REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE YAHIA FARES DE MEDEA



Faculté de **T**echnologie

Département du **G**énie **E**lectrique

Mémoire de fin d'études de Master

Filière : Télécommunications

Spécialité : Systèmes de Télécommunications

Simulation des algorithmes « *Dynamic Framed Slotted ALOHA* » pour les systèmes RFID

Présenté par :

Saidani Mohamed

Safidine Chemssedine

Proposé et Dirigé par :

Dr. Kalache Mohamed Aissa

Dr. Gacemi Abderrezak

Année universitaire 2020-2021

Dédicace 1

Tout d'abord, je tiens à remercier DIEU,

De m'avoir donné la force et le courage de mener à bien ce modeste travail.

Je tiens à dédier cet humble travail à :

A ma tendre mère et mon très cher père

A mes frères et ma sœur

A mon binôme

A mes meilleurs amis

Tous ceux qui m'aiment et que j 'aime

Saidani Mohamed

Dédicace 2

Je dédie ce modeste travail

A ma Chère Mère

A mon Père Dont le mérite

Les sacrifices et les qualités humaines m'ont permis de vivre ce jour

A mes Frères et ma sœur

A tous les gens m'aiment

Safidine chemssedine

ملخص

في الآونة الأخيرة، اكتسبت تقنية تحديد الهوية بموجات الراديو الكثير من الاهتمام بسبب إمكانيات التطبيق غير المحدودة والتكاليف المنخفضة للغاية. تستخدم أنظمة تحديد الهوية بموجات الراديو على نطاق واسع لغرض التعرف على الأشياء المادية بشكل فريد. واحدة من أهم مشكلات الأداء مع هذه الأنظمة هي حل التصادم الذي يحدث عند وجود علامات متعددة في نفس الوقت في منطقة القراءة لقارئ واحد. تتصادم العلامات وتلغي بعضها البعض. تقدم هذه المذكرة محاكاة لخوارزميات مضادة للتصادم وتقارن أداء خوارزميات *ALOHA* وتعتبر هذه الأخيرة الأكثر استخداما والأكثر فعالية.

Résumé

Récemment, la technologie d'identification via radio fréquence (RFID) a attiré beaucoup d'attention grâce à ses possibilités d'application illimitées et leurs coûts très bas. Les systèmes RFID sont largement utilisés à fin d'identifier des objets matériels de manière unique. Un des problèmes de performances les plus importants de ces systèmes est de résoudre la collision qui survient quand plusieurs tags sont présents au même moment dans la zone contient un seul lecteur. Les tags entrent en collision et annulent les unes les autres. Cette thèse présenter la simulation des algorithmes d'anticollision et compare les performances des algorithmes *Dynamic Framed slotted ALOHA*. Ces derniers sont considérés les plus utilisés et les plus efficaces

Mots-Clés : Identification par radiofréquence, Algorithme d'anticollision, *Dynamic Framed slotted ALOHA*

Abstract

Recently, radio frequency identification technology (RFID) has attracted a lot of attention due to its unlimited application possibilities and very low costs. RFID systems are widely used to uniquely identify material objects. One of the most important performance problems of these systems is to solve the collision that occurs when several tags are present at the same time in the reading area of a single reader. The tags collide and cancel each other out. This thesis presents a simulation of the anti-collision algorithms and compares the performance of Dynamic Framed Slotted ALOHA algorithms. These latter are the most used and the most efficient.

Keywords: Radio frequency identification, Anti-collision algorithm, Dynamic Framed slotted ALOHA.

Remerciements

Je tiens à remercier toutes les personnes qui ont contribué au succès de mon mémoire et qui m'ont aidée lors de la rédaction de ce mémoire.

Je voudrais dans un premier temps remercier, notre promoteur Dr. Kalache Mohamed Aissa, pour sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à alimenter ma réflexion.

Mes parents, pour leur soutien constant et leurs encouragements.

Table des Matières

Résumé I
Remerciements II
Table des matières III
Liste des figures V
Liste des tableaux VII
Liste des abréviations VIII
Introduction générale 1
CHAPITRE I : Généralité sur RFID 3
Introduction : 3
I.1. Les systèmes d'identification automatique 3
I.1.1 Le code barre 3
I.1.2 La reconnaissance optique de caractères 4
I.1.3 Les Smart cards 5
I.1.3.1 Les cartes mémoire 5
I.1.3.2 Les cartes à microprocesseur 6
I.1.4 Les systèmes RFID 6
I.1.4.1 Historique 6
I.2 Fonctionnement d'un système RFID 7
I.2.1 Composition d'un système RFID 7
I.2.1.1 Le tag 8
I.2.1.2 Le lecteur 8
I.2.2 Les fréquences 9
I.2.3 La portée et le couplage 10
I.2.4 La mémoire 11
I.2.5 La sécurité 11
I.2.6 Les collisions 12
I.3 RFID active ou passive 12
I.4 Champ proche ou champ lointain 14
I.5 Applications 17
Conclusion 17
Chapitre II : Algorithmes d'anticollision 19
Introduction 19
II.1 Classification des algorithmes d'anticollision 19
II.2 Algorithmes probabilistes et déterministes 22
II.2.1 Les algorithmes déterministes 22
II.2.1.1 Binary search 23
II.2.1.2 Le Protocole de Query Tree : 26
II.2.1.2.1 Query Tree 27
II.2.1.2.2 Query Tree Shortcutting 28
II.2.1.2.3 Query Tree Short-Long 29
II.2.1.2.4 Improved QT 30
II.2.2 Les algorithmes probabilistes 30
II.2.2.1 Basic ALOHA 31
II.2.2.2 Slotted ALOHA 31
II.2.2.2.1 Slotted ALOHA avec Muting ou Slowdown 32
II.2.2.2.2 Slotted ALOHA avec Early End 32
II.2.2.2.3 Slotted ALOHA avec Early End et Muting/Slow Down 33

II.2.2.3 Basic framed slotted ALOHA 33
II.2.2.4 Dynamic Framed Slotted ALOHA Algorithm (DFSA) 35
II.2.2.4.1 EPC Génération 2 Protocole 35
II.2.2.4.2 DFSA Basé dans un Tag Estimation Fonctions 39
II.3 Hybrid Anti-collision Algorithmes : 45
II.3.1 Framed Query Tree Algorithme 45
II.3.2 Query Tree ALOHA Algorithme 47
Conclusion 48
ChapitreIII: Implémentation et simulation 50
Introduction 50
III.1 Les critères d'évaluation 50
III.1.1 Système d'efficacité (throughtput) 50
III.1.2 rapidité d'identification des tags. 50
III.2 EPC G2 Air Interface: 50
III.2.1 Lien Tag-Lecteur 51
III.2.2 Backscatter Link 56
III.2.3 Les paramètres du temps 56
III.3 Modèle de simulation 58
III.4 Résultats de simulation 58
Conclusion 62
Conclusion générale 63
Bibliographie 64

Liste des figures

Figure I.1 Un code barre EAN (European Article Number) 4
Figure I.2 Structure d'un code barre EAN.....4
Figure I.3 les éléments principaux d'un system RFID 8
Figure I.4 Ici, le transpondeur et l'antenne sont fins et souples, collés sur un autocollant.....8
Figure I.5 Sur cette photo, on distingue bien le transpondeur au centre, entouré d'une antenne de cuivre bobinée 8
Figure I.6 Schémas de principe des tags a) passives et b) actives.....14
Figure I.7 Définition des zones de champs proche et lointain.....15
Figure I.8 Fréquences couramment utilisées en RFID 16

Figure II.1 Tag collision dans un système passive RFID.....19
Figure II.2 Classification des algorithmes d'anticollision.....20
Figure II.3 Schéma de la méthode fréquentielle.....21
Figure II.4 Schéma de la méthode spatial 22
Figure II.5 Exemple de signal numérique 24
Figure II.6 Exemple de déroulement de l'algorithme Binary Search.....26
Figure II.7 Tree diagramme de protocole de QT.....28
Figure II.8 Tree diagramme de protocole de QT shortcutting de Tableau II.2 29
Figure II.9 Slotted ALOHA avec un protocole de muting 33
Figure II.10 Slotted ALOHA avec Early End 33
Figure II.11 Le protocole de FSA.....34
Figure II.12 BFSA avec un protocole de muting 34
Figure II.13 L'algorithme Q [14] 36
Figure II.14 Le protocole de EPC G2 anti-collision.....37
Figure II.15 Throughput de framed slotted ALOHA de frame size F = 256.....39
Figure II.16 Un exemple de processus d'identification dans l'algorithme FQT 46
Figure II.17 Un exemple de processus d'identification dans l'algorithme QT-ALOHA 48

Figure III.1 – Codage PIE 51
Figure III.2 – Forward link preamble et frame-sync [40] 52
Figure III.3 – La structure de la trame de la commande "Query" 54
Figure III.4 – La structure de la commande 'Query Rep' 54
Figure III.5 – La structure de la commande 'Query Adjust'.....55
Figure III.6 – La structure de la commande 'ACK' 55
Figure III.7 – Structure de la trame de la réponse à la commande 'ACK'.....56
Figure III.8 – Synchronisation des liaisons de la norme EPC G2.....57
Figure III.9 – systèmes d'efficacité des algorithmes ALOHA en fonction de nombre des tags en petite échelle.....59
Figure III.10 – Zoom de système d efficacité des algorithmes ALOHA en fonction de nombre des tags en petite échelle.....59
Figure III.11 – la rapidité des algorithmes ALOHA en fonction de nombre des tags en petite échelle 60
Figure III.12 – systèmes d'efficacité des algorithmes ALOHA en fonction de nombre des tags en grand échelle 61

Figure III.13 – Zoom de système d efficacité des algorithmes ALOHA en fonction de nombre des tags en grand échelle 61
Figure III.14 – Rapidités des algorithmes ALOHA en fonction de nombre des tags en grand échelle 62

Liste des tableaux

Tableau I.1 – Les différentes fréquences d'émission.....9

Tableau II.1 – Communication entre le Reader et le tag avec le protocole de QT ...28
Tableau II.2 – Example de protocole de QT shortcutting.....29
Tableau II.3 – Example de protocole de IQT.....30
Tableau II.4 – Taille optimale des trames pour une gamme de tags donnée [18].....40

Tableau III.1 – Minimum et maximum valeurs de T_1, T_2 et T_3 spécifié dans [21] ...57
Tableau III.2 – Air interface of EPC G2 standard parameters used in our simulations 57

.

Liste des abréviations

ACK ACKnowledge. ASC Allocated Slot Counter.

BFSA Basic Framed Slotted ALOHA.

CE Collision Estimator.

CRC Cyclic Redundancy Check.

DFSA Dynamic Framed Slotted ALOHA.

DR Divide Ratio.

DS-CDMA Direct Sequence Code Division Multiple Access.

EDFSA Enhanced Dynamic Framed Slotted ALOHA.

EOF End-Of-Frame.

EPC Electronic Product Code.

EPC G2 EPC Generation 2.

FPGA Field Programmable Gate Arrays.

FSA Framed Slotted ALOHA.

HF High Frequency.

ID IDentification.

IoT Internet of things.

IQT Improved Query Tree.

ISO International Organization for Standardization.

LF Low Frequency.

MAC Medium Access Control.

MAP Maximum A Posteriori.

MSK Minimum-Shift Keying.

PA Pure ALOHA. PC Protocol Control.

PDF Probability Density Function.

PIE Pulse Interval Encoding.

PSK Phase Shift Keying.

PW Pulse Width.

Q Quadrature.

QT Query Tree.

QT-sl Query Tree short-long.

RF	Radio Frequency.
RFID	Radio Frequency IDentification.
RN16	Random Number 16.
SA	Slotted ALOHA.
SOF	Start-Of-Frame.
Tari	Type A Reference Interval.
TDMA	Time-Division Multiple Access.
TS	Tree Splitting.
UHF	Super High Frequency.
UHF	Ultra High Frequency.

Introduction générale

La technologie RFID (*Radio Frequency IDentification*) — ou identification par fréquence radio — fait partie des technologies d'identification automatique, au même titre que la reconnaissance optique de caractères ou de codes barre. Le but de ces technologies est de permettre l'identification d'objets ou d'individus par des machines.

La technologie RFID a la particularité de fonctionner à distance, sur le principe suivant : un lecteur émet un signal radio et reçoit en retour les réponses des tags — ou *tags* — qui se trouvent dans son champ d'action. Il existe une variété presque infinie de systèmes RFID ; différents types de mémoire, différentes fréquences, différentes portées, différents types d'alimentation.

La technologie RFID est utilisée depuis longtemps et à large échelle, notamment dans les secteurs de la logistique, la protection contre le vol ou encore l'identification des animaux. Pendant longtemps, le prix des tags RFID, leur encombrement ainsi que le manque de normalisation ont limité leur développement.

En 2005, après des années de recherche, de miniaturisation et d'efforts de normalisation, la technologie RFID vit une étape majeure de son développement. D'une part, on sait maintenant produire des tags de moins d'un demi-millimètre. D'autre part, les industriels viennent de se mettre d'accord pour adopter le standard EPC [1], qui s'apprête à compléter puis remplacer les codes barre pour créer un « Internet des objets ». Maintenant que la technologie est au point, il reste encore le problème des coûts de fabrication, qui toutefois ne cessent de baisser.

Les entreprises technologiques et les industriels ont largement financé la recherche, notamment par le biais du *Auto-ID Center* au *Massachusetts Institute of Technology* (MIT). L'identification sans contact est devenue un champ de recherche interdisciplinaire indépendant, qui mêle des domaines tels que les technologies radio, les technologies des semi-conducteurs, la protection des données, la cryptographie ou la téléinformatique. Le marché des systèmes RFID est en très forte croissance, de l'ordre de 30% par an. On prévoit que les ventes de systèmes RFID passeront de 900 millions $US en 2000 à 2650 millions $US en 2005 [2] et que ce nombre explosera quand les *tags* RFID marqueront chaque objet vendu par la grande distribution.

La technologie RFID, alors qu'elle est en train d'investir la vie quotidienne, reste méconnue du grand public. La documentation disponible reste relativement succincte et

les informations sont parfois contradictoires. Les enjeux sont pourtant majeurs, car la RFID peut bouleverser notre vie quotidienne. Pour toutes ces raisons, autant que par intérêt pour les systèmes d'identification et les disciplines qui y sont afférentes, nous avons choisi de nous pencher sur la technologie RFID et tenté d'en synthétiser les aspects les plus saillants.

Le problème majeur des systèmes RFID est la collision des tags. Il se produit lorsque plusieurs tags tentent de communiquer tous en même temps avec un seul lecteur. Pour gérer le problème de collision de tags, deux classes d'algorithmes d'anticollision existent : Algorithme à base ALOHA et *Tree based.* Dans cette thèse on va présenter ces derniers algorithmes. Les algorithmes à base ALOHA sont adopté dans les standards tel que le EPC, et il est l'algorithme le plus utiliser. L'objectif de cette thèse et d'évaluer et comparer les performances des algorithmes d'anticollision à base ALOHA.

Cette thèse est organisée en trois chapitres :

- Le premier chapitre analyse le cadre théorique de quelques termes qu'on a besoin dans notre thème : les systèmes RFID et la couche physique du protocole EPC Global.

- Le deuxième chapitre explique les diffèrent types des protocoles d'anti-collisions.

- Le dernier chapitre présente les résultats de simulation des protocoles d'anti-collisions á base ALOHA par MATLAB.

Enfin, nous avons clôturé notre mémoire par une conclusion générale.

Chapitre I : Généralité sur RFID

Introduction

Le système RFID est un système général complet qui comprend toutes les activités appropriées pour fournir des informations et efficace aux besoins des clients avec la satisfaction de tout le monde dans le domaine de la communication sans fil. Parmi toutes les applications des RFID, il en est une qui d'adopter par Les principaux groupes industriels, c'est l'attribution d'un code individuel à chaque objet, l'Électronique Product Code (EPC) Il s'agit d'une tentative ambitieuse de créer un réseau global, normalisé, permettant d'étiqueter et de suivre tout ce qui peut être expédié, stocké ou vendu. En fait, l'EPC permet virtuellement de numéroter tous les objets du plan été. Le but est bien sûr pour les industriels d'anticiper la demande, de réduire les stocks, de prévoir les pénuries, bref, les perspectives économiques sont énormes. La norme EPC a été élaborée en étroite collaboration avec le monde académique, par l'Auto-ID Center.

Ce chapitre est consacré à connaitre les différentes techniques d'identifications, que leur importance a diminuée par rapport à celle de RFID. Nous reconnaîtrons le principe de fonctionnement du RFID, les différents types de cette technique, et on parlera leurs normes internationales. La dernière section aborde les déférentes fréquences d'émission et leur application.

I.1. Les systèmes d'identification automatique

Les systèmes RFID font partie des technologies d'identification automatique, que l'on appelle aussi AIDC (*Automatic Identification and Data Capture*). Voici quelques-unes de ces technologies, qu'il est important de connaître pour comprendre le fonctionnement des systèmes RFID.

I.1.1 Le code barre

Omniprésents dans notre vie quotidienne, les codes barre dominent les systèmes d'identification automatique depuis plus de 20 ans.

Le code barre est un code binaire représenté par une s´séquence de barres vides et de barres pleines, larges ou étroites, disposées parallèlement. La séquence peut être interprétée numériquement ou alpha numériquement. Elle est lue par balayage optique au laser, c'est-`a-dire d'après la différence de réflexion du rayon laser par les barres noires

et les espaces blancs. On utilise actuellement une dizaine de types de codes barre différents, sans compter les codes barre `a deux dimensions — mais s'agit-il encore de codes barre ?

Le code barre le plus courant est le code EAN (*European Article Number*) figure I.1, crée pour répondre aux besoins de l'industrie alimentaire en 1976. Le code EAN est une ´évolution de l'UPC (*Universal Product Code*) américain, introduit aux Etats-Unis´ d`es 1973 ; UPC et EAN sont compatibles entre eux.



Fig I.1 *– Un code barre EAN (European Article Humber).*

Identifiant de Pays		Identifiant de l'entreprise					Numéro d'article du Fabricant					CD
0	4	1	2	3	4	5	0	8	1	5	0	9
FRG		Nom de la compagnie 1 nom de la route 80001 Munich					Lapin au Chocolat 100g					

Fig I.2 *– Structure d'un code barre EAN.*

Le code EAN est composé de 13 chiffres : l'identifiant du pays, l'identifiant de la société, le numéro de l'objet chez le fabriquant et un numéro de contrôle.

Malgré son grand âge, le code barre conserve des avantages importants comme son coût quasiment nul et sa large diffusion. En revanche, il présente plusieurs inconvénients : il est fragile, doit être lu de manière optique et peut être remplacé par quelqu'un de mal intentionné. De plus, il ne peut pas être modifiée à distance, contient peu d'informations et n'a bien sûr aucune capacité de traitement de données.

I.1.2 La reconnaissance optique de caractères

La reconnaissance optique de caractères (*Optical Character Recognition*, OCR) est utilisée depuis les années 1960. Elle fonctionne avec des polices de caractères conçues pour être lisibles aussi bien par les hommes que par les machines. On l'utilise aujourd'hui dans le domaine administratif et les services bancaires, notamment pour l'encaissement de moyens de paiement, tels que les chèques ou les bulletins de versement.

Si les système OCR ne sont pas plus répandus, c'est notamment dû à la complexité des lecteurs et à leur prix élevé.

I.1.3 Les Smart cards

Une *smart card* (« carte intelligente » ou « carte à puce »), est un système électronique de stockage de données, éventuellement avec une capacité de traitement (carte microprocesseur) qui, par commodité, est incorporé dans une carte en plastique de la taille d'une carte de crédit. Les premières *smart cards* sont apparues en 1984, sous la forme de cartes téléphoniques prépayées. Pour fonctionner, les *smart cards* doivent être placées dans un lecteur, qui entre en contact avec la surface de contact de la *smart card*. Le lecteur fournit à la *smart card* l´énergie et la pulsation d'horloge. Les transferts de données entre le lecteur et la carte se font par une interface série bidirectionnelle (port E/S).

Un des principaux avantages des *smart cards* est que les données qui y sont stockées peuvent être protégées contre les accès (lecture et/ou écriture) non d´désires. Les *smart cards* simplifient et sécurisent de nombreux services, à commencer par les transactions financières. En 1992, 200 millions de *smart cards* avaient déjà été produites dans le monde. En 1995, ce chiffre était de 600 millions, dont 500 millions de cartes mémoire et 100 millions de cartes à microprocesseur. Le marché des *smart card* représente donc un des secteurs a plus forte croissance de l'industrie micro-électronique.

Les *smart cards* comptent plusieurs inconvénients, basés sur la nécessite du contact et des manipulations : elles sont vulnérables à la corrosion et la poussière. Les lecteurs qui sont utilisés fréquemment (cabines téléphoniques, automates à billets...) tombent en panne et sont chers à entretenir. De plus, les lecteurs accessibles au public ne peuvent pas être protégés contre le vandalisme.

On distingue deux types de *smart cards* : les *cartes mémoire* et les *cartes à microprocesseur*.

I.1.3.1 Les cartes mémoire

Les cartes mémoire fonctionnent sur le principe d'une machine a états ; on accède à la mémoire — généralement une EEPROM — selon une logique séquentielle. Elles peuvent contenir des algorithmes de sécurité simples. Les fonctionnalistes des cartes mémoire sont généralement optimisées pour une application spécifique. Les applications sont assez rigides, mais les cartes sont bon marché ; on les utilise dans des applications à large échelle, l'a ou le coût est un facteur essentiel.

I.1.3.2 Les cartes à microprocesseur

Cartes à microprocesseur Comme leur nom l'indique, les cartes à microprocesseur contiennent un microprocesseur, connecté à une mémoire segmentée (segments ROM, RAM et EEPROM).

La ROM Elle comprend un système d'exploitation pour le microprocesseur. Le contenu de la ROM est inséré lors de la fabrication de la puce ; il est identique sur toutes les puces issues du même lot et ne peut pas être modifie.

L'EEPROM Elle contient les données et le programme relatif à l'application. La lecture et l'écriture de cette zone mémoire est contrôlée par le système d'exploitation et se fait après la fabrication.

La RAM C'est la mémoire temporaire de travail du microprocesseur. Les données stockées dans la RAM sont perdues quand l'alimentation électrique est interrompue.

Les cartes à microprocesseur sont très pratiques, puisque l'on peut facilement intégrer plusieurs applications différentes dans une même carte (multi-applications).

Les cartes à microprocesseur sont surtout utilisées dans les applications qui demandent un certain niveau de sécurité, comme les cartes pour téléphones GSM ou les cartes de type « porte-monnaie électronique ». La possibilité de programmer les cartes à microprocesseur favorise l'adoption rapide de nouvelles applications.

I.1.4 Les systèmes RFID

Les systèmes RFID sont très proches des *smart cards*. Comme sur les *smart cards*, les données sont stockées sur une puce électronique (*tag*). Cette puce peut être de type « machine a états » ou contenir un microprocesseur, elle peut avoir différents types de mémoire. Par contre, à la différence des *smart cards*, il n'y a pas de contact physique entre la puce et le lecteur ; l'alimentation électrique de la puce se fait par induction électromagnétique. Les données sont aussi transmises selon ce principe, ainsi que par réflexion des ondes radio. C'est bien de l`a que vient le nom de cette technologie : *Radio Frequency IDentification*.

