pour les joueurs est un autre défi pour cette approche. Un autre défi, est la collision entre clusters des jeux. Les messages d'un cluster lors de l'évolution du jeu peuvent interférer avec les messages d'un autre cluster. Donc on doit éviter aussi ce type de collision [19].

### 3.5.2. Protocoles de Routage Pour La Radio Cognitive

Avant de parler du routage dans les réseaux cognitifs, on doit d'abord mettre le point sur la relation entre les couches 2 (MAC) et 3 (NET). Parce que la couche MAC s'intéresse à la communication entre deux nœuds voisins, elle offre à la couche NET l'élément de base pour une communication multi hop. A ce propos, il existe des protocoles de routage créés en fonction des protocoles MAC, ce type de protocoles s'appelle le protocole de routage orienté MAC. De même, les protocoles MAC qui prennent en considération les différentes routes décidées par un protocole de routage s'appellent les protocoles MAC orienté routage [19]. Un exemple du protocole orienté routage est présenté dans [24].

Principalement, on peut diviser les protocoles de routage pour les réseaux cognitifs en deux catégories :

#### *Conception découplée*

D'une façon générale, les couches MAC et NET sont indépendantes. La couche MAC s'intéresse à la communication directe entre nœuds, et la couche NET s'intéresse à la structure générale du réseau. On appelle cette conception typique pour les réseaux : une conception découplée [25]. On cite deux travaux appartenant à cette catégorie : (1) le travail fait par C. Xin [43], et (2) le travail fait par R. Pal. [44].

L'utilisation de ce type de conception est pour réutiliser les méthodes de routage conçues pour les réseaux non-cognitifs. Le problème est que ces derniers diffèrent des réseaux cognitifs au niveau de la couche MAC (comme la variabilité des canaux existants et la taille des bandes passantes qui n'est pas fixe). Pour cela, on doit régler ce type de problèmes séparément de la couche NET, en proposant un protocole MAC indépendamment des stratégies utilisées dans la couche NET [25].

#### *Conception Couplée*

Pour s'adresser directement aux problèmes du routage pour les réseaux cognitifs comme l'optimisation du chemin, une interaction entre les couches MAC et Net doit exister. La couche hybride doit être capable de calculer le chemin de routage à prendre, et elle doit être capable aussi d'assurer la communication directe entre les nœuds qui forment ce chemin. Ce type de conception est appelé une conception couplée. Généralement, cette conception assure la qualité de service, mais elle doit utiliser des algorithmes plus complexes qui engendrent un surcoût important que les chercheurs tendent à minimiser [25]. On cite le travail de Q. Wang et H. Zheng [45], et le travail de G. Cheng et Al. [46].

# 4. Réseau logiciel

Une autre nouvelle technologie qu'on doit aussi mentionner est le réseau logiciel. Contrairement à la radio logiciel, qui est l'origine et la base de la radio cognitive, Le réseau logiciel n'a aucune relation avec le réseau cognitif. Cette technologie s'intéresse plus à la structuration des réseaux.

Dans ces jours, et depuis longtemps, le paramétrage des switches et des routeurs est le facteur le plus important qui définit le réseau. Avec la demande continue de la connexion vers internet, et le passage graduel vers l'internet des objets, ce facteur devient de plus en plus difficile à le manier. La raison principale de cette complexité est la diversité trouvée dans la production des routeurs, où chacun installe son propre firmware avec des commandes plus ou moins différentes aux autres.

Cette complexité est doublée avec l'expansion du réseau qui nécessite chaque fois des modifications sur l'architecture pour au moins maintenir le seuil minimum de performance souhaitée. A cause de la technologie actuelle où on utilise des routeurs moins flexibles, cette tâche devient de plus en plus difficile et compliquée.

Le réseau logiciel est un paradigme de réseautage qui est venu pour répondre à ces difficultés d'infrastructure des réseaux utilisés actuellement [51].

## 4.1. Définition du réseau logiciel

Le réseau logiciel est une architecture de réseau qui a été créée pour répondre aux limites des réseaux IP traditionnels face aux changements qui sont entrain d'apparaitre dans ce domaine.

Le principe du réseau logiciel est qu'au lieu de régler et contrôler chaque composant réseau séparément (forcer manuellement une architecture d'interaction), on les relie tous avec un contrôleur central qui sera capable de communiquer les commandes de control avec ces composants. En utilisant les termes techniques, on dit qu'on sépare le niveau de données qui représente les tâches relatives au transfert des paquets du niveau du control qui représente les tâches de configuration et paramétrage des composants réseau.

Plus précisément, le réseau logiciel est une architecture à trois niveaux [51] :

### 4.1.1. Niveau de données

Ce niveau représente les composants actuels du réseau. Ces composants n'ont que deux tâches :

- Envoyer leurs états à la couche supérieure et recevoir les commandes de control.
- Exécuter le transfert des données (selon les commandes envoyées). Comme la programmation de flux assure une très bonne flexibilité, et il est seulement limité aux capacités des tables de flux implémentées, le transfert sera basé sur le flux au lieu de la destination dans les réseaux logiciels.

### 4.1.2. Niveau de control

Ce niveau est représenté par une entité extérieure qui fournit les technologies des serveurs, dans laquelle on implémente un système d'exploitation des réseaux (NOS), appelé aussi le contrôleur. Le contrôleur est une plateforme logicielle qui propose les outils et les abstractions des composants réseaux nécessaires pour faciliter leur programmation ou leur configuration.

Dans les systèmes d'exploitation traditionnels, on doit toujours avoir cette interface intermédiaire qui joue le rôle du traducteur des commandes logiques données par le système d'exploitation (comme l'accès à un fichier) pour la couche physique (emplacement de la tête d'un disque dans le cylindre, la piste et le secteur voulu). Cette interface est définie par l'ensemble des pilotes donnés par les constructeurs du matériel et installé sur le système d'exploitation.

Dans le cas des réseaux logiciels, cette interface sera représentée par des protocoles de communication qui fournissent la possibilité d'exécution des commandes logiques par les composants du réseau. Un exemple de ce type de protocoles est le protocole OpenFlow développé par l'université du Stanford, et qui est le protocole le plus connu dans le domaine des réseaux logiciels.

Aussi, comme pour les systèmes d'exploitation traditionnels où on fait des abstractions sur les composants physiques de la machine pour faciliter leurs accès à l'utilisateur (comme les systèmes des fichiers), les contrôleurs doivent être capable de donner une vue globale sur le réseau dans la façon la plus abstraite (comme la représentation par graphe) pour faciliter aux programmeurs du réseau de choisir les relations et les topologies nécessaires pour atteindre les objectifs voulus.

### 4.1.3. Niveau de gestion

Le niveau de gestion représente l'ensemble d'applications réseaux à exécuter par le niveau de données (le routage, les pare-feu, les moniteurs, etc…). Une application de gestion

définit les politiques utilisées qui sont transmises aux composants du réseau via un protocole de control comme OpenFlow.

La figure 18 montre le schéma général d'un réseau logiciel.

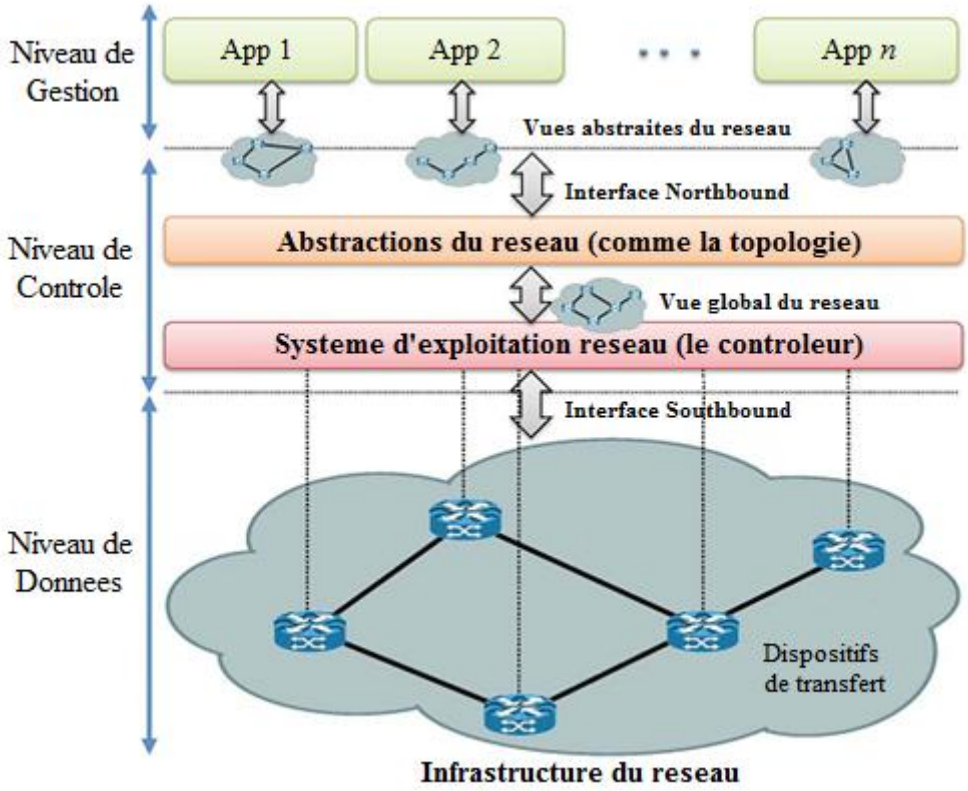



**Figure 18 :** *Schéma général du réseau logiciel. [51]*

## 4.2. Architecture du réseau logiciel

L'architecture du réseau logiciel est représentée par une composition de couches (figure 19). Chaque couche a ses propres tâches. Dans ce qui suit, nous allons détailler chaque couche en commençant par la couche la plus basse (couche infrastructure) vers la couche la plus haute (couche application réseau).

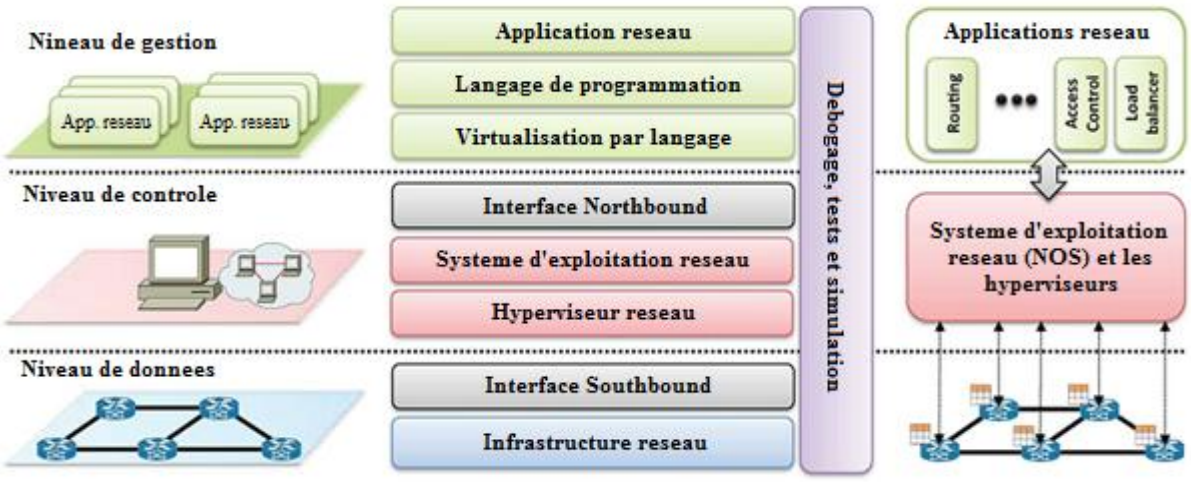



**Figure 19 :** *Architecture du réseau logiciel. [51]*

### 4.2.1. Couche infrastructure

L'infrastructure du réseau logiciel est similaire à celle des réseaux traditionnels. La seule différence réside dans l'enlèvement de l'intelligence et la prise des décisions des dispositifs qui composent le réseau ; en laissant ces dispositifs sans tâche autre que le transfert des données selon les règles données par le contrôleur qui s'en chargera de la partie « intelligence » du réseau.

Comme nous l'avons mentionné précédemment, les composants doivent être installés sur la même interface Southbound (Comme OpenFlow) pour permettre au contrôleur de communiquer les règles et les politiques de transfert avec eux [51].

### 4.2.2. Couche interface Southbound

Les interfaces Southbound sont le pont qui connecte le contrôleur avec les dispositifs du transfert. Ce qui les rend les vrais séparateurs du contrôle et le transfert de données. Cependant, ces interfaces restent toujours liées aux dispositifs.

L'industrie a heureusement accepté l'intégration de ce type d'interfaces dans les dispositifs. Car elles fournissent l'interopérabilité entre les différentes marques, et les différentes technologies. Ce qui permettre aux fabricants de concentrer moins sur ce sujet et focaliser plus sur l'amélioration de la performance de leurs produits. Cette interopérabilité entre les différents dispositifs de différentes marques, qui fournissent l'option d'utilisation du protocole OpenFlow, a été déjà montrée.

A l'instant, OpenFlow est l'interface la plus dominante, à cause de ses spécifications communes pour l'intégration de l'option « activation d'OpenFlow » dans les dispositifs de

transfert, et aussi pour le canal de communication utilisé entre le contrôleur et ces dispositifs. Le protocole OpenFlow fournit trois informations importantes au contrôleur :

- Des messages d'évènement lorsque les dispositifs détecte un changement des liens ou des ports de communication.
- Des informations périodiques ou à la demande sur l'état de flux.
- Des messages avec paquets si le dispositif ne reconnait pas le flux, ou il y a une règle explicite dans la table des flux qui force l'envoi des paquets au contrôleur s'il connait ce flux.

Il est à noter qu'il n'existe pas qu'OpenFlow comme interface Southbound. Il y a aussi d'autres interfaces qui sont créées pour atteindre certains objectifs. Nous citons : ForCES, POF, OVSDB, OpFlex et d'autres [51].

### 4.2.3. Couche hyperviseur du réseau

La virtualisation dans nos jours joue un rôle immense sur le plan technique et sur le plan commercial. Et malgré son évolution dans les domaines de stockage et de calcul, la virtualisation des réseaux reste presque intouchable. Cela est causé par la configuration des réseaux qui reste toujours statique.

On peut définir les exigences principales d'un réseau par deux dimensions : la topologie, et l'espace d'adressage. A l'instant, il est difficile pour une topologie physique de supporter la multitude des demandes d'applications et de services qui est l'objectif principale de la virtualisation. De même, il est clair que même si on installe une machine virtuelle sur un serveur pour opérer sur le même réseau, la machine doit toujours exister sur la même adresse dans l'infrastructure physique. Donc, pour fournir une virtualisation complète, les réseaux doivent fournir des propriétés similaires à celles du calcul ou du stockage. D'où, les réseaux logiciels entrent dans la scène avec ces propriétés introduites par les abstractions proposées par les interfaces Southbound.

Plusieurs solutions de virtualisation des réseaux logiciels sont proposées, et qu'elles sont testées dans des scenarios réels. Nous mentionnons à titre d'exemple : FlowVisor, AutoSlice, et NVP qui est proposé par la fameuse maison VMWare [51].

### 4.2.4. Couche système d'exploitation du réseau

Les systèmes d'exploitation traditionnels fournissent des abstractions pour l'accès aux dispositifs du bas-niveau et des mécanismes de sécurité. A l'instant, on reste toujours configurer et gérer les réseaux en utilisant le bas-niveau représenté par un ensemble d'instructions

spécifique aux dispositifs ou par des NOS fermés et licenciés (Comme Cisco IOS). Dans ce cadre, les réseaux logiciels sont promis pour faciliter la gestion des réseaux et la résolution des problèmes de réseautage par des moyens du contrôle logique et centralisé, offerts par le contrôleur.

Un contrôleur ou un NOS est un élément essentiel dans l'architecture des réseaux logiciels. Il supporte les applications qui génèrent les configurations et les instructions à transmettre aux dispositifs basées sur les politiques définies par l'opérateur du réseau. Il y a deux axes de développement de NOS :

- *Les NOS centralisés* : Ce sont des NOS installés sur un serveur qui gère tout le réseau. On cite NOX-MT, Maestro, Beacon, Meridian, ProgrammableFlow et Rosemary comme exemples.
- *Les NOS distribués* : Ce sont des NOS installés sur des nœuds choisis selon les clusters formés dans le réseau. Dans le but de partager les tâches et diminuer le risque d'échec de réseau trouvé dans l'architecture centralisée. A titre d'exemple, nous citons Onix, HyperFlow, ONOS, PANE, SmartLight et Fleet [51].

### 4.2.5. Couche interface Northbound

Le NOS peut offrir une interface pour les développeurs d'applications. Cette interface fournit des abstractions sur l'ensemble d'instructions à bas-niveau utilisé dans l'interface Southbound. Les interfaces Northbound ouvertes et standardisées jouent un rôle crucial dans la promotion de la portabilité et l'interopérabilité des applications dans des différentes plateformes. Elles jouent un rôle similaire à celui de POSIX dans les systèmes d'exploitation traditionnels.

Bien que la couche interface Southbound ait déjà l'OpenFlow comme une proposition acceptée par la majorité des actionnaires des réseaux logiciels, la couche interface Northbound reste toujours ouverte à la recherche et le développement. Et les chercheurs sont en accord que c'est tôt pour cette couche d'avoir un standard. L'expérimentation sur des différents contrôleurs sera la base pour trouver l'interface commune pour toutes ou la majorité des applications. Les interfaces Northbound existantes aujourd'hui sont liées aux NOS proposés. Elles ont leurs définitions spécifiques et elles restent indignes d'être des standards. Nous citons à titre d'exemple : NOSIX et SFNet [51].

### 4.2.6. Couche virtualisation par langage

L'objectif de cette couche est d'avoir la possibilité de voir le réseau d'une autre façon que sa vue globale fournie par le NOS. La simplification de vue engendra une simplification énorme

pour le développement des applications. Dans ce contexte, les langages de programmation qui fournissent ce type de virtualisation sont vus comme des langages du haut niveau. On cite à titre d'exemple le langage Pyretic qui introduit les « objets réseau » comme concept d'abstraction. Chaque objet réseau consiste à une topologie du réseau, et un ensemble de politiques (règles) associé à elle. Les objets réseau cachent simultanément ces informations et offrent les services demandées [51].

### 4.2.7. Couche langage de programmation

Comme pour les assembleurs, l'utilisation des instructions des interfaces Southbound force les développeurs de passer leur temps dans les détails au lieu de le passer dans la résolution des problèmes des réseaux. Les abstractions données par les langages du haut niveau peuvent aider les développeurs pour mieux concentrer sur leurs applications. En réseau logiciel, ces langages sont utilisés pour :

- Créer des abstractions à haut niveau.
- Assurer une meilleure productivité.
- Promouvoir la modularisation et la réutilisation du code.
- Encourager le développement de la virtualisation du réseau.

Il existe beaucoup de langages dont nous citons : Pyretic, Procera, NetCore, Flowlog, HFT et Maple [51].

### 4.2.8. Couche application réseau

Les applications réseaux implémentent la logique du contrôle qui va être traduite en commandes installées sur les dispositifs du transfert. Puisqu'on peut utiliser les réseaux logiciels dans n'importe quel environnement, on peut trouver ou développer une variété d'applications qui résolvent des problèmes comme le routage, la sécurité, et même la consommation de l'énergie.

Beaucoup d'applications sont faites en basant sur le réseau logiciel. On peut les classer en cinq catégories : La gestion du trafic, le sans fil, mesure et suivi, la sécurité, et le réseautage des centres de données. La table 1 montre quelques applications réalisées selon ces catégories [51].