I.1.4.1 Historique

1940 La notion de RFID (identification par fréquences radio) date de la 2 -ème guerre mondiale ; il est l'eau développement de la radio et du radar. Pour savoir si les avions qui arrivaient dans l'espace aérien britannique étaient amis ou ennemis, les alliés plaçaient

dans leurs avions d'imposantes balises, ou transpondeurs, afin de répondre aux interrogations de leurs radars. Ce système, dit IFF (*Identify : Friend or Foe* ; de nos jours, le contrôle du trafic aérien reste basé sur ce principe), est la première utilisation de la RFID. La première étude dont on dispose sur le sujet est un travail Harry Stockman [3], qui sera suivi notamment par les travaux de F. L. Vernon [4] et ceux de D.B. Harris [5]. Ces deux derniers articles sont considérés comme les fondements de la RFID et décrivent les principes qui sont toujours utilisés aujourd'hui.

1970 Durant les années 1960 et 1970, les systèmes RFID restent une technologie confidentielle, a usage militaire pour le contrôle d'accès aux sites sensibles, notamment dans le secteur nucléaire.

1980 Les avancées technologiques permettent l'apparition du *tag* passif. L'absence de source d'énergie embarquée rend le *tag* moins couteux. Le *tag* récit son énergie par le signal du lecteur. Les distances de lecture obtenues sont de quelques centimètres.

A la fin des années 1970, la technologie se répand dans le secteur privé. Une des premières applications commerciales est l'identification de bétail en Europe. D`es le début des années 1980, plusieurs sociétés européennes et américaines se mettent à fabriquer des *tags* RFID.

1990 Début de la normalisation pour une interopérabilité des équipements RFID.

2004 L'Auto-*ID Center* du MIT devient *EPCglobal*, une organisation dont le but est de promouvoir la norme EPC (*Electronic Product Code*) — sorte de super code barre stocker dans un *tag* RFID —, élaborée par les universitaires et adoptée par l'industrie.

Pour un historique complet et très bien documenté de la RFID, voir [6].

I.2 Fonctionnement d'un système RFID

On a tendance à réduire les systèmes RFID au rôle de « codes barre du futur ». Mais les systèmes RFID existent en d'innombrables variantes ; il n'est pas possible de les présenter toutes. Nous allons plutôt décrire leur principe général, puis quelques façons de les distinguer, ce qui devrait donner un bon aperçu de leur diversité et de leur fonctionnement.

I.2.1 Composition d'un système RFID

Un système RFID se compose de deux éléments : l'étiquette (*tag*) et le lecteur (figure I.3).

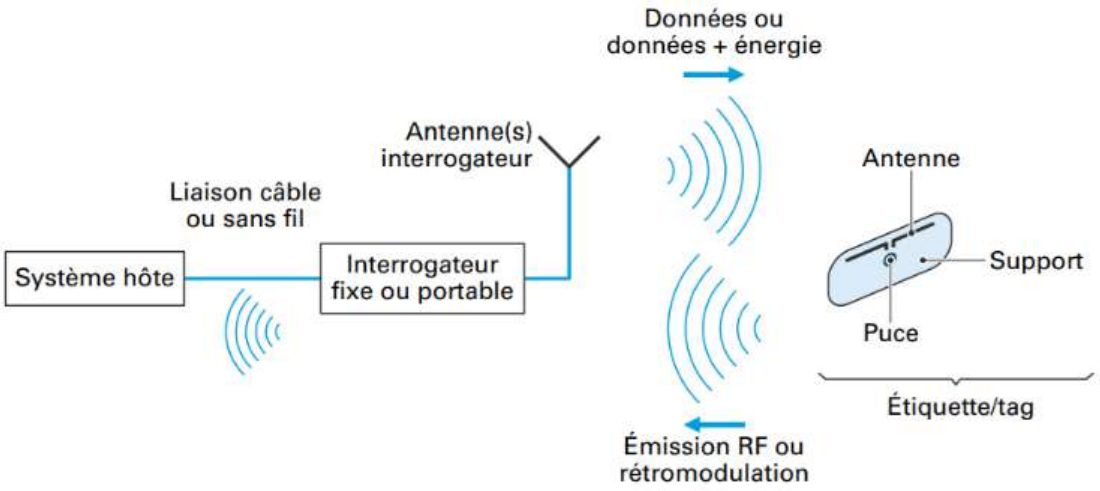


Fig I.3 *-les élément principaux d'un system RFID.*

I.2.1.1 Le tag

Le *tag* (étiquette) — appelé aussi transpondeur, pour *trans*mettre–ré*pondeur* — comprend une puce, dotée d'une mémoire, reliée à une antenne bobinée. Le plus souvent, le *tag* est collé sur un film en plastique ou moulé dans une carte au format carte de crédit.

On trouve des *tags* RFID de toutes formes et de toutes tailles. En voici quelques-uns, présentés aux figures I.4 et I.5.

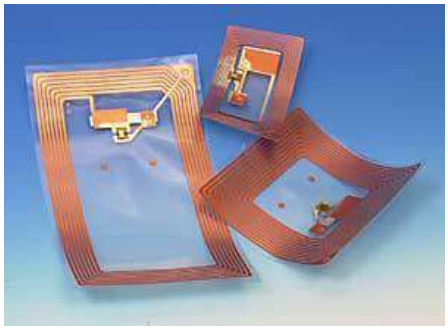


Fig I.4 *– Ici, le transpondeur et l'antenne sont fins et souples, collés sur un autocollant.*

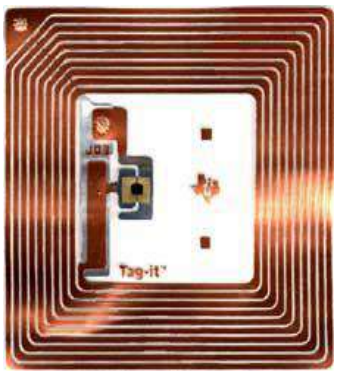


Fig I.5 *– Sur cette photo, on distingue bien le transpondeur au centre, entouré d'une antenne de cuivre bobinée.*

I.2.1.2 Le lecteur

Le lecteur, selon la technologique utilisée, peut lire mais aussi écrire des données sur le *tag*. Il émet des ondes radio et des champs magnétiques, puis écoute les réponses des *tags* qui se trouvent dans son champ de lecture. Le lecteur contient typiquement un

module radio (émetteur et récepteur) et une interface de contrôle. La plupart des lecteurs fournissent une interface supplémentaire de type RS 232 pour transférer les données reçues a d'autres systèmes (PC, robots, etc.).

I.2.2 Les fréquences

Les systèmes RFID génèrent et réfléchissent des ondes électromagnétiques ; ce sont donc des *systèmes radio* et, à ce titre, ils sont soumis à une législation stricte. Les systèmes RFID doivent notamment veiller à ne pas perturber le fonctionnement des autres *systèmes radio* : (télévision, services de secours, services radio maritimes et aériens, téléphones mobiles, etc.). On ne peut, en principe, utiliser que les plages de fréquences spécifiquement réservées aux applications industrielles, scientifiques ou médicales. Ces plages de fréquences sont appelées ISM (*Industrial–Scientific– Medical*).

En plus des fréquences ISM, on utilise les plages de fréquences en dessous de 135 kHz (< 400 kHz pour l'Amérique du Nord, du Sud et le Japon). En effet, en raison de la basse fréquence de l'horloge, le transpondeur nécessite peu d'énergie, ce qui favorise une portée élevée. Le taux de pénétration dans l'eau et les matériaux non métalliques est aussi meilleur.

Tableau I.1 – *Les différentes fréquences d'émission.*

	LF	HF	UHF	Microwave
Plages de Fréquence	125 - 134KHz	13.56 MHz	866 - 915MHz	2.45 GHz
Sources d'énergie	Courant induit	Courant induit	Piles ou batterie intégrée	Piles ou batterie intégrée
Distance de lecture	> 1 M	1M	1 à 10 M	10 à 100 M
Type De tag	Passive	Passive	Passive ou active	Actives
Applications	Billetterie, Identification animalière, Contrôle d'accès	Bibliothèque, localisation bagages, location	Suivi de flotte de véhicule, logistique, Accès sécurisé	Contrôle d'accès, logistique militaire, Péage automatique

Plus la fréquence est élevée, plus la vitesse d'horloge permettra des calculs rapides, comme des applications cryptographiques. De même, une bande passante plus large permettra des débits plus élevés. Par contre, la portée se trouve réduite et la transmission est plus sensible aux interférences.

Les principales plages de fréquences utilisées par les systèmes RFID sont les basses fréquences (125 et 134.5 kHz) et les fréquences ISM : 6.78 MHz, 13.56 MHz, 27.125 MHz, 40.68 MHz, 433.92 MHz, 869.0 MHz, 915.0 MHz (pas en Europe), 2.45 GHz, 5.8 GHz et 24.125 GHz.

La plage de fréquences la plus utilisée est de loin 13.56 MHz (haute fréquence). Suivent ensuite 134.5 kHz (basse fréquence), puis 2.45 GHz (micro-ondes) et 868/915 MHz (UHF).

I.2.3 La portée et le couplage

On distingue trois types de couplages entre le lecteur et le *tag*. Le couplage est étroitement lié à la fréquence et à la portée du système, qui peut varier de quelques millimètres a plus de 15 m.

Close coupling systems (« systèmes à couplage rapproché ») Ces systèmes ont une portée très faible, jusqu'à 1 cm. Ils fonctionnent avec des champs électromagnétiques, jusqu'à 30 MHz. L'énergie disponible est importante et permet d'utilisation d'un microprocesseur. On retrouve donc ces systèmes dans des applications qui utilisent le chiffrement, comme le verrouillage de portes et les cartes avec des fonctions de paiement. Ces systèmes sont de moins en moins importants sur le marché.

Remote coupling systems (« systèmes à couplage distant ») La portée de ces systèmes va jusqu'à 1 mètre. Ils fonctionnent aussi avec des champs électromagnétiques, par induction. Ces systèmes représentent plus de 90% des systèmes RFID vendus actuellement. Les fréquences généralement utilisées sont 135 kHz et 13.56 MHz. Quelques applications particulières (Eurobalise) fonctionnent à 27.125 MHz.

Long-range systems (« systèmes longue portée ») Ces systèmes portent à plus d'1 mètre. Les *tags* sont trop éloignés pour fonctionner par induction. En revanche, ils réfléchissent les ondes radio. Ces systèmes fonctionnent en UHF à 868 MHz (Europe), 915 MHz (USA) et sur les micro-ondes à 1.5 GHz et 5.8 GHz. La portée des transpondeurs passifs est de 3 mètres, tandis que les transpondeurs actifs, qui comportent une batterie, atteignent plus de 15 mètres.

Les avancées scientifiques permettent chaque année de diminuer la quantité d'énergie nécessaire à l'alimentation des tags. C'est pour les mêmes raisons qu'un PDA peut fonctionner avec des piles alors qu'il a la même puissance de calcul qu'un PC du milieu des années 1980. Chaque année, les systèmes RFID peuvent être conçus avec une meilleure portée, puisque les tags consomment moins à la même fréquence.

I.2.4 La mémoire

Les capacités mémoire des transpondeurs RFID vont normalement de quelques bytes à plusieurs kilobytes. Le cas des « transpondeurs 1 bit » est particulier : une information binaire permet de signaler au lecteur deux états : « le transpondeur est dans le champ » et « le transpondeur n'est pas dans le champ ». C'est peu, mais c'est suffisant pour des fonctions de surveillance ou de signalement. Un transpondeur 1 bit n'a pas besoin de puce électronique, il peut être fabriqué à un coût très bas. Les transpondeurs 1 bit sont généralement utilisés en *Electronic Article Surveillance* (EAS) pour protéger les biens dans les magasins et les entreprises. Les marchandises qui n'ont pas encore été payées portent un *tag*, qui sera détecté par le lecteur installé à la sortie du magasin.

Les mémoires peuvent être en lecture seule, mais aussi en lecture/écriture. Dans les systèmes très simples, les données du transpondeur, en général un simple numéro de série, sont écrites sur la puce lors de la fabrication et ne peuvent pas être modifiées.

A l'inverse, dans les systèmes plus complexes, le lecteur peut écrire des données sur le transpondeur. On utilise principalement trois types de mémoire :

Les EEPROMs (*Electrically Erasable Programmable Read-Only Memory*) consomment beaucoup d'énergie durant les opérations d'écriture et le nombre de leurs réécritures est limité — typiquement entre 100'000 et 1'000'000 fois.

Les FRAMs (*Ferromagnetic Random Access Memory*) consomment 100 fois moins d'énergie que les EEPROMs, et le temps d'écriture est 1000 fois plus bref. Pour l'instant des problèmes de fabrication ont retardé la diffusion des FRAMs sur le marché.

Les SRAMs (*Static Random Access Memory*), particulièrement répandues dans les systèmes à micro-ondes, sont très rapides à écrire. Par contre, la rétention des données nécessite une source d'énergie permanente, fournie par une batterie auxiliaire.

Dans les systèmes programmables, les accès mémoire doivent être autorisés par la puce elle-même, en lecture comme en écriture.

I.2.5 La sécurité

Les enjeux relatifs à la sécurité des *tags* RFID sont nombreux et importants. Le concepteur de systèmes RFID rencontre les mêmes problèmes que celui qui conçoit des cartes à puces : comment empêcher qu'un attaquant puisse lire, modifier ou fabriquer, voire dupliquer les données d'un *tag*, de manière à perturber le système et éventuellement

obtenir frauduleusement un accès à un bâtiment ou un service ? Il s'agit d'un problème complexe qui ne connaît pas de réponses parfaitement satisfaisantes.

Dans le cas des systèmes RFID, le fait que les échanges de données se font par ondes radio, ou champs électromagnétiques, ajoute une dimension supplémentaire au problème. D'une part, les menaces qui pèsent sur les *smart cards* avec contact physique sont renforcées par le fait qu'un attaquant peut interagir avec un *tag* RFID à distance, sans même que son propriétaire légitime ne s'en rende compte. D'autre part, il existe une nouvelle menace, qui pose un défi de taille aux concepteurs : comment empêcher un attaquant d'intercepter la communication radio, pour ensuite « rejouer » la transaction en imitant le *tag* original, un peu à la manière d'Ali Baba devant la grotte des voleurs ?

Il est facile d'imaginer des scénarios catastrophiques pour les acteurs qui utiliseraient des systèmes de paiement électronique mal sécurisés...

Lors de la conférence de sécurité informatique *Blackhat 2004*, un consultant allemand, Lukas Grunwald, a présenté le logiciel libre RFDump, qui permet de lire et de modifier différents types de *tags* RFID. Il existe également une bibliothèque de fonctions en langage C écrite par le français Loïc Dachary, libre elle aussi, pour dialoguer avec les *tags* RFID.

I.2.6 Les collisions

Les premiers *tags* RFID, fabriques dans les années 1960, fonctionnaient sur de basses fréquences. De nos jours, des fréquences élevées, typiquement sur la bande UHF, améliorent la capacité d'un lecteur à lire plusieurs *tags* se trouvant dans son champ de lecture, car le taux de transfert est plus grand et les données de chaque *tag* sont transmises plus rapidement, réduisant la probabilité de collision entre les données de chacun des *tags* — typiquement lors de la lecture des produits dans un chariot de supermarché.

Pour réduire les chances que deux *tags* émettent en même temps (collision), les *tags* utilisent un protocole anticollision pour contrôler la fenêtre de temps durant laquelle chacun répond ; ce temps dépend du numéro unique des tags.

I.3 RFID active ou passive

Dans les systèmes RFID actifs, le tag possède une puce électronique ayant un émetteur RF. La communication entre l'interrogateur et le tag peut donc se faire comme dans n'importe quel système pair à pair, en utilisant des protocoles full duplex par exemple. Généralement, l'énergie rayonnée par l'interrogateur et captée par le tag n'est

pas suffisante pour alimenter correctement la puce électronique. Les systèmes actifs doivent donc prévoir l'embarquement d'une source d'énergie propre au tag. Ajouté au fait que la puce possède son propre circuit d'émission, cela peut augmenter fortement le coût du tag RFID. La norme ISO/IEC 18000-7 prévoit le fonctionnement de systèmes actifs à 433 MHz. Avec de tels systèmes, la portée de communication entre un interrogateur et un tag peut atteindre sans difficulté la centaine de mètres. Le principe de fonctionnement des systèmes RFID passifs repose sur la modulation de l'onde provenant de l'interrogateur et partiellement réfléchie par le tag. Quels que soient les fréquences ou les modes de couplage, le moyen utilisé pour réaliser cette rétro modulation, consiste à commuter une charge (impédance) placée en parallèle entre la puce électronique et l'antenne du tag. Le signal réfléchi par le tag vient alors se superposer au signal provenant de l'interrogateur. Dans le cas, très majoritairement rencontré, des tags passives ne possédant pas de source d'énergie embarquée, le rapport entre la puissance du signal émis par l'interrogateur (pour alimenter la puce) et la puissance du signal rétro modulé par tag peut atteindre 60 dB. L'interrogateur doit donc présenter une bonne sensibilité pour détecter et décoder l'information issue du tag. La difficulté de ces systèmes consiste donc à trouver la meilleure charge permettant de créer de fortes variations de signal réfléchi sans pour autant pénaliser l'alimentation du circuit. La figure I.6 schématise les grandes différences entre tags actifs et passifs, séparant ces notions de la présence ou non de source d'énergie embarquée par tag. Dans la majorité des cas, la distance de communication entre un tag passif et son interrogateur est limitée par la distance de télé alimentation (sauf dans les cas, de plus en plus rares, où l'interrogateur n'a pas sensibilité nécessaire). Une manière d'augmenter cette distance est d'ajouter au tag une source d'énergie propre. Cette source d'énergie va permettre d'alimenter le circuit de la puce électronique sans pour autant devoir capter l'énergie issue du signal RF transmis par l'interrogateur. Cette source d'énergie peut également servir à alimenter d'autres systèmes électroniques associés au tag RFID comme des capteurs. Le tag RFID peut alors récupérer de l'information issue de ces capteurs, la stocker dans une zone mémoire particulière de la puce électronique sans pour autant être dans le champ rayonné par un interrogateur. Ces systèmes, appelés BAP (Battery Assisted Passive), se comportent comme des systèmes passifs sans source d'énergie une fois cette source épuisée.

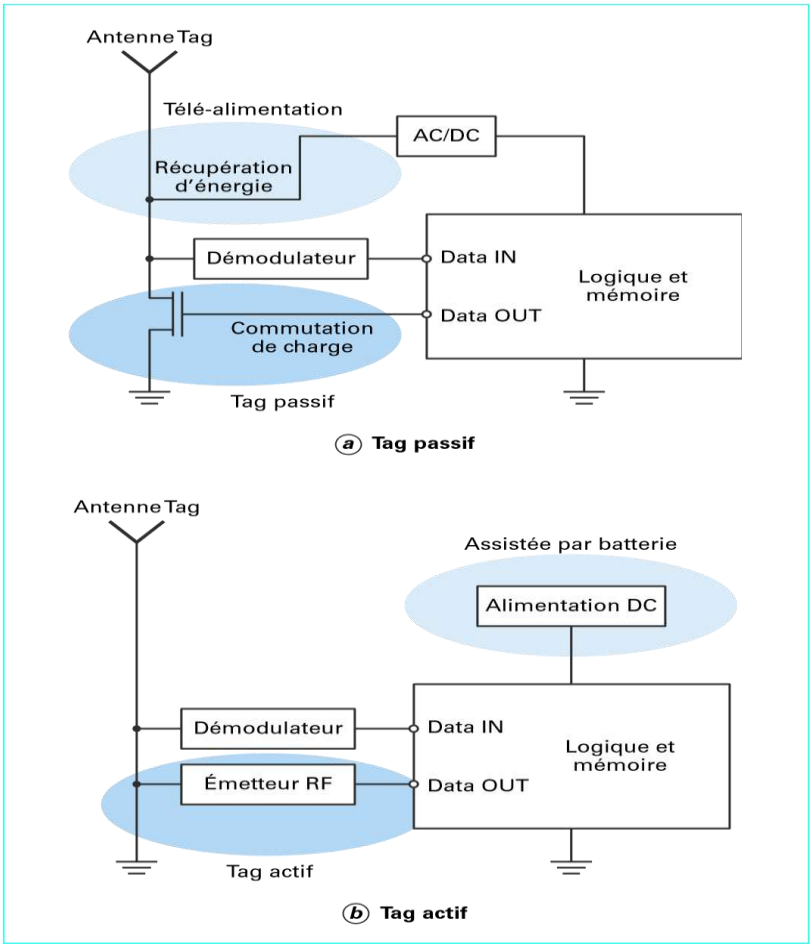


Fig I.6 *-Schémas de principe des tags a) passives et b) actives.*

I.4 Champ proche ou champ lointain

Les systèmes RFID passifs peuvent fonctionner à différentes fréquences. Dans un premier temps, l'interrogateur doit émettre un signal permettant la télé alimentation de la ou des tags présentes à proximité. Pour rayonner et de la même manière recevoir un signal radio, il faut se poser la question de l'antenne la plus adaptée. Le concepteur a le choix entre deux grandes familles d'antennes : les antennes fermées (boucles) ou ouvertes (dipôles). Les premières vont plutôt créer un champ magnétique dans leur entourage proche alors que les secondes créeront plutôt un champ électrique. Au fur et à mesure que l'on s'éloigne de la structure rayonnante, le champ électromagnétique se forme et les célèbres équations de Maxwell permettent de relier champs magnétique et électrique. On parle alors de champ formé ou champ lointain. La distance à laquelle on peut considérer que le champ est formé dépend de la fréquence du signal et des dimensions de l'antenne.

La figure I.**7** résume les ordres de grandeur des extensions des zones de champ proche et de champ lointain. Les limites dépendent de *D*, la plus grande dimension de l'antenne

rayonnant le champ électromagnétique et de λ la longueur d'onde du signal. Il n'est pas dans l'objectif de cette thèse de détailler plus en avant ces notions et le lecteur pourra se référer aux ouvrages [1] et [2].