**Table 1 :** *Applications réalisées pour les réseaux logiciels. [51]*

| Catégorie | Application | Objectif | Contrôleur | Southbound |
|---|---|---|---|---|
| Gestion du trafic | ElasticTree | Routage energy-aware | NOX | OpenFlow |
| | FlowQoS | QoS pour les réseaux de large bande | POX | OpenFlow |
| | MicroTE | Gestion de trafic avec un overhead minimal. | NOX | OpenFlow |
| | OpenQoS | Routage dynamique | Floodlight | OpenFlow |

| | | | | |
|---|---|---|---|---|
| | | avec QoS pour les applications multimédia | | |
| | Pythia | Gestion de trafic pour les applications big data. | OpenDaylight | OpenFlow |
| | QoS for SDN | QoS sur les réseaux hétérogènes | Floodlight | OpenFlow |
| | OSP | Recouvrement rapide | NOX | OpenFlow |
| | ALTO VPN | VPN à la demande | NMS | SNMP |
| **Sans fil** | AeroFlux | Contrôle pour les WLAN hiérarchique et extensible | Floodlight | OpenFlow, Odin |
| | CROWD | Le chevauchement des puces LTE et WLAN | ___ | OpenFlow |
| | CloudMAC | Traitement externalisé des MACs d'un WLAN. | ___ | OpenFlow |
| | FAMS | Système VLAN flexible basé sur l'OpenFlow | ProgrammableFlow | OpenFlow |
| | MobileFlow | Modèle basé sur les flux pour les réseaux mobiles | MobileFlow | SDMN API |
| | Odin | WLAN programmable et virtualisé. | Floodlight | OpenFlow, Odin |
| | OpenRoads | Contrôle de chemin des données en utilisant OpenFlow. | FlowVisor | OpenFlow |
| | SoftRAN | Equilibrage de charge et gestion d'interface | ___ | Femto API |
| **Mesure et suivi** | BISmark | Mesure active et passive | Procera | OpenFlow |
| | DCM | Suivi distribué et coactif du trafic | DCM Controller | OpenFlow |
| | OpenNetMon | Suivi des paramètres de QoS | POX | OpenFlow |
| | OpenSample | Mesure par échantillonnage à faible latence. | Floodlight | sFlow modifié |
| | OpenSketch | Niveau séparé pour la mesure des données. | OpenSketch | OpenSketch sketches |
| | OpenTM | Outil d'estimation de la matrice du trafic. | NOX | OpenFlow |
| | PaFloMon | Outils du suivi passif définis par l'utilisateur. | FlowVisor | OpenFlow |
| | PayLess | Plateforme du suivi à temps réel basée sur les requêtes. | Floodlight | OpenFlow |
| **Sécurité** | CloudWatcher | Plateforme pour la surveillance des clouds | NOX | OpenFlow |
| | DDoS detection | Détection et atténuation des attaques | NOX | OpenFlow |

| | | | | |
|---|---|---|---|---|
| | Elastic IP & Security | IP élastique et groupes de sécurité basés sur SDN | NOX | OpenFlow |
| | FlowNAC | Contrôle d'accès au réseau basé sur le flux | NOX | OpenFlow |
| | OrchSec | Architecture pour développer les applications de sécurité | *Tous* | Flow-RT, OpenFlow |
| | SANE | Fortification de la politique de sécurité | SANE Controller | SANE header |
| | SDN RTBH | Atténuation des attaques de DoS | POX | OpenFlow |
| | VAVE | Validation de l'adresse source avec la vue globale | NOX | OpenFlow |
| **Réseautage des centres de données** | Big Data Apps | Optimisation d'utilisation du réseau | ___ | OpenFlow |
| | CloudNaaS | Des primitives de réseautage pour les applications Cloud. | NOX | OpenFlow |
| | FlowComb | Prédire le taux d'activité ou la charge des applications | NOX | OpenFlow |
| | FlowDiff | Détecter les problèmes opérationnels. | FlowVisor | OpenFlow |
| | LIME | Migration directe des réseaux | Floodlight | OpenFlow |
| | NetGraph | Requêtes graphe pour la gestion des réseaux. | ___ | OpenFlow, SNMP |
| | OpenTCP | Adaptation d'un protocole TCP dynamique et programmable. | ___ | ___ |

### 4.3. Défis des réseaux logiciels

Avec tout le développement survenu au réseau logiciel, la recherche reste toujours active en essayant d'arriver au réseau logiciel complet, et plusieurs challenges existent encore. Nous résumons ces défis dans les points ci-dessous :

- *Conception des switches :* Principalement, la recherche essaye de trouver des solutions concernant l'implémentation hétérogène des différents switches, la maximisation de la capacité des tables de flux, l'amélioration des performances des switches qui utilisent OpenFlow, l'évolution de l'aspect Hardware, et la conception des switches spécifiques aux réseaux logiciels.

- *Plateformes des contrôleurs :* La recherche reste ouverte pour l'amélioration de la modularité et la flexibilité, l'interopérabilité et la portabilité des applications, et la délégation des contrôles.
- *Tolérance des erreurs :* La séparation en deux niveaux du réseau logiciel met en question la tolérance des erreurs. Principalement, la recherche essaye de trouver des solutions pour les cas où les contrôleurs tombent en panne, et aussi le délai de retard pour restructurer le réseau lorsqu'un dispositif tombe en panne.
- *Extensibilité :* Dans les réseaux à grande échelle, le contrôleur doit être capable de traiter des millions de flux par second sans compromettre la performance du réseau. De plus il y a un retard causé par le processus de configuration du flux. Ces deux sont les obstacles principaux pour l'amélioration de l'extensibilité des réseaux logiciels.
- *Evaluation de performance :* Avec l'expansion de l'intégration des réseaux logiciels, l'expérimentation sur cette technologie grandisse. Ce qui ouvre les portes à des nouveaux défis concernant les limites des performances des réseaux logiciels. Bien que la simulation et l'expérimentation soient les outils les plus utilisé pour évaluer les performances, La modélisation analytique a ses propres bénéfices. Pouvoir évaluer un modèle sans passer par l'expérimentation ou la simulation peut gagner du temps. Une autre question qui se pose aussi et qui reste ouverte à la recherche est « quel est le nombre de contrôleurs recommandé pour une topologie réseau et quels sont leurs emplacement ? ».
- *Sécurité :* Comme toute technologie du réseau, la sécurité est un défi qui existera toujours. Pour le réseau logiciel, les chercheurs ont identifié 7 vecteurs de danger (table 2) et 6 types d'attaques sur les réseaux qui se basent sur l'OpenFlow (table 3).

**Table 2 :** *Les vecteurs de danger trouvés dans les réseaux logiciels. [51]*

| Vecteur de danger | Description | Spécifique aux SDN ? | Conséquences sur les SDN |
|---|---|---|---|
| **Vecteur 1** | Faux flux du trafic dans le niveau de données | Non | Une porte ouverte pour les attaques DDoS |
| **Vecteur 2** | Exploitation de vulnérabilité des dispositifs de transfert. | Non | Possibilité d'inflation d'attaque. |
| **Vecteur 3** | Exploitation de vulnérabilité des contrôleurs centralisés | Oui | Exploiter les contrôleurs logiquement centralisés |
| **Vecteur 4** | Exploitation du positionnement des contrôleurs | Oui | Le compromis d'un contrôleur peut compromettre tout le réseau. |
| **Vecteur 5** | Exploitation de la similarité entre les systèmes d'exploitation traditionnels et les contrôleurs. | Oui | Développement et implémentation des applications malveillantes sur les contrôleurs |

| | | | |
|---|---|---|---|
| **Vecteur 6** | Exploitation de vulnérabilité des stations administratives | Non | Possibilité d'inflation d'attaque. |
| **Vecteur 7** | Manque de confiance en ressources d'analyse et de traitement contre les attaques. | Non | Impact négatif sur la détection des erreurs et le recouvrement rapide. |

**Table 3 :** *Les types d'attaques sur l'utilisation d'OpenFlow. [51]*

| Attaque | Propriété de sécurité attaquée | Exemple |
|---|---|---|
| **Spoofing** | Authentification | Spoofing des adresses IP et MAC |
| **Falsification** | Intégrité | Installation des fausses règles |
| **Répudiation** | Non-répudiation | Modification de l'adresse source. |
| **Divulgation d'information** | Confidentialité | Attaques sur un canal latéral pour détecter la configuration des règles de flux. |
| **Déni de service** | Disponibilité | Surcharger le contrôleur par les demandes de flux. |
| **Elévation des privilèges** | Autorisation | Pénétration sans autorisation au contrôleur. |

- *Autres Défis :* Il y a aussi d'autres types de défis rencontrés par les réseaux logiciels. Comme toute nouvelle technologie, son implémentation peut engendrer des problèmes sociaux, des employeurs peuvent perdre leurs postes. Dans cet égard, l'implémentation du réseau logiciel peut être confrontée par une certaine résistance. Et la recherche doit trouver des compromis pour ce genre de problèmes. Aussi, avec l'utilisation graduelle des nouvelles technologies comme le Cloud, les réseaux logiciels doivent être capables de les gérer.

## Conclusion

Dans ce chapitre nous avons présenté les notions de bases des réseaux cognitifs. Nous avons défini un réseau cognitif comme un réseau qui peut percevoir, s'adapter et estimer les futurs états de son environnement. Le réseau cognitif est composé des nœuds appelés cognitifs. Ces nœuds implémentent une conception des agents sur des plateformes physiques appelées radios logicielles.

Il existe aussi une nouvelle technologie de la même classe. Mais qui s'intéresse plus à la structuration des réseaux. Cette technologie nommée le réseau logiciel, a trouvé son chemin dans l'industrie, et plusieurs avancements sont faits à ce propos.

Les technologies de l'intelligence des réseaux sont très ouvertes pour la recherche. Plus particulièrement pour les réseaux cognitifs, plusieurs solutions sont présentées pour concevoir

des plateformes qui facilitent aux nœuds la détection et le partage des bandes spectrales libres. On note plus particulièrement, l'approche du Canal de Control Commun, qui est la plus utilisée, et qui permet aux nœuds cognitifs de communiquer et négocier l'allocation des canaux avant la transmission des données.

# Chapitre 2 :
# Optimisation multicritère des réseaux cognitifs

## Introduction

Comme nous avons vu dans le chapitre 1, plusieurs éléments entrent dans la structuration des réseaux cognitifs. La recherche dans ce domaine est toujours ouverte des deux côtés : physique et logiciel. Pour ce qui nous concerne, on concentre plus sur les réseaux cognitifs ad hoc. Car c'est là où on peut confronter réellement les besoins hétérogènes des nœuds dans un même réseau.

Plusieurs travaux sont faits concernant les réseaux cognitifs. Et plusieurs propositions d'architectures sont données dans ces dernières années. Dans cette section nous visons les travaux faites en considérant le multicritère. Pour cela, nous avons fait une recherche sur un intervalle de 100 articles dans la fameuse bibliothèque d'IEEE. Les mots clés utilisés sont : « cognitive radio », et « multi objective ». De plus, on a limité la recherche que pour les 10 dernières années (2006 à 2015). Les papiers sont ordonnés par pertinence. Et nous nous intéressons que par les articles qui traitent l'allocation des ressources et le routage unicast pour des critères techniques. Donc les articles comme [65] et [70] sont exclus bien qu'ils traitent le multicritère dans les réseaux cognitifs.

Le problème d'optimisation multicritère a ses propres règles concernant la recherche de la meilleure solution. La comparaison entre deux solutions n'est pas évidente car on ne connait pas sur quel critère elle va se baser. Pour cela, plusieurs méthodes sont proposées. Et elles sont classées en deux catégories : La catégorie qui essaye de garder toutes les solutions « non comparables » qui dominent tout l'espace de solutions, On appelle les méthodes qui appartiennent à cette catégorie les méthodes Pareto. Et la catégorie des méthodes Non-Pareto qui essayent de transformer la fonction objective multicritère en une fonction uni-critère pour permettre la comparaison entre les solutions. Dans ce rapport, nous allons classer les 13 travaux trouvés selon ces deux catégories.

## 1. Travaux basés sur les méthodes non-Pareto

Dans cette catégorie, les chercheurs essayent de transformer les problèmes avec multiple critères en un problème plus simple et avec un seul critère. Cette transformation rendra le

problème solvable avec les méthodes déjà utilisées dans l'optimisation de la performance des réseaux classiques.

Nous présentons 6 travaux. Les travaux [58], [62], et [63] sont basés sur la théorie des jeux. Les travaux [64] et [67] utilisent des algorithmes évolutionnaires comme méthodes d'optimisation. Tandis que le travail [66] essaye de modéliser son problème pour permettre l'utilisation des heuristiques ou des méta-heuristiques connues.

Dans l'article [58], les auteurs proposent une solution pour le problème d'optimisation du profit de la bande passante tout en maintenant une distribution juste du spectre entre les nœuds. Ils modélisent ce problème comme un jeu. Pour encourager la coopération entre les nœuds, la fonction d'utilité, calculée en utilisant les deux critères, est basée sur les prix et le crédit. Le nœud qui veut transmettre un paquet doit payer un prix, tandis que celui qui le reçoit gagne plus de crédit.

Dans l'article [62], les auteurs proposent une solution intéressante qui voit le réseau cognitif par les yeux des utilisateurs primaires (UP). La question cherchée est comment va un UP partager sa part du spectre avec les utilisateurs secondaires (US) sans dégrader son gain, sa qualité de service, et minimiser les délais d'attente des US ? Dans ce travail, les auteurs utilisent l'apprentissage par renforcement pour répondre à cette question. Le multicritère dans leur solution est transformé en uni-critère, en proposant une fonction objective qui englobe tous les critères de la question posée.

Les auteurs de l'article [63] proposent aussi une solution basée sur l'apprentissage par renforcement. Mais, différemment au problème visé par [62], les auteurs considèrent les liens réseau entre les nœuds. Ils veulent que les US puissent trouver leurs tables de routage dans un environnement avec une utilisation dynamique du spectre. Les critères considérés sont deux métriques de la qualité de service. Leur solution est basée sur le concept MORL (Multi Objective Reinforcement Learning). Mais car ils ne veulent qu'avoir une seule solution au lieu d'un ensemble de solutions, Il force un critère d'être une contrainte. Ce qui rend leur problème monocritère.

L'article [64] traite le même problème mais avec des critères différents. Les auteurs cherchent une topologie qui assure l'optimisation des deux critères : le débit, et les délais de transmission. Pour cela, ils utilisent l'algorithme de colonie de fourmis en mettant ces deux objectifs dans une seule fonction objective globale.

Le travail [66] traite aussi le choix de la route et le partage du spectre dans un réseau cognitif. Il considère deux critères : le débit, et la collision avec les UP. Comme dans le travail [63], les auteurs mettent le nombre de collisions comme une contrainte (il ne doit pas dépasser 1% à 3% de la transmission), ce qui rend le problème monocritère. La solution proposée est composée de deux étapes : La première étape est la recherche de la route qui a prouvé une meilleure performance pour le débit. La deuxième étape consiste à affecter les canaux libres aux liens qui composent la route pour maximiser le débit.

Les auteurs de l'article [67] proposent une architecture adaptative pour l'allocation des ressources. Elle est basée sur l'algorithme génétique. Les auteurs traitent 5 critères : la consommation de l'énergie, le bit error rate (BER), les délais de transmission, les interférences, et le débit. Pour la comparaison entre les solutions, les auteurs attribuent à chaque critère un poids, et ils proposent une fonction de fitness calculée par la somme pondérée de ces critères.

## 2. Travaux basés sur les méthodes Pareto

Les méthodes Pareto sont les méthodes les plus utilisées dans l'optimisation multicritère. Dans ces méthodes, on ne cherche pas une seule solution, mais tout un ensemble de solutions qui ne sont dominées par aucun autre élément dans l'espace de solutions étudié. On appelle cet ensemble le front de Pareto, d'où le nom « méthodes Pareto ». Pour ceux qui cherchent une seule solution, ils doivent former des préférences sur les critères. Et selon ces préférences, ils choisissent depuis le front de Pareto trouvé, la solution qui leur convient.

Il est presque reconnu comme une règle principale, pour les chercheurs, que l'allocation des ressources et le routage dans les réseaux cognitifs sont des problèmes d'optimisation multicritère. Donc, ce n'est pas surprenant de trouver des travaux qui essayent d'utiliser les méthodes Pareto pour résoudre ces problèmes. Dans notre recherche, nous avons trouvé 7 travaux qui appartiennent à cette catégorie. Les travaux [56], [57], [59], [60], [61] et [68] essayent de modifier l'algorithme génétique pour faire l'optimisation multicritère. Seul le travail [69] qui sort de la bulle en s'abstenant d'utiliser un algorithme évolutionnaire.

Le problème visé par l'article [56], est la recherche des meilleurs paramètres de la radio pour répondre aux multiples besoins des nœuds cognitifs. Les auteurs essayent d'améliorer l'algorithme génétique utilisé pour l'optimisation multicritère NSGA-II en ajoutant les caractéristiques des systèmes immunitaires. La simulation montre que leurs solution MIGA est mieux que NSGA-II.

Dans l'article [57], les auteurs cherche à optimiser l'utilisation du spectre. Ils argumentent que sous la condition d'un débit fixe, avoir une partie vide de la bande passante veut dire avoir une partie du spectre non exploitée. Du même, le manque de la bande passante engendre une augmentation de la force de transmission pour couvrir ce manque. Ils voient le problème comme optimisation des deux critères : la bande passante, et la force de transmission. Et ils proposent une solution similaire à [56] appelée MMGA.

Les auteurs de l'article [59] proposent une solution qui prend aussi la force de transmission et la bande passante en considération. La solution est une méta-heuristique évolutionnaire qui modifie la sélection des individus (solutions) utilisés pour le croisement et la mutation par deux étapes parallèles : La première appelée « clonage adaptative », qui crée plusieurs clones pour les individus de la sélection. Le nombre de clones produits pour chaque individu est calculé par son utilité (ou fitness). La deuxième étape appelée « recherche de voisinage », ajoute au groupe de sélection les voisines des individus appartenant à ce groupe. Après avoir le front de Pareto voulu par la méta-heuristique, Les auteurs utilisent l'algorithme flou pour choisir la solution la plus préférable.

Comme pour [56], [57] et [59], les auteurs de [60] proposent une amélioration pour l'algorithme génétique, en ajoutant une étape supplémentaire basée sur la recherche local ; dont le but est de raffiner les individus trouvés et de garder trace des solutions déjà utilisées. Les auteurs ne spécifient pas l'ensemble de critères à considérer, mais ils spécifient les paramètres qui forment les solutions, et qu'ils sont : la force de transmission, le type de modulation, et l'index de modulation. Dans leur simulation, les auteurs spécifient 3 critères : la consommation de l'énergie, le débit, et le BER.

Une autre solution hybride utilisée dans l'optimisation multicritère des réseaux cognitifs, est celle proposée dans l'article [61]. La solution utilise l'algorithme génétique basée sur la théorie du quantum (QGA). Et elle traite les trois objectifs communs : le débit, l'énergie, et le BER.

Dans l'article [68], les auteurs veulent inclure la probabilité de résiliation forcée, dans l'allocation du spectre, avec 3 autres fonctions objectives utilisées pour quantifier la qualité de service. La solution proposée est basée sur l'algorithme MODE pour la recherche du front de Pareto. Comme la plupart des méthodes Pareto, les auteurs n'offrent pas un mécanisme pour le choix de la solution parmi le front.

Dans l'article [69], les auteurs considèrent un réseau cognitif où les nœuds peuvent gagner de l'énergie depuis les fréquences radio quand ils sont en veille. Ils proposent une plateforme d'optimisation multicritère basée sur l'approche des poids de Tchebycheff. Ils veulent optimiser 3 critères : le gain d'énergie depuis les fréquences radio, la force de transmission, et les interférences, tout en prenant en compte une communication sécurisée. Donc, les auteurs mettent des contraintes sur les canaux pour focaliser le problème de sécurité.

## Conclusion

On a deux classes de propositions qui tiennent en compte l'aspect multicritère dans les réseaux de radios cognitives. La première classe est celle dite non-Pareto. Les propositions appartenant à cette classe essayent de transformer le problème multicritère vers un problème uni-critère. Cela est fait soit par englober tous les critères dans une seule fonction objective, ou bien par concentrer sur un critère en mettant les autres en contraintes. La deuxième classe est celle des modèles qui profitent directement des notions de l'optimisation multi-objective. Ils trouvent d'abord un ensemble de solutions appelé front de Pareto qui « domine » tout l'espace de solutions. Puis, selon leurs préférences, ils choisissent une qui donne les meilleurs résultats.

Nous remarquons que peu de propositions parmi celles trouvées qui traitent le routage. De plus, aucune de ces dernières ne tient en compte les besoins actuels des nœuds lors de la recherche de la meilleure structuration. Cela conduit à avoir les mêmes résultats même si les besoins des nœuds diffèrent. Bien que les modèles étudiés soient performants, la question reste toujours ouverte : Est-ce que ces solutions satisfont les besoins actuels des nœuds ?