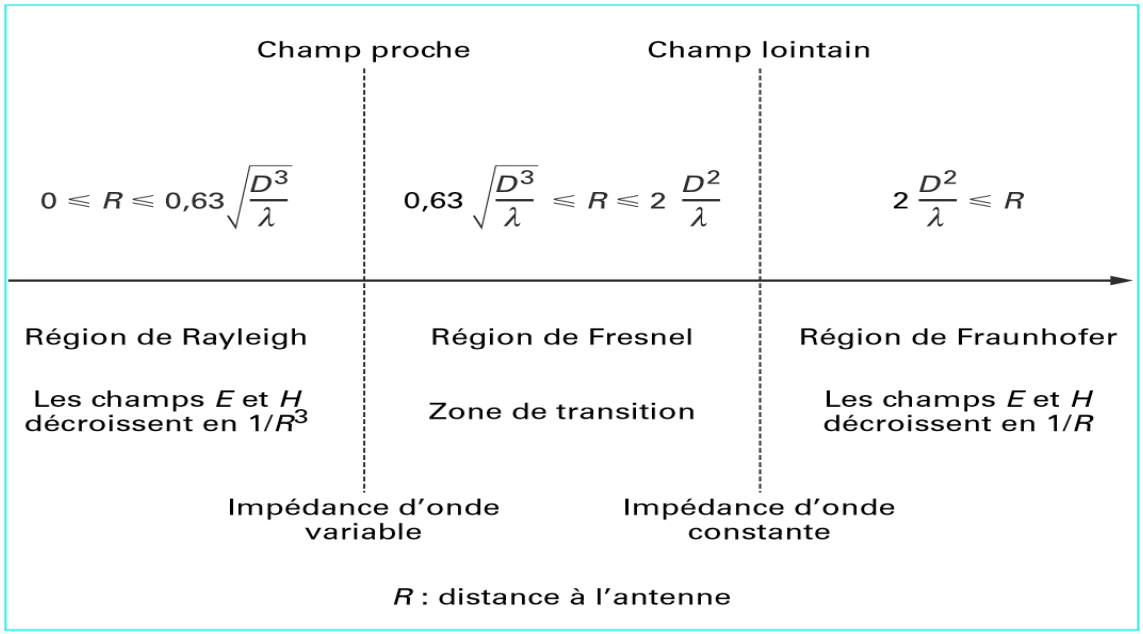


Fig I.7 *– Définition des zones de champs proche et lointain*

Prenons l'exemple de système RFID fonctionnant **à 13,56 MHz**. Dans l'air (ou dans le vide), la longueur d'onde associée à cette fréquence est de plus de 22 m. Vu les champs maximaux que l'on est autorisé à rayonner, les distances auxquelles on pourra lire un tag ne dépasseront jamais 1 à 2 m. Cela signifie que les systèmes RFID fonctionnant à cette fréquence seront toujours dans la zone de champ proche. On peut alors se poser la question du type d'antenne. Allons-nous rayonner principalement un champ électrique (antenne dipôle) ou un champ magnétique (antenne boucle). Pour rayonner correctement un champ électrique, une antenne doit avoir des dimensions proches de la longueur d'onde. Avec des tags de l'ordre de 22 m, aucune application RFID n'aurait pu voir le jour à 13,56 MHz. Le choix est donc par défaut celui des **antennes boucle** créant un champ magnétique.

Si on travaille à présent **à 900 MHz**, la longueur d'onde est d'environ 33 cm dans l'air libre. Les puissances que l'on peut rayonner dans cette gamme de fréquence permettent d'envisager des distances de communication comprises classiquement entre 5 et 10 m. Nous sommes cette fois dans la zone où le champ électromagnétique est formé. On peut véritablement parler de propagation d'onde entre l'interrogateur et tag. Se pose encore une fois le choix entre des antennes boucle ou des antennes de type dipôle. Cette fois, la taille du dipôle optimal est de l'ordre de λ/2 c'est-à-dire 15 cm. Cette taille est tout à fait

compatible avec les contraintes des applications RFID. Les équations de Maxwell, valables dans le cas du champ lointain, indiquent que pour une onde électromagnétique se propageant en espace libre, le rapport entre l'amplitude du champ électrique et celle du champ magnétique est constant. Ce rapport est égal à l'impédance du milieu de propagation et vaut, pour le vide, 377 Ω. La valeur de ce rapport montre que le choix des **antennes de type dipôle électrique** est tout à fait logique.

Pour être tout à fait complet, nous pouvons remarquer que la notion de champ proche peut parfaitement s'appliquer pour des systèmes RFID UHF. En effet, si le tag se rapproche de l'antenne de l'interrogateur, le champ n'est pas encore formé et nous sommes dans un régime de champ proche. Dans ce cas, on peut très bien utiliser une antenne boucle pour récupérer le champ magnétique comme dans les applications RFID HF.

Les bandes de fréquence dans lesquelles peuvent fonctionner les systèmes RFID font partie des bandes non soumises à licence. Ces bandes, réservées aux applications industrielles, scientifiques et médicales (bandes ISM), si elles ne sont pas soumises à licence, ne sont utilisables qu'en respectant scrupuleusement des gabarits d'émission (largeur de bande autorisée, puissance ou champ maximal à ne pas dépasser, taux d'occupation à respecter). Les utilisateurs de ces bandes veilleront à respecter les règlements et régulations propres à chaque région du globe où sera déployée l'application RFID. Ceci sans compter les règlements liés à la sécurité sanitaire et à l'exposition humaine aux rayonnements non ionisants (la RFID n'utilise pas encore de rayons X ou gamma !).

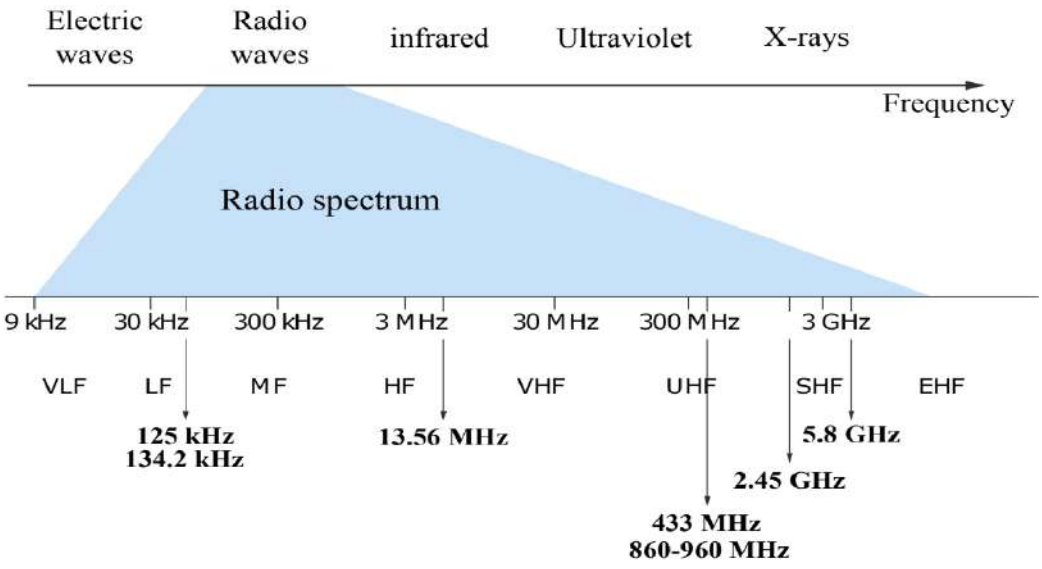


Fig I.8 *– Fréquences couramment utilisées en RFID.*

I.5 Applications

Les systèmes RFID sont utilisés depuis plusieurs années, dans des applications relativement classiques. Récemment, les évolutions technologiques ont favorisé leur apparition dans des domaines moins classiques, ce qui soulève d'importantes questions relatives au respect de la vie privée, de la protection des données et des libertés individuelles.

On a déjà évoqué, en examinant les différents types de systèmes RFID, certaines applications courantes comme les systèmes anti-vol dans les magasins, le contrôle d'accès, la logistique, l'identification du bétail et des animaux de compagnie. La RFID sert également à identifier les bagages dans les aéroports, les livres dans les bibliothèques, les documents dans les entreprises, à faire payer les conducteurs aux péages, à relever les données de capteurs, à effectuer des micropaiements, à emprunter les transports publics à Londres, Paris ou Venise ou les remontées mécaniques dans les stations de ski. La plupart de ces utilisations sont largement documentées sur Internet, nous ne les décrirons pas en détail ici.

Ces applications se généralisent rapidement. La guerre en Irak a été l'occasion pour l'armée américaine d'utiliser les systèmes RFID à large échelle. Tous les tickets vendus pour la prochaine coupe du monde de football, qui se tiendra en 2006 en Allemagne, seront porteurs de *tags* RFID. Il existe même des machines à laver capables d'indiquer quel est le meilleur programme de lavage en fonction du linge qui se trouve à l'intérieur.

Les *tags* RFID peuvent aussi faire bien plus que simplement renvoyer un numéro ; ils peuvent par exemple être associes à des capteurs et mémoriser les températures de stockage d'une viande, de manière à prévenir les risques d'avarie. Ils peuvent aussi profiter de capacités de calcul pour trouver des applications encore insoupçonnées.

Conclusion

L'adoption générale et rapide d'une technologie (les ordinateurs personnels, les téléphones portables) se produit au moment où cette technologie fonctionne bien, ou les prix sont suffisamment bas et la demande assez grande. Cette heure a sonné pour la RFID. Il reste bien sûr des défis (prix, standardisation, sécurité) mais les systèmes RFID sont sur le point de se répandre massivement.

L'utilisation des RFID pour le suivi des hommes, des animaux et des produits va permettre d'optimiser de nombreux processus industriels, de minimiser les stocks et

d'améliorer un certain nombre de services tels que les transports publics par exemple. D'autres usages sont encore à inventer, car on présente souvent le *tag* RFID comme une « étiquette intelligente » ou le « successeur du code barre » mais les RFID sont capables de bien plus que ça.

« L'internet des objets », résultat d'années de recherche, pose de nombreux défis notamment logiciels, à la fois dans le traitement et l'analyse des montagnes de données qui seront disponibles que dans l'adaptation des infrastructures existantes au réseau global EPC.

La technologie RFID présente toutefois des problèmes techniques mais aussi des problèmes de sécurité et des problèmes de respect de la sphère privée. Ces problèmes, inhérents à toute technologie, sont cette fois particulièrement importants, de par les possibilités de suivi de chaque objet et de chaque personne. Il est aussi dans l'intérêt des industriels de résoudre ces problèmes, sans quoi la RFID pourrait rencontrer une éraction de rejet de la part des consommateurs.

Nous pensons que, comme d'autres sciences capables de changer le monde (nucléaire, manipulations génétiques), la RFID va susciter des débats passionnés, avant d'être encadrée par des législations particulières. Seulement, pour que le débat ait lieu, il est nécessaire que le grand public connaisse puis comprenne cette technologie et soit sensibilisé à ses dangers, ce qui suppose encore un long travail de vulgarisation.

Chapitre II : Algorithmes d'anticollision

Introduction

Dans le cas où plusieurs objets qui portent une tag RFID ou tag RFID sont présents simultanément dans le champ du lecteur une collision se provoque (Figure II.1), la superposition des signaux envoyées par ces tags produira un seule signale. La restauration des signaux des tags est difficile ou presque impossible. Il s'agit de la problématique dite de collisions des tags. L'algorithme d'anticollision doit être appliquer dans le système pour la lecture peut gérer les collisions des tags.

Il existe deux grandes classes de méthodes d'algorithmes anticollision : des méthodes dites déterministes (*Tree Based*) et des méthodes probabilistes (ALOHA Based) ; Toutes deux sont basées sur des protocoles de la couche MAC. Dans cette thèse on va concentrer sur les méthodes à base d'ALOHA

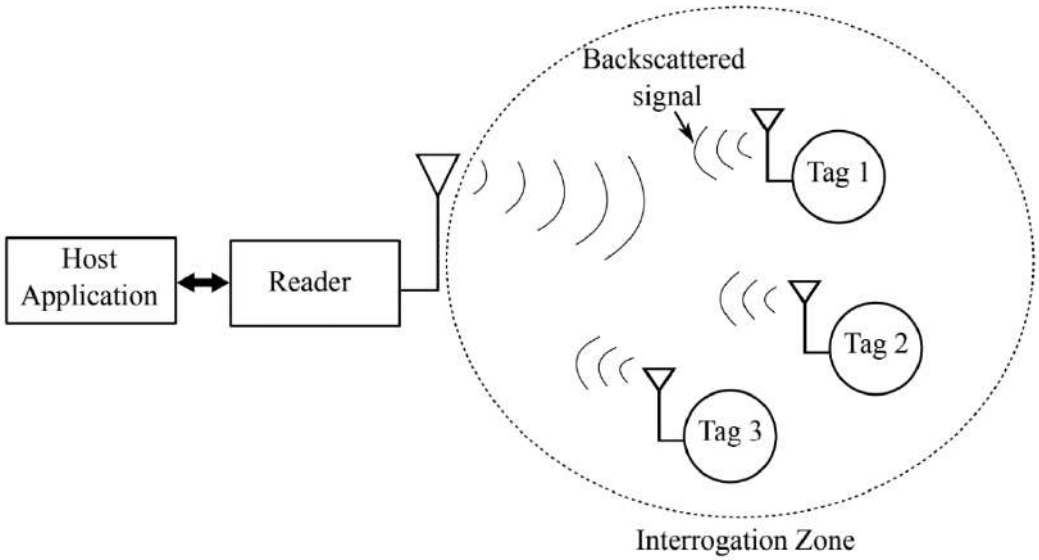


Fig II.1 *– Tag collision dans un système passive RFID.*

II.1 Classification des algorithmes d'anticollision

Les systèmes d'anticollision sont connus aussi sous le nom de systèmes d'accès multiples, il s'agit d'un logiciel installé sur le lecteur qui a pour rôle la séparation les signaux des différents tags qui sont dans le champ d'action du lecteur et ainsi gérer la collision dans un système à faible puissance et à faible complexité de calcul à fin identifier les tags avec précision, le tout en un minimum de temps.

L'évolution des systèmes RFID a conduit aussi l'évolution et la diversité des algorithmes et des méthodes d'anticollision. Chacune de ces méthodes à ces avantages et ces inconvénients voici un schéma qui illustre la classification de ces méthodes et algorithmes suivie d'une brève description de ces derniers.

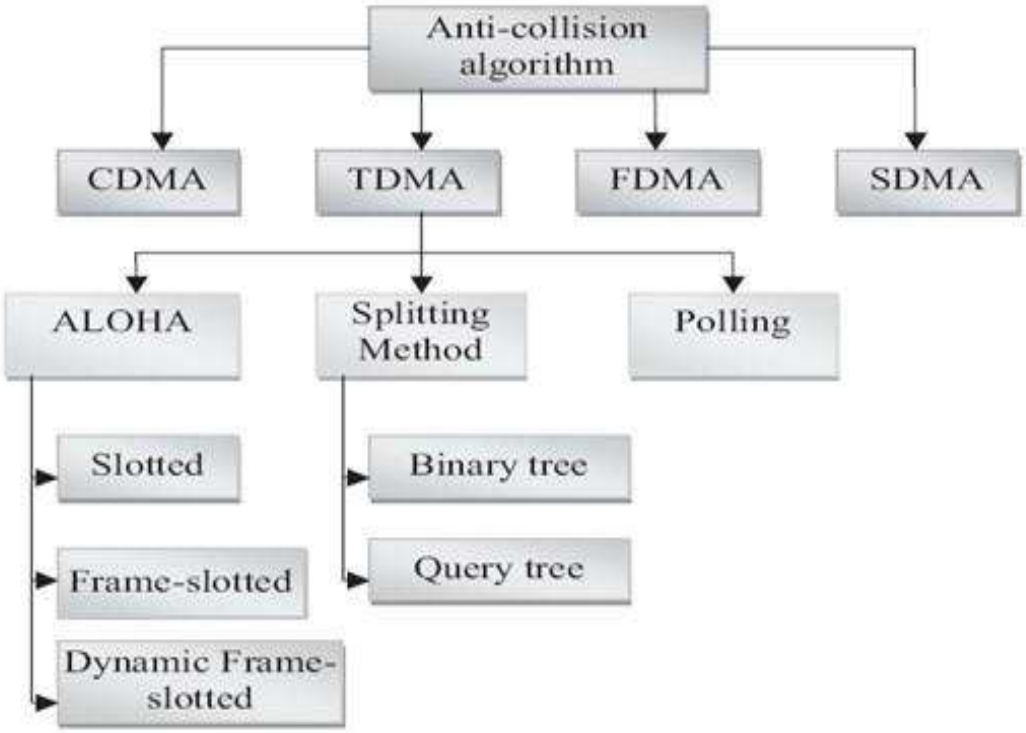


Fig II.2 *– Classification des algorithmes d'anticollision.*

- **Méthode fréquentielle FDMA** : pour Frequency Division Multiple Access, dans cette méthode le lecteur est composé de plusieurs antennes fonctionnent sur des fréquences différentes, une antenne est utilisée pour l'envoi des commandes et/ou l'énergie aux différentes tags qui sont présentes dans le champ d'action du lecteur , les autres antennes sont utilisées pour la réception des données des tags, tel que chaque antenne est allouée à une seule tag, donc le lecteur peut communiquer simultanément avec plusieurs tags (au maximum le nombre d'antenne qu'il contient) sur des fréquences différente [11].

- **Méthode TDMA** : Pour Time Division Multiple Access, c'est la plus couramment utilisée, consiste à diviser le temps de lecture en suite d'intervalles de temps, la mise en disposition de canal sera divisée entre ces intervalles. Il existe plusieurs variantes de cet algorithme, les plus utilisés sont les ALOHA-Based et Tree-Based

- Algorithmes qui seront détaillées dans la section suivante [11].

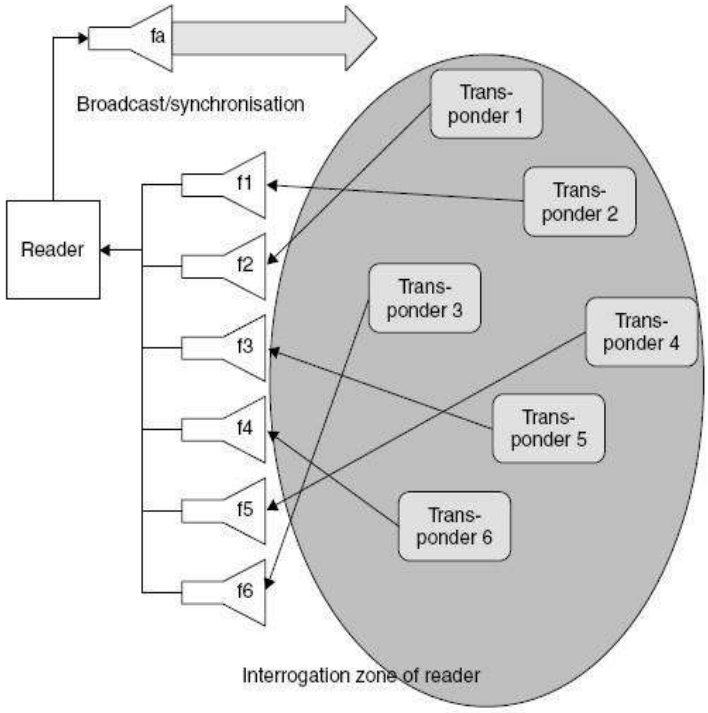


Fig II.3 *– Schéma de la méthode fréquentielle.*

- **Méthode CDMA** (Code Division Multiple Access) : C'est une méthode de multiplexage qui utilise la technique d'étalement de spectre qui permet à plusieurs liaisons numériques d'utiliser la même fréquence porteuse. Elle est surtout utilisée dans les réseaux de téléphonie mobile, la télécommunication spéciale, militaire essentiellement, et dans les systèmes de navigation par satellite [12].

Le principe de cette technique repose sur l'utilisation d'une séquence de signale pseudo aléatoire unique à chaque tag. Le tag multiplie cette séquence avec l'information quelle veut envoyer au lecteur, dans le cas d'une collision le lecteur reçois un seule signale qui provient de plusieurs tags.

Pour séparer les données envoyées par une tag précise le lecteur doit connaitre au préalable la séquence que ce tag a utilisée pour coder son signale, ainsi il n'aura qu'à multiplier le signale reçu par cette séquence, le signale résultant contiendra une seule onde basse fréquence qui représente la donnée originale, à savoir l'identificateur du tag.[7]

CDMA et le reste des techniques d'étalement de spectre sont difficiles et couteuse à implémenter dans des systèmes RFID car ces méthodes nécessitent un système et un schéma de codage très complexe.

- **Méthode spatiale SDMA** : pour Scape Division multiple Access dans cette méthode la lecture est doté d'une antenne directionnelle ou une antenne intelligente, afin de permettre au lecteur de couvrir petit à petit son entourage (l'espace) pour communiquer avec un seul tag à un moment donné. Cette méthode possède un inconvénient majeur, la présence de deux marqueurs à faible distance l'un de l'autre rend cette méthode inefficace [11].

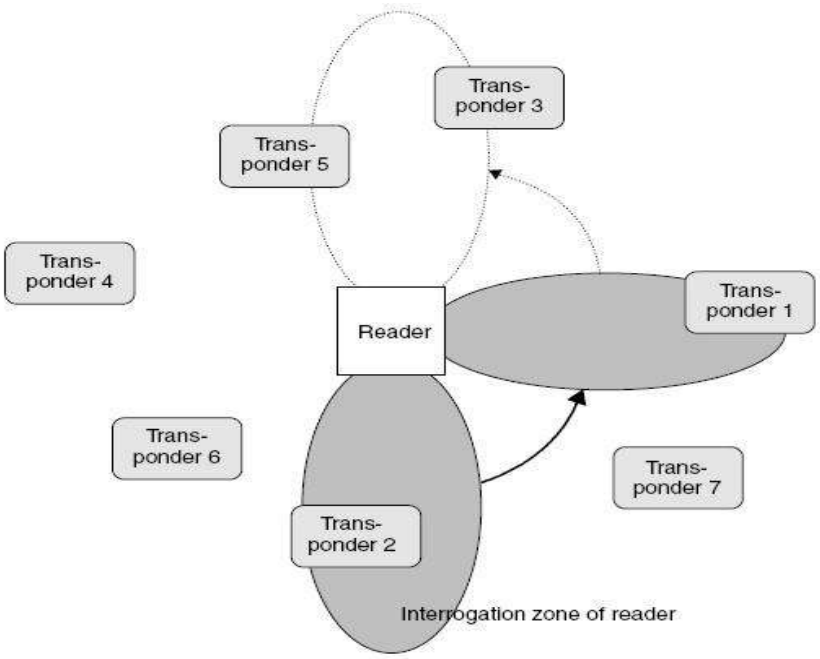


Fig II.4 *– Schéma de la méthode spatial.*

II.2 Algorithmes probabilistes et déterministes

Dans la classe TDMA on distingue deux grandes familles d'algorithmes d'anticollision, les algorithmes probabilistes et les algorithmes déterministes.

II.2.1 Les algorithmes déterministes

Dans le cas des algorithmes déterministes, on est sûr d'identifier tous les tags dans un temps donné. Dans ces algorithmes le lecteur envoie des commandes aux tags qui se trouvent dans son champ de lecture, ces commandes dépondent des identifiants des tags. Les algorithmes d'anticollision déterministes utilisent les arbres en tant que structure de base comme l'algorithme de l'arbre binaire (Binary Tree Algorithm) et l'algorithme de l'arbre de requêtes (Query Tree Algorithm).

Dans l'algorithme de l'arbre binaire, quand le lecteur détecte une collision il divise les tags en deux groupes et fait la même chose pour chaque groupe ainsi formant un arbre binaire jusqu'à identifier tous les tags.

Pour l'algorithme de l'arbre de requêtes, le lecteur envoie à chaque tour de boucle un préfix d'identificateur, toutes tags qui a un identificateur ainsi préfixé vas répondre au lecteur par son identificateur, si le lecteur détecte une collision il ignore ce qu'il vient de recevoir et rajoute un bit au préfixe formant un arbre de préfixes qui a pour feuilles les identificateurs des tags identifiées.

L'efficacité des algorithmes déterministes est affectée par la longueur des identificateurs et leurs distributions mais aussi le nombre élevé des tags dans la zone de lecture [10] [8].

Ce type d'algorithmes a été largement utilisé dans les réseaux classiques. Mais la plupart des suppositions faites pour le réseau classique ne s'applique pas sur les systèmes RFID ceci dû à diverses limitations tel que la mémoire et la capacité de calcule.[12]

Pour passer outre ces limitations, quelques variantes de l'algorithme d'anticollision Binary-Tree émergée des réseaux classiques ont été présentées.