Les travaux basés sur le front de Pareto sont mieux placés pour répondre à cette question. Car ils donnent tout un ensemble (front de Pareto) de solutions qui varient dans les axes des critères (ou des besoins). Cette variation permet au réseau de choisir une solution depuis cet ensemble qui satisfait le plus les besoins des nœuds. Dans ce cas, comment va-t-on modéliser ces besoins, et comment choisir en se basant sur ces besoins la solution la plus préférable pour notre réseau ?

# Chapitre 3 :
# Interaction robuste et efficace des systèmes autonomes et hétérogènes à base d'infrastructure réseau cognitive

## Introduction

Dans les chapitres précédents, nous avons présenté la radio cognitive comme le module de base de notre système. Comme il est expliqué, la radio cognitive est le système le plus autonome : à cause de son interaction directe avec son environnement, et le plus hétérogène : à cause de sa flexibilité envers le choix du canal (ainsi le choix des protocoles de communication) et aussi, à cause de ses besoins applicatifs qu'ils sont propres à lui. Pour ces raisons, nous modélisons l'interaction des systèmes autonomes et hétérogènes par un réseau à radio cognitif. L'objectif deviendra donc la recherche d'une architecture interaction robuste et efficace dans ce réseau.

Pour illustrer le problème visé par notre architecture, supposons la situation suivante :

Soit un nœud *A* qui veut envoyer *k* paquets à un nœud *B*. Il doit donc choisir un nœud voisin *X* et un canal libre *C* pour les envoyer, sachant que :

- *A* veut transmettre ses paquets à *B* dans les meilleurs délais.
- *A* ne donne pas beaucoup d'importance à l'énergie qu'il va consommer. Mais le nœud *X* veut économiser son énergie.
- *A* veut que ses paquets arrivent en toute sécurité.
- *A* ne veut pas qu'il soit interféré lorsqu'il est en train d'envoyer ses paquets.

Dans ce cas, comment savoir quel est le meilleur choix *(X, C)* pour le nœud *A* ?

Dans cet exemple, les nœuds du réseau ont leurs propres préférences concernant les quatre critères décrits (délai de transmission, énergie consommée, le niveau de sécurité voulu, et la qualité du lien). Un nœud qui veut transmettre des données, doit évaluer chaque choix (voisin, canal). Et selon ses préférences et aussi les préférences de ses voisins, il cherche l'action qui lui donne le maximum du gain pour ses propres critères.

Bien qu'on a trouvé plusieurs solutions pour des problèmes similaires au notre, Aucune de ces recherches traitent le problème d'hétérogénéité au niveau des besoins. En effet, aucun des travaux parle de la notion « besoin multicritère » ou essaye de proposer un modèle mathématique pour la représenter. Dans notre solution, nous essayons de le faire, tout en gardant à l'esprit la possibilité de l'améliorer, et la relier aux scenarios réels. Nous nous concentrons plus sur les deux couches 2 et 3 du modèle OSI. La représentation de ces besoins doit être calculée dans les couches supérieures. De plus bien qu'on ne représente que 4 critères dans ce projet, le modèle proposé peut être généralisé, car on définit chaque besoin séparément des autres. Ce qui donne à notre solution l'extensibilité voulue sur la multitude des critères inclus. Nous croyons que si la recherche finit par trouver le chemin entre les activités de l'utilisateur et la représentation proposée, nous nous approcherons de plus en plus vers une radio cognitive plus complète.

A la lumière de ces arguments, nous nous proposons deux modèles de solution : un qui sera centralisé basé sur le réseau logiciel, qui utilise un solveur de problèmes multicritères. Et un autre qui est distribué basé sur la théorie des jeux. L'objectif est d'utiliser le premier comme un modèle de comparaison et voir si notre proposition basée sur la théorie des jeux (le deuxième modèle) peut arriver aux mêmes résultats d'un solveur de problème multicritère.

Dans cette partie, nous présentons d'une façon abstraite le problème avec la représentation de l'environnement et des besoins. Puis nous décrivons le modèle centralisé basé sur le réseau logiciel. Après, nous présentons notre proposition distribuée qui est basé sur l'apprentissage par renforcement ; un concept de solution utilisé dans la théorie des jeux.

# 1. Spécifications générales

- On modélise notre réseau par un graphe générale $G\ (M, N, E)$, tel que :
  - $M = \{m_1, m_2, \ldots, m_M\}$ est l'ensemble des utilisateurs primaires (PU).
  - $N = \{n_1, n_2, \ldots, n_N\}$ est l'ensemble des utilisateurs secondaires (SU) qu'ils sont nos joueurs.
  - $E = \{e_1, e_2, \ldots, e_L\}$ est l'ensemble des liens radio entre les SUs.
- Chaque PU $m_i$ est connu par le canal qu'il utilise $c_i \in C = \{c_1, c_2, \ldots, c_M\}$ tel que $c_i$ est caractérisé par un quadruplet (la bande passante $b$, le taux d'énergie consommée $e$, la classe de sécurité $s$, la distance couverte $d$).
- Les besoins d'un nœud $n$ sont représentées par un quadruplet $B_n = (\alpha, \beta, \gamma, \delta)$ qui représente la distribution d'importance accordée à un critère par rapport à un autre $(\alpha+\beta+\gamma+\delta=1)$.
- Un paquet $q$ est caractérisé par sa taille $L_q$ et son niveau de sécurité voulu $s_q$.

# 2. Modèle centralisé

Dans le modèle centralisé, nous voulons rapprocher le problème visé à un problème déjà connu. L'objectif est d'avoir la possibilité d'utiliser les solutions pour des problèmes similaires dans la résolution de notre problème. Avoir une vision globale sur le réseau est un des avantages de la structure centralisé. Elle permet l'intégration de la solution, si on peut atteindre l'objectif décrit.

## 2.1. Modélisation du problème

On peut diviser notre problème en deux sous-problèmes. Le premier est la distribution des canaux entre les nœuds. Et le deuxième est le routage.

### 2.1.1. Distribution des canaux

Avant de chercher le meilleur chemin entre une source et une destination, on doit tout d'abord avoir la possibilité de choisir un. Donc, au premier lieu, on doit construire des liens fixes entre les nœuds du réseau. Ces liens sont définis par les canaux de communication utilisés entre les nœuds. Pour faire, on affecte à chaque nœud dans le réseau un canal. Donc, on peut facilement voir ce sous-problème comme un problème de coloriage de graphe (coloriage des sommets). Mais au lieu de chercher à minimiser le nombre de sommets voisins ayant la même couleur, on cherche à minimiser une fonction du coût multi-objective. Ce qui nous mène à l'optimisation multicritère pour un problème classique.

Dans l'optimisation multicritère, l'objectif est de trouver un ensemble de solutions où aucune ne soit dominée par une autre appartenant dans l'espace de solutions. On dit qu'une solution domine une autre, si la première est meilleure que l'autre quoi que ce soit le critère ou l'objectif comparé. L'ensemble de ces solutions est appelé le front de Pareto.

Pour notre problème, une solution consiste à un vecteur de N cases. Chaque case représente un nœud, et elle contient une couleur (un canal). La taille de l'espace de solutions égale à $M^N$ solutions possibles. Et comme nous l'avons dit, l'objectif est de trouver une seule solution. Donc, trouver et s'arrêter au front de Pareto ne résout pas le problème. On doit choisir, depuis cet ensemble, la solution qui satisfait le plus les besoins des nœuds. Pour cela, l'optimisation multicritère propose la notion de préférence, et des méthodes de classement de solutions pour répondre aux tels obstacles.

Dans ce qui suit, nous allons définir la fonction multicritère à optimiser. Puis, nous parlons de la solution préférée et comment la trouver.

#### 2.1.1.1. Fonction objective

On définit la fonction objective en une fonction multicritère comme suit :

$$Pour\ chaque\ solution\ x, f(x) = \left(Ct_{bp}(x), Ct_{en}(x), Ct_{se}(x), Ct_{in}(x)\right) \in \mathbb{R}^4$$

Tel que :

- $Ct_{bp}(x)$ est le coût total qui représente la perte de bande passante causée par la solution *x*.
- $Ct_{en}(x)$ est le coût total de l'énergie consommée lors d'utilisation de la solution *x*.
- $Ct_{se}(x)$ est le nombre de liens non sécurisés lors d'utilisation de la solution *x*.
- $Ct_{in}(x)$ est le nombre de possibilités d'interférence.

Une contrainte qu'on doit ajouter sur la solution ***x***, est que chaque nœud doit être capable de transmettre des paquets vers au moins un nœud.

##### A. Coût Total de Bande Passante

Soit un nœud ***i*** qu'on veut lui affecter un canal depuis la liste ***C***. Soit $\boldsymbol{b_{max}}$ la bande passante maximale qu'on peut l'avoir depuis ***C***. On propose le coût du choix d'un canal ***c*** par :

$$Ct_{bpi} = (1 - \frac{b_c}{b_{max}})$$

Le coût total de la bande passante est la somme de tous les coûts partiels du choix des canaux pour tous les nœuds. C'est-à-dire :

$$Ct_{bp} = \sum_{i=1}^{N} Ct_{bpi}$$

##### B. Coût Total d'Energie

Le calcul de ce coût est fait comme pour la bande passante. Donc on propose les coûts partiel et total par les deux équations :

$$Ct_{eni} = \frac{e_c}{e_{max}}$$

Et :

$$Ct_{en} = \sum_{i=1}^{N} Ct_{eni}$$

### C. Nombre de liens Moins Sécurisés

Pour le problème de sécurité, on suppose qu'on connait les positions des zones de danger, leurs surfaces, et leurs niveaux de danger. On définit alors une zone de danger $\boldsymbol{Z_i}$ par une centre ***f***, un diamètre ***l***, et un niveau ***ss***.

Soit un nœud ***n*** qui va utiliser un canal ***c*** ***(b,e,s,d)*** choisi pour communiquer avec ***y*** (***y*** est coloré par ***c***). On note ***d(n,f)*** la distance entre le nœud et le centre de zone $\boldsymbol{Z_i}$. On propose le calcul du coût partiel de ***n*** du choix du canal ***c*** pour le nœud ***y*** par :

$$\begin{cases} C_{sey} = 1\ , si\ d(n,f) \leq d + l, et\ ss > s, et\ d(n,y) \leq d \\ C_{sey} = 0, sinon \end{cases}$$

$$Ct_{sen} = \sum_{y=1}^{N} C_{sey}$$

Car on ne veut non seulement assurer des liens sécurisés, mais aussi assurer les liens les plus sécurisés, on ajoute au coût total la moyenne du niveau de sécurité assurée par les canaux choisis ($s_{c_i}$) :

$$Ct_{se} = \sum_{i=1}^{N} Ct_{sei} + \frac{1}{N} \sum_{i=1}^{N} s_{c_i}$$

### D. Nombre de possibilités d'interférence

On dit qu'il y a une possibilité d'interférence entre deux nœuds affectés au même canal, si les nœuds sont voisins, ou bien s'il existe un voisin commun qui peut les atteindre en utilisant le même canal.

Pour trouver cette possibilité, on teste un canal ***c*** pour un nœud ***n***, s'il y a au moins deux nœuds couverts par le champ du ***c***, et qu'ils utilisent le même canal ***c***, ou bien ***c*** = $\boldsymbol{c_n}$ le canal

affecté à ***n*** couvre un autre nœud utilisant le même canal, alors on dit qu'il y a une possibilité d'interférence. Le coût total $\boldsymbol{Ct_{in}}$ sera la somme de ces possibilités.

#### 2.1.1.2. Recherche de la solution préférée

La fonction objective est une fonction des coûts qu'on cherche à les minimiser. La première étape dans l'optimisation de cette fonction, est de chercher le front de Pareto. Car le problème est NP-complet (Coloriage de graphe), il n'existe pas une solution linéaire qui résout ce problème. Mais, il y a des heuristiques et méta-heuristique qui ont prouvé leur efficacité dans la recherche du front de Pareto. Pour notre problème, nous utilisons la méthode NSGA-II [52].

Après la génération d'un pseudo front de Pareto, on doit classer ses éléments selon une préférence. Le calcul de cette préférence dépend des besoins des nœuds. Soit la figure 20 ci-dessous de deux nœuds avec des besoins selon 2 critères.

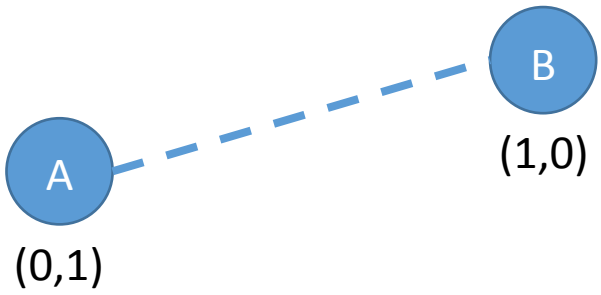



**Figure 20 :** *Exemple du réseau de deux nœuds avec un conflit de besoins.*

Comment calculer le besoin global ou la préférence dans ce cas ? Il est clair que la préférence doit être un compromis entre les deux besoins. Pour cela, nous avons choisi la moyenne pour la calculer, car :

- Elle assure le compromis entre les besoins en conflit.
- Elle montre la direction générale des besoins en accord.

On donne l'équation de la préférence globale par :

$$(\alpha, \beta, \gamma, \delta) = \frac{1}{N}\sum_{i=1}^{N}(\alpha_i, \beta_i, \gamma_i, \delta_i)$$

Pour ce modèle, nous allons utiliser la méthode décrite dans la section 3.3.2 qui viendra. Pour choisir la meilleure solution en utilisant le pseudo front de Pareto, et la préférence globale.

### 2.1.2. Calcul des table de routage

Le but de la distribution des canaux est de fournir une structure de réseau stable, qui satisfait d'une façon moyenne les besoins des nœuds. Avoir une telle structure, facilitera la recherche des chemins entre deux nœuds. Soudainement, on a un poids (multicritère) lors du choix du saut, car chaque saut est affecté à un canal. Ce qui donne la possibilité de transformer le problème de routage en un problème du plus court chemin, qui est un problème simple qui peut être résolut par l'algorithme de Djikstra. Les seuls défis sont : comment présenter la distance qui est basé sur des valeurs multicritères ? Et comment déterminer quel est le meilleur chemin (comment comparer entre deux distances multicritères) ?

Avant toute chose, nous devons définir la distance. Dans notre cas, nous définissons le meilleur chemin pour la transmission des paquets comme un chemin qui :

- Fournit une bande passante maximale.
- Ne consomme pas beaucoup d'énergie.
- Assure une sécurité maximale.
- A un nombre de sauts minimal.

Car ce sont les critères qui détermine l'utilité du chemin, la distance qui présente cette utilité, doit être basée sur ces critères. Dans ce cas, la distance entre A et B est composée de 4 paramètres :

- Le minimum de bande passante assuré entre A et B.
- La somme d'énergie consommée.
- La somme des sauts non sécurisés.
- Le nombre de sauts effectués.

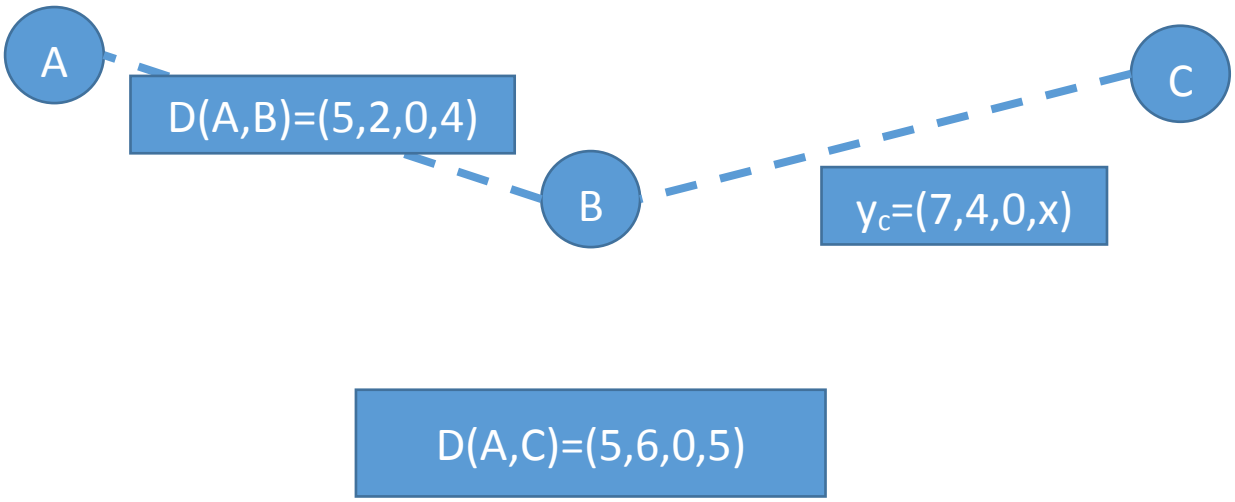



**Figure 21 :** *Exemple de calcul de distance.*

Le calcul de cette distance est illustré par l'exemple de la figure 21. Soit un nœud A qui veut communiquer avec un nœud C. Sachant qu'on a calculé déjà la distance entre A et un nœud B. Et que le nœud B peut communiquer directement avec le nœud C en utilisant le canal $y_c$ affecté à C. Quel est la distance entre A et C en passant par B ?

Dans cet exemple, nous avons la distance D (A,B) = (5,2,0,4). Cela veut dire que la bande passante existante entre A et B égale à 5, la somme d'énergie au long du chemin égale à 2, Le chemin est totalement sécurisé, et on effectue 4 sauts pour arriver à B. Aussi, on a $y_c$, le canal avec les caractéristiques suivantes : (7,4,0,X). (X pour la distance réelle couverte par $y_c$ qu'elle n'a pas d'importance pour le calcul). En suivant les règles données, on a :

D (A,C) = (min(5,7), 2+4, 0, 4+1) = (5,6,0,5).

Pour le troisième paramètre, l'utilisation du canal $y_c$ n'engendre aucun risque de sécurité car il n'y a pas de zones de danger. Pour cette raison, quoi que ce soit le besoin de sécurité demandé pour le paquet, le nombre de sauts non sécurisés reste 0.

Maintenant qu'on a une méthode pour calculer les distances entre les nœuds. Un problème surmonte : Comment comparer entre deux chemins ? Supposons pour un instant, qu'il y a un autre chemin de A vers C avec une distance (3,3,0,4). Quel est la meilleure distance ? (5,6,0,5) ou (3,3,0,4). Pour répondre à cette question nous devons voir les critères utilisés. On a deux critères qu'on veut toujours optimiser : l'énergie, et le nombre de sauts. Et les deux autres (bande passante, et sécurité) sont relatifs aux besoins du nœud transmetteur (ou initiateur de la communication). Donc, le chemin préféré pour le transmetteur A sera celui qui suit la préférence ($\alpha_A$,1,$\gamma_A$,1). Tel que : $\alpha_A$ et $\gamma_A$ sont calculées depuis les besoins du nœud A. Après avoir la préférence, une utilisation de la méthode de classement décrite dans la section 3.3.2 suffira pour trouver le meilleur chemin.

Bien que nous parlions d'un seul chemin, l'algorithme de Djikstra est conçu pour créer l'arbre de couvrement minimal, qui est le cheminement de A vers tous les autres nœuds. Si nous connaissons le niveau de sécurité voulu pour tous les paquets, on peut facilement donner des tables de routages complètes aux nœuds, ce qui est plus efficace que le routage à la demande.

## 2.2. Intégration de la solution

La modélisation du problème et les solutions proposées, nécessitent un cerveau centralisé qui peut voir tout le réseau et tout l'environnement, pour pouvoir exécuter ces solutions. On veut aussi, garder l'aspect ad-hoc des nœuds. Pour ces raisons, le réseau logiciel est l'architecture parfaite pour cette solution. On peut profiter de la structure du canal de contrôle commun pour faire la différence entre le canal utilisé pour la communication avec le contrôleur, et les canaux de données qui vont être distribués entre les nœuds.

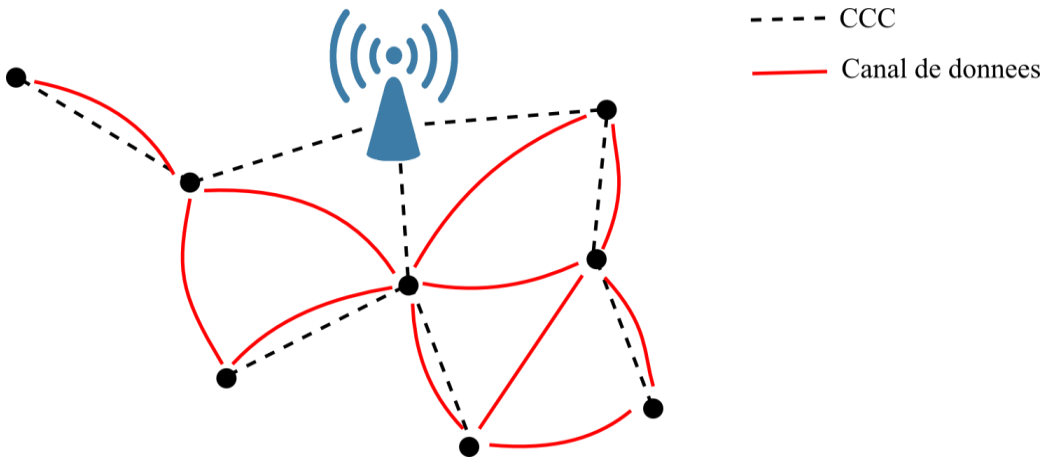



**Figure 22 :** *Architecture générale du modèle centralisé.*

Le contrôleur sera le moniteur de notre réseau. Il sera le responsable à collecter les informations sur l'état des nœuds, et selon cet état, il va choisir le meilleur partage du spectre entre ces nœuds. De plus, il va être le responsable de choisir la meilleure route entre deux nœuds qui veulent être connectés. Donc on peut résumer son fonctionnement par la figure ci-dessous.

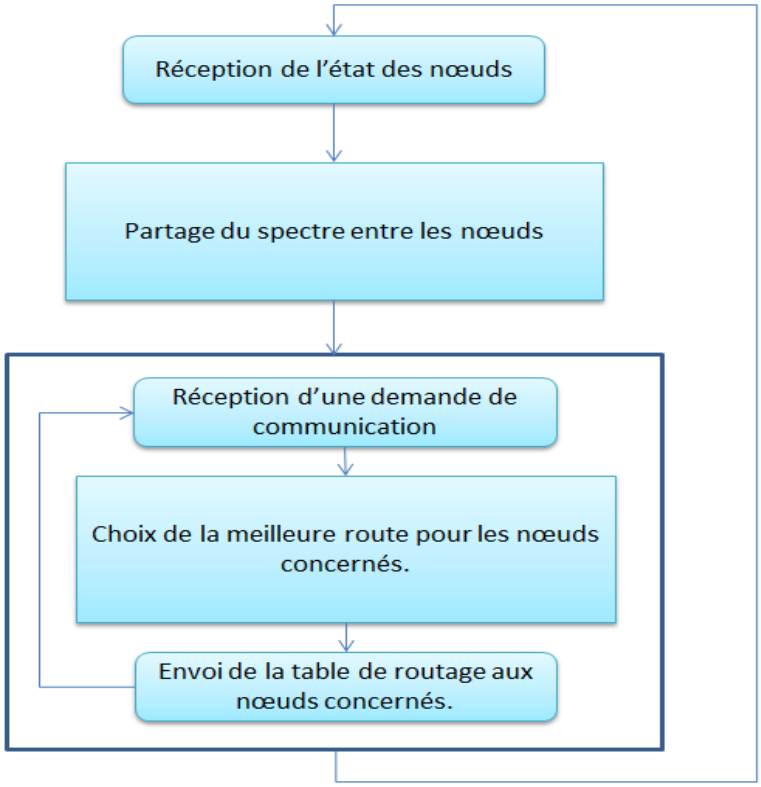



**Figure 23 :** *Les tâches du contrôleur.*

## 3. Modèle distribué

Dans notre modèle distribué, on modélise le réseau en un jeu répété, où chaque nœud est un joueur, avec un ensemble *A* d'actions. Chaque action est définie par le voisin qu'on veut lui envoyer les paquets, et le canal dans lequel on veut les transmettre. Dans chaque tour, le joueur évalue les actions selon ses préférences (besoins), selon son environnement et selon les tendances des autres joueurs, il calcule les gains (ou bien les coûts) pour ces actions, puis il choisit l'action la plus favorable pour lui.

L'article [53] traite le même problème mais en se limitant à un seul critère : le délai de transmission. Dans notre solution nous utilisons le même modèle du jeu. Mais, au lieu de se limiter à un seul critère, nous proposons une solution multicritère.

Dans notre réseau nous ne voulons pas chercher la solution optimale pour ce contexte multicritère. Mais on veut chercher une solution qui satisfait les besoins des nœuds. Si les nœuds veulent un délai de transmission court, le modèle doit être capable de réagir ainsi. Si les nœuds veulent que tous les critères soient optimisés, le modèle doit être capable aussi à le faire.

Dans la littérature de la théorie des jeux, Pour atteindre un équilibre de Nash, Il existe deux concepts de solution utilisés dans les jeux répétés : le jeu fictif, et l'apprentissage par

renforcement. Dans le premier concept, Le joueur construit des croyances sur les stratégies « souvent » utilisées par les autres joueurs. Et selon ces croyances, il choisit sa meilleure action. Dans le deuxième concept, Le joueur choisit son action en se basant sur ses propres expériences. Il n'anticipe pas les actions et il n'a pas de croyances sur les autres joueurs. L'idée est que les joueurs tendront à choisir l'action qui a prouvé son efficacité (utilité) précédemment.

Bien que dans l'article [53] les auteurs ont utilisé le jeu fictif comme un concept de solution, Nous utilisons l'apprentissage par renforcement. La raison de ce choix est que dans le jeu fictif, les joueurs construisent des probabilités sur les actions de leurs voisins. Le calcul de ces probabilités nécessite une interaction plus active entre les joueurs pour savoir qu'elles sont leurs actions et améliorer les croyances, ce qui peut engendrer des obstacles pour la performance globale du réseau. De plus, la fonction d'utilité devient plus complexe pour notre cas, à cause du multicritère. Alors que, si on choisit l'apprentissage par renforcement comme concept, on gagne un peu de marge sur le partage d'information, car les joueurs se basent plus sur leurs propres expériences. Aussi, car c'est l'expérience qui définit l'utilité des actions, on peut toujours garder une fonction multicritère pour les expériences. Ce qui veut dire que pour une action, on peut calculer l'expérience gagnée pour chaque critère isolement. Et on peut se relier sur les méthodes de l'optimisation multicritère pour choisir la meilleure action selon les préférences des joueurs.

## 3.1. Spécifications

- Chaque nœud *n* aura deux matrices essentielles : $R_n$ qui représente ses ressources, et la matrice $S_n$ qui représente les canaux libres à utiliser (Spectrum Opportunity Matrix) :

$$R_n(i,j) = \begin{cases} 1 & Si\ le\ lien\ e_i\ existe\ et\ le\ canal\ c_j\ est\ libre. \\ 0 & Sinon. \end{cases}$$

$$S_n(i,j) = \begin{cases} 0 & Si\ l'utilisation\ du\ canal\ c_j\ dans\ le\ lien\ e_i \\ & interfere\ avec\ le\ PU. \\ 1 & Sinon. \end{cases}$$

- On considère un voisinage à deux sauts. Et les nœuds voisins partagent leurs préférences qui doivent être prises en considération si un nœud veut communiquer. Donc, un nœud *n* aura un tableau $P_n$ tel que $P_n(i) = B_i$.

### 3.2. Calcul des actions permises $A_n$

Dans la littérature, les chercheurs obligent que deux nœuds ne doivent pas s'interférer, même s'ils sont secondaires [53]. Bien que cela simplifie le problème des interférences, cette proposition va causer des problèmes de communication entre les nœuds secondaires lorsque les ressources deviendront très limitées. Du coup, les nœuds vont attendre tout leur quantum pour pourvoir au moins compéter sur ces ressources, alors qu'ils peuvent juste utiliser un protocole « classique » pour le partage du même canal (comme le CSMA/CA).

Pour cela, nous considérons que seules les actions qui n'interfèrent pas avec les PUs sont permises. Donc, on a la matrice des actions $A_n$ calculée par le produit par élément :

$$A_n = R_n \otimes S_n$$

Lorsque le nœud *n* détecte un PU, il doit :

- Changer son action.
- Mettre à jour $R_n$.
- Notifier les voisins pour qu'ils mettent à jour leurs $S_i$.

### 3.3. Choix de l'action

Dans l'apprentissage par renforcement, il existe deux types de décisions : une qui permet au joueur de gagner de l'expérience sur les actions non choisies auparavant (Exploration), et une autre qui permet au joueur d'utiliser ses expériences pour choisir la meilleure action (Exploitation). Dans les deux cas, le joueur va mettre à jour les expériences gagnées par son choix. Une des approches les plus connues au domaine des réseaux cognitifs dans l'apprentissage par renforcement est l'ε-greedy. Cette méthode propose ε comme une probabilité très petite d'exécution d'une action d'exploration. Et le calcul de l'expérience reçue est la moyenne entre l'expérience précédente qui est multiplié par (1 – ε) la probabilité de non exploration et le gain reçu par l'exécution de l'action multiplié par ε (on considère chaque action exécutée comme une exploration dans le calcul des expériences) [54].

L'équation générale de l'expérience sera :

$$Q_n^{t+1}(a) = (1 - \varepsilon)Q_n^t(a) + \varepsilon r_n^{t+1}(a)$$

On peut dire que $\varepsilon$ désigne l'influence de la nouvelle action sur l'expérience. Si $\varepsilon = 1$, le nœud *n* négligera ses expériences du passé, le gain reçu dans la nouvelle action sera son expérience.

Pour accélérer la convergence vers l'équilibre de Nash, on initialise les expériences avec des gains maximaux (ou des coûts minimaux). Ce qui oblige les joueurs de toujours choisir les actions inexplorées au début.

### 3.3.1. Exploration

Dans cette phase, le nœud choisit une action d'une façon ***aléatoire***, dans le but d'avoir des expériences et améliorer ses connaissances. On note $Q_{n,x}$ la matrice des paiements locaux ou des Q-valeurs (qui représente l'expérience) du joueur *n* pour un critère *x*. Chaque case dans cette matrice représente une action *a (i, c)*, et elle contient un vecteur de valeurs pour chaque destination *y* (si c'est nécessaire). On notera $Q^t_{n,x}((i,c),y)$ cette valeur dans un temps $t \in T = \{0,1,2,\ldots\}$.

#### 3.3.1.1. Q-valeurs du critère α (délai de transmission)

Le temps de transmission est le critère le plus important dans le routage en CRN, et plusieurs travaux sont faits concernant ce critère par rapport aux autres. Car le temps de transmission d'un paquet n'est connu que lors de son arrivé à sa destination (c.à.d. que lorsque le paquet prend tout le chemin), on doit séparer l'expérience gagnée pour le délai de transmission en deux : l'expérience propre au nœud α', et l'expérience des voisins α''.

Pour l'expérience du nœud, nous utilisons l'équation de calcul de Q-valeur pour le temps *t+1* :

$$Q^{t+1}_{n,\alpha'}(i,c) = (1-\varepsilon)Q^t_{n,\alpha'}(i,c) + \varepsilon d^{t+1}_{n,k}(i,c)$$

On donne l'équation du temps de transmission d'un paquet *k* depuis un nœud *n* par l'action *(i, c)* comme suit [55]:

$$d_{n,k}((i,c)) = (1+err)\frac{L_k}{b_c} + \varphi_{n,i}$$

Où $\varphi_{n,i}$ est la durée prise par le paquet *k* depuis son arrivé à *n* jusqu'à son sortie vers le voisin *i*, et *err* est la probabilité d'avoir une erreur de transmission.

De plus, les nœuds partagent leurs services à leurs voisins, en leur donnant le minimum délai assuré pour une destination *y* (expérience des voisins). Le nœud *n* garde les services proposés par les voisins *i* dans une matrice $Q^t_{n,\alpha''}(i,y)$. Le calcul du service est simple, on donne son équation par :

$$Service^t_i(y) = Q^t_{i,\alpha''}(i,y) = \min\left(Q^t_{i,\alpha'}(j,c) + Q^t_{i,\alpha''}(j,y)\right) * (buffer+1)$$

Où *buffer* représente le nombre de paquets qui reste dans le buffer. Si le buffer est vide, le délai proposé par le nœud i n'inclura que le paquet que les nœuds veulent transmettre. Par cette variable, on assure que les joueurs vont choisir un chemin moins congestionné pour leurs paquets, ce qui va surement améliorer les délais de transmission.

On peut dire donc que l'expérience du nœud *n* pour une action *(i, c)* pour envoyer des paquets à une destination *y* est :

$$Q^t_{n,\alpha}((i,c),y) = Q^t_{n,\alpha'}(i,c) + Q^t_{i,\alpha''}(i,y)$$

#### 3.3.1.2. Q-valeurs du critère β (consommation d'énergie)

Pour l'énergie, le nœud connait déjà les coûts, donc il ne va pas les apprendre. Mais néanmoins, il doit calculer ses Q-valeurs. Comme il est indiqué dans le premier exemple, il doit tenir en compte les préférences sur l'énergie du nœud qu'il veut communiquer avec.

Contrairement au critère $\alpha$, où on choisit l'action selon la destination, on ne s'intéresse que par ce que le voisin désire. Si on consomme *e* d'énergie lors de l'utilisation d'un canal, le coût sera *2e*, car le voisin va l'utiliser lui aussi. Mais si ce voisin ne donne pas d'importance à ce critère, on peut donc penser que le coût sera que *e*. On veut minimiser l'énergie, éviter les nœuds qui donnent plus d'importance à ce critère sera bénéficial. Dans ce cas, on doit utiliser les préférences des voisins pour déterminer les coûts d'énergie.

Car la somme des besoins égale à 1, il existe un petit piège qu'on doit l'éviter. Pour le montrer, on donne un exemple où on considère 2 critères *(a,b)* : soit deux nœuds avec des besoins *(0,1)* et *(0.5,0.5)*. Bien que le premier donne une importance absolue à *b* par rapport à *a* contrairement au deuxième, ils accordent tous les deux une importance maximale à *b*. Donc on peut dire que le deuxième nœud accorde une importance totale à *a* et *b*. alors que le premier néglige totalement *a*.

Pour connaitre l'importance d'un critère pour un nœud, on ne peut pas se relier sur les valeurs des besoins. Dans cet exemple, on doit trouver le **taux d'importance** de *b* par rapport à *a*. Dans ce cas, on doit juste diviser *a* sur *b*. On trouve alors *(0,1)* et *(1,1)*. De même pour les besoins *(0.25,0.75)* et *(0.75,0.25)*, on trouve les taux d'importances *(0.33,1)* et *(1,0.33)*. Pour généraliser, on divise les préférences sur le critère le plus préférable (besoin maximal) pour trouver les taux d'importance.

Soit $\theta_\beta(x)$ le ***taux d'importance*** de l'énergie pour un nœud *x*. On donne le coût d'une action *(i, c)* comme suit :

$$Q_{n,\beta}^{t+1}(i,c) = Q_{n,\beta}^{t}(i,c) = \left(1 + \theta_{\beta}(i)\right) e_c$$

#### 3.3.1.3. Q-valeurs du critère γ (sécurité)

Dans ce projet, Le problème de sécurité visé est l'évitement des attaques passives. Dès qu'on détecte une menace, le nœud doit évaluer son niveau du danger, et choisir le canal le plus sécurisé pour envoyer les données. Pour cela, on suppose que les nœuds ont un mécanisme pour détecter et évaluer le danger.

Le critère de la sécurité prend les caractéristiques des deux critères mentionnés avant. Tandis qu'il y a un apprentissage à faire, il y a aussi la connaissance du nœud sur le niveau de sécurité assurée $s \in S = \{1,2,3,\ldots,K\}$ (Le niveau 0 veut dire une sécurité extrême) par un canal lors de la transmission des paquets.

Dans notre cas, les zones de danger apparenteront aléatoirement. Quand un nœud détecte une telle zone, il évalue son niveau et avertit ses voisins. Pour un nœud *n* l'information peut être inconnue donc il doit s'adapter à ces évènements. Il est clair que si le nœud envoie un paquet vers une zone de danger, les nœuds relais vont changer leurs actions pour assurer sa sécurité. Ce qui va surement affecter la performance par rapport aux autres critères.

Comme pour la matrice des interférences avec les PUs, on considère une matrice de sécurité qui ne permettra que les actions qui assure le niveau voulu par l'émetteur du paquet. Aussi, le niveau de sécurité minimal (le plus grand nombre dans S) assuré par une route sera propagé entre les relais pour améliorer leurs connaissances (comme pour le délai de transmission).

On considère donc :

$$SE_n(i,j) = \begin{cases} 0 & Si\ l'utilisation\ du\ canal\ c_j\ dans\ le\ lien\ e_i \\ & ne\ securise\ pas\ le\ paquet. \\ 1 & Sinon. \end{cases}$$

$$Q_{n,\gamma}^{t+1}(a,y) = (1-\varepsilon) Q_{n,\gamma}^{t}(a,y) + \varepsilon r_{n,\gamma}^{t+1}(a,y)$$

$$r_{n,\gamma}^{t+1}((i,j),y) = \max_{i \in n \to y} (s_j)$$

#### 3.3.1.4. Q-valeurs du critère δ (interférences)

Dans le critère des collisions, le nœud veut choisir l'action qui lui donne un minimum d'interférence avec les autres nœuds. Quand deux nœuds envoient des paquets en même temps, les récepteurs ne vont pas pouvoir détecter le signal. Si un nœud aura un timeout en attendant un acquittement, il utilise alors le mécanisme du CSMA/CA pour retransmettre leurs paquets, et il incrémente le nombre d'échecs de transmission.

Comme pour l'énergie, les nœuds ne sont pas consternés par leurs récepteurs finaux des paquets mais par leurs voisins directs. Donc, on considère l'équation des Q-valeurs comme suit, tel que le paiement reçu *r* est le nombre d'échecs de transmission apparues pour l'action *a* :

$$Q_{n,\delta}^{t+1}(a) = (1-\varepsilon)Q_{n,\delta}^{t}(a) + \varepsilon r_{n,\delta}^{t+1}(a)$$

### 3.3.2. Exploitation

Dans cette phase qui est souvent exécutée, les nœuds choisissent leurs meilleures actions. Cela sera fait en utilisant les Q-tables qui sont mises à jour chaque fois qu'on choisit une action (aléatoire pour exploration ou meilleure pour exploitation). Le problème dans notre cas est comment savoir quelle est la meilleure action avec une expérience multicritère ?

Car on parle du multicritère, trouver la meilleure action n'est pas une tâche simple même que lorsqu'on a les préférences du nœud. Pour faire, on doit utiliser une méthode de classement proposée pour l'optimisation multicritère.