II.2.1.1 Recherche binaire

Dans cet algorithme la collision est résolue en diminuant progressivement lest bit qui sont entrés en collision, le code Manchester est utilisé pour la codification des informations échangées entre le lecteur et les tags, dans ce code le 0 logique est représenté par un front montant et le 1 logique est représenté par un front descendant [14].

À la réception de l'information le récepteur (qui peut être le lecteur ou l'étiquette) reçoit les bits de cette information de poids fort vers le poids faible.

La figure II.5 un exemple de codification de 00100011011 en code Manchester. Le code Manchester permet au lecteur de savoir si une collision s'est produite et quel est le bit exacte de la collision, quand un lecteur reçoit un code d'identification, alors d'après le codage Manchester il devrait y avoir une transition de signale après chaque état (bit), si ce n'est pas le cas ça vaudra dire qu'il y a une erreur de transmission, dans notre cas les signaux se sont superposées causant une collision.

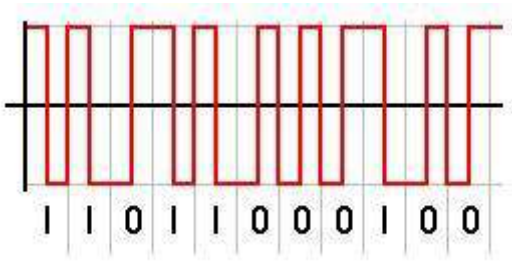


Fig II.5 *– Exemple de signal numérique.*

Voici les commandes utilisées dans cet algorithme :

- REQUEST(UID) : c'est une commande d'identification sera envoyée par le lecteur aux tags qui se trouvent dans son champ d'action, contient en paramètre un numéro de série, après la réception de la commande chaque tag va vérifier si son identifiant est inférieur ou égale à ce numéro de série, si le cas le tag va envoyer son identifiant unique, sinon elle va ignorer la commande.

- SELECT(ID) : cette commande sera envoyée au tag identifié par le lecteur, elle possède en paramètre le numéro de l'identifiant du tag identifiée et va permettre à ce tag de répondre aux prochaines commandes qui seront envoyées par le lecteur tel que les commandes de Lecteur/Ecriture et d'autre commandes sauf la commande REQUEST, à la réception de la commande chaque tag qui se trouve dans le champ d'action de lecteur va comparer son identifiant avec le numéro passé en argument.

- LECTURE : cette commande va être traitée par tag sélectionnée par la commande SELECT, elle permet à ce tag d'envoyer des informations qui sont stockées dans sa mémoire.

- ECRITURE (Data) : cette commande va être traitée par tag sélectionnée par la commande SELECT, elle contient des données à écrire dans la mémoire du tag.

- UNSELECT : cette commande va être traitée par tag sélectionnée par la commande SELECT, elle permet de désélectionner le tag ça veut dire qu'elle va plus répondre aux prochaines commandes envoyées par le lecteur, même pas la commande REQUEST, le tag sera mué, comme s'il n'existe pas dans le champ de lecteur.

Pour commencer la lecteur va envoyer une commande REQUEST à toutes les tags qui se trouvent dans son champ de lecture, cette commande à comme paramètre le numéro maximum qu'une tag peut avoir c'est le numéro de série le plus élevé, ça veut dire que si la taille des identificateurs des tags est sur n bits tous ces n bits de ce numéros seront à 1,après la réception de la commande chaque tag va envoyer son identifiant , car toutes les

identifiants des tags sont inférieurs ou égale à numéro envoyé par le lecteur , chaque identifiant d'une tag sera codé en Manchester et sera envoyé par des ondes radiofréquence.

Après l'envoi des identificateurs, le lecteur reçoit un seul signal, on peut avoir deux cas, soit le message est bien reçu donc aucune collision ne s'est produite, donc une tag est identifiée, le lecteur va envoyer à cette tag la commande SELECT, puis il va envoyer à cette tag des commande LECTEUR ou ECRITURE selon les besoins , quand le lecteur veut terminer la communication avec cette tag il va la désélectionner à l'aide de la commande UNSELECT, pour qu'il peut identifier d'autres tags et cette tag ne répond plus aux prochaines commandes, le deuxième cas lorsque une collision est produite, le lecteur va deviser les tags qui ont répondues à la commande REQUEST précédente en deux groupes selon le bit de collision, tel que seules les tags qui ont le bit de collision 0 vont répondre au prochaine commande REQUEST, le lecteur va envoyer la commande REQUEST avec comme paramètre le numéro de série qui va permettre au le lecteur de faire cette division, le numéro dépend de la position de bit de collision on peut avoir trois cas : On suppose que la taille des identificateurs des tags est sur n bits

- Si le bit de collision est le premier bit lu par le lecteur, ça veut dire le bit de poids fort des identifiants des tags, dans ce cas le numéro qui sera envoyé par le lecteur contient le bit 0 comme poids fort et tous les autres bits seront complétés par 1.

- Si le lecteur à déjà lu des bits avant la collision et le bit de collision n'est pas le dernier bit (n'est pas le bit de poids faible), donc les tags qui ont répondues à la commande REQUEST précédente ont des bits communs, dans ce cas le lecteur va envoyer un numéro qui contient au début les bits communs lus sans collision puis le bit de collision va être remplacés par 0 et un ou les bits restants vont être complétés par des 1.

- Si le bit de collision est le bit de poids faible des identifiants des tags, donc le lecteur a déjà lu des bits dans ce cas il va envoyer un numéro qui contient les bits lus et le bit de collision sera remplacé par 0.

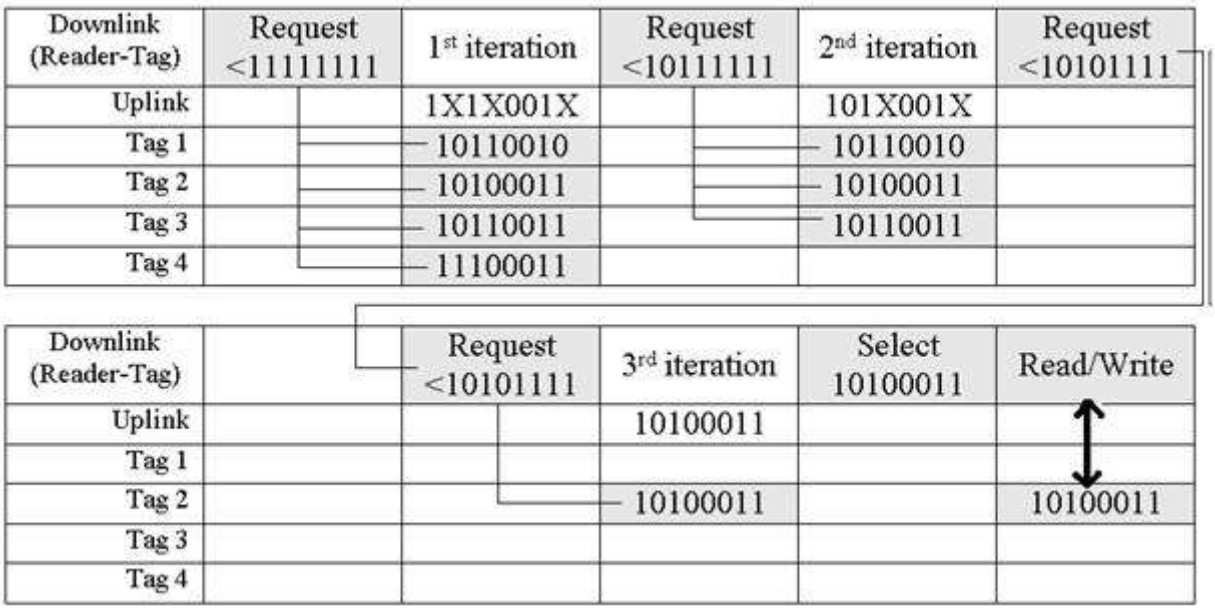


Fig II.6 *– Exemple de déroulement de l'algorithme Binary Search.*

Le lecteur va continuer à diviser les tags qui répondent à la commande REQUEST jusqu'à l'identification d'une tag, après la communication avec tag identifiée le lecteur va retourner aux collisions précédentes de dernière collision vers la première et à chaque collision il va envoyer la commande REQUEST qui contient le même numéro qui a envoyé déjà sauf le bit de collision sera remplacé par 1,si une collision est produite il va procéder de la même façon que la première fois, le lecteur continue comme ça jusqu'à l'identification de toutes les tags [13][8]. La figure II.6 est un exemple d'identification d'un tag en utilisant l'algorithme Binary Search.

II.2.1.2 Le Protocole de Query Tree

Dans les variantes de découpage d'arbres, les tags nécessitent un générateur de nombres aléatoires et un compteur, ce qui les rend coûteuses et complexes sur le plan informatique. Les protocoles Query Tree (QT) surmontent ces problèmes en stockant les informations relatives à la construction des arbres au niveau du lecteur, et les tags doivent uniquement disposer d'un circuit de correspondance des préfixes. Le protocole Query Tree est un protocole sans mémoire car il n'exige pas que les tags soient équipés d'une mémoire supplémentaire inscriptible sur la puce. Il existe de nombreuses variantes de protocoles Query Tree existent [15]-[17]. Nous en présentons quelques-unes dans les sections suivantes.

II.2.1.2.1 Query Tree

Le protocole *Query Tree* a été proposé par Law et al [15]. Dans ce protocole, chaque tag a un préfixe correspondant. Le lecteur transmet une requête aux tags contenant une chaîne de bits q. Un tag dont le préfixe d'identification correspond à q répondra au lecteur. Si un seul tag répond, le *tag* est identifié avec succès. Si plusieurs *tags* répondent simultanément, le lecteur émet à nouveau une requête avec une chaîne de bits q plus longue, avec un bit supplémentaire, 0 ou 1, ajouté à q, c'est-à-dire q0 ou q1. Évidemment, les tags avec le préfixe q sont divisées en deux sous-groupes q0 et q1. La procédure de division sera exécutée de manière répétée jusqu'à ce que chaque *tag* dans la zone d'interrogation soit identifiée avec succès. L'algorithme du protocole QT peut être repris par l'algorithme 1.

Le tableau II.1 ci-dessous présente un exemple de protocole QT. Nous supposons qu'il y a quatre tags avec les IDs 0001, 0011, 1010 et 1101. Où est la chaîne vide. La figure II.7 présente le diagramme en arbre de cet exemple. Nous pouvons remarquer que la longueur des ID des tags affecte le délai d'identification du protocole QT. Law et al. [15] proposent également de nombreuses variantes du protocole QT. Elles sont présentées ci-dessous.

Algorithme 1 : L'algorithme de QT

Reader

1: $Q = \{0, 1\}$
2: Broadcast query q_1 to the tags.
3: **if** the tag reply is string w **then**
4: update Q to be $\{q_2, \ldots, q_l\}$
5: **end if**
6: **if** a collision is detected **then**
7: set Q to be $\{q_2, \ldots, q_l, q_1 0, q_1 1\}$
8: **else if** there is no reply **then**
9: update Q to be $\{q_2, \ldots, q_l\}$
10: **end if**
11: Go to 2 until Q is empty

Tag

1: **if** $w_1 w_2 \ldots w_n = q$ **then**
2: tag sends string w to the reader
3: **end if**

Tableau II.1 *– Communication entre le Reader et le tag avec le protocole de QT.*

	Slot 1	Slot 2	Slot 3	Slot 4	Slot 5
Query(q)	/	0	1	00	01
Response	Collision	Collision	Collision	Collision	Empty
Q	0,1	1,00,01	00,01,10,11	01,10,11,000,001	10,11,000,001
	Slot 6	Slot 7	Slot 8	Slot 9	
Query(q)	10	11	000	001	
Response	1010	1101	0001	0011	
Q	11,000,001	000,001	001	/	

II.2.1.2.2 Query Tree raccourcis

Lorsque le lecteur reçoit un slot de collision, deux requêtes q0 et q1 sont ajoutées dans sa Queue Q. L'algorithme recherche d'abord la requête q0. S'il s'avère qu'il y a un slot vide, c'est-à-dire aucun tag avec le préfixe q0, alors nous savons qu'il y a au moins deux tags avec le préfixe q1. Par conséquent, le lecteur saute la requête q1 et insère les requêtes q10 et q11 [15].

Le tableau II.2 présente un exemple pour cette variante. Nous supposons qu'il y a quatre tags avec des ID uniques 0101, 0111, 1010 et 1101. Dans l'emplacement 4, le lecteur détecte un emplacement vide avec la requête q = 00, ce qui signifie que le lecteur sait qu'il existe au moins deux tags avec le préfixe '01'. Par conséquent, le Par conséquent, le lecteur ignore la requête '01' et insère '010' et '011' dans Q. La Figure II.7 montre son tree diagramme.

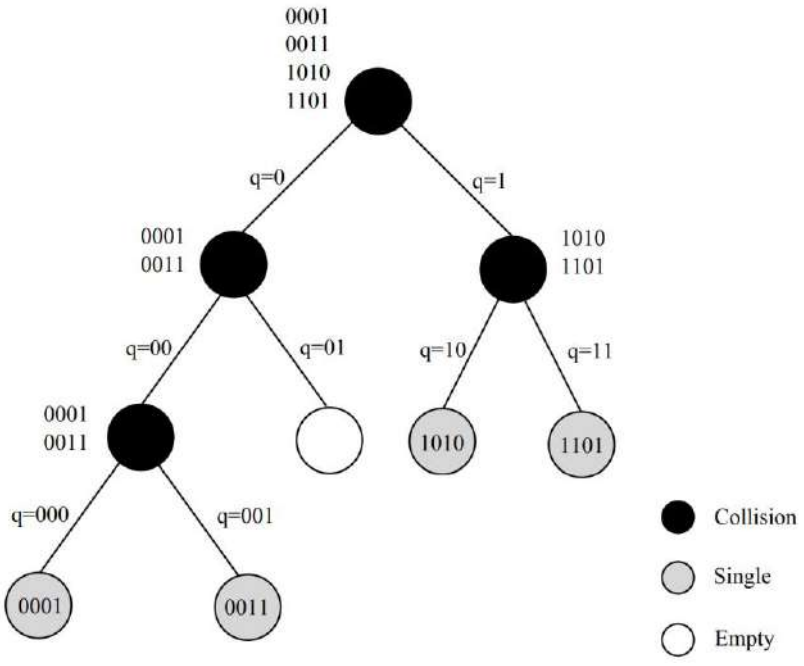


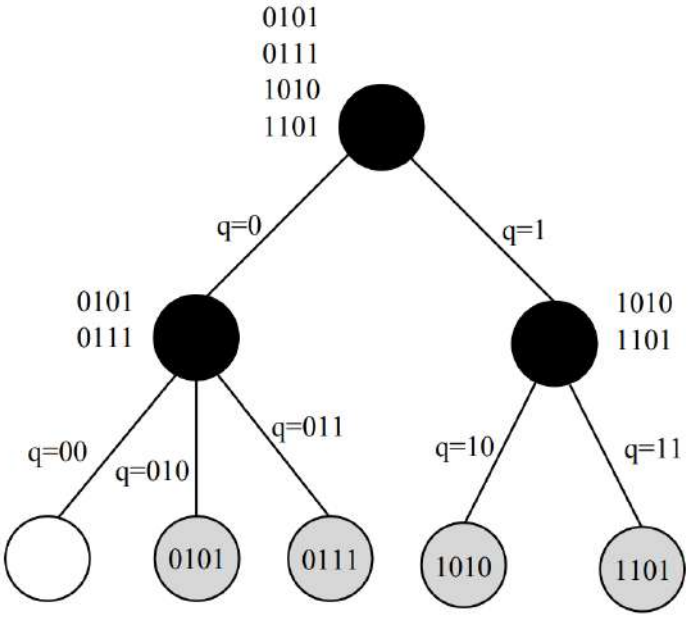
Fig II.7 *– Tree diagramme de protocole de QT.*

Tableau II.2 *– Example de protocole de QT shortcutting.*

	Slot 1	Slot 2	Slot 3	Slot 4
Query(q)	/	0	1	00
Response	Collision	Collision	Collision	Empty
Q	0,1	1,00,01	00,01,10,11	10,11,010,011
	Slot 5	Slot 6	Slot 7	Slot 8
Query(q)	10	11	010	011
Response	1010	1101	0101	0111
Q	11,000,001	010,011	011	/

II.2.1.2.3 Query Tree Short-Long

Le protocole QT-sl (query tree short-long) [15] réduit le nombre de bits transmis par les tags. Les tags, ce qui réduit le délai d'identification. Nous notons que dans les protocoles QT, un grand nombre de réponses binaires des tags se terminent par des collisions. Bits transmis par les tags finissent en collisions. Pour minimiser ces gaspillages, nous pouvons avoir deux types de requêtes de la part du lecteur. Les requêtes courtes n'induisent que des réponses de 1 bit de la part des tags, tandis que les requêtes longues induisent des réponses de 1 bit de la part des tags. Les tags, tandis que les requêtes longues induiront les identifiants complets des tags.



.

Fig II.8 *– Tree diagramme de protocole de QT shortcutting de Tableau II.2.*

Le lecteur enverra une requête longue uniquement lorsqu'il sait qu'un seul tag correspond au préfixe. Il faut noter qu'avec le protocole QT-sl, nous un bit supplémentaire pour spécifier si la requête est courte ou longue.

II.2.1.2.4 QT Améliorant

Le protocole QT Améliorant (IQT) a été proposé par Zhou et al [16]. Ce protocole a amélioré le protocole QT, en réduisant le nombre de bits transmis des tags au lecteur. La principale caractéristique du protocole IQT est que le lecteur surveille les réponses des tags bit par bit. Si une collision se produit sur un bit particulier, le lecteur signale aux tags qu'ils doivent cesser de transmettre. Dans ce protocole, les tags renvoient les bits restants de leurs identifiants si le préfixe de la requête correspond au préfixe de leurs identifiants. Le lecteur, une fois qu'il a détecté un bit de collision, donne aux tags le signal "stop" pour arrêter la transmission. Le nombre de bits envoyés sera ainsi réduit. Le tableau II.3 montre un exemple d'IQT dans lequel quatre tags ont les ID 0001, 0011, 1000 et 1100. L'algorithme de sélection des requêtes dans IQT n'est pas le même que celui de QT. Par exemple, dans le slot 2, après que le lecteur ait reçu un bit 0 et un bit de collision. Au lieu d'insérer les requêtes 00 et 01. Le lecteur insère 000 et 001.

Tableau II.3 *– Example de protocole de IQT.*

	Slot 1	Slot 2	Slot 3	Slot 4	Slot 5	Slot 6	Slot 7
Query(q)	/	0	1	000	001	10	11
Response	x	0x	x	1	1	00	00
Identified				0001	0011	1000	1100
Q	0,1	1,000,001	000,001,10,11	001,10,11	10,11	11	/

II.2.2 Les algorithmes probabilistes

La famille des probabilistes comme ALOHA, Slotted-ALOHA, Framed-slotted-ALOHA et le Dynamic Framed-slotted-ALOHA se basent sur un principe probabiliste. En ALOHA le lecteur demande aux tags de choisir au hasard un intervalle de temps dans le quelle elles enverront leurs identificateurs, mais laisse la liberté aux tags de choisir la taille de cet intervalle. En Slotted-ALOHA, Framed-Slotted-ALOHA et Dynamic Framed-Slotted-ALOHA le lecteur demande aux tags de choisir au hasard un intervalle de temps appelé slot dans le quelle elle envoie son identificateur et la durée de slot est choisi par le lecteur. [8].

Cette façon de faire ne garantit pas l'identification de tous les tags dans une période spécifique, et peut aussi engendrer des cas de famine.

II.2.2.1 Basic ALOHA

C'est le plus simple des algorithmes probabilistes et des algorithmes d'anticollision, il a étais développé en 1970, il est utilisé pour lire le contenu des tags seulement ,dans cet algorithme le lecteur envoie une seul commande de lecture pendant toute la période de lecture des tags, cette commande sera envoyée au début de l'algorithme aux tags qui se trouvent dans le champ d'action du lecteur, pour que ces tags commencent à envoyer leurs identifiants, après la réception de la commande chaque *tag* choisit aléatoirement un intervalle du temps [n.m] qui sera le temps de transmission de l'identifiant, mais ce choix n'est pas synchronisé avec le lecteur ni avec les autres tags. La transmission de l'identifiant de chaque *tag* commence au début de l'intervalle du temps choisit n et se termine à la fin de l'intervalle choisit m [8] [11].

Dans cet algorithme on peut avoir deux types de collision, la première se produit lorsque deux ou plusieurs tags choisissent le même intervalle de temps, donc ces tags vont envoyer leurs identifiants au même temps par des ondes radiofréquence, donc ces ondes vont se superposer et le signal reçu par le lecteur contiendra une information incorrecte, cette collision est appelée collision totale. Le deuxième cas de collision se produit lorsque une *tag* communique son identificateur au lecteur et qu'une autre *tag* commence à envoyer le sien, il y'aura une collision entre les ondes radio de la première *tag* et la deuxième, dans ce cas le lecteur aura déjà lu au moins un bit de la première *tag*, cette collision se produit car les tags ont choisies deux intervalles de temps qui s'intersecté, on aura deux intervalles [m, n] et [m',n'] avec soit m' ou n' incluse dans [m, n], ce type de collision est appelé collision partiel.[9] Dans les deux cas de collision, après que le *tag* est terminée l'émission de son identificateur elle ne recevra pas d'accusé de réception de la part du lecteur, donc elle déduira qu'il y'a eu une erreur de transmission (dans ce cas une collision) et rechoisira aléatoirement un autre intervalle de transmission.

Si aucune erreur n'est survenue, le *tag* reçoit un accusé de réception positive. Cet algorithme est répété jusqu'à ce que tous les tags soient bien identifiés.

II.2.2.2 Slotted ALOHA

Cet algorithme est une amélioration de l'algorithme basic aloha, le cycle de lecture est divisé en intervalle de temps de même duré appelées slot, le nombre de slots est limité et est choisis au début de cycle de lecture et donc :

$$\text{La durée de cycle de lecture} = \text{Nombre de slots} \times \text{la durée d'un slot} \qquad \text{(II.1)}$$

La différence entre les slots c'est le numéro d'ordre de slot dans le cycle de lecture. Au début, le lecteur envoie une commande de lecture aux tags qui se trouvent dans son champ d'action, après la réception de la commande, chaque *tag* va choisir aléatoirement un numéro de slot parmi les numéros de slot proposées par le lecteur dans la commande de lecture puis va attendre son tour pour envoyer son identifiant ça veut dire elle va attendre le début de son slot (intervalle de transmission).

Le point fort de cet algorithme c'est qu'on ne peut pas avoir la collision partielle car il n'y a pas d'intersection entre les slots, mais on peut avoir des collisions totale dans le cas où deux ou plusieurs tags choisissent le même numéro de slot et dans ce cas les tags qui sont entrées en collision vont choisir aléatoirement à la fin de leurs slots un autre numéro slot puisqu'ils n'ont pas reçu d'accusé de réception et ce slot doit être bien sûr supérieur au slot précédent et inférieur ou égale au dernier slot de cycle de lecture [8].

Dans le cas où le lecteur a réussi à identifier un tag il lui enverra un accusé de réception pour qu'elle n'envoie pas une autre fois son identificateur, cette dernière ne répondra pas aux prochaines commandes de lecture s'il y'aura autres cycle de lecture. On peut avoir aussi des slots qui n'ont pas étais choisis par aucune *tag* et dans ce cas le lecteur va chômer dans tout la période ces slots.