Soit le nœud *n* qui veut envoyer *k* paquets vers un nœud *y*. La première étape est de déterminer les actions permises avec l'équation donnée précédemment toute en prenant en considération la matrice de sécurité :

$$A_n = R_n \otimes S_n \otimes SE_n$$

Pour chaque action $A_n(i,c) = 1$, on a le quadruplet :

$$(Q_{n,\alpha}(i,c,y), Q_{n,\beta}(i,c), Q_{n,\gamma}(i,c,y), Q_{n,\delta}(i,c)).$$

Car on veut exécuter l'action avec le minimum de toutes ces valeurs, on doit au début utiliser la méthode d'élimination des stratégies dominées (théorie des jeux) pour avoir l'ensemble de Pareto des actions. Puis, on applique la méthode de classement suivante qui prend en charge les préférences du nœud :

On transforme ces quadruplets qui caractérisent les actions en quadruplets de gains$(g_\alpha, g_\beta, g_\gamma, g_\delta)$. Le calcul de ces gains, va montrer les taux de satisfactions du nœud pour chaque critère s'il choisisse une action par rapport à une autre dans son ensemble de Pareto. Une action qui donne une satisfaction de 100% (gain = 1) pour un critère est l'action qui a la Q-valeur minimal dans cet ensemble. On peut donner les équations de calcul des gains comme suit :

Pour un critère *x*, et l'ensemble des actions *A* :

$$\begin{cases} g_x = 1 \; si \; Q_{n,x} = \min\limits_{(a \in A)} Q_{n,x}(a) \\ g_x = \dfrac{\min\limits_{(a \in A)} Q_{n,x}(a)}{\max\limits_{(a \in A)} Q_{n,x}(a)} \; si \; Q_{n,x} = \max\limits_{(a \in A)} Q_{n,x}(a) \end{cases}$$

Par résolution de l'équation de la droite, on trouve :

$$g_x = 1 - \frac{Q_{n,x} - \min\limits_{(a \in A)} Q_{n,x}(a)}{\max\limits_{(a \in A)} Q_{n,x}(a)}$$

Après avoir les gains associés aux actions, les joueurs doivent ordonner ces actions selon leurs gains, et selon leurs besoins. Car on veut que la quantification des actions reste toujours en quadruplets (pas de transformation en uni-critère), les préférences seront déterminées par la dominance. On dit qu'une action ***a*** est plus préférable qu'une autre ***b*** si ***a*** réponds aux besoins du nœud plus que ***b.***

On propose cet algorithme pour classer les actions et trouver la meilleure :

- Trouver ***X*** l'ensemble de Pareto de ***A***.
- Calculer les gains des éléments de ***X***.
- Calculer les taux d'importance des critères pour le nœud.
- Ordonner les éléments de ***X*** selon les préférences du joueur :
    - Calculer les ***facteurs de gain*** selon l'ordre d'importance des critères (gain du critère le moins important / gain de plus important). Si les critères on le même taux d'importance, alors un ordre de priorité doit être défini.
    - Faire la différence entre le taux et le facteur.
    - Ordonner ces différences en ordre croissant (le couple avec la plus petite différence est le plus préférable) selon l'ordre d'importance.
- La meilleure action est la première dans l'ordre.

Donnons un exemple pour illustrer comment trouver la meilleure action un ensemble où les éléments sont multicritères avec cette méthode :

Soit un ensemble des actions A = {1 : (18,5), 2 : (10,10), 3 : (19,11), 4 : (5,18), 5 : (14,17)}. Dans cet exemple, on veut maximiser (Et non pas minimiser, mais le raisonnement est

le même), donc pour ces couples on trouve l'ensemble de gains G = {(0.95, 0.28), (0.53, 0.56), (1, 0.61), (0.26, 1), (0.74, 0.94)}. Et supposons que les besoins du joueur ayant cet ensemble d'actions sont donner par le couple (0.5, 0.5).

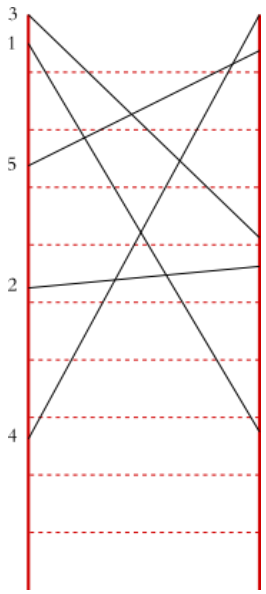
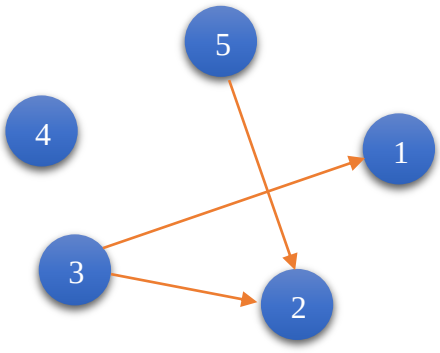
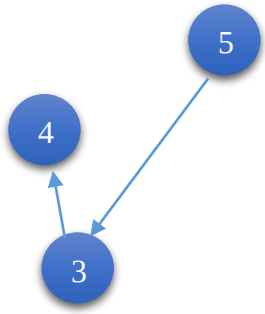
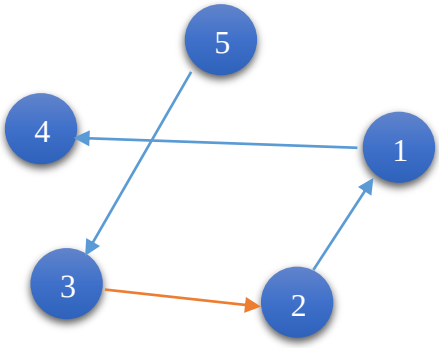



**Figure 24 :** *Une représentation géométrique des gains.*

En appliquant cet algorithme dans notre exemple, on aura le déroulement suivant :

**Table 4 :** *Deroulement de la methode de classement proposee.*

| | | |
|---|---|---|
| Initialement |   | L'ensemble de Pareto = {3,4,5}<br><br>Les taux d'importance :<br><br>(1,1) (car les préférences sont (0.5,0.5))<br><br>L'ordre d'importance sera indifférent (Dans ce cas on priorise le critère 1 sur le critère 2 |
| Apres l'exécution de l'algorithme | | Les facteurs de gains :<br><br>3 : (1, 0.61) - 4 : (1, 3.85) |

| | | |
|---|---|---|
| |  | 5 : (1, 1.27)<br><br>La différence avec (1,1) :<br>3 : (0, 0.39) - 4 : (0, 2.85)<br>5 : (0, 0.27)<br><br>L'ordre sera donc :<br>5 > 3 > 4 |
| Si on applique l'algorithme d'une manière itérative en tirant la meilleure action dans chaque itération.<br><br>L'ordre sera donc :<br>5 > 3 >> 2 > 1 > 4 |  | Observation : La seule vraie dominance qui nous reste dans cet ordre est celle entre 3 et 2. |

Si on regarde cet exemple, on remarque que c'est plus préférable de choisir l'action 5 car elle nous donne plus d'équilibre concernant les besoins (0.5, 0.5), puis l'action 3. Bien que 2 n'est pas dominée par 1 ou 4 mais elle est plus favorable qu'eux. Car elle nous donne plus d'équilibre. Et bien que 2 nous donne plus d'équilibre que 5 et 3, la dominance de 3 et de 5 sur 2 conclut que le joueur devra choisir 5 et 3 et non pas 2. Le seul point à noter est la préférence de l'action 1 sur l'action 4. Par somme pondérée 4 doit être supérieure à 1 (0.5*(0.95 + 0.28) < 0.5*(0.26 + 1)). Cela est le résultat de l'ordre d'importance entre critères. Car on priorise le 1$^{er}$ critère sur le 2$^{eme}$, on finit par avoir 1 > 4.

### 3.4. Schéma du nœud

Après avoir expliqué notre contribution, on doit maintenant l'intégrer dans l'architecture du nœud. Pour cela, nous utilisons la plateforme décrite dans le chapitre 1 (section 3.4.2). De

plus, on garde le même schéma proposé dans l'article [53]. Le schéma général du nœud sera donc celui donné dans la figure :

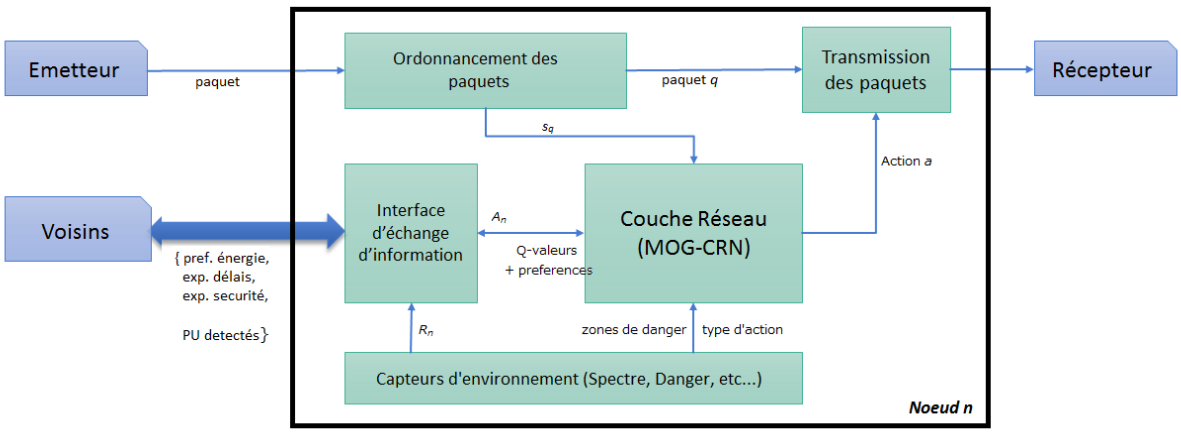



**Figure 25 :** *Schema general du noeud.*

Comme nous pouvons le voir dans ce schéma, le calcul de l'action à exécuter est fait dans le bloc MOG-CRN. Les entrées de ce bloc sont l'ensemble des actions permises $A_n$, les distances des zones de danger, Le type d'action (exploitation ou exploration), et le niveau de sécurité minimal requis par le paquet à envoyer. Selon ces informations, combinées avec les expériences du nœud et les préférences des voisins, le nœud choisit son action comme il est décrit précédemment (section 3.3.2). La matrice S, qui sera multipliée par la matrice R (multiplication élémentaire), est calculée en utilisant l'information « PUs détectés » inter-changée entre les voisins. Cette information est représentée par un vecteur logique de taille M (le nombre de PU / Canaux dans le réseau).

La durée de vie des nœuds est divisée en quantums. Chaque quantum est composé de trois phases : la phase de détection, la phase de coordination, et la phase de transmission. La première phase consiste à l'extraction des informations depuis les capteurs, et former selon ces informations les objets nécessaires (la matrice R, le vecteur de zones de danger, et le type d'action). La deuxième phase consiste à deux étapes : la première est d'utiliser toutes les informations partagées par les nœuds voisins pour calculer les objets nécessaires (ensemble d'action permises et la mise à jour des Q-tables). La deuxième est de concevoir les paquets d'échange d'information (les paquets de contrôle) selon ces objets pour les diffuser aux voisins. Dans la troisième phase, les nœuds utilisent les données mises à jour dans la deuxième phase pour envoyer/recevoir les paquets de données. Quel que soit la phase, si le nœud reçoit un paquet de contrôle il le garde dans un buffer spécial pour l'utiliser dans la première étape de la deuxième phase.

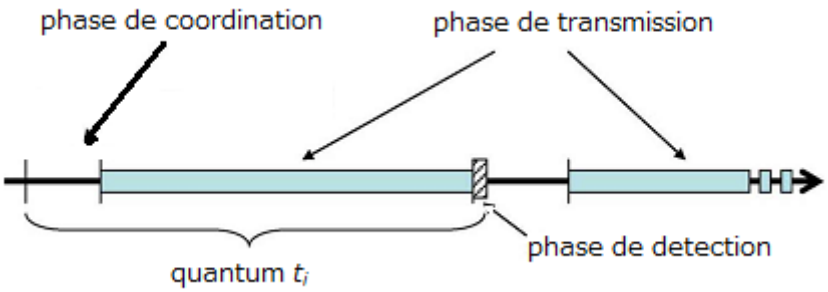



**Figure 26 :** *Representation du quantum.*

## 3.5. Algorithmes des évènements

Pour notre solution, nous avons six évènements à modéliser : la transmission du paquet de contrôle (CTRL), la réception de ce paquet, le traitement des paquets de contrôle, la transmission des paquets de données (DATA), la réception des paquets de données, et le choix de l'action.

On a deux groupes de paquets, un qui sera transmis dans le canal de contrôle commun, et l'autre dans un canal de données. Pour le premier groupe (paquets CCC), on a deux types de paquets : le paquet de contrôle (CTRL), et le paquet de demande d'envoi (RTS). Pour le deuxième groupe (paquets données), on a trois types de paquets : le paquet de permission d'envoi (CTS), le paquet de donnée (DATA), le paquet d'acquittement (ACK).

La structure du paquet CCC se compose de 5 champs :

- Adresse de destinataire DEST.
- Adresse de l'émetteur SRC.
- Type de paquet TYPE (un paquet CTRL ou un paquet RTS).
- Les données à envoyer dans ce paquets DTA (le canal et le nombre de paquets à envoyer si TYPE = RTS, et les informations à partager avec les voisins si TYPE = CTRL).
- Le nombre de sauts que le paquet doit fait dans le réseau SAUT (nécessaire pour les paquets CTRL, il définit de niveau de voisinage).


**Figure 27 :** *Structure du paquet CCC.*

La structure du paquet de données de compose de 8 champs :

- Le destinataire DEST.
- L'émetteur SRC.
- Le type de paquet TYPE (CTS, DATA, ou ACK)
- Le destinataire final DEST_F (nécessaire pour les paquets DATA).
- L'émetteur original SRC_O (nécessaire pour les paquets DATA).
- Le numéro séquentiel du paquet à transmettre N_SEQ (N'est pas nécessaire pour les paquets CTS).
- Les données du paquet DTA (si TYPE = ACK, on envoi le numéro de paquet suivant à transmettre).
- Le niveau de sécurité minimal demandé par l'ensemble de paquets minSE.

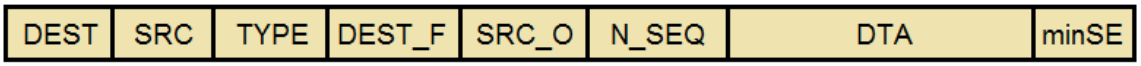



**Figure 28 :** *Structure du paquet de données.*

Dans ce qui suit, nous présentons les algorithmes de ces six évènements avec l'utilisation de ces paquets.

### 3.5.1. Transmission du paquet CTRL

Le paquet CTRL est un paquet diffusé sur le canal de contrôle commun (CCC) par le nœud dans le but de partager les informations nécessaires avec le voisinage. La partie donnée de ce paquet se compose de :

- L'importance de l'énergie pour le nœud.
- Le pire délai de transmission que le nœud peut assurer pour envoyer un paquet vers une destination.
- L'expérience gagnée au niveau de sécurité lors de l'envoi vers une destination.
- La présence des utilisateurs primaire dans le voisinage.

Donc l'algorithme de cet évènement est le suivant :

| *Algorithme 1* : Transmission des paquets CTRL |
|---|
| - *dta* ← **Nouvelle structure donnée** ;<br>- *dta*.prefEN ← Taux d'importance de l'énergie pour le nœud actuel ;<br>- *dta*.maxDL ← vecteur des délais maximaux vers toutes les destinations ;<br>- *dta*.maxSE ← vecteur d'expérience pour la sécurité du nœud actuel ;<br>- *dta*.vecPU ← vecteur de détection des utilisateurs primaires ;<br>- *pck* ← **Nouveau paquet CCC vide** ;<br>- *pck*.DEST ← ADR_DIFF ; /* adresse de diffusion */ |

- *pck*.SRC ← adresse du nœud actuel ;
- *pck*.TYPE ← CTRL ;
- *pck*.DTA ← *dta* ;
- *pck*.SAUT ← le niveau de voisinage ; /* dans notre cas, un voisinage de 2 sauts */
- **Régler l'antenne pour transmission sur le CCC** ;
- **Attendre un temps aléatoire** ; /* les nœuds partagent le même canal */
- **Transmission (*pck*)** ;

### 3.5.2. Réception du paquet CTRL

Lors de la réception d'un paquet CTRL, le nœud vérifie combien de sauts il le reste. Si le nombre de sauts restants est diffèrent de 0, il le diffuse. Puis il met le paquet dans un buffer spécial *buffer_ccc*.

*Algorithme 2* : Réception des paquets CTRL

- *pck* ← **Le paquet CTRL reçu** ;
- *pck*.SAUT ← *pck*.SAUT – 1 ;
- *buffer_ccc*[case_vide] ← *pck* ;
- **Si** *pck*.SAUT ≠ 0 **Alors**
  - **Régler l'antenne pour transmission sur le CCC** ;
  - **Attendre un temps aléatoire** ;
  - **Transmission (*pck*)** ;
- **Fin Si** ;

### 3.5.3. Traitement des paquets CTRL

Lors de la phase de coordination, le nœud utilise tous les paquets existants dans buffer_ccc pour mettre à jour les Q-tables la matrice S.

*Algorithme 3* : Traitement des paquets CTRL

- **Pour chaque** *pck* **dans** *buffer_ccc* **faire**
  - $Q^{t}_{n,\alpha''}(pck.\mathrm{SRC},:) \leftarrow pck.\mathrm{DTA.maxDL}$;
  - $Q^{t}_{n,\beta}(pck.\mathrm{SRC},:) \leftarrow (1 + pck.\mathrm{DTA.prefEN}) * e_c$ ; /* $e_c$ : vect. d'nrg des canaux */
  - Sauvegarder *pck*.DTA.maxSE dans une matrice pour les expériences des voisins ;
  - $S_n(pck.\mathrm{SRC},:) \leftarrow pck.\mathrm{DTA.vecPU}$ ;
  - **Vider la case de *buffer_ccc*** ;
- **Fin Pour** ;

### 3.5.4. Choix de l'action

Avant de transmettre un ensemble de paquets, les nœuds doivent chercher l'action appropriée. Une action est définie par le voisin auquel le nœud va transmettre l'ensemble de paquets, et le canal utilisé pour la transmission. Comme nous l'avons précisé, le choix de l'action est influencé par le type de décision à faire (exploration ou exploitation), les expériences reçues

par les choix précédents, et l'ensemble des actions permises. Ce choix est fait dans le bloc MOG-CRN.

*Algorithme 4* : MOG_CRN (Choix d'action)

- Trouver la matrice SE, selon la sécurité requise par l'ensemble de paquets ;
- **A ← R .* S .* SE** ; /* A est l'ensemble des actions permises, (.*) = mul. élémentaire*/
- Trouver la décision D selon le nombre de quantum exécuté et la probabilité ε.
- **Si** D = Exploitation **Alors**
  - Depuis les Q-tables, former des quadruplets d'expérience pour chaque action dans A ;
  - Transformer le quadruplet de besoins en quadruplet de préférences ;
  - Ordonner les besoins selon ces préférences ;
  - Transformer les quadruplets d'expériences en quadruplets de gain pour les actions ;
  - Trouver l'ensemble de Pareto *Pr* dans A ;
  - Ordonner les actions dans *Pr* par leurs gains, on respectant l'ordre des besoins ;
  - **Choisir la première action dans l'ordre** ;
- **Sinon** /* Si D = Exploration */
  - **Choisir aléatoirement une action depuis A** ;
- **Fin Si** ;

### 3.5.5. Transmission des paquets de données

Après choisir le voisin et le canal, le nœud commence à transmettre ses paquets. Pour faire, il doit passer par plusieurs étapes de communication. Le nœud doit envoyer un paquet RTS sur le CCC à son voisin, pour demander la permission de lui transmettre les paquets, en l'indiquant le canal voulu et le nombre de paquets. Après un certain temps, si le voisin ne répond pas à la requête par un paquet CTS sur le canal de transmission voulu, le nœud doit arrêter la communication et cherche une autre action. Sinon, Il commence à envoyer ses paquets DATA. Pour chaque ensemble de paquets envoyé, il attend un acquittement (ACK), qui l'indique le prochain paquet à envoyer. Après un certain nombre d'échecs successifs pour le premier acquittement qu'il attend, le nœud abandonne le canal et cherche une autre action pour retransmettre les données. Si la transmission est faite en succès, le nœud saura que le récepteur reçoit sans aucun problème, il complètera donc l'envoi des paquets jusqu'à la réception du dernier paquet ACK. A la fin, le nœud libère le canal, il calcule les coûts de communication, puis il met à jour ses Q-tables selon ces coûts. La figure 29 montre le passage entre les états pour le transmetteur des données.