Dans le cas où une collision s'est produite dans le dernière slot de cycle de lecture, le lecteur va envoyer une autre commande de lecture aux tags qui n'ont pas été identifiées lors du cycle de lecture précédent (les tags qui n'ont pas reçues d'accusés de réception), ainsi un autre cycle de lecture est lancé et n'importe quel cycle lancé sera divisé en slots et le nombre de slots sera aléatoire, qui veut dire qu'il dépend pas des cycles précédents, mais la taille des slots et la procédure de lecture des *tags* resteront toujours les mêmes, on continue comme ça jusqu'à l'identification de toutes les tags.[10] [33], [34], [35].

II.2.2.2.1 Slotted ALOHA avec Muting ou Slowdown

Cette variante a le même principe de fonctionnement que l'ALOHA pur avec inhibition ou ralentissement, mais fonctionne en slotted manner (Fig II.9).

II.2.2.2.2 Slotted ALOHA avec Early End

Un lecteur ferme un slot s'il ne détecte aucune transmission à son début. Le lecteur utilise deux commandes : start-of-frame (SOF) et end-of-frame (EOF). La première est utilisée pour démarrer un cycle de lecture et la seconde est utilisée par le lecteur pour

fermer un slot vide. La figure II.10 montre comment la fin anticipée est utilisée pour clôturer un slot vide.

II.2.2.2.3 Slotted ALOHA avec Early End et Muting/Slow Down

Lorsqu'un tag est identifié, le lecteur envoie une commande de sourdine ou de ralentissement au tag. Le tag s'arrête ou ralentit pour envoyer son ID dans les slots suivants. En outre, comme l'exige la fonction En outre, comme l'exige la fonction de fin anticipée, les emplacements vides sont fermés de manière anticipée à l'aide de la commande EOF.

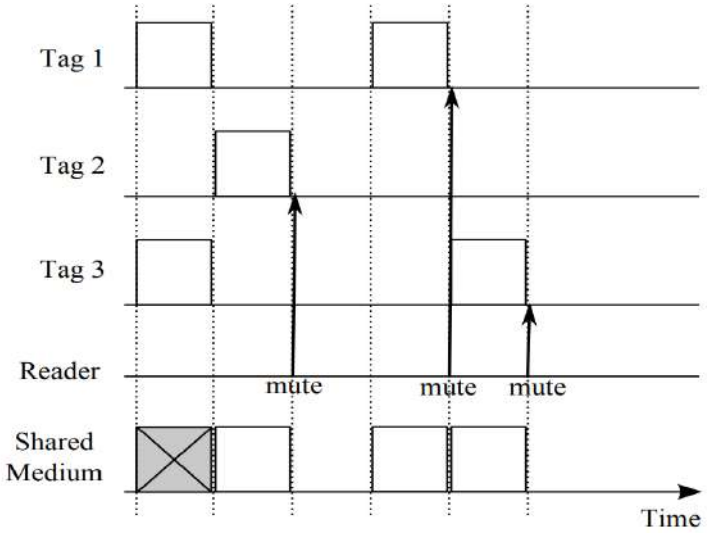


Fig II.9 *– Slotted ALOHA avec un protocole de muting.*

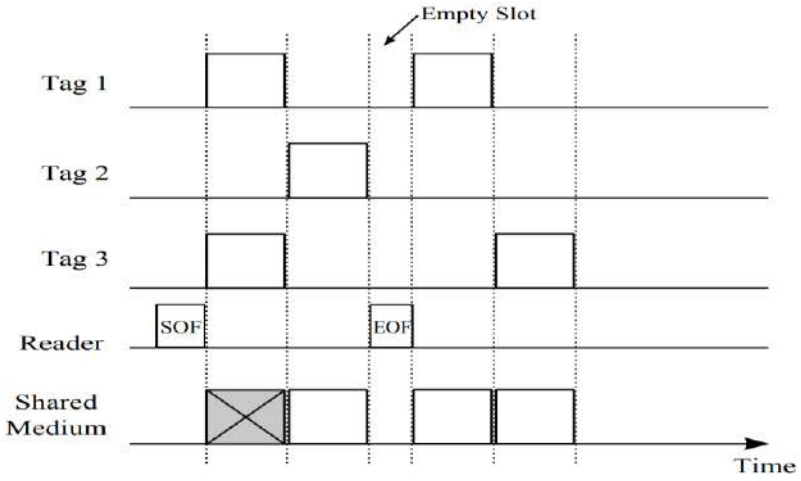


Fig II.10 *– Slotted ALOHA avec Early End.*

II.2.2.3 Cadre de base fendu ALOHA

On introduit dans cet algorithme la notion de frame, un frame est un cycle de lecture de durée fixe, n'importe quel frame lancé par le lecteur aura la même durée que le premier frame. Chaque frame est divisé en suites d'intervalles de temps de mêmes durés appelées

slots, puisque la durée de cycle de lecture (frame) est fixée donc tous les cycles de lecture aurons le même nombre de slots. Chaque frame est lancé par la commande de lecture [8].

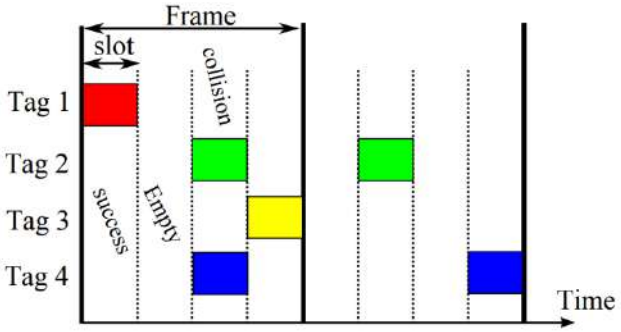


Fig II.11 *– Le protocole de FSA.*

Dans cet algorithme il ne peut y avoir que la collision totale pour la même raison que l'algorithme *slotted ALOHA*, les commandes de lecture envoyés par le lecteur contiennent la durés du frame et le nombre de slot possibles, après la réception de la commande chaque *tag* va choisir un numéro de slot aléatoirement parmi les numéros proposé par le lecteur puis elle vas attendre le début de son slot pour commencer à envoyer son identifiant unique, à la fin de chaque frame le lecteur va envoyer des accusés de succès aux tags identifiées pour que ces tags ne répondent pas aux prochaines commandes de lecture, ça veut dire dans le prochaine frame . En cas où une collision est produite, les tags qui sont rentrées en collision doivent attendre le prochain frame, ça veut dire la prochaine commande de lecture.

Dans cet algorithme pour que le lecteur envoie une autre commande de lecture aux tags il suffit qu'au moins une collision se produise dans le frame précédent, le lecteur continue comme ça jusqu'à ce qu'il n'y ait aucun frame dans lequel s'est produite une collision [9][10] On peut aussi être dans le cas où un ou plusieurs slots qui n'ont pas choisi par aucun tag et pendant ces slots le lecteur va chômer. (Voir figure II.11 et figure II.12)

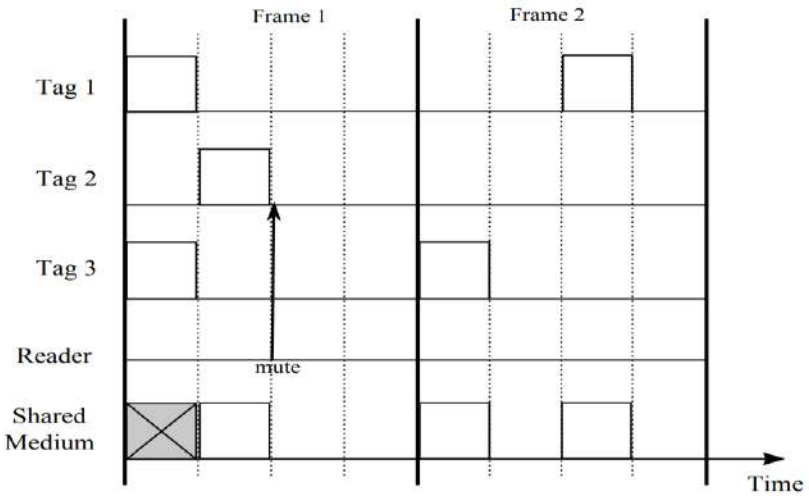


Fig II.12 *– BFSA avec un protocole de muting.*

II.2.2.4 Dynamic Framed Slotted ALOHA Algorithme

Le problème des variantes de BFSA est l'utilisation d'une taille de trame constante. Si le nombre de tags est supérieur à la taille de la trame, un grand nombre d'emplacements de collision se produiront dans un tel protocole. Inversement, si le nombre de tags est inférieur à la taille de la trame, le lecteur connaîtra de nombreux créneaux vides. Il est donc nécessaire d'ajuster la taille de la trame de façon dynamique pour qu'elle corresponde au nombre de tags. La taille optimale du cadre est liée au nombre de tags [18]. C'est pourquoi la majorité des protocoles ALOHA dynamiques à créneaux encadrés (DFSA) utilisent une fonction d'estimation des tags pour faire varier la taille de la trame. Les fonctions d'estimation des tags estiment le nombre de tags sur la base de certaines informations de la trame précédente, telles que la taille de la trame F, le nombre de slots en collision c_k, de slots réussis c_1 (slots remplis avec un tag) et de slots vides c_0 [19],[20], [37]-[39].

II.2.2.4.1 EPC Génération 2 Protocole

Le protocole EPC de génération 2 (EPC G2) [14] est largement utilisé dans la gamme des ultra-hautes fréquences (UHF) pour les communications à 860-960 MHz, où la première publication de la norme par EPCglobal en 2004. Le protocole EPC G2 a adopté le protocole DFSA pour résoudre le problème des collisions. Ce protocole utilise une technique simple d'ajustement de trame. Le lecteur diffuse d'abord une commande "Query", qui contient un paramètre Q (un nombre entier compris entre 0 et 15). Il fixe ensuite la taille de sa trame à 2^Q.

Les tags choisissent alors un emplacement au hasard de 0 à 2^Q -1 et le mettent dans leur compteur. Le tag qui choisit zéro comme compteur d'emplacements répondra au lecteur avec un 'RN16' (nombre aléatoire de 16 bits). Si le lecteur reçoit un slot réussi de la part du tag (slot rempli d'un RN16), il transmettra un message au lecteur. Un RN16), il transmet une commande 'ACK', contenant le RN16 reçu. Après cela, le *tag* renvoie son ID (EPC) au lecteur. L'état du tag acquitté passe à l'état 'acknowledged'. L'état 'acknowledged', sachant que tous les tags initialisent leur état à l'état 'arbitré'. Ensuite, le lecteur émet une commande 'Query Rep' pour initier un autre slot.

Le paramètre 'Query Rep' demande aux tags de décrémenter leur compteur d'emplacements d'une unité. Les tags dont le compteur d'emplacements est égal à zéro renvoient leur RN16 au lecteur. Dans chaque slot, le lecteur ajuste le paramètre Q. Si un emplacement vide se produit, le lecteur diminue Q d'une constante C et envoie une commande 'Query Adjust' ou 'Query Rep'. Par contre, si un slot de collision se produit, le

lecteur augmente Q de la même constante et envoie une commande 'Query Adjust' ou 'Query Rep'. Ici, 0,1 < C < 0,5. Lorsque Q est grand, la valeur de C est petite, tandis que lorsque Q est petit, la valeur de C est plus grande. Avant le début de la trame suivante, en envoyant la commande 'Query Adjust', Q est arrondi. La commande 'Query Adjust' augmente ou diminue Q de 1 ou le laisse inchangé. A la réception de la commande 'Query Adjust', les tags choisissent une nouvelle valeur aléatoire pour leurs compteurs d'emplacements dans la plage (0, 2^Q - 1). Ce cycle se répète jusqu'à ce que tous les tags soient identifiés. L'algorithme 1 décrit en détail ce protocole.

Le choix du paramètre Q est considéré comme crucial et un algorithme Q (voir figure II.13) est utilisé pour ajuster sa valeur. Ici, le lecteur arrondit Q_{fp}, qui est une représentation en virgule flottante de Q et la définit comme la nouvelle valeur de Q [14]. Lorsque la trame est terminée, le lecteur émet la commande "Query Adjust" sans modifier Q. Cet algorithme est peu coûteux en termes de calcul et converge rapidement vers la taille de trame optimale. La figure II.14 montre l'exécution du protocole EPC G2 pour deux trames.

Lee et al. [21] ont montré que cette technique peut encore être améliorée comme suit. Premièrement, pour réduire les frais généraux de signalisation de l'algorithme, ils proposent d'envoyer la valeur de Q aux tags uniquement lorsqu'elle change, c'est-à-dire lorsque l'arrondi donne une nouvelle valeur de Q.

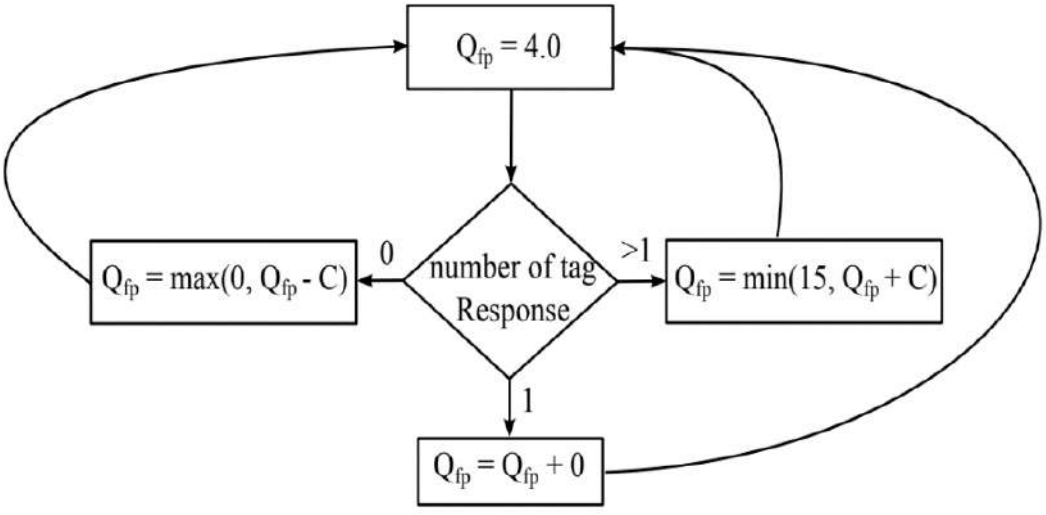


***Fig II.13** –L'algorithme Q [14].*

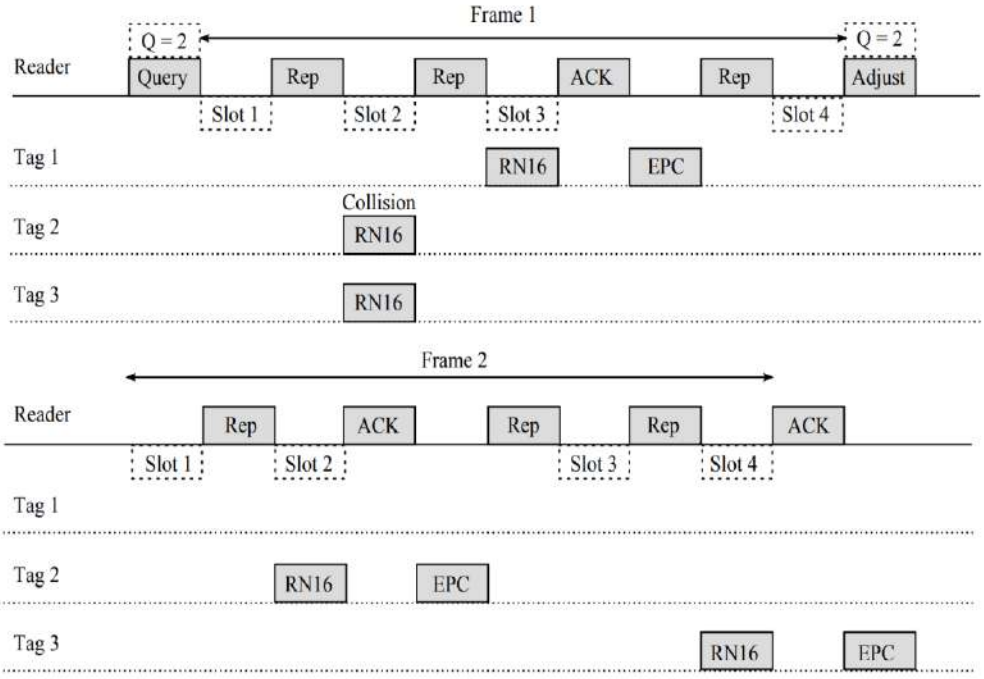


Fig II.14 *– Le protocole de EPC G2 anticollision.*

Deuxièmement, un lecteur peut utiliser différentes constantes et échelles pour mettre à jour la valeur de Q. Plus précisément, ils proposent de diminuer Q de Ci pour chaque emplacement vide, et d'augmenter Q de Cc pour les emplacements avec collisions. Si le nombre de tags, n, est connu, ces constantes sont calculées comme suit :

$$C_c = -0{,}0491 \times \ln n + 0{,}53 \quad \text{(II.2)}$$

$$C_c = (e-2) \times C_c \quad \text{(II.3)}$$

Sinon, ils sont calculés comme suit :

$$C_c = \text{toute valeur comprise entre } 0{,}1 \; et \; 0.5 \quad \text{(II.4)}$$

$$C_i = (e-2) \times C_c \quad \text{(II.5)}$$

Où e est la constante mathématique 2.71828. . ..

Dans [22], les auteurs proposent un changement dynamique de la constante C liée au paramètre Q après chaque ajustement de Q, ces changements suivent le concept mentionné dans [14] qui stipule que "lorsque Q est grand, le lecteur utilise une petite valeur de C, et une plus grande valeur de C pour un petit Q". C'est alors calculé comme suit :

$$C = 0{,}49 - Q \times 0{,}0255 \quad \text{(II.6)}$$

Algorithme 2 L'algorithme de EPC G2

```
All tags initialize their state to 'arbitrate' sate
Q_fp ← 4
Reader sends 'Query'
Every tag generates RN16 & slot counter
for inventory procedure do
  for current frame do
    if slot counter = 0 then
      Tag replies its RN16
    end if
    if a single tag replies then
      Reader sends 'ACK' with RN16
      if RN16 received by tag = RN16 saved in tag with 'arbitrate' sate then
        Tag sends EPC to reader
        Tag switches to the 'acknowledged' state
      end if
      Reader sends 'Query Rep'
    else if multiple tags reply then
      Q_fp ← min (15, Q_fp + C)
      Q ← round(Q_fp)
      if frame size adjusted then
        Reader sends 'Query Adjust'
        Every tag generates slot counter
      else
        Reader sends 'Query Rep'
      end if
    else if no tag replies then
      Q_fp ← max (0, Q_fp − C)
      Q ← round(Q_fp)
      if frame size adjusted then
        Reader sends 'Query Adjust'
        Every tag generates slot counter
      else
        Reader sends 'Query Rep'
      end if
    end if
    if tag receives 'Query Rep' then
      slot counter ← slot counter - 1
    end if
  end for
end for
```

II.2.2.4.2 DFSA Basé dans un Tag Estimation Fonctions

La taille de trame optimale est celle qui promet l'efficacité ou le débit maximum du système, où le débit est le rapport entre les tags lus avec succès dans une trame et la taille F de la trame. Et la taille F de la trame. Théoriquement, la taille optimale de la trame est égale au nombre de tags dans la zone d'interrogation du lecteur [23]. Zone d'interrogation du lecteur [23]. Par conséquent, il suffit d'estimer le nombre de tags. La figure II.15 montre le débit théorique en fonction du nombre de tags lorsque la taille de trame F est fixée à 256. Fixée à 256. Ce débit théorique peut être calculé comme suit $a_1^{F,n}/F$, avec n et $a_1^{F,n}$ (voir l'équation (II.10)) sont respectivement le nombre de tags et le nombre attendu de slots réussis. [24]. Nous pouvons voir que le débit maximal peut être atteint lorsque le nombre de tags et la taille de la trame sont approximativement les mêmes et qu'il est de 36,8%. Dans ce qui suit, nous allons décrire les principales méthodes d'estimation des tags.

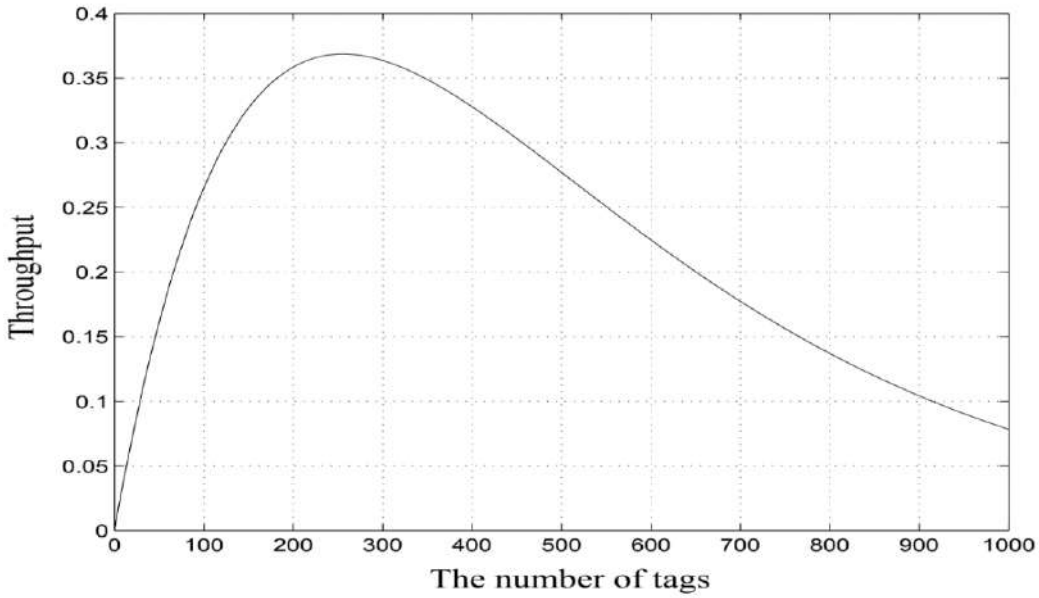


***Fig II.15** – Throughput de framed slotted ALOHA de frame size F = 256.*

- **La méthode de Schoute [25]** La méthode de Schoute estime le nombre de tags comme :

$$n_{est} = 2.39 \times c_k + c_1 \qquad \text{(II.7)}$$

Où n_{est} est le nombre estimé de tags, c_k est le nombre de slots entrés en collision et c_1 le nombre de slots réussis.