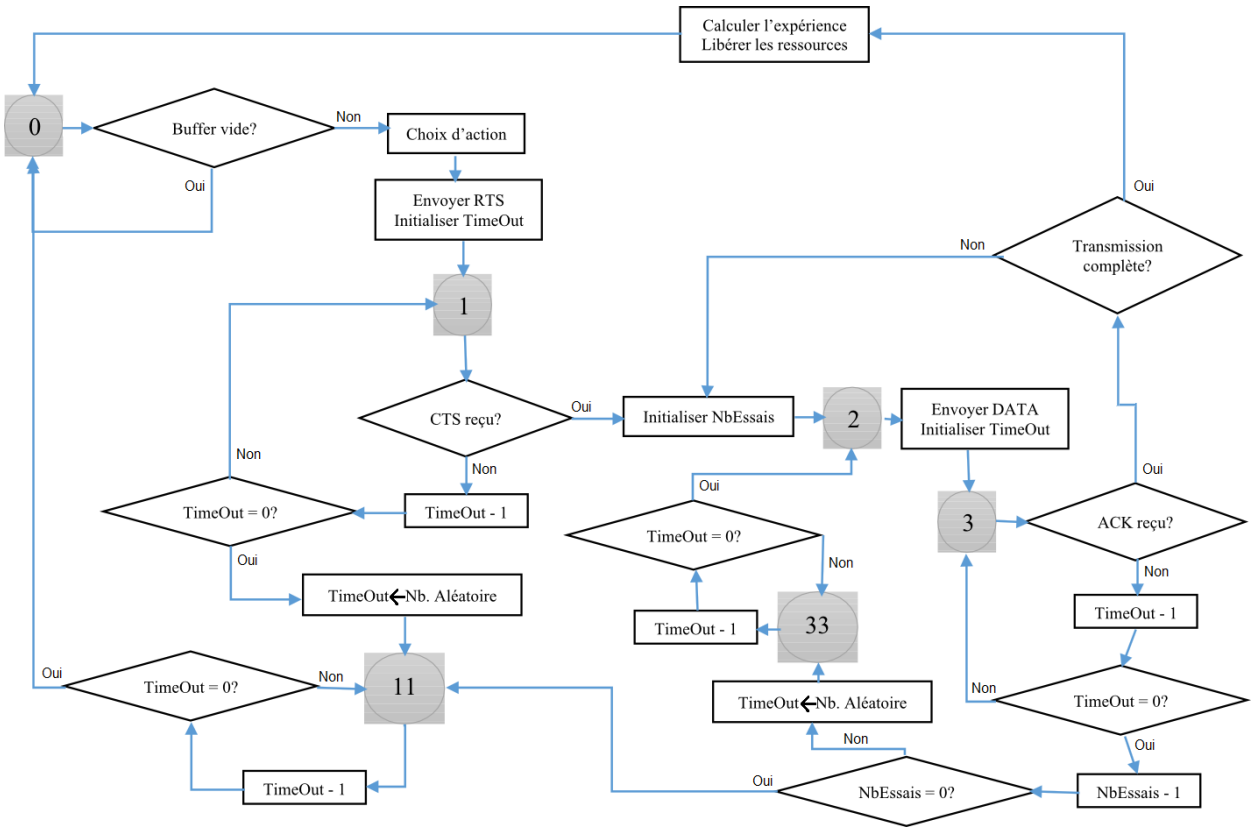



**Figure 29 :** *Organigramme de transmission de données.*

### 3.5.6. Réception des paquets de données

Dans le côté récepteur, le traitement est presque le même. S'il reçoit un RTS dans son état d'inactivité, il reconfigure son antenne pour être près à la communication, puis il envoie un CTS. Pour chaque ensemble de paquets reçu, il envoie un paquet ACK et il attendra un certain temps. Si aucun paquet n'arrive, le nœud libère le canal en rejetant tous les paquets reçus. Si on arrive à recevoir le dernier paquet, on envoie le dernier paquet et on attend un certain temps pour assurer que le transmetteur a reçu le dernier acquittement. Si aucun paquet n'est retransmis, le nœud ajoute l'ensemble de paquets dans son buffer en marquant le temps de son arrivé, et il se repositionne dans son état initial de réception. La figure 30 montre ce passage entre les états de réception.

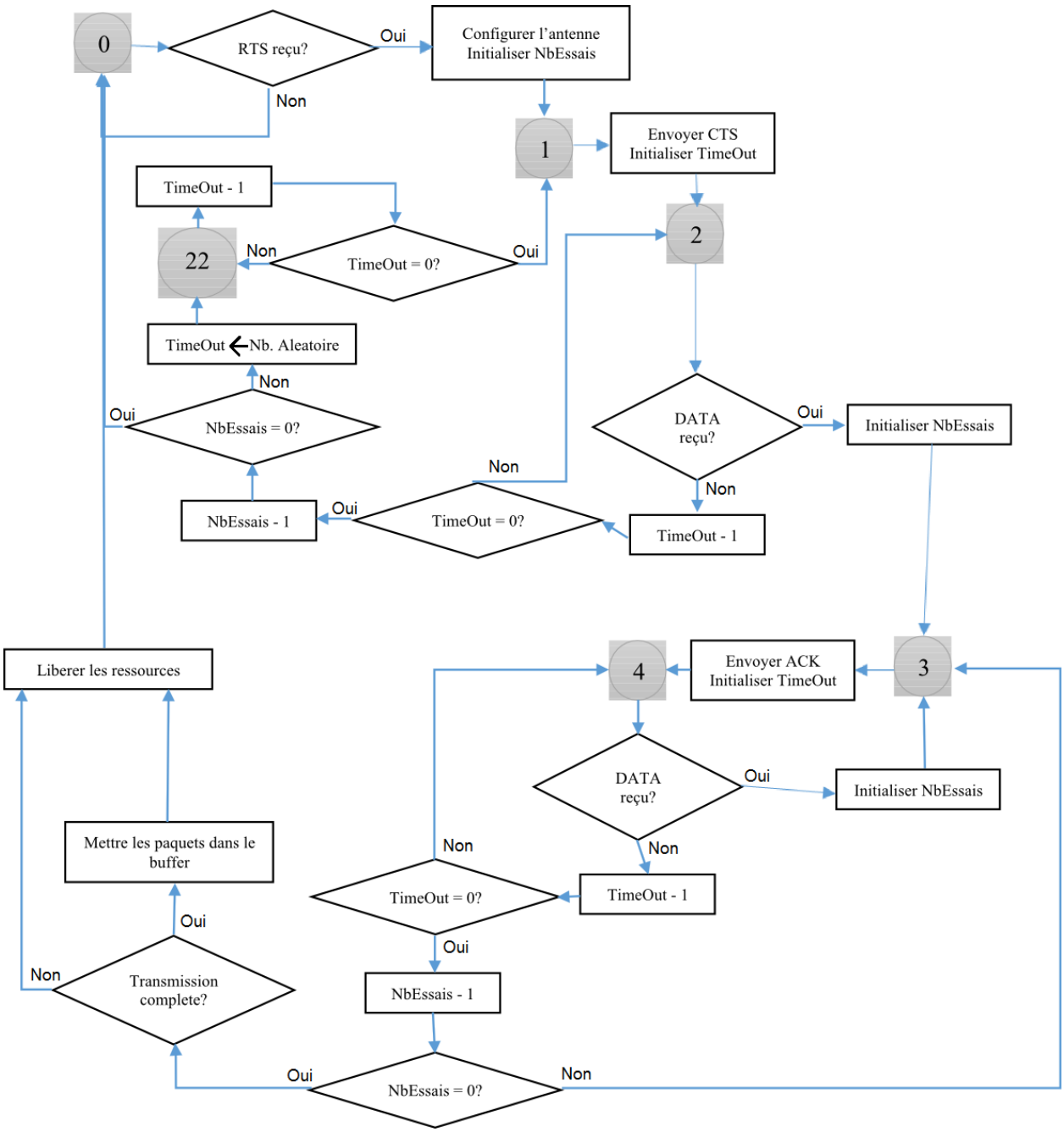



**Figure 30 :** *Organigramme de réception de données*

# Conclusion

Dans ce chapitre, nous avons proposé deux solutions pour l'interaction des systèmes autonomes et hétérogènes, représentés principalement par des nœuds qui intègrent une radio cognitive.

La première solution est centralisée et basée sur le réseau logiciel. Son objectif est d'étudier la possibilité de transformer le problème visé en un, ou plusieurs problèmes connus qui ont des solutions efficaces. Une chose qu'on a démontré en utilisant l'optimisation multicritère,

et la théorie des graphes comme base. La deuxième solution, qui est distribuée et basée sur la théorie des jeux, consiste à une solution qui veut exploiter le noyau de la radio cognitive. En éliminant toute dépendance des points d'accès ou des contrôleurs, les nœuds cherchent à se situer dans le réseau et satisfaire leurs propres besoins.

# Chapitre 4 :
# Simulation et résultats

## Introduction

Dans ce chapitre, nous présentons les modèles de simulation pour les deux solutions proposées. Nous essayons d'introduire les conditions des expérimentations pour les deux modèles, et la comparaison des résultats obtenus. A la fin, nous essayons de tirer des conclusions selon nos observations sur ces résultats.

Il est à noter que le choix de la simulation est due au fait que c'est la méthode la plus utilisée par les chercheurs concernant la structuration des réseaux à radios cognitives. La raison principale est que cette technologie n'est pas encore mise en implémentation dans la pratique. Et peu de travaux qui ont essayé de l'implémenter.

## 1. Outils utilisés

Pour les réseaux, il existe beaucoup de plateformes de simulation. Nous citons à titre d'exemple NetSim, TinyOS, et la fameuse plateforme et langage de programmation MatLab ; qui et l'outil utilisé pour construire nos simulations.

Nous utilisons MatLab R2013a comme plateforme. Avec la modélisation à temps discret, l'objet système proposé par la plateforme, et un langage de programmation mis en place pour être facile, l'utilisateur peut simuler presque tout. Et avec les outils de construction des graphes et des charts, la plateforme MatLab facilite la visualisation des résultats.

## 2. Conditions de l'expérimentation

Car le problème est multicritère, l'objectif de cette simulation est de voir le comportement des deux modèles : centralisé et distribué, pour des différents types de besoins des nœuds. Pour cela, nous utilisons un seul benchmark du réseau : celui utilisé dans l'article [53].

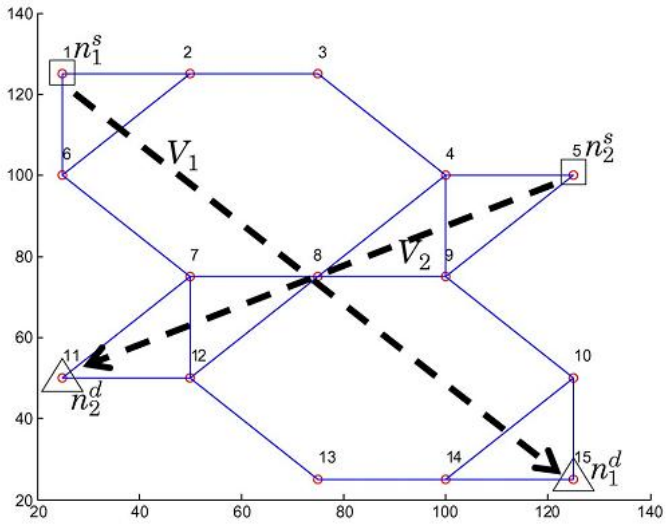



**Figure 31 :** *Topologie réseau utilisée dans la simulation. [53]*

La simulation est faite avec un temps discret. Chaque itération ou instant consomme ***ts*** = 50ms. Dans ces 50ms, les nœuds passent d'un état vers un autre. Le réseau est composé de 15 nœuds répartis dans un espace de 100x100 $m^2$, 6 canaux de données crées aléatoirement et un canal CCC. La table 5 ci-dessous montre les paramètres de ces canaux :

**Table 5 :** *Liste des canaux utilisés*

| ***Canal*** | ***B. passante (Kb/ts)*** | ***Energie (mJ)*** | ***Niv. sécurité*** | ***Distance (M)*** |
|---|---|---|---|---|
| ***CCC*** | **800** | **-** | **-** | **36** |
| ***C1*** | 1320 | 132.78 | 3 | 36 |
| ***C2*** | 1800 | 136.06 | 4 | 36 |
| ***C3*** | 1045 | 115.9 | 1 | 52.364 |
| ***C4*** | 2280 | 136.69 | 4 | 36 |
| ***C5*** | 1045 | 127.46 | 2 | 36 |
| ***C6*** | 950 | 114.45 | 0 | 57.6 |

Il est à noter que nous avons essayé de se rapprocher le plus possible de la réalité lors de la création des canaux. Si le canal offre une bande passante basse en traversant une distance petite (une modulation basse pour une grande fréquence), alors le niveau de sécurité va être mauvais, et l'énergie consommée sera relativement grande (augmentation de la force de transmission pour atteindre les voisins). Aussi, nous ignorons l'énergie consommée et le niveau de sécurité offerts par le CCC.

Pour les interférences, nous nous intéressons plus sur les collisions des paquets de données. Pour cela, nous supposons que les paquets CTS et ACK peuvent interférer avec les paquets DATA. Mais le contraire n'est pas valide. Aussi, La raison pour les collusions des paquets qui passent par le CCC, est pour simuler les trous d'information causés par ces collisions lorsque les

nœuds diffusent leurs expériences au même temps, ou pour simuler le comportement du réseau lors des échecs de communication.

Pour le modèle centralisé, on met la population à 100 individus. L'algorithme génétique NSGA-II s'exécute sur 200 générations. Aussi, on met 10% comme taux de mutation pour le modèle centralisé, et 10% aussi comme taux d'exploration pour le modèle distribué. Lors que le contrôleur calcule la solution, on configure les nœuds et on commence la simulation. Pour le cas du modèle distribué, la simulation démarre avec un état vide, sans aucune solution prédéfinie.

Les nœuds émetteurs essayent d'envoyer 300 fichiers dans une demi-heure, la taille de chaque fichier est 5000 Kb. Chaque lien établi entre deux nœuds ne sera coupé que lorsque tout le fichier est envoyé. Dans le cas d'un échec de transmission d'une partie du fichier, tout le fichier sera rejeté. La raison pour ces contraintes est d'utiliser la bande passante au maximum (nous ne changeons pas le lien pour envoyer les paquets du même fichier) tout en gardons à l'esprit l'aspect sécurité (chaque fichier a un niveau de sécurité prédéfini) et les pires échecs possibles pour la transmission des fichiers (perte totale des paquets).

Aussi, nous allons exécuter la simulation 6 fois : 4 fois où on donne de l'importance à un seul critère, une fois avec une importance équitable entre les critères, et une fois avec une génération aléatoire des besoins pour les nœuds. Dans chaque test, nous allons présenter la performance du réseau pour chaque critère.

Dans chaque test, on calcule :

- La bande passante moyenne allouée au réseau
- Le débit moyen reçu par les nœuds puis.
- L'énergie moyenne consommée par les nœuds.
- La sécurité moyenne assurée par les nœuds.
- La moyenne du nombre d'échecs de transmission par seconde dans le réseau.

# 3. Résultats obtenus

## 3.1. Test 1 : Importance totale au débit

Dans ce test, nous supposons que tous les nœuds ont un besoin total pour le débit. Pour cela, on leurs attribue tous le quadruplet (1,0,0,0). La table 6 montre les résultats pour la simulation des deux modèles :

**Table 6 :** *Résultats de la simulation lorsque des noeuds demandent plus de débit.*

| ***Modèle*** | **Bande Passante** | **Débit** | **Energie** | **Sécurité** | **Erreurs de transmission** |
|---|---|---|---|---|---|
| *Centralisé* | 1778 | 818.84 | 669.61 | 2.98 | 3.88 |
| *Distribué* | 1378 | 872.73 | 1544.6 | 2.50 | 8.89 |

Le modèle centralisé essaye d'augmenter le débit en utilisant deux principes : maximiser l'utilisation des canaux avec les grandes bandes passantes, et minimiser le nombre de sauts. Alors que le modèle distribué essaye de le faire par la maximisation de l'utilisation des canaux aux grandes bandes passantes en évitant les congestions. Malgré cette différence de perspective, les deux modèles proposent presque le même débit moyen, avec un peu d'avantage pour le modèle distribué. Car le chemin des paquets est fixé, la performance dégrade avec le temps pour le modèle centralisé. La figure (32.a) montre cette dégradation. Alors que le deuxième modèle essaye toujours d'améliorer jusqu'à la stabilisation (Figure 32.b), ce qui le met dans une meilleure performance. Il est aussi clair que la mauvaise gestion de la congestion dans le modèle centralisé a causé une mauvaise exploitation de la grande bande passante maximisée. Par contre, même si la bande passante offerte par le modèle distribué n'est pas optimisé comme pour le modèle centralisé, la stabilité offerte par l'algorithme de l'apprentissage par renforcement, avec la bonne gestion des congestions, rend le système plus efficace lors de l'exploitation de la bande passante.

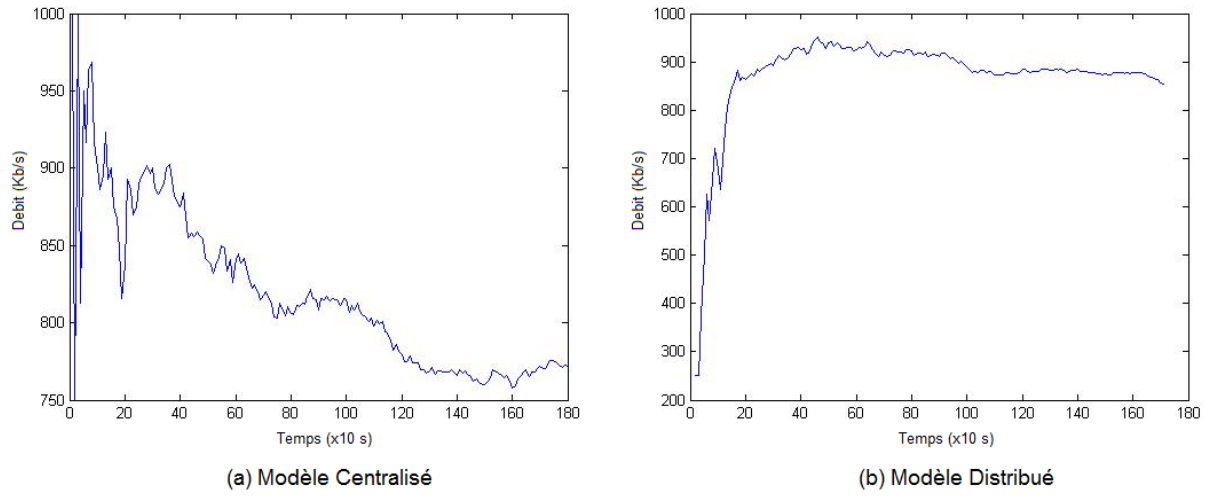



**Figure 32 :** *Evolution du débit dans le temps.*

Les interférences aussi jouent une importance majeure pour le débit. Peut-être que si on inclut ce critère dans l'optimisation, le modèle centralisé peut performer mieux.

Comme on a expliqué dans la conception, le classement de la meilleure solution est trouvé par deux principes : la dominance, et la priorité des besoins. Dans notre cas, il est à noter qu'on priorise l'énergie de la sécurité (l'ordre des besoins indique les priorités si les besoins sont équivalents). Dans ce test, on remarque une grande différence entre les deux modèles pour l'énergie (Figure 33). Le modèle centralisé performe beaucoup mieux. La raison est le nombre de saut. Le modèle distribué se bat pour trouver les routes, ce qui rend tout le réseau toujours en activité, et c'est ce qui consomme de l'énergie. Alors que le modèle centralisé utilise une route fixe qui minimise le nombre de sauts, avec une minimisation prédéfinie de l'énergie consommable par les nœuds lors de l'affectation des canaux. Cela rend le modèle centralisé performe mieux que le modèle distribué lors de la consommation l'énergie.

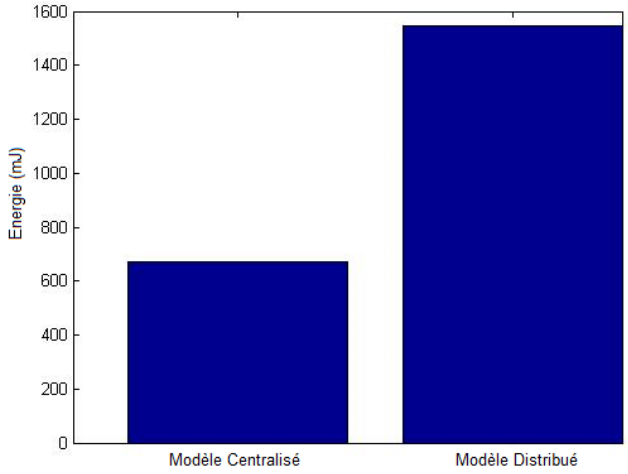



**Figure 33 :** *L'énergie moyenne consommé par les nœuds..*

La sécurité est le troisième critère dans notre contexte. Les deux modèles proposent un niveau entre 2 et 3, avec une proposition meilleure par le modèle distribué. Tandis que le modèle centralisé chercher une meilleure solution pour tous les critères (Figure 34), L'algorithme du choix de la solution dans l'ensemble de Pareto trouvé lui rend un peu vulnérable avec cet ordre de besoins. Par contre, le modèle distribué cherche la solution action par action avec une stricte contrainte sur la sécurité qui, bien qu'il perd de l'énergie. Le choix de la route n'est pas déterminé par ce qu'il veut le nœud source des paquets, mais par la route la plus sécurisée, même si elle est longue.