- **Lower bound** Cette méthode a été proposée par Vogt [18], [26]. Elle est basée sur l'observation qu'une collision implique au moins deux tags. Par conséquent, une limite inférieure sur la valeur du nombre estimé de tags peut être obtenue par :

$$n_{est} = 2 \times c_k + c_1 \tag{II.8}$$

- **L'inégalité de Chebyshev** Vogt [18], [26] a proposé une deuxième méthode basée sur l'inégalité de Chebyshev et visant à minimiser la distance entre un vecteur de résultat de lecture réel < (c_0, c_1, c_k) > et le résultat calculé théoriquement < $a_0^{F,n}$, $a_1^{F,n}$, $a_k^{F,n}$ > tel que représenté avec l'équation (II.9)

$$n_{est} = min_n \left| \begin{pmatrix} a_0^{F,n} \\ a_1^{F,n} \\ a_k^{F,n} \end{pmatrix} - \begin{pmatrix} c_0 \\ c_1 \\ c_k \end{pmatrix} \right| \tag{II.9}$$

Où $a_0^{F,n}$, $a_1^{F,n}$, $a_k^{F,n}$ sont les valeurs attendues des slots vides, des slots réussis et des slots de collision, respectivement, avec une taille de trame F et un nombre de tags n. Le nombre attendu de slots remplis avec r tags répondants $a_r^{F,n}$ est représenté par l'équation (II.10) [18]. L'expression (II.10) suppose que le nombre r de tags dans un slot est distribué de manière binomiale.

$$a_r^{F,n} = F \binom{n}{r} \left(\frac{1}{F}\right)^r \left(1 - \frac{1}{F}\right)^{n-r} \tag{II.10}$$

La méthode de l'inégalité de Tchebychev améliore la précision de l'estimation du nombre de tags, par rapport aux méthodes de la borne inférieure et de Schoute. Cependant, elle nécessite une charge de calcul assez importante.

Vogt [18] a également proposé une série de tailles de trame promettant des délais d'identification plus faibles pour une portée de tag donnée. Elles sont présentées dans le tableau II.4.

Tableau II.4 *– Taille optimale des cadres pour une gamme de tags donnée [18].*

Frame size F	16	32	64	128	256
N_{low}	1	10	28	57	130
N_{high}	9	27	56	129	∞

- **La méthode de Cha [23]** Cette méthode estime le nombre de tags en calculant le rapport entre le nombre de slots avec collisions C_k et la taille de la trame F, et est donnée par :

$$C_{ratio} = 1 - \left(1 - \frac{1}{F}\right)^{n} \left(1 + \frac{n}{F-1}\right) \quad \text{(II.11)}$$

Où n est le nombre de tags à estimer. Une estimation du numéro du tag est obtenue après une image en résolvant l'équation non linéaire $C_{ratio} = \frac{C_k}{F}$.

- **La méthode de Khandelwal [27]** Khandelwal et al. Proposent d'estimer le nombre de tags en utilisant :

$$n_{est} = \frac{\log \frac{C_0}{F}}{\log\left(1 - \frac{1}{F}\right)} \quad \text{(II.12)}$$

Où c_0 est le nombre de slots vides.

La méthode d'estimation des tags de Khandelwal ne peut être invoquée lorsque $c_0 = 0$. Dans ce cas, le nombre d'estimation des tags est obtenu par la borne inférieure $(2c_k + c_1)$. Enfin, Khandelwal et al. Ont proposé de fixer la taille de trame optimale à $F_{opt} = 1{,}943 \times n_{est}$.

- **La méthode de Kodialam [28**] Kodialam et al. Ont proposé une fonction d'estimation qui calcule le nombre prévu de slots vides et de slots de collision et $a_0^{F,n}$et $a_k^{F,n}$. Les équations résultantes sont ensuite utilisées pour dériver deux estimateurs, appelés estimateur zéro (ZE) et estimateur de collision (CE) :

$$ZE = e^{-\left(\frac{n_0}{F}\right)} = \frac{C_0}{F} \quad \text{(II.13)}$$

$$CE = 1 - \left(1 + \frac{n_k}{F}\right) e^{-\left(\frac{n_k}{F}\right)} = \frac{C_k}{F} \quad \text{(II.14)}$$

Où n_0 est l'estimation du nombre de tags obtenue à partir de ZE, et n_k est l'estimation du nombre de tags calculée à partir de CE, respectivement. Les valeurs de c_0 et c_k sont obtenues en observant le nombre de slots vides et de slots de collision dans une trame donnée. Si $n_0 < n_k$, alors le tag estimé est n_0, sinon il est n

- **La méthode de Chen [29**] Chen et Lin introduisent deux fonctions d'estimation : Chen-I et Chen-II. Dans la première, les auteurs calculent la probabilité d'avoir exactement k tags dans m slots.

$$p(k,m) = \frac{(-1)^m F!\, n!}{m!\, F^n} \times \sum_{j=m}^{min(F,(n/k))} (-1)^j \frac{(F-j)^{n-jk}}{(j-m)!\,(F-j)!\,(n-jk)!\,(k!)^j} \quad \text{(II.15)}$$

À l'aide de l'équation (2.14), les auteurs calculent la probabilité d'avoir exactement m slots avec zéro réponse du tag (k = 0). La valeur réelle de m est c0, qui est obtenue à partir du retour d'information du lecteur. L'équation de probabilité est ensuite résolue pour la valeur de n, qui est le numéro estimé du tag. Dans la seconde, c'est-à-dire Chen-II, la fonction calcule le nombre attendu de slots vides et de slots de succès en utilisant l'équation (II.10). La combinaison des résultats conduit à l'équation suivante:

$$n_{est} = \left(F - a_0^{F,n} - 1\right) \frac{a_1^{F,n}}{a_0^{F,n}} \quad \text{(II.16)}$$

Puis, le n_{est} est calculé en supposant que les nombres de slots vides et de slots réussis observés dans un cycle de lecture sont identiques à leurs attentes, c'est-à-dire que $a_0^{F,n} = c_0$ et $a_1^{F,n} = c_1$.

- **La méthode de Eom [30]** La méthode d'Eom estime le nombre de tags par slot de collision γk. Elle estime donc le nombre de tags. En supposant que la taille de la trame est suffisamment grande, l'estimation de γk se fait par les équations itératives suivantes:

$$\beta_k = \frac{F}{\gamma_{k-1} c_k + c_1} \quad \text{(II.17)}$$

$$\gamma_k = \frac{1 - e^{-1/\beta_k}}{\beta_k \left(1 - \left(1 + \frac{1}{\beta_k}\right) e^{-1/\beta_k}\right)} \quad \text{(II.18)}$$

Cette équation itérative converge vers le nombre de tags par slot de collision (γ_k). Où β_k désigne le rapport entre la taille du cadre et la quantité de tags après la k^{th} Le n_{est} d'estimation d'Eom du nombre de tags, est donné par :

$$n_{est} = \gamma_k c_k + c_1 \quad \text{(II.19)}$$

Starting with $\beta_1 = \infty$ and $\gamma_1 = 2$, γ_k* is the value that obeys the following condition:

$$|\gamma_{k*-1} - \gamma_{k*}| < \varepsilon \quad \text{(II.20)}$$

- **Maximum a posteriori (MAP)** Dans [31], [32] les auteurs proposent une méthode précise pour estimer la quantité de tags basée sur une distribution de probabilité a posteriori. La distribution exacte de probabilité a posteriori est dérivée et une règle de décision pour l'estimation du nombre de tags est proposée, appelée la règle de probabilité maximale a posteriori. Supposons que le résultat possible dans chaque slot est p_e pour vide, p_s pour réussi, et p_c pour collision. Comme la probabilité est soumise à la condition $p_e + p_s + p_c$ =1.

La probabilité que dans F slots, le slot vide se produise c0 fois, le slot réussi se produise c_1 fois. Et que le slot de collision se produise c_k fois :

$$P(c_0, c_1, c_k) = \frac{F!}{c_0!\, c_1!\, c_k} p_e^{c_0} p_e^{c_1} p_e^{c_k} \tag{II.21}$$

p_e, p_s et p_c peuvent s'écrire sous la forme d'une distribution binomiale (l'équation II.10)

$$p_e = \left(1 - \frac{1}{F}\right)^n \tag{II.22}$$

$$p_s = \frac{n}{F}\left(1 - \frac{1}{F}\right)^{n-1} \tag{II.23}$$

$$p_c = 1 - p_e - p_s \tag{II.24}$$

Par conséquent, le nombre estimé de tags est déterminé de telle sorte que la probabilité P (c_0, c_1, c_k) soit maximisée. Par conséquent, la règle de décision de la méthode d'estimation des tags est la suivante :

$$n_{est} = argmax\big(P(c_0, c_1, c_k)\big) \tag{II.25}$$

L'algorithme du DFSA basé sur les méthodes d'estimation des tags qui suivent la norme EPC G2 est présenté dans l'algorithme 3.

Algorithme 3 L'algorithme de DFSA

All tags initialize their state to 'arbitrate'
Reader sends 'Query' command containing Q
while no all tags are identified **do**
  $F \leftarrow 2^Q$
  $c_0 \leftarrow 0; c_1 \leftarrow 0; c_k \leftarrow 0$
  all tags generate its RN16 and slot counter (0 to $F-1$)
  for slot $\leftarrow 1$ **to** F **do**
    for each tag **do**
      if slot counter $= 0$ **and** state = ' arbitrate' **then**
        tag replies its RN16
      end if
    end for
    if a collision slot is detected **then**
      $c_k \leftarrow c_k + 1$
    else if an empty slot is detected **then**
      $c_0 \leftarrow c_0 + 1$
    else if a success slot is detected **then**
      $c_1 \leftarrow c_1 + 1$
      reader sends 'ACK'
      for each tag **do**
        if ACK = RN16 **then**
          tag sends EPC
          tag updates its state to 'acknowledged'
        end if
      end for
    end if
    if slot $< F$ **then**
      reader sends 'Query Rep'
      all tags decrement their slot counter by one
    end if
  end for
  reader calculates the estimated tags number (n_{est}) using F, c_0, c_1, and c_k
  $F_{opt} \leftarrow n_{est} - c_1$
  reader updates F_{opt} to be in form of $2^{Q_{opt}}$
  if $Q_{opt} = Q + 1$ **then**
    reader sends 'Query Adjust' containing of: "$Q_{opt} = Q - 1$"
  else if $Q_{opt} = Q - 1$ **then**
    reader sends 'Query Adjust' containing of: "$Q = Q - 1$"
  else if $Q_{opt} = Q$ **then**
    reader sends 'Query Adjust' containing of: "no change to Q"
  else
    reader sends 'Query' containing of Q_{opt}
  end if
  $Q \leftarrow Q_{opt}$
end while

II.3 Hybrid Anti-collision Algorithmes

Dans l'algorithme FS-ALOHA et QT expliqué ci-dessus, lorsqu'il y a beaucoup tags qu'un lecteur veut identifier, il existe beaucoup tags qui vont répondre en même temps et donc beaucoup de collisions ont lieu. En fin de compte, le temps d'identification des tags est plus long. L'algorithme FQT (Framed query tree) et l'algorithme QT-ALOHA (query tree ALOHA) présentés dans ce document sont des formes hybrides de l'algorithme FS-ALOHA et de l'algorithme QT.[36]

II.3.1 Framed Query Tree Algorithme

L'algorithme FQT divise les balises de manière aléatoire en unités de trame. Et dans cette unité, les balises sont identifiées en utilisant l'algorithme QT pour elles. L'opération réelle est la suivante :

Lorsqu'il demande aux tags de transmettre leur ID, le lecteur envoie également une taille d'époch̀e, c'est-à-dire le nombre total de trames. Ensuite, chaque *tag* décide aléatoirement de sa propre trame participante et ne répond que lorsque le lecteur interroge sa propre trame. Le lecteur exécute son processus d'identification dans chaque trame en utilisant l'algorithme QT. Cependant, le lecteur transmet aux tags le numéro de la trame ainsi que le préfixe de l'ID. Chaque *tag* confirme si le numéro de trame est le même que le numéro de trame qu'elle a choisi, et si c'est le cas, elle transmet son propre ID en regardant le préfixe et il est cohérent, comme dans l'algorithme QT existant. Après avoir identifié tous les tags dans une trame par le processus d'identification de l'algorithme QT dans la trame, le lecteur passe à la trame suivante. Ce processus est exécuté répétitivement pour chaque trame.

L'exemple de la Figure II.16 montre le processus d'identification de l'algorithme FQT lorsque le nombre des tags à identifier est de 8 et la taille de l'époque est de 4. Pour commencer, un lecteur transmet la taille de l'époque aux tags, puis chaque tag sélectionne sa propre trame de façon aléatoire. Trame au hasard. Pour trois tags qui ont sélectionné la trame 1, l'algorithme QT est utilisé pour identifier les tags. Pour identifier les tags. Par la suite, les tags sont identifiés en passant aux trames suivantes par la même méthode.

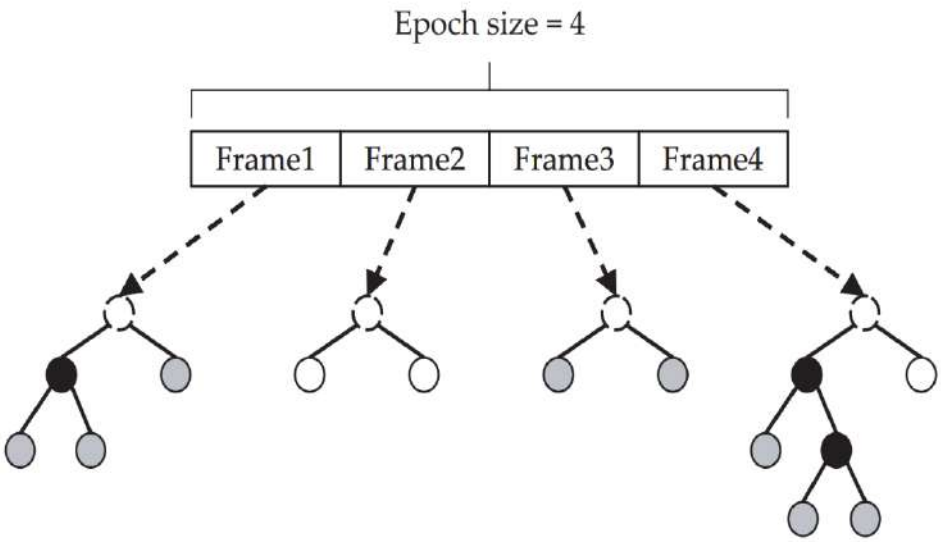


Fig II.16 *– Un exemple de processus d'identification dans l'algorithme FQT.*

Déterminer la taille d'époque la plus appropriée est très important pour améliorer les performances. Par intuition, une taille d'époque pour laquelle la profondeur de l'arbre ne dépasse pas 2 lors de l'exécution de l'algorithme QT présente les meilleures performances. La raison en est que lors de l'exécution de " 0 ", " 1 " dans la file d'attente initiale de l'algorithme QT, le cas où deux tags au total, un tag commençant par "0" et une autre commençant par "1", sont identifiées est le cas le plus idéal. La trame 3 est exactement le meilleur cas. En supposant que les tags à identifier est N et que la taille de l'époque est ES, l'idéal ES idéal peut être exprimé comme suit :

$$N = 2 \times ES \tag{II.26}$$

Ceci peut être vérifié par la simulation. Mais il y a un gros problème dans ce cas. En effet, il est difficile de déterminer une taille d'époque appropriée dès le début, car la procédure d'identification des tags est lancée sans connaître N, c'est-à-dire le nombre des tags à identifier. C'est pourquoi l'algorithme FQT final utilise le FFT (First Frame Test). Le FFT commence à partir d'une petite taille d'époque et arrête son processus d'identification lorsque la première trame présente des collisions dépassant un seuil de collision, puis reprend son identification des tags en augmentant la taille de l'époque. Comme il est supposé que tous les tags soient divisés aléatoirement en cadres, si de nombreuses collisions se produisent dans le premier cadre, les cadres restants sont plus susceptibles d'avoir une telle tendance.

Comme le montre le meilleur cas ci-dessus, l'image II.16 de la figure II.16, un arbre peut avoir la meilleure performance lorsque ses balises sont deux et que la profondeur est de 1. Performance lorsque ses balises sont au nombre de deux et que la profondeur

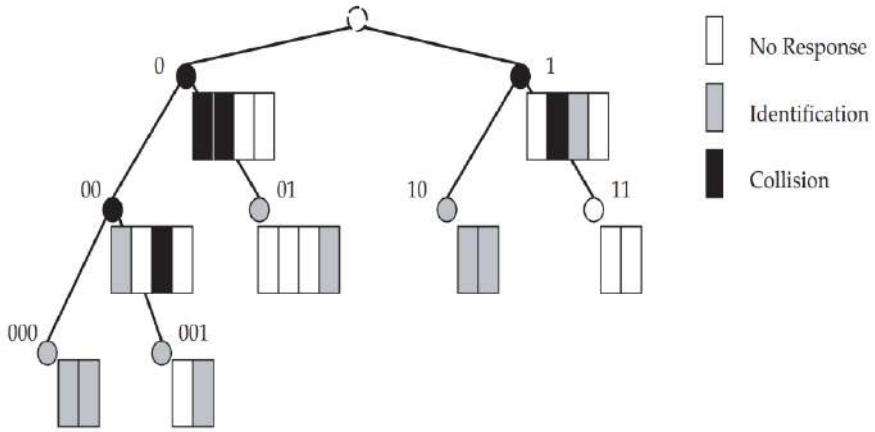
est de 1. Ainsi, un seuil est seuil est nécessaire pour empêcher l'arbre de devenir plus profond que cela. Le seuil de collision est une constante basée sur le concept que plus il y a de collisions, plus l'arbre devient profond. Suite à des simulations effectuées de nombreuses fois, nous avons vérifié qu'une taille d'époque appropriée est approchée plus rapidement que dans tous les autres cas lorsque ce seuil de collision est fixé. Autres cas lorsque ce seuil de collision est fixé à 3. Dans le cas où il est inférieur à 3, même si la taille de l'époque est adéquate, la taille de l'époque devient grande et peut donc être ignorée. Être laissée de côté. Au contraire, lorsqu'il est supérieur à 3, la taille de l'époque devient trop grande et la vitesse de passage à la vitesse supérieure peut être réduite. Trop grande et donc la vitesse d'augmentation de la taille de l'époque devient trop lente. Le seuil de collision est donc supposé être de 3 dans ce document.

II.3.2 L'algorithme Query Tree ALOHA

L'algorithme QT-ALOHA est une autre forme hybride de FS-ALOHA et de l'algorithme QT. L'algorithme FQT met essentiellement en œuvre l'algorithme FS-ALOHA et utilise l'algorithme QT pour le processus d'identification des tags. D'autre part, dans l'algorithme QT-ALOHA l'algorithme QT est une image globale, tandis que le processus d'identification de *tag* réelle progresse avec l'algorithme FS-ALOHA. Ce fonctionnement est le suivant :

Lorsqu'il demande aux tags de transmettre leur ID, le lecteur envoie un préfixe et une taille de trame en même temps. Ensuite, seuls les tags qui sont cohérentes avec leur propre préfixe procèdent à l'algorithme FS-ALOHA avec la taille de trame transmise. Au cours de l'algorithme FS-ALOHA, si une collision se produit, même dans un seul slot, elle est interprétée comme une collision de l'algorithme QT, puis un nouveau préfixe est créé et placé dans la file d'attente. A ce stade, la différence avec l'algorithme QT est de calculer une taille de trame [3]. Est de calculer une taille de trame [3] à transmettre ensuite et de la placer dans la file d'attente. File d'attente.

La figure II.17 est un exemple de l'algorithme QT-ALOHA. On suppose que le nombre des tags à identifier est de 8 et que la taille de la trame initiale commence à 4. Il est supposé que le nombre des tags à identifier est de 8 et que la taille initiale de la trame est de 4. Tour, le lecteur transmet aux tags un préfixe "0" et une taille de trame de 4. Et des collisions se sont produites dans la trame. Ensuite, "00" et "01" sont introduits dans la file d'attente et la taille de la trame est déterminée comme étant 4 par le processus de calcul de la taille de la trame. Ensuite, "1" est retiré de la file d'attente. Lorsque la file d'attente est vide, l'algorithme se termine.



Query	0	1	00	01	10	11	000	001
Frame Size	4	4	4	4	2	2	2	2
Response	collision	collision	collision	identified	identified	no res.	identified	identified

Fig II.17 *– Un exemple de processus d'identification dans l'algorithme QT-ALOHA.*

Conclusion

On vient de détailler dans ce chapitre les classes d'algorithmes d'anticollision et qui sont divisés en deux classes : (i) les algorithmes déterministes et (ii) les probabilistes de la famille TDMA.

Les probabilistes algorithmes sont basés sur ALOHA. Ce dernier réduit la probabilité de collision en transmettant les réponses des tags à un moment aléatoire. Les avantages des protocoles DFSA ou EDFSA sont l'adaptabilité dynamique aux nombres variables de tags qui permet de réduire les récurrences de collision et de slot vide. de réduire les récurrences de collision et de slot vide. Au contraire des protocoles PA, SA et BFSA au contraire.

D'autre part, l'anti-collision des tags basée sur l'arbre est basée sur la division des tags en sous-groupes de façon répétée selon les IDs des tags jusqu'à ce qu'il ne reste qu'un seul tag dans un sous-groupe. De manière répétée, en fonction des ID des tags, jusqu'à ce qu'il ne reste plus qu'un seul tag dans un sous-groupe. L'un des inconvénients de ces protocoles, par rapport à ceux basés sur ALOHA, est que leur efficacité dépend du nombre de bits utilisés pour les tags. Du nombre de bits utilisés pour l'identification des tags. En outre, ils doivent également redémarrer le processus d'identification si un nouveau tag entre dans la zone d'interrogation du lecteur. Dans le protocole QT, les tags n'ont pas besoin d'être

équipés d'une puce supplémentaire. N'ont pas besoin d'être équipées d'une mémoire supplémentaire. Contrairement aux protocoles TS qui nécessitent des ressources de mémoire.

On présentera dans le chapitre suivant les différentes simulations qu'on a mené et de l'implémentation de notre programme.

Chapitre III: Implémentation et simulation

Introduction

Ce chapitre est consacré à la réalisation et simulation des différents algorithmes d'anticollision DFSA présentés dans le chapitre précédant. Ensuite, nous décrivant le déroulement de la simulation par logiciel MATLAB. Finalement nous allons comparer les performances de ces algorithmes en termes de système d'efficacité et la rapidité d'identification des tags.

III.1 Les critères d'évaluation

III.1.1 Système d'efficacité (throughtput)

Le débit est le rapport entre les tags lus avec succès et la taille F de la trame dans une trame, ou le nombre de tags lus avec succès par slot.

$$\text{System d'efficacité} = \frac{\text{Nombre de slots remplir par un tag}}{\text{Nombre de slots total}} \qquad \text{(III.1)}$$

III.1.2 Rapidité d'identification des tags

La rapidité d'identification des tags est le rapport entre le nombre total des tags et le temps nécessaire pour identifier tous les tags. Elle est exprimée en [tags/seconde].

$$\text{rapidité} = \frac{\text{Nombre total des tags}}{\text{temps nécessaire pour identifier tous les tags}} \qquad \text{(III.2)}$$

III.2 EPC G2 Air Interface

Dans cette thèse, nous avons supposé que tous les protocoles anti-collision simulés sont basés sur la norme EPC G2 [40]. Nous avons évalué ces protocoles en fonction du temps nécessaire pour identifier tous les tags. Par conséquent, pour calculer ce temps, nous devons définir la couche physique du lien lecteur-tag et du lien tag-lecteur de l'EPC G2, ainsi que la structure de trame de leurs commandes au lecteur et de leur réponse au tag.