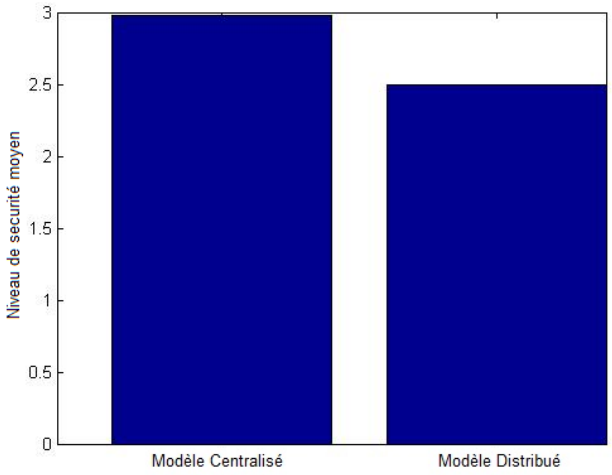



**Figure 34 :** *Niveau de sécurité moyen assuré pour les paquets envoyés.*

Aussi, on note que le nombre d'échecs de transmission est beaucoup plus moins dans le modèle centralisé que pour le modèle distribué (Figure 35). Comme l'énergie, cela est dû au manque des routes pour le modèle distribué, alors que le modèle centralisé minimise le nombre de sauts, ce qui minimise les interactions.

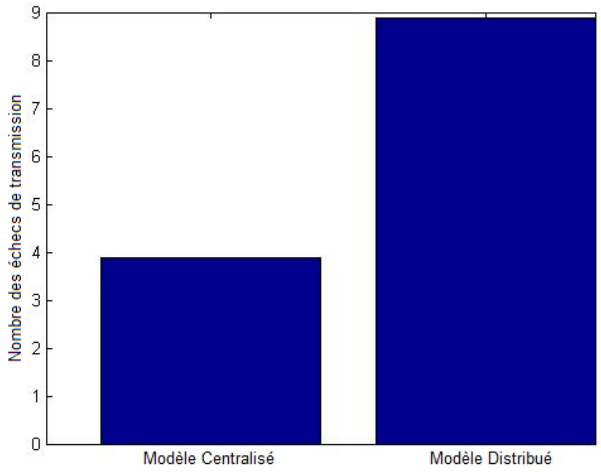



**Figure 35 :** *Nombre moyen d'échecs de transmission par second.*

## 3.2. Test 2 : Importance totale à la consommation d'énergie

Dans ce test, nous supposons que tous les nœuds ont un besoin total pour la consommation de l'énergie. Pour cela, on leurs attribue tous le quadruplet (0,1,0,0). La table 7 montre les résultats pour la simulation des deux modèles :

**Table 7 :** *Résultats de la simulation lorsque les noeuds demandent moins de consommation d'énergie.*

| ***Modèle*** | **Bande Passante** | **Débit** | **Energie** | **Sécurité** | **Erreurs de transmission** |
|---|---|---|---|---|---|
| *Centralisé* | 1088,7 | 1019.4 | 449.49 | 1.11 | 2.02 |
| *Distribué* | 1370.8 | 665.04 | 1472.4 | 2.10 | 10.40 |

Dans ce test, on remarque que le modèle centralisé performe mieux que le test 1. La seule différence est que le premier test optimise beaucoup mieux la bande passante. On note aussi que le réseau exploite beaucoup mieux la bande passante offerte. Les canaux 3 et 6 jouent un rôle pivot dans la solution avec le diamètre du champ qu'ils couvrent et l'énergie minimale qu'ils offrent. On remarque aussi l'amélioration au niveau de la transmission qui, avec cette solution, profite du grand champ des canaux qui ne consomme pas beaucoup d'énergie.

Contrairement au test 1, le test 2 montre que la contrainte du voisinage selon le canal CCC associée au modèle distribué, limite la performance. On remarque que la bande passante n'est pas bien exploitée et on a plus d'échecs de transmission qu'au test 1. La cause principale de cette dégradation de performance revient au grand champ des canaux qui donnent moins d'énergie à consommer. Ce qui provoque une augmentation des échecs de transmission, et éventuellement une diminution du débit moyen.

Sauf pour la bande passante moyenne, on remarque la dominance totale du modèle centralisé sur le modèle distribué (Figure 36). Comme il est expliqué, les inconvénients du modèle distribué dans ce test pèsent plus. Et cela se montre sur les résultats trouvés.

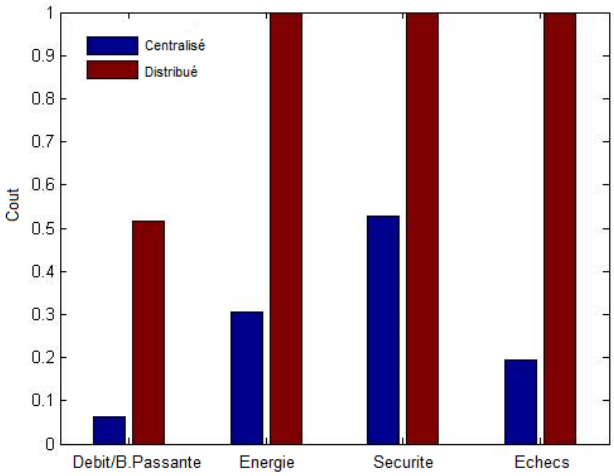



**Figure 36 :** *Les couts de performance pour le test 2.*

On doit aussi noter que les canaux qui ne consomment pas de l'énergie sont à faible fréquence. Le niveau de sécurité est calculé en se basant sur la même caractéristique. Donc il est logique d'avoir un bon niveau de sécurité lors de l'optimisation de l'énergie, ou vice versa.

## 3.3. Test 3 : Importance totale à la sécurité

Dans ce test, nous supposons que tous les nœuds ont un besoin total pour la sécurité. Pour cela, on leurs attribue tous le quadruplet (0,0,1,0). La table 8 montre les résultats pour la simulation des deux modèles :

**Table 8 :** *Résultats de la simulation lorsque les nœuds demandent une communication sécurisée.*

| ***Modèle*** | **Bande Passante** | **Débit** | **Energie** | **Sécurité** | **Erreurs de transmission** |
|---|---|---|---|---|---|
| *Centralisé* | 1088.7 | 1028 | 447.89 | 1.09 | 1.94 |
| *Distribué* | 1399.5 | 664.11 | 1475.4 | 2.13 | 10.06 |

Dans ce test, on peut qu'à confirmer les résultats du test 2. Les résultats du test 2 et test 3 sont presque les mêmes à cause de la similarité des relations sécurité-fréquence et énergie-fréquence.

## 3.4. Test 4 : Importance totale aux interférences

Dans ce test, nous supposons que tous les nœuds ont un besoin total pour la minimisation du nombre des échecs de transmission. Pour cela, on leurs attribue tous le quadruplet (0,0,0,1). La table 9 montre les résultats pour la simulation des deux modèles :

**Table 9 :** *Résultats de la simulation lorsque les nœuds demandent une transmission sans erreurs.*

| ***Modèle*** | **Bande Passante** | **Débit** | **Energie** | **Sécurité** | **Erreurs de transmission** |
|---|---|---|---|---|---|
| *Centralisé* | 1143.3 | 679.15 | 380.26 | 0.68 | 3.59 |
| *Distribué* | 1372.1 | 682.27 | 1494.7 | 2.11 | 9.53 |

Pour le modèle centralisé, Bien qu'on veuille améliorer la transmission, on a fini par améliorer l'énergie et la sécurité. La méthode utilisée n'est pas parfaite. Parmi les $6^{15}$ solutions possibles, on a un front de 100 solutions. Et on peut donc rater des solutions mieux que ce que cette simulation a trouvé.

Pour la même raison, avoir une solution meilleure que ce qu'on a trouvé dans test 1 pour l'optimisation du nombre des échecs de transmission dans le modèle distribué, revient à proposer une méthode plus rapide que ce qu'on utilise. Il est clair que la stabilité causée par les premières expériences de la performance (figure 37) du modèle distribué limite un peu l'optimisation ou la convergence vers une solution beaucoup plus meilleure.

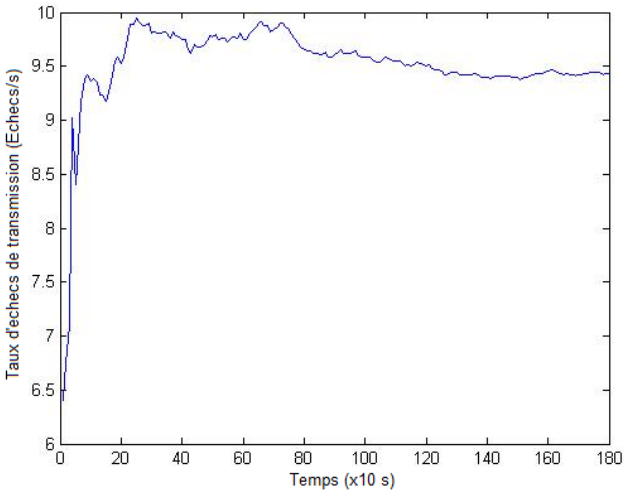



**Figure 37 :** *Evolution du nombre d'échecs dans le temps.*

Comme la figure le montre, après presque 800 secondes, le réseau commence l'amélioration jusqu'à la seconde 1400 où il se stabilise. Les mauvaises performances au début pèsent un peu dans les performances finales.

## 3.5. Test 5 : Importance équitable entre les critères

Dans ce test, nous supposons que tous les nœuds ont un besoin équitable entre les critères. Pour cela, on leurs attribue tous le quadruplet (0.25,0.25,0.25,0.25). La table 10 montre les résultats pour la simulation des deux modèles :

**Table 10 :** *Résultats de la simulation lorsque les nœuds ont un besoin équitable entre les critères.*

| ***Modèle*** | **Bande Passante** | **Débit** | **Energie** | **Sécurité** | **Erreurs de transmission** |
|---|---|---|---|---|---|
| *Centralisé* | 1671 | 723 | 378.07 | 1.24 | 4.11 |
| *Distribué* | 1404.1 | 790.94 | 1488.6 | 2.18 | 10.08 |

L'objectif de ce test est pour trouver l'architecture la plus équilibré. Comme il est montré par la table 10, le modèle centralisé est plus performant que le modèle distribué lors de la recherche de cette architecture. Il assure une bonne bande passante, une consommation d'énergie minimale, un très bon niveau de sécurité (entre 1 et 2) et il minimise les erreurs de transmissions à un seuil acceptable par rapport au modèle distribué.

Comme nous l'avons déduit par les tests précédents, la congestion joue un rôle très important pour l'amélioration du débit. Bien que le modèle centralisé soit plus performant dans ces critères, il a failli de proposer un bon débit même avec la prise en compte du nombre de sauts. Le débit du modèle distribué est mieux, malgré la différence de bande passante entre les deux modèles.

## 3.6. Test 6 : Besoins aléatoires des nœuds

Dans ce test, nous affectons aux nœuds des besoins aléatoirement. L'objectif est d'étudier la convergence des deux modèles vers les solutions qui essayent de satisfaire les besoins hétérogènes des nœuds. La table 11 prouve les résultats trouvés dans le test 5. Les figures 38 et 39 montrent respectivement la convergence du modèle centralisé et du modèle distribué sur les 3 critères : Débit, énergie, et taux d'échecs de transmission.

**Table 11 :** *Résultats des deux modèles lorsque les besoins sont aléatoirement attribués aux nœuds.*

| ***Modèle*** | **Bande Passante** | **Débit** | **Energie** | **Sécurité** | **Erreurs de transmission** |
|---|---|---|---|---|---|
| *Centralisé* | 1531 | 758.03 | 439.19 | 1.83 | 3.28 |
| *Distribué* | 1363.4 | 774.12 | 1563.6 | 2.33 | 9.63 |

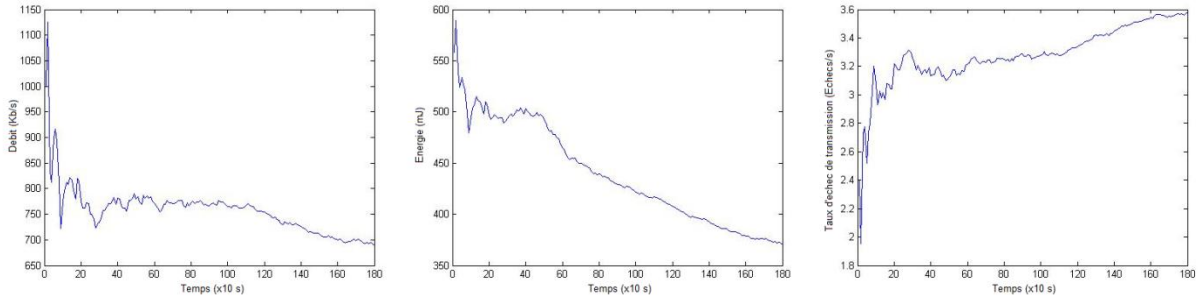



**Figure 38 :** *Evolution du débit, énergie et taux des échecs dans le temps pour le modèle centralisé.*

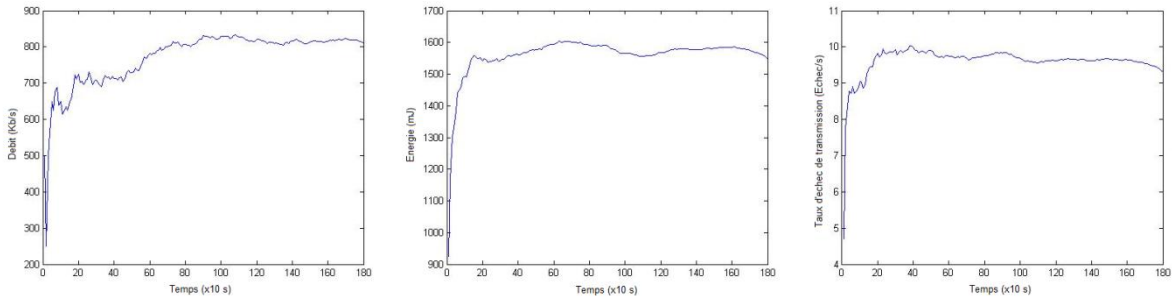



**Figure 39 :** *Evolution du débit, énergie et taux des échecs dans le temps pour le modèle distribué.*

Comme on peut le remarquer dans les deux figures. Le modèle centralisé commence avec une bonne performance au niveau de débit et échecs de transmission. On remarque que la dégradation des performances et le taux d'échecs de transmission suivent la même trajectoire. Cette similarité nous montre encore une fois l'importance de la congestion pour ces deux critères. Avec la dégradation du débit, on a aussi une diminution de l'énergie consommée. Car les nœuds qui ne peuvent pas relier leurs paquets vont conserver leur énergie jusqu'à ce qu'ils puissent les transmettre. D'autre part, le modèle distribué commence avec un état initiale vide,

les nœuds ne vont pas se mettre à communiquer jusqu'à ce qu'ils reçoivent les premiers paquets. Leurs choix d'action après, est déterminé par leurs expériences. Pour cette raison, on ne peut pas voir une amélioration pour l'énergie ou le taux d'échecs, mais plutôt une stabilisation dans une valeur.

On peut aussi conclure que bien que le modèle centralisé présente une performance meilleure dans le temps de simulation, le modèle distribué offre de la robustesse. La stabilité du réseau dans ce modèle le rend plus fiable par rapport au modèle centralisé qui ne peut pas nous assurer qu'il ne va pas continuer sa dégradation de performance.

# Conclusion

Dans ce chapitre, nous avons montré notre expérimentation dans 12 simulations : 6 pour le modèle centralisé et 6 pour le modèle distribué. Pour chaque pair de simulations nous avons étudié le comportement de réseau pour un type de besoin multicritère des nœuds. La table 12 résume ces tests (C = Centralisé, D = Distribué) :

**Table 12 :** *Les moyennes des performances des modeles dans les 6 tests.*

| | **Débit** | | **Energie** | | **Sécurité** | | **Echecs** | |
|---|---|---|---|---|---|---|---|---|
| | **C** | **D** | **C** | **D** | **C** | **D** | **C** | **D** |
| **Test 1** | 818.84 | 872.73 | 669.61 | 1544.6 | 2.98 | 2.5 | 3.88 | 8.89 |
| **Test 2** | 1019.4 | 663.04 | 449.49 | 1472.4 | 1.11 | 2.1 | 2.02 | 10.4 |
| **Test 3** | 1028 | 664.11 | 447.89 | 1475.4 | 1.09 | 2.13 | 1.95 | 10.06 |
| **Test 4** | 679.15 | 682.27 | 380.26 | 1494.7 | 0.68 | 2.11 | 3.59 | 9.54 |
| **Test 5** | 723 | 790.94 | 378.07 | 1488.6 | 1.24 | 2.18 | 4.11 | 10.08 |
| **Test 6** | 758.03 | 774.12 | 439.19 | 1563.6 | 1.83 | 2.33 | 3.28 | 9.63 |
| | | | | | | | | |
| **Moyenne** | 837.74 | 741.2 | 460.75 | 1506.6 | 1.4883 | 2.225 | 3.1383 | 9.7667 |
| **Ecart-type** | 151.15 | 85.302 | 107.42 | 38.214 | 0.8196 | 0.15909 | 0.93606 | 0.53365 |

Nous avons conclu que les deux modèles proposés ont des avantages et des inconvénients, qui apparaissent pour chaque simulation. Principalement, le problème de congestion qui n'est pas traité dans le modèle centralisé, et qui fait partie de l'équation de minimisation des délais de transmission dans le modèle distribué, a permis à ce modèle d'offrir un débit plus grand que celui offert par le modèle centralisé.

L'affectation des canaux joue un rôle très important dans le problème de congestion. Un nœud qui reçoit avec une bande passante grande et transmet dans une petite bande passante est susceptible de devenir un point de congestion. De même pour le choix des routes, si le même nœud est toujours choisit pour relier les paquets, il va surement devenir un point de congestion. Ces conclusions sont confirmées par les tests 2 et 3 où on affecte des canaux de petites bandes

passantes pour améliorer la consommation d'énergie et la sécurité. Dans ce cas, nous avons remarqué le saut de performance dans le modèle centralisé.

D'une autre part, le modèle distribué assure les optimisations des critères voulus, mais avec une efficacité médiocre. La performance du modèle distribué dans le débit et la sécurité est relativement bonne. Mais pour l'énergie et les échecs de transmission, ce modèle faillit de donner une performance proche à celle du modèle centralisé. Le premier critère est dû à l'ignorance des nœuds de la meilleure route (qui change souvent pour éviter les congestions), ce qui rend le réseau plus actif. Et le deuxième, à cause de la définition du voisinage selon le CCC et non pas par les canaux de données. Depuis la table 12, nous pouvons remarquez depuis les moyennes et les écart-types de la performance des deux modèles, que bien que le modèle centralisé est plus performant, le modèle distribué offre une meilleure robustesse. Cela nous assure qu'au cas de changement des besoins des nœuds, nous ne risquons pas d'avoir une chute de performance.

# Conclusion générale

## 1. Conclusion

Dans ce rapport, nous avons parlé des systèmes autonomes et hétérogènes. Nous avons montré que la recherche est de plus en plus intéressée par ces systèmes. Un travail qui a ouvert tout un domaine de recherche dans le réseau, et qui a essayé de concevoir un dispositif qui permet l'existence de ces systèmes dans une même machine, est le travail de Joseph Mitola. Où il a présenté la radio logicielle, puis la radio cognitive. Un dispositif qui permet aux machines de choisir leurs protocoles de communication selon leurs besoins.