III.2.1 Lien Tag-Lecteur

Lien Tag-Lecteur utilise le codage PIE (pulse interval encoding) pour transmettre les commandes du lecteur. Dans ce codage, la durée des données 0 est égale à Tari, et la durée des données 1 est égale à 2×Tari, comme le montre la figure III.1. À 2×Tari, comme le montre la figure III.1 Tari est l'intervalle de temps de référence pour la liaison avant, ses valeurs est comprise entre 6,25µs et 25µs. Selon la norme EPC G2, la valeur maximale de la largeur d'impulsion (PW) indiquée dans la figure III.2 est de 2×Tari. Largeur d'impulsion (PW) indiquée sur la figure III.1 doit être de :

$$\text{Max } (0.265\text{Tari}, 2\mu s) \qquad \text{(III.3)}$$

Et la valeur minimale de PW est égale à 0,525Tari. Dans ce travail, nous choisissons Tari et PW comme : 6.25µs et 0.5Tari, respectivement.

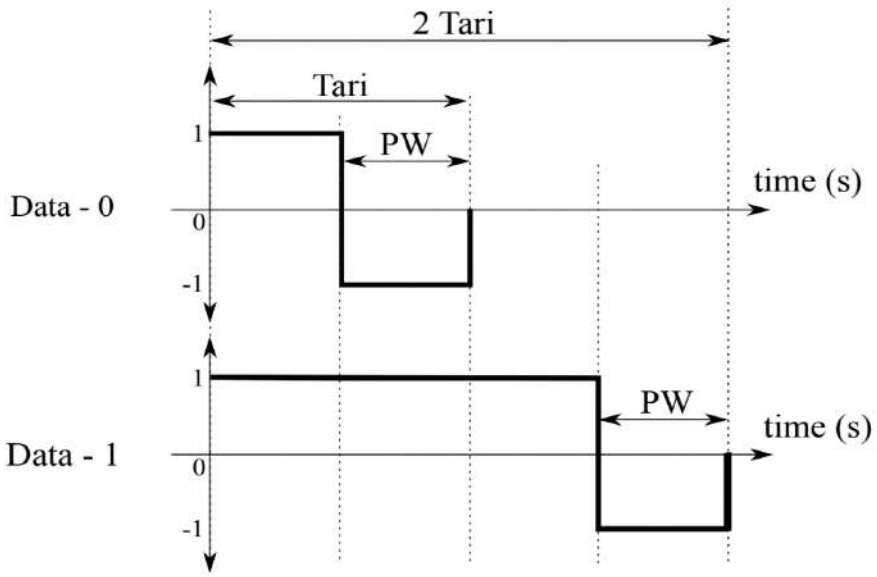


Fig III.1 *– Codage PIE.*

Les commandes codées sont précédées d'un champ de préambule ou de synchronisation de trame. La commande "Query" commence par un préambule, tandis que les autres commencent par une synchronisation de trame. La figure III.2 montre le préambule et la synchronisation de trame de la liaison aller. Le préambule et la synchronisation de trame comprennent un délimiteur de début de longueur fixe, un symbole de données 0 et un symbole de calibrage du lecteur au *tag* (RTcal). Toutefois, le préambule comprend en plus un symbole de calibration *tag* -lecteur (TRcal).

Le *tag* mesure la longueur du Trai envoyé dans le préambule, qui est une information utile pour le décodage PIE. RTcal est la longueur d'un symbole de données 0 plus la longueur d'un symbole de données 1 (RTcal = 0length + 1length). Un tag mesure la longueur de RTcal et l'utilise pour décoder le signal du lecteur. Le lecteur doit spécifier un débit de données de liaison par rétrodiffusion en utilisant le TRcal. La longueur de cette dernière est également mesurée par *tag*. Les valeurs du délimiteur, du RTcal et du TRcal fixées dans ce travail sont respectivement de 12,5µs, 3Tari et 33,3µs.

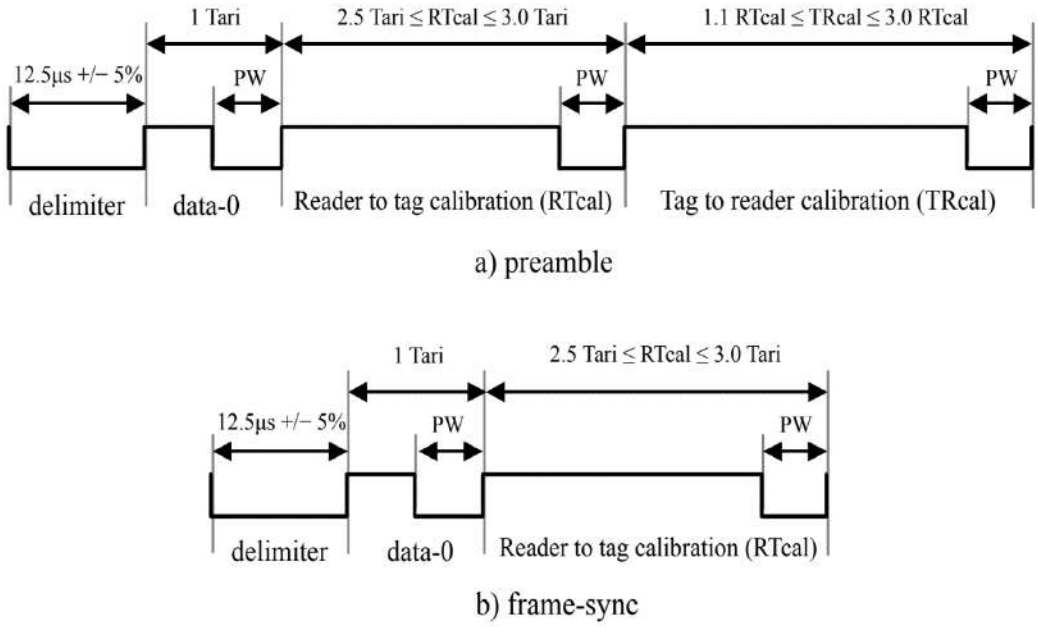


Fig III.2 *– Forward link preamble et frame-sync [40].*

Dans cette sous-section également, nous avons défini la structure de trame des commandes importantes du lecteur utilisées dans la procédure d'identification du tag, comme indiqué dans la sous-section qui sont : Les commandes 'Query', 'Query Rep', 'Query Adjust' et 'ACK'.

La commande 'Query' déclenche et spécifie une ronde d'inventaire. La commande 'Query' comprend les champs suivants :

- Le **code de commande** de "Query" est "1000".

- **DR (divide ratio) :** Il définit le débit de données de la liaison par rétrodiffusion et contient un bit. Le site valeurs possibles de DR sont : 64/3 et 8. Les bits '0' et '1' signifient respectivement DR = 8 et DR = 64/3. Respectivement. Dans ce travail, nous choisissons DR = 64/3 (c'est-à-dire le bit '1').

- **M (cycles par symbole) :** Il a deux bits et définit également le débit de données de la liaison de rétrodiffusion et le type de modulation (FM0 ou Miller). Les chaînes de bits '00', '01', '10' et '11' indiquent respectivement M=1 (FM0), M=2 (Miller), M=4 (Miller) et M=8 (Miller). Dans notre simulation, nous avons supposé que les tags soient codés par un codage FM0 (c'est-à-dire que la chaîne de bits = '00').

- **TRext :** Il choisit si le tag utilise le préambule de liaison par rétrodiffusion avec une tonalité pilote. Ou non. La tonalité pilote est constituée des 12 premiers zéros du préambule. Si la tonalité pilote est utilisée, le bit TRext est égal à '1', sinon il est égal à '0'. Dans ce travail, nous avons utilisé la tonalité pilote dans le préambule (c'est-à-dire TRext = '1').

- **Sel :** Le tag doit mettre en œuvre un flag sélectionné, SL, qu'un lecteur peut affirmer ou désaffirmer en utilisant une commande 'SELECT' [f 21]. Flag (SL ou SL) en utilisant une commande 'SELECT' [f 21]. Si le champ Sel = '00' ou '01' cela signifie que, tous les tags répondent, sinon si Sel = '10', les tags dont le flag SL est désactivé (SLf) répondent et si Sel = '11', les tags dont le flag SL est désactivé (SLf) répondent. Sel = '11' les tags avec leur flag SL assert (SL) répondent. Dans ce travail, nous choisissons Sel='00' (i.e. tous les tags répondent).

- **Session et cible :** Les tags doivent fournir 4 sessions (désignées par S0, S1, S2 et S3). Les tags participent à une et une seule session au cours d'un cycle d'inventaire. Un tag maintiendra un flag inventorié indépendant pour chacune de ses quatre sessions en utilisant une commande 'Select'. Pour chacune de ses quatre sessions. Chaque flag inventorié a deux valeurs, désignées par A et B. Au début de chaque cycle d'inventaire, un lecteur choisit d'inventorier un tag. Chaque cycle d'inventaire, un lecteur choisit d'inventorier les tags A ou B dans l'une des quatre sessions. Sessions. Le champ session permet de choisir une session pour le cycle d'inventaire ('00' : S0, '01' : S1, '10' : S2, '11' : S3). Le champ cible permet de choisir si les tags dont le flag inventorié est A ou B participent à la ronde d'inventaire ('0' : S0). Participent au cycle d'inventaire ('0' : A, '1' : B). Dans ce travail, nous avons supposé dans toutes les simulations. Que, le lecteur sélectionne la session S0 et que la valeur de son flag inventorié est A pour tous les tags (i.e. Session='00' et Target='0').

- **Q :** Ce champ est une valeur entière variant entre 0 et 15. Il représente la taille de la trame F (F = 2Q). La chaîne de bits du champ Q est égale à (Q)2 (par exemple, si Q = 4, alors la chaîne de bits = '0100'). String='0100').

- **CRC-5 :** Il protège une "requête" et est calculé sur le premier bit du code de commande jusqu'au dernier bit Q. Si un tag reçoit une 'Query' avec une erreur CRC-5, il doit traiter la commande comme étant invalide. Le CRC-5 est un contrôle de redondance cyclique de 5 bits avec un polynôme de x 5+x 3+1.

La structure de la trame de la commande "Query" de la norme EPC G2 utilisée dans notre simulation est illustrée à la figure III 3. Est présentée à la figure III.3.

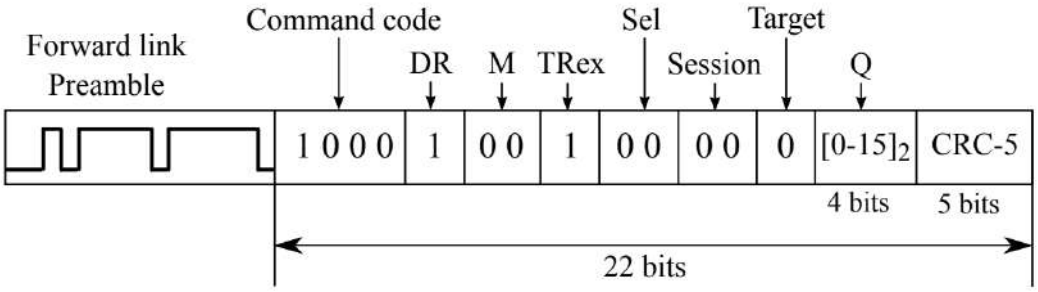


Fig III.3 *– La structure de la trame de la commande "Query".*

La commande 'Query Rep' demande aux tags de décrémenter leur compteur de slots d'une unité. La commande 'Query Rep comprend le champ suivant (voir Figure III.4) :

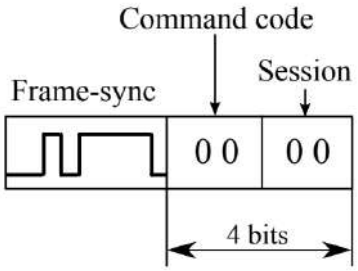


Fig III.4 *– La structure de la commande 'Query Rep'.*

- Le code de commande de "Query Rep" est "00".

- **La session** corrobore le numéro de session pour le tour d'inventaire. Si un tag reçoit un 'Query Rep' dont le numéro de session est différent de celui de la 'Query' qui a initié la trame, il doit ignorer la commande.

La commande 'Query Adjust' permet d'ajuster Q en l'augmentant ou en le diminuant d'une unité ou de ne pas le modifier. La commande "Query Adjust" comprend les champs suivants (voir Figure III.5) :

- Le **code de commande** de 'Query Adjust' est '1001'.
- Le champ de **session** est le même que le précédent.
- **UpDn :** Ce champ détermine si et comment le tag ajuste Q, comme suit :

- '110' : Incrémente Q (c'est-à-dire Q = Q + 1).

- '000' : Pas de changement de Q.

- '110' : Diminue Q (c'est-à-dire Q = Q - 1).

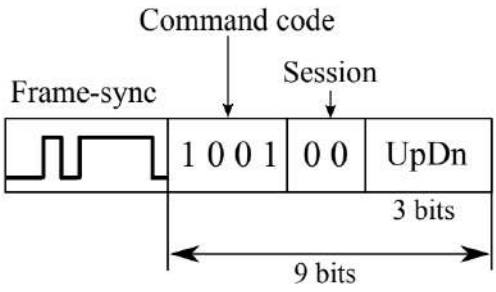


Fig III.5 *– La structure de la commande 'Query Adjust'.*

La commande 'ACK' contient le numéro aléatoire de 16 bits (RN16) reçu du tag. Le lecteur envoie cette commande au tag pour accuser réception de celle-ci. Si le RN16 du tag correspond au RN16 émis dans la commande 'ACK' (un ACK réussi), le tag répond à son EPC. L"ACK' comprend les champs suivants : Le code de commande, égal à '01', et le champ RN16, égal au RN16 décodé reçu (voir la figure III.6).

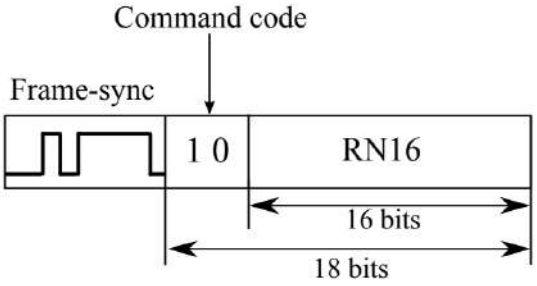


Fig III.6 *– La structure de la commande 'ACK'.*

III.2.2 Backscatter Link

Le codage des données de la liaison rétrodiffusée utilisé dans notre simulation est le codage FM0. Ce codage est le codage FM0 est précédé d'un préambule de liaison rétrodiffusion de 18 bits. Préambule de liaison de rétrodiffusion.

Le débit de données de la liaison de rétrodiffusion BLF est donné en utilisant le champ DR de la commande 'Query' et la durée TRcal du préambule de la liaison aller. Et la durée TRcal du préambule de la liaison aller. Cette dernière étant mesurée par le tag. Par conséquent, le BLF est spécifié comme suit :

$$BLF = \frac{DR}{TRcal} \tag{III.4}$$

Dans notre simulation, nous avons fixé DR à 64/3 et TRcal à 33,3µs. Par conséquent, BLF = 640kbit/s.

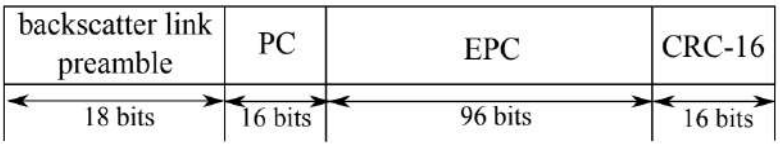


Fig III.7 *– Structure de la trame de la réponse à la commande 'ACK'.*

La réponse du tag des commandes 'Query', 'Query Adjust' ou 'Query Rep' est un RN16. Réponse de la commande 'ACK' utilisée dans notre simulation est composée de : 16 bits de contrôle de protocole (PC) [40], 96 bits de code de produit électronique (EPC) [40], et 16 bits de CRC-16 avec un polynôme de xxx. Polynôme de x 16 + x 12 + x 5 + 1. La figure III.7 montre la structure de trame de la réponse à la commande 'ACK'. De la commande 'ACK'.

III.2.3 Les paramètres du temps

Le temps entre la commande du lecteur et le tag est défini comme T_1, tandis que le temps entre le tag et la commande du lecteur est défini comme T_2. Lecteur est défini comme T_2. T_3 est le temps entre les commandes du lecteur, comme le montre la Figure III.8. Selon l'EPC G2, les valeurs maximales et minimales de ces temps sont résumées comme suit 115Annexe dans le tableau III.1. Où, Tpri est la période d'un symbole FM0, qui est égale à 1/BLF, et FrT est la tolérance du débit de données de la liaison de

rétrodiffusion spécifiée dans [40]. Dans la simulation, nous avons fixé T_1, T_2 et T_3 à $RTcal$, $3T_{pri}$ et $2RTcal$, respectivement.

Tableau III.1 *– Minimum et maximum valeurs de T_1, T_2 et T_3 spécifié dans [40].*

	Minimum	Maximum
T_1	$Max(RTcal, 10T_{pri}) \times (1 - \lvert FrT \rvert) - 2\mu s$	$Max(RTcal, 10T_{pri}) \times (1 - \lvert FrT \rvert) - 2\mu s$
T_2	$3T_{pri}$	$20T_{pri}$
T_3	$2RTcal$	$Max(RTcal, 10T_{pri}) \times (1 - \lvert FrT \rvert) - 2\mu s$

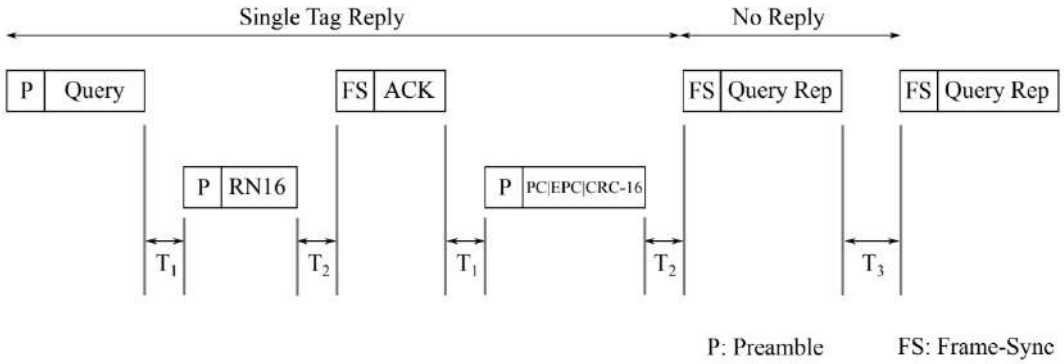


Fig III.8 *– Synchronisation des liaisons de la norme EPC G2.*

Les paramètres de l'interface aérienne de la norme EPC G2 utilisés dans notre simulation sont résumés dans le tableau suivant :

Tableau III.2 *– Air interface of EPC G2 standard parameters used in our simulations.*

	Parameters	Values
Forward link	Coding	PIE
	Tari [µs]	6.25
	PW [µs]	3.125
	Delimiter [µs]	12.5
	RTcal	3 *Tari*
	TRcal [µs]	33.3
	Duration of preamble [µs]	70.8
	Duration of frame-sync [µs]	37.5
Backscatter link	Coding	FM0
	DR	64/3
	Data rate (*BLF*)	640 kbit/s
	Preamble length [bits]	18
	EPC length [bits]	96
Link timing [µs]	T_1	18.75
	T_2	4.6875
	T_3	37.5

III.3 Modèle de simulation

Dans cette thèse on a réalisé sous logiciel Matlab les algorithmes DFSA, BFSA et EPC. Ces algorithmes suivre le standard EPC G2 pour la communication entre le tag et le lecteur. La communication entre tag et lecteur dans ce standard est le suivant : le lecteur envoie la command 'Query', qui contient le paramètre *Q* avec 2^Q est la taille de trame. Ce command initialise une nouvelle trame. 'Qurey Rep' initialise un nouveau slot.

Dans la simulation, on simule les derniers algorithmes cites plusieurs itérations en fonction de nombres de tags. L'algorithme d'anti-collision termine l'exécution lorsqu'il identifie tous les tags pour chaque itération. Le système d'efficacité et la rapidité montre dans les figures ci-dessus sont obtenu par compter Le système d'efficacité et la rapidité durant un itération. Apres les valeurs finales de ces derniers sont données par la valeur moyenne pour toutes les itérations. Dans cette travaille on choisit 10000 itérations pour petite échelle de nombre de tags et 1000 itérations pour grand échelle.

Dans la simulation, on a fixé la taille de trame initiale a la valeur de 8 slots. On suppose que le canal de communication entre tag et lecteur est idéal sans bruit. Car cette travaille et base sur la comparaison des algorithmes dans la couche MAC pas couche physique. Aussi on suppose que la détection de collision est parfaite dans tous les algorithmes.

III.4 Résultats de simulation

Dans ce travail on a évalué les performances des algorithmes d'anti-collision FSA suivant : BFSA, EPC G2 et DFSA basé sur l'estimation de nombre de tag et on a comparé leur performance entre eux en termes de système d'efficacité et rapidité. Les figures suivantes représentent les résultats de simulation.

Figure III.9 représente le système d'efficacité en fonction de nombre des tags au petite échelle (i.e. de 10 à 100 avec un pas de 10) des algorithmes suivants : EPC G2 avec c = 0.2, 0.3 et 0.4 et BFSA pour taille de trame 64 et 256 et DFSA base sur les méthodes d'estimations des tags suivant : Cha, Chebyshev, Kandelwal, Kodialam et Eom. On peut remarquer que, les algorithmes DFSA base sur Eom, Kodialam et Cha ont presque les mêmes résultats et ils donnent les meilleurs résultats par rapport les autres algorithmes. En plus, l'algorithme BFSA avec F=256 donne les mauvaises performances. La valeur maximale du système d'efficacité est fournie par l'algorithme DFSA Kandelwal, et elle égale à 0.4224 pour n = 10. Les valeurs moyennes du système d'efficacité de 10 à 100

tags de l'algorithme DFSA base sur Eom, Kodialam, Cha et Chebyshev sont égale à 0.3566, 0.3574, 0.3542 et 0.3493 successivement.

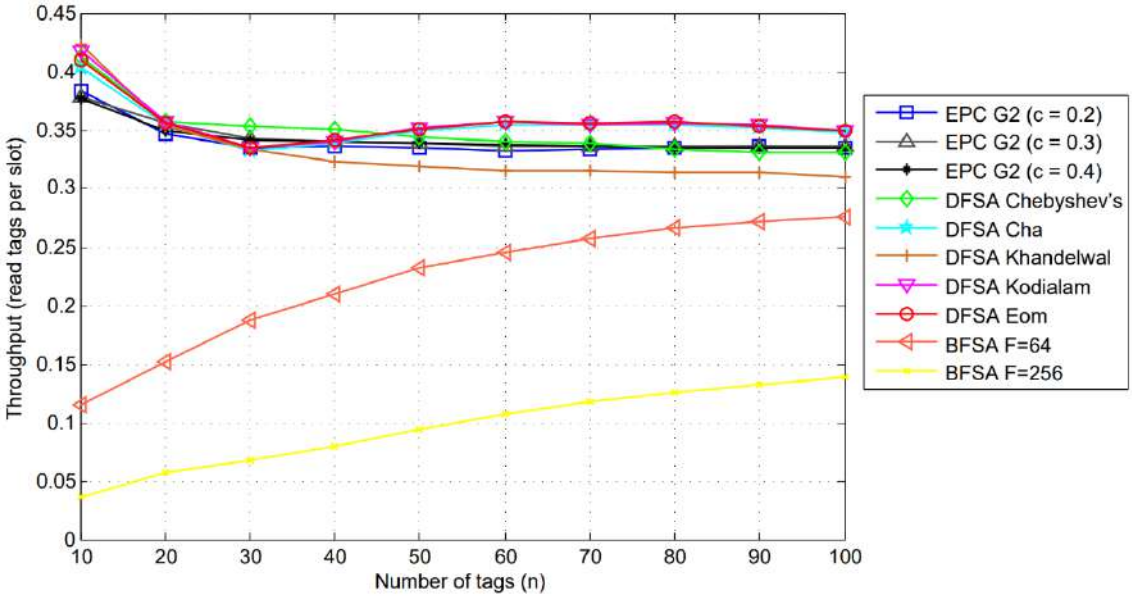


Fig III.9 *– systèmes d'efficacité des algorithmes ALOHA en fonction de nombre des tags en petite échelle.*

Figure III.10 représente un zoom sur l'échelle de système d efficacité de la figure III.9. On remarque que, l'algorithme DFSA Kodialam et Eom sont mieux que les autre pour nombre de tag supérieur ou égale a 50 tags et l'algorithme DFSA Chebyshev et le meilleur pour $30 \leq n \leq 40$.