Nous avons aussi mettre à la lumière l'orientation de la recherche dans ces systèmes qui sont représentés par les réseaux de radios cognitives. Et que beaucoup de recherche est faite sur les 3 premières couches du modèle OSI. Les chercheurs proposent beaucoup de solutions. Que ce soit pour l'optimisation d'un seul critère. Ou bien l'optimisation de plusieurs critères. Et nous avons pu conclure que malgré ces efforts, les chercheurs n'essayent pas d'inclure les besoins des nœuds dans leurs solutions.

Dans notre travail, nous voulons avancer un pas de plus et étudier le comportement des réseaux cognitifs si nous considérons les besoins de nœuds. Nous avons présenté notre modèle pour les besoins. Puis, nous avons proposé deux modèles d'interactions : un modèle centralisé basé sur les réseaux logiciels, et un autre modèle totalement distribué basé sur l'apprentissage par renforcement. Le premier modèle est conçu pour permettre l'intégration des solutions qui existent dans la littérature. Tandis que le deuxième est une proposition pure par notre part.

L'expérimentation dans ces deux modèles a mis à la lumière, la robustesse du modèle distribué par rapport au modèle centralisé qui montre plus d'efficacité. Les deux modèles peuvent être améliorés pour performer mieux. Pour le modèle centralisé, l'inclusion de la congestion et la modification périodique des affectations des canaux et des tables de routage peuvent améliorer l'efficacité et permettre la robustesse manquante. D'autre part, La définition du voisinage selon les canaux choisis pour la transmission, et non pas selon le CCC, peut améliorer l'efficacité du modèle distribué.

## 2. Perspectives

Malgré ce travail que nous sommes fières de le faire. Beaucoup de suppositions sont mis pour lui permettre d'exister. L'introduction des canaux prédéfinis, l'environnement de simulation, et le nombre de nœud dans le réseau sont des paramètres supposés. Il reste donc à confirmer ces résultats dans les cas où les canaux et leurs propriétés ne sont pas connus par les nœuds. Et que même si le nombre de nœuds change le réseau reste stable. De plus, il reste à étudier les résultats dans le monde réel.

Aussi, la difficulté de sujet et le temps court de travail nous a laissé de tirer que les résultats nécessaires. Beaucoup de tests qui restent à faire dans d'autres conditions, nous citons :

- Le cas de mouvement des nœuds.
- Le cas de mouvement des utilisateurs primaires.
- Le cas de changement de niveau de sécurité posé sur les paquets.
- Le cas de mouvement des zones de danger.
- Le cas d'apparition et disparition soudaines des nœuds.
- Le cas où les nœuds changent leurs besoins.

D'autre part, nous avons juste proposé un modèle pour les besoins. Nous n'avons pas donné comment les besoins sont calculés depuis les couches supérieures. Où quel protocole de communication va-t-il être utilisé lors du choix du canal.

Toutes ces questions, et peut-être d'autres, restent sans réponses par ce projet.

## 3. Orientation

La recherche pour nous ne s'arrête pas dans ce modèle. Car, comme nous l'avons dit, il reste encore des questions à répondre. Dans le futur, nous pensons à améliorer les deux modèles proposés, puis les tester avec des conditions plus sévères. Nous pensons aussi à étudier la possibilité d'un modèle hybride qui essaye de tirer les avantages gagnés par les deux modèles.

Le domaine des réseaux cognitifs est un domaine de futur qui est toujours ouvert à la recherche. Et à ce propos, nous voulons continuer de faire évoluer nos propositions jusqu'à leurs limites.

# Acronymes

AGC – Automatique Gain Control

ANF – Agence Nationale des Fréquences.

ASIC – Application-Specific Integrated Circuit

API – Application Programming Interface

BER – Bit Error Rate

CAN – Convertisseur Analogique Numérique

CCC – Common Control Channel

CCM – CORBA Component Model

CDMA – Code Division Multiple Access

CNA – Convertisseur Numérique Analogique

CORBA – Common Object Request Broker Architecture

CSMA/CA – Carrier Sense Multiple Access with Collision Avoidance

DAB – Direct Access Based

DDoS – Distributed Denial of Service

DSA – Dynamic Spectrum Allocation

DSP – Digital Signal Processor

FCC – Federal Communications Commission

FHS – Frequency Hopping Sequence

FPGA – Field-Programmable Gate Array

GPP – General Purpose Processor

GSM – Global System for Mobile communication

IF – Intermediate Frequency

IP – Internet Protocole

JTRS – Joint Tactical Radio System

KP – Knowledge Plan

LLC – Logical Link Control

LNA – Low-Noise Amplifier

LO – Local Oscilliator

MAC – Medium Access Control (deuxième couche du modèle OSI)

MCPA – Multi-Carrier Power Amplifier

MORL – Multi-Objective Reinforcement Learning

NCG – Node Contention Graph

NET – NETwork layer (troisième couche du modèle OSI)

NOS – Network Operating System

OSI – Open Systems Interconnection

PHY – PHYsical layer (première couche du modèle OSI)

POMDP – Partially Observable Markov Decision Process

POSIX – Portable Operating System Interface for uniX

QoS – Quality of Service

RF – Radio Frequency

RSP – Receive Signal Processor

R&TTE – Radio and Telecommunication Terminal Equipment directive

SCA – Software Communication Architecture

SDN – Software Defined Network

SDR – Software Defined Radio

SP – Split Phase

UP (ou PU) – Utilisateurs Primaires (Primary Users)

US (ou SU) – Utilisateurs Secondaires (Secondary Users)

# Bibliographie

[1] Jos Akhtman, *Heterogeneous Networking: An Enabling Paradigm for Ubiquitous Wireless Communications*, Proceedings of the IEEE - Vol. 98, No. 2, February 2010.

[2] Haoming Li, Javad Hajipour, Alireza Attar, and Victor C. M. Leung, *Broadband Wireless Access with Fiber-Connected Massively Distributed Antennas Architecture*, IEEE Wireless Communications, June 2011.

[3] Kenneth Bannister, Gianni Giorgetti, Sandeep K.S. Gupta, *Wireless Sensor Networking for "Hot" Applications: Effects of Temperature on Signal Strength, Data Collection and Localization*, ACM-HotEmNet 2008.

[4] Matthew N. O. Sadiku and Cajetan M. Akujuobi, *Software-defined radio: a brief overview*, IEEE Potentials, October/November 2004.

[5] P. Johnson, *New Research Lab Leads to Unique Radio Receiver*, E-Systems Team – Vol. 5, No. 4, May 1985.

[6] J. Mitola III, *Software Radios: Survey, Critical Evaluation and Future Directions*, IEEE 1992.

[7] Raymond J. Lackey and Donald W. Upmal, *Spreakeasy: The Military Software Radio*, IEEE Communications Magazine, May 1995.

[8] Nutaq's Blog, *A short history of software-defined radio (SDR) technology*, http://blog.nutaq.com/blog/short-history-software-defined-radio-sdr-technology.

[9] Vanu Website, *Anywave Base Station*, http://www.vanu.com/products/anywave/

[10] Brad Brannon, *Software-Defined Radio*, in Farid Dowla (ed.) , *Handbook of RF & Wireless Technologies*, Oxford UK: Newnes, p 133-181, 2004.

[11] Friedrich K. Jondral, *Software-Defined Radio – Basics and Evolution to Cognitive Radio*, EURASIP Journal on Wireless Communications and Networking 2005:3, 275-283, April 2005.

[12] Joseph Mitola III, *Software Radio Architecture: Object-Oriented Approaches to Wireless Systems Engineering*, John Wiley & Sons, New York, USA, 2000.

[13] Dola Saha, Dirk Grunwald and Douglas Sicker, *Wireless Innovation Through Software Radios*, ACM SIG COMM Communication Review – Vol. 39, No. 1, January 2009.

[14] Tore Ulversoy, *Software Defined Radio: Challenges and Opportunities*, IEEE Communications Surveys & Tutorials – Vol. 12, No. 4, Fourth Quarter 2010.

[15] Vanu G. Bose, *The Impact of Software Radio on Wireless Networking*, Mobile Computing and Communications Review – Vol. 3, No. 1, 1999.

[16] Joseph Mitola III, *Cognitive Radio Architecture: The Engineering Foundations of Radio XML*, John Wiley & Sons, New York, USA, 2006.

[17] Joseph Mitola III, *Cognitive Radio – An Integrated Agent Architecture for Software-Defined Radio*, Doctorat Dissertation, Royal Institute of Technology (KTH), Sweden, May 2000.

[18] George Dimitrakopoulos et Al. *Cognitive Radio – Spectrum and Radio Resource Management*, Wireless World Research Forum, 2004.

[19] C. Zou, C. Chigan and Z. Tian, *Development and Trends in Medium Access Control Design for Cognitive Radio Networks*, in Yang Xiao (ed.) and Fei Hu (ed.) , *Cognitive Radio Networks*, p 127-174, Taylor & Francis Group, New York, 2009

[20] A. De Domenico, E. Clvanese Strinati and M. Di Benedetto, *A Survey on MAC Strategies for Cognitive Radio Networks*, IEEE Communications Surveys & Tutorials – Vol. 14, No. 1, First Quarter 2012.

[21] B. Wang and K. J. Ray Liu, *Advances in Cognitive Radio Networks: A Survey*, IEEE Journal of Selected Topics in Signal Processing – Vol. 5, No. 1, February 2011.

[22] J. So and N. Vaidya, *A Multi-Channel MAC Protocol for Ad Hoc Wireless Networks*, Technical report, Dept. of Computer Science, University of Illinois, Urbana-Chamapign, 6, 2003.

[23] H. Zheng and C. Peng, *Collaboration and Fairness in Opportunistic Spectrum Access*, IEEE, 2005.

[24] L. Hyungkeun, L. Jang-Yeon and C. Jin-Woong, *An Adabtative MAC Protocol Based on Path-Diversity Routing in Ad Hoc Networks*, in 9th Advanced Communication Technology Conference – Vol. 50, p 975-985, June 2002.

[25] C. Hu, M. Lifson and Y. Xiao, *Routing in Cognitive Radio Networks*, in Yang Xiao (ed.) and Fei Hu (ed.) , *Cognitive Radio Networks*, p 257-276, Taylor & Francis Group, New York, 2009

[26] I. F. Akyildiz, W. Y. Lee, M. C. Vuran, and S. Mohanty, *Next generation dynamic spectrum access/cognitive radio wireless networks: A survey*, Computer Networks, 50:2127–2159, 2006

[27] A. Raniwala, K. Gopalan, and T. cker Chiueh, *Centralized channel assignment and routing algorithms for multichannel wireless mesh networks*, SIGMOBILE Mob. Comp. Comm. Rev., 8(10.2):50–65, 2004

[28] P. Kyasanur and N. Vaidya, *Routing in multichannel multi-interface ad hoc wireless networks*, Tech. rep., UIUC, Dec. 2004

[29] G. Cheng,W. Liu, Y. Li, and W. Cheng, *Spectrum aware on-demand routing in cognitive radio networks*, Proc. of IEEE DySPAN 2007

[30] Z. Tang and J.J. Garcia-Luna Aceves, *Hop-Reservation Multiple Access (HRMA) for Ad Hoc Networks*, in Proc. IEEE INFOCOM, vol. 1, pp. 194–201, Mar. 1999

[31] A. Tzamaloukas and J.J. Garcia-Luna-Aceves, *Channel-Hopping Multiple Access*, in Proc. IEEE International Conference on Computer Communication and Network (IC3N ’00), Oct. 2000

[32] A. Tzamaloukas and J.J. Garcia-Luna-Aceves, *Channel-Hopping Multiple Access with Packet Trains for Ad Hoc Networks*, In Proceedings IEEE Mo- MuC ’00, Oct. 2000

[33] P. Bahl, R. Chandra, and J. Dunagan, *SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad hoc Wireless Networks*, in Proc. of ACM MobiCom, 2004

[34] Hoi-Sheung, W. So, J. Walrand, and M. Jeonghoon, *MCMAC: A Parallel Rendezvous Multi-Channel MAC Protocol*, in Proc. IEEE WCNC 2007, pp. 334–339, Mar. 2007

[35] X. Jing and D. Raychaudhuri, *Spectrum Co-existence of IEEE 802.11b and 802.16a Networks Using Reactive and Proactive Etiquette Policies*, Mobile Networks and Applications, vol. 11, issue 4, pp. 539–554, Aug. 2006

[36] C. Cordeiro, K. Challapali, D. Birru, and N. Sai Shankar, *IEEE 802.22: The First Worldwide Wireless Standard Based on Cognitive Radios*, in Proc. IEEE DySPAN 2005, pp. 328–337, Nov. 2005

[37] C. Cordeiro and K. Challapali, *C-MAC: A cognitive MAC protocol for multichannel wireless networks*, in Proc. Symposium on Dynamic Spectrum Access Networks (DySPAN’07), pp. 147–157, April 2007

[38] W. Hu et Al., *COGNITIVE RADIOS FOR DYNAMIC SPECTRUM ACCESS - Dynamic Frequency Hopping Communities for Efficient IEEE 802.22 Operation*, IEEE Communication Magazine, vol. 45, No. 5, pp. 80–87, May 2007

[39] D. Willkomm et Al., *Double Hopping: A new Approach for Dynamic Frequency Hopping in Cognitive Radio Networks*, in Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2008 (PIMRC ’08) , pp. 1–6, September 2008

[40] C. M. Cordeiro, K. Challapali, and D. Birru, *IEEE 802.22: An Introduction to the First Wireless Standard based on Cognitive Radios*, J. Commun., Special Issue from selected papers from DySPAN 2005, vol. 1, no. 1, pp. 328–337, April 2006 (Invited Paper).

[41] Q. Zhao, L. Tong, and A. Swami, *Decentralized cognitive mac for dynamic spectrum access*, in Proc. Symposium on Dynamic Spectrum Access Networks (DySPAN’05), Baltimore, MD, USA, pp. 224–232, November 2005

[42] F. Wang, O. Younis, and M. Krunz, *GMAC: A game-theoretic MAC protocol for mobile ad hoc networks*, in Proc. Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks 2006, pp. 1–9, April 2006

[43] C. Xin, *A novel-layered graph model for topology formation and routing in dynamic spectrum access networks*, Proc. of IEEE DySPAN 2005

[44] R. Pal., *Efficient Routing Algorithms for Multi Channel Dynamic Spectrum Access Networks*, Proc. of IEEE DySPAN 2007, pp. 288–291, 2007

[45] Q. Wang and H. Zheng. *Route and spectrum selection in dynamic spectrum access networks*. In IEEE CNCC, 2006.

[46] G. Cheng,W. Liu, Y. Li, andW. Cheng, *Spectrum aware on-demand routing in cognitive radio networks*, Proc. of IEEE DySPAN, 2007

[47] Fortuna, Carolina, and Mihael Mohorcic. "*Trends in the development of communication networks: Cognitive networks.*" Computer networks 53.9, 2009: p. 1354-1376

[48] Mahmoud, Qusay H. "*Cognitive networks*." John Wiley& Sons Ltd, 2007: 57-71.

[49] Mitola, Joe. "*The software radio architecture*." Communications Magazine, IEEE 33.5, 1995: 26-38.

[50] R.W. Thomas, L.A. DaSilva, A.B. MacKenzie, *Cognitive networks*, in: Proceedings of the First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, Baltimore, MD, USA, November 8–11, 2005.

[51] Kreutz, Diego, Fernando MV Ramos, P. Esteves Verissimo, Christian Esteve Rothenberg, Siamak Azodolmolky, and Steve Uhlig. "*Software-defined networking: A comprehensive survey.*" proceedings of the IEEE 103, no. 1 (2015): 14-76.

[52] http://www.mathworks.com/matlabcentral/fileexchange/10429-nsga-ii--a-multi-objective-optimization-algorithm

[53] Jashni, Behrouz, Ali Tadaion, and Farid Ashtiani. "*Dynamic link/frequency selection in multi-hop cognitive radio networks for delay sensitive applications."* In Telecommunications (ICT), 2010 IEEE 17th International Conference on, pp. 128-132. IEEE, 2010.

[54] Yau, Kok-Lim Alvin, Peter Komisarczuk, and Paul D. Teal. "*Enhancing network performance in distributed cognitive radio networks using single-agent and multi-agent reinforcement learning*." In Local Computer Networks (LCN), 2010 IEEE 35th Conference on, pp. 152-159. IEEE, 2010.

[55] Shiang, Hsien-Po, and Mihaela Van der Schaar. "*Distributed resource management in multihop cognitive radio networks for delay-sensitive transmission.*" Vehicular Technology, IEEE Transactions on 58, no. 2 (2009): 941-953.

[56] L. Yong, J. Hong, H. Y. Qing. *Design of Cognitive Radio Wiereless Parameters Based on Multi-objective Immune Genetic Algorithm*. In : Communications and Mobile Computing, 2009. CMC'09. WRI International Conference on. IEEE, 2009. p. 92-96.

[57] Y. Bao, J. Hong, H. Y. Qing, et al. *Multi-objective optimization of power control and resource allocation for cognitive wireless networks*. In: Computer and Information Science, 2009. ICIS 2009. Eighth IEEE/ACIS International Conference on. IEEE, 2009. p. 70-74.

[58] Y. Zhihui, S. Keqin. *Multi-objective cooperative algorithms based on game theory in cognitive radio*. In: 2009 International Conference on Wireless Communications & Signal Processing. 2009. p. 1-4.

[59] J. Hong, Y. Bao, L. Qiang et al. *Multi-objective optimization of cross-layer configuration for cognitive wireless network*. In: Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on. IEEE, 2009. p. 379-383.

[60] A. EL-SALEH, M. ISMAIL, M.A.M ALI et al.*Development of a cognitive radio decision engine using multi-objective hybrid genetic algorithm*. In : Communications (MICC), 2009 IEEE 9th Malaysia International Conference on. IEEE, 2009. p. 343-347.

[61] H. WANG et L. GUO. *Multi-objective optimization of cognitive radio in clonal selection quantum genetic algorithm*. In : Measuring Technology and Mechatronics Automation (ICMTMA), 2010 International Conference on. IEEE, 2010. p. 740-743.

[62] A. ALSARHAN et A. AGARWAL. *Spectrum sharing in multi-service cognitive network using reinforcement learning*. In : Cognitive Wireless Systems (UKIWCWS), 2009 First UK-India International Workshop on. IEEE, 2009. p. 1-5.

[63] K. ZHENG, H. LI, R.C. QIU et al. *Multi-objective reinforcement learning based routing in cognitive radio networks: Walking in a random maze*. In : Computing, Networking and Communications (ICNC), 2012 International Conference on. IEEE, 2012. p. 359-363.

[64] Q. ZHANG, Q. HE, et P. ZHANG. *Topology reconfiguration in cognitive radio networks using ant colony optimization*. In : Vehicular Technology Conference (VTC Fall), 2012 IEEE. IEEE, 2012. p. 1-5.

[65] R. SAMANO-ROBLES et A. GAMEIRO. *Joint spectrum selection and radio resource management based on multi-objective portfolio optimization for cognitive radio networks*. In : Future Generation Communication Technology (FGCT), 2012 International Conference on. IEEE, 2012. p. 22-27.

[66] L. LAI, J. WANG, A. HUANG et al. *Routing and resource allocation with collision constraint in multi-hop cognitive radio networks*. In : Globecom Workshops (GC Wkshps), 2012 IEEE. IEEE, 2012. p. 974-979.

[67] I. ALQERM et B. SHIHADA. *Adaptive multi-objective Optimization scheme for cognitive radio resource management*. In : Global Communications Conference (GLOBECOM), 2014 IEEE. IEEE, 2014. p. 857-863.

[68] K.K. Anumandla, B. Akella, S.L. Sabat, S.K. Udgata. *Spectrum Allocation in Cognitive Radio Networks using Multi-Objective Differential Evolution Algorithm*. In: Signal Processing and Integrated Networks (SPIN), 2015 2nd International Conference on. IEEE, 2015. p. 264 – 269.

[69] D. NG, E. LO et R. SCHOBER. *Multi-objective resource allocation for secure communication in cognitive radio networks with wireless information and power transfer*. In : IEEE Transactions on Vehicular Technology vol. 65, no. 5, p. 3166 - 3184, May 2016

[70] Y. JIE, A.E. KAMAL. *Multi-Objective Multicast Routing Optimization in Cognitive Radio Networks*. In IEEE WCNC'14 Track 3 (Mobile and Wireless Networks), p.2576 - 2581, 2014.