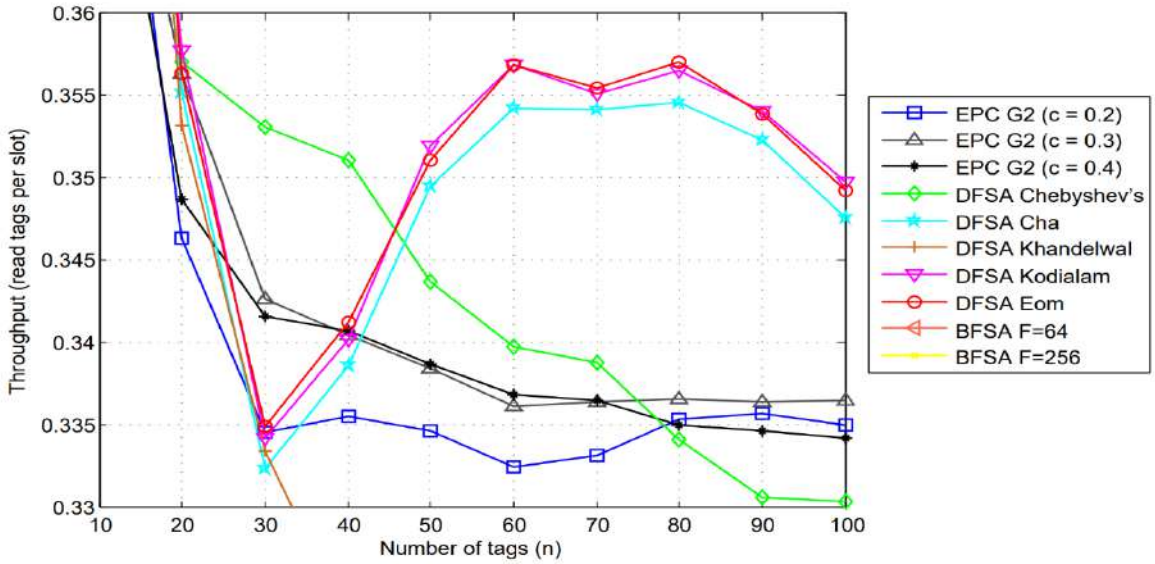


Fig III.10 *– Zoom de système d efficacité des algorithmes ALOHA en fonction de nombre des tags en petite échelle.*

Figure III.11 représente la rapidité en fonction de nombre de tag au petite échelle (i.e. de 10 à 100 avec un pas de 10) des algorithmes suivants : EPC G2 avec c = 0.2, 0.3 et 0.4 et BFSA pour taille de trame 64 et 256 et DFSA (Cha, Chebyshev, Kandelwal, Kodialam et Eom). On peut remarquer que, les algorithmes DFSA base sur Eom, Kodialam et Cha ont presque les mêmes résultats et ils donnent les meilleurs résultats par rapport les autres algorithmes. En plus, les algorithmes BFSA sont les mauvaises par rapport les autres. Les valeurs moyennes du système d'efficacité de 10 à 100 tags de l'algorithme DFSA base sur Eom, Kodialam, Cha et Chebyshev sont égale à 1174, 1174, 1174 et 1162 tags/second successivement.

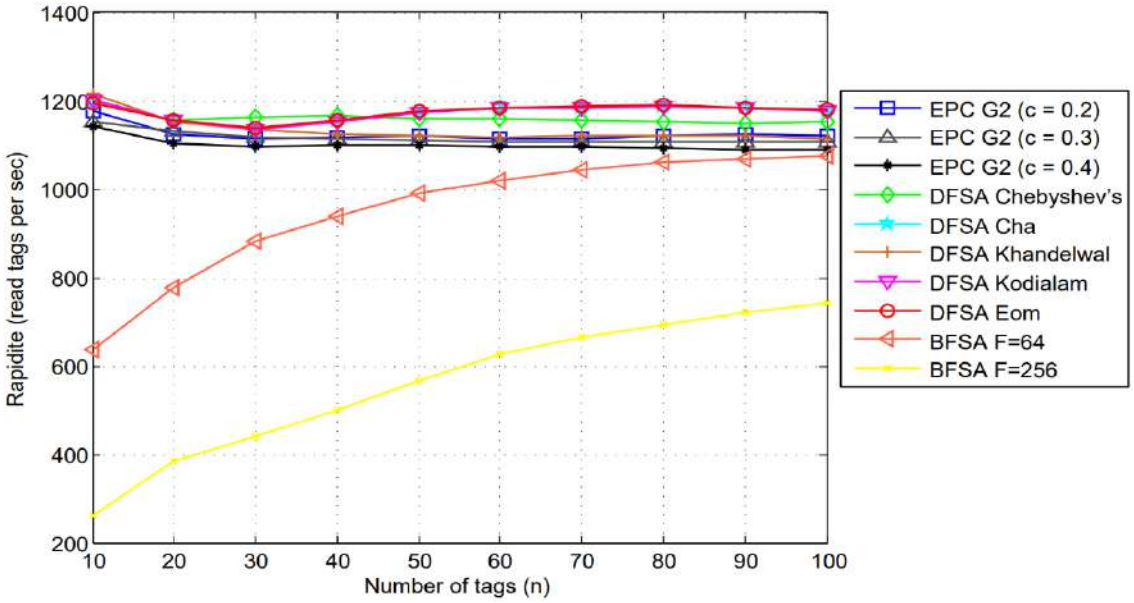


Fig III.11 *– la rapidité des algorithmes ALOHA en fonction de nombre des tags en petite échelle.*

Figure III.12 représente le système d'efficacité en fonction de nombre des tags à la grande échelle (i.e. de 100 à 1000 avec un pas de 100) des algorithmes suivants : EPC G2 avec c = 0.2, 0.3 et 0.4 et BFSA pour taille de trame 64 et 256 et DFSA base sur les méthodes D'estimations des tags suivants : Cha, Chebyshev, Kandelwal, Kodialam et Eom. On peut remarquer que, tous les algorithmes sauf BFSA et DFSA Kandelwal ont presque les mêmes résultats et ils donnent les meilleures performances par rapport les autres algorithmes. Les valeurs moyennes du système d'efficacité de 100 à 1000 tags des algorithmes DFSA base sur Eom, Kodialam, Cha et Chebyshev et EPC G2 (c=0.2, c=0.3, c=0.4) sont égale à 0.3421, 0.3421, 0.3415, 0.3367, 0.3399, 0.3376, 0.3349 successivement. Figure III.14 représente un zoom sur l'échelle de système d efficacité de la figure III.13.

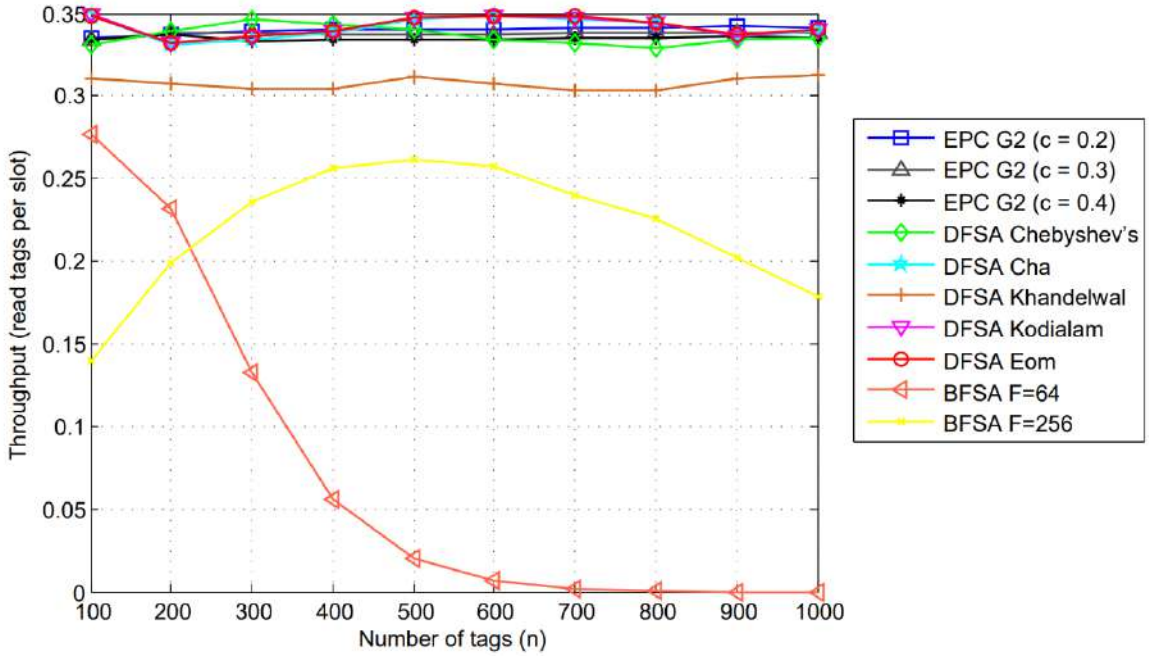


Fig III.12 *– systèmes d'efficacité des algorithmes ALOHA en fonction de nombre des tags en grand échelle.*

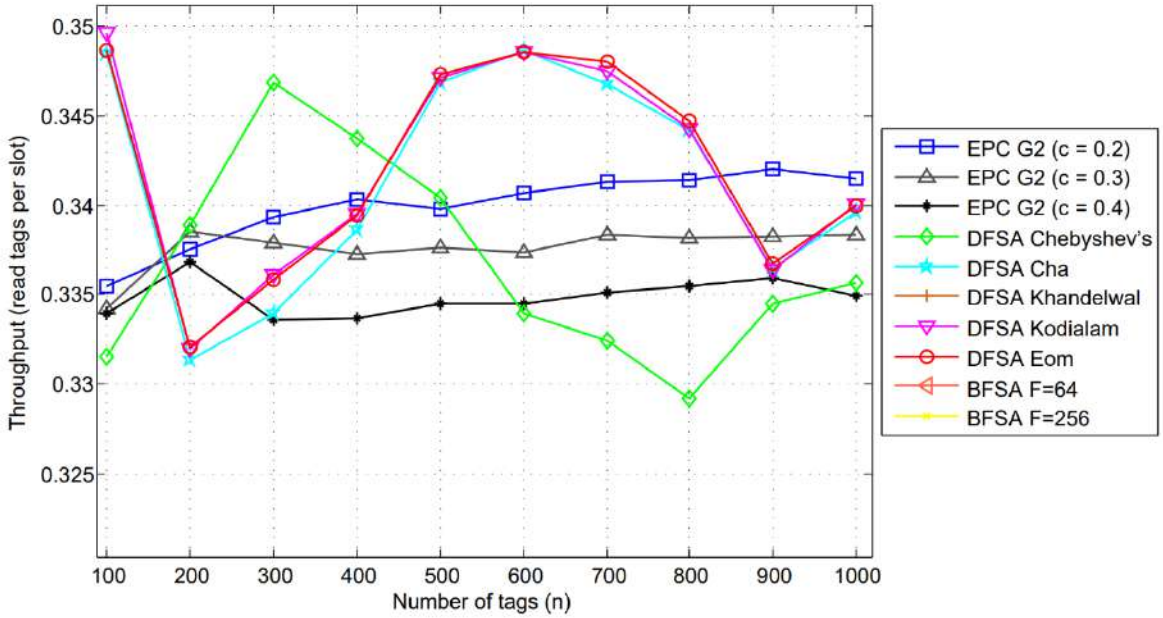


Fig III.13 *– Zoom de système d efficacité des algorithmes ALOHA en fonction de nombre des tags en grand échelle.*

Figure III.14 représente la rapidité en fonction de nombre de tag au grand échelle (i.e. de 100 à 1000 avec un pas de 100) des algorithmes suivants : EPC G2 avec c = 0.2, 0.3 et 0.4 et BFSA pour taille de trame 64 et 256 et DFSA (Cha, Chebyshev, Kandelwal, Kodialam et Eom). On peut observer que, les algorithmes DFSA base sur Eom, Cha,

Kodialam et Chebyshev surpasse les autres algorithmes. En plus, les algorithmes BFSA sont mauvais que les autres. Les valeur moyenne de la rapidité des algorithmes DFSA base sur Eom, Cha, Kodialam et Chebyshev sont :

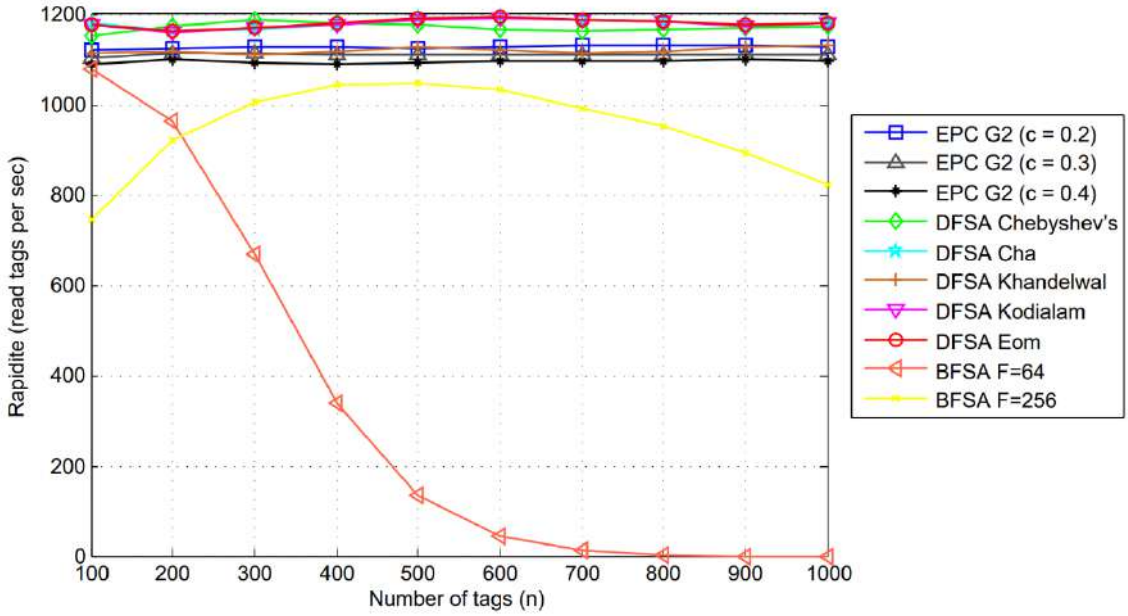


Fig III.14 *– Rapidités des algorithmes ALOHA en fonction de nombre des tags en grand échelle.*

Conclusion

Dans ce dernier chapitre nous avons simulé sur MATLAB et comparé les performances des algorithmes anti-collision DFSA et BFSA en termes de système d'efficacité et rapidité en fonction de nombre de tas en petite et grande échelle. Les résultats de simulation montrent que les algorithmes DFSA base sur Eom, Kodialam et Cha surpasse les autres algorithmes en termes de système d'efficacité et rapidité pour petite échelle. Dans le cas de grand échelle tous les algorithmes sauf BFSA et DFSA Kandelwal donnent les meilleurs résultats que les autres.

Conclusion générale

Le principal problème rencontré avec les systèmes RFID au cours du processus d'identification est la collision des étiquettes, Ceci a encouragé les chercheurs à développer divers protocoles anti-collision de la couche MAC tels que : ALOHA et basé sur les arbres. Dans cette thèse, nous avons décrit différents protocoles anti-collision de la couche MAC tels que : Les protocoles anti-collision basés sur ALOHA et sur les arbres. Dans le protocole basé sur ALOHA, nous avons présenté DFSA (EPC G2 et estimation des tags basée sur), qui sont l'adaptabilité dynamique à nombre variable de tags, Nous avons également présenté quelques protocoles basés sur les arbres comme : Les protocoles TS, QT, BS et K-TAS. Les inconvénients de ces derniers par rapport à ceux basés sur ALOHA est que l'efficacité de leur système dépend du nombre de bits de l'ID des tags et qu'ils doivent relancer le processus d'identification si un nouveau tag entre dans le champ d'interrogation du lecteur. Le processus d'identification si une nouvelle étiquette entre dans le champ d'interrogation du lecteur.

Nous avons comparé les performances des protocoles anti-collision ALOHA et nous avons constaté que les algorithmes DFSA base sur Eom, Kodialam et Cha surpasse les autres algorithmes en termes de système d’efficacité et rapidité pour petite échelle. Dans le cas de grand échelle tous les algorithmes sauf BFSA et DFSA Kandelwal donnent les meilleurs résultats que les autres.

Bibliographie

[1] EPCglobal http://www.epcglobalinc.org/

[2] Klaus Finkenzeller, RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification, Second Edition, John Wiley & Sons, Ltd., England, 2003, ISBN 0-470-84402-7 Site web: http://rfid-handbook.com/english/

[3] Harry Stockmaby, Communication by Means of Reflected Power, Proceedings of the IRE, pp. 1196–1204, October 1948

[4] F.L. Vernon, Jr., Application of the Microwave, Homodyne, IRE Transactions on Antennas and Propagation AP-4, 110 (1952)

[5] D. B. Harris, Radio transmission systems with modulatable passive responder, Brevet.

[6] Historique de la RFID

http://www.aimglobal.org/technologies/rfid/resources/shrouds_of_ time.pdf

[7] Rohatgi Anil. Rfid anti-collision system using the spread spectrum technique. Technical report, Georgia Institute of Technology Propagation Group.

[8] Majid Alotaibi, Adam Postula, and Marius Portmann. Tag anti-collision algorithms in rfid systems—a new trend. WSEAS Transactions on Communications, 8(12) :1216–1232, 2009.

[9] Selwyn Piramuthu. Anticollision algorithm for rfid tags. In Proc. of conference on Mobile and Pervasive computing (CoMPC), 2008.

[10] Shu-qin Geng, Wu-chen Wu, Li-gang Hou, and Wang Zhang. Anti-collision algorithms for multi-tag rfid. Radio Freq Ident Fundamen Appl Bring Res Practice, pages 103–112, 2010.

[11] Dong-Her Shih, Po-Ling Sun, David C Yen, and Shi-Ming Huang. Taxonomy and survey of rfid anti-collision protocols. Computer communications, 29(11) :2150–2166, 2006.

[12] Wang Jianfang. A novel anti-collision backtracking algorithm based on binary-tree search in uhf. 2010.

[13] Leian Liu, Zhenhua Xie, Jingtian Xi, and Shengli Lai. An improved anti-collision algorithm in rfid system. In Mobile Technology, Applications and Systems, 2005 2nd International Conference on, pages 5–pp. IEEE, 2005.

[14] Klaus Finkenzeller and Rachel Waddington. RFID Handbook: Radio-frequency identification fundamentals and applications. John Wiley, 1999.

[15] C. Law, K. Lee, and K. Siu, "Efficient memoryless protocol for tag identification," in Proceedings of the 4th international workshop on Discrete algorithms and methods for mobile computing and communications, 2000, pp. 75–84.

[16] F. Zhou, C. Chen, D. Jin, C. Huang, and H. Min, "Evaluating and optimizing power consumption of anti-collision protocols for applications in RFID systems," in Proceedings of the 2004 international symposium on Low power electronics and design, 2004, pp. 357– 362.

[17] J. H. Choi, D. Lee, and H. Lee, "Query tree-based reservation for efficient RFID tag anti-collision," IEEE Communications Letters, vol. 11, no. 1, pp. 85–87, 2007.

[18] H. Vogt, "Multiple object identification with passive RFID tags," in IEEE International Conference on Systems, Man and Cybernetics, vol. 3, 2002, pp. 1–6.

[19] J. R. Cha and J. H. Kim, "Dynamic framed slotted ALOHA algorithms using fast tag estimation method for RFID system," in CCNC 2006. 2006 3rd IEEE Consumer Communications and Networking Conference, 2006., IEEE, vol. 2, 2006, pp. 768–772.

[20] Z. Huang, R. Xu, C. Chu, Z. Li, Y. Qiu, J. Li, Y. Ma, and G. Wen, "A novel cross layer anti-collision algorithm for slotted aloha-based uhf rfid systems," IEEE Access, vol. 7, pp. 36 207–36 217, 2019. doi: 10.1109/ACCESS.2019.2900739.

[21] D. Lee, K. Kim, and W. Lee, "Q+-algorithm: An enhanced RFID tag collision arbitration algorithm," in International Conference on Ubiquitous Intelligence and Computing, Springer, 2007, pp. 23–32.

[22] M. A. Kalache and L. Fergani, "Performance's comparison of RFID anti-collision algorithms," in International Conference on Multimedia Computing and Systems (ICMCS), 2014, pp. 808–813.

[23] J. R. Cha and J. H. Kim, "Novel anti-collision algorithms for fast object identification in RFID system," in 11th International Conference on Parallel and Distributed Systems (ICPADS'05), IEEE, vol. 2, 2005, pp. 63–67.

[24] S. Lee, S. Joo, and C. Lee, "An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification," in The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2005, pp. 166–172.

[25] F. Schoute, "Dynamic frame length ALOHA," IEEE Transactions on Communications, vol. 31, no. 4, pp. 565–568, 1983.

[26] H. Vogt, “Efficient object identification with passive RFID tags,” in International Conference on Pervasive Computing, Springer, 2002, pp. 98–113.

[27] G. Khandelwal, K. Lee, A. Yener, and S. Serbetli, “ASAP: A MAC protocol for dense and time-constrained RFID systems,” EURASIP journal on Wireless Communications and Networking, vol. 2007, no. 1, p. 018 730, 2007.

[28] M. Kodialam and T. Nandagopal, “Fast and reliable estimation schemes in RFID systems,” in Proceedings of the 12th annual international conference on Mobile computing and networking, 2006, pp. 322–333.

[29] W. T. Chen and G. Lin, “An efficient anti-collision method for tag identification in a RFID system,” IEICE transactions on communications, vol. 89, no. 12, pp. 3386–3392, 2006.

[30] J. Eom and T. Lee, “Accurate tag estimation for dynamic framed-slotted ALOHA in RFID systems,” IEEE Communications Letters, vol. 14, no. 1, pp. 60–62, 2010.

[31] W. Chen, “An accurate tag estimate method for improving the performance of an RFID anticollision algorithm based on dynamic frame length ALOHA,” IEEE Transactions on Automation Science and Engineering, vol. 6, no. 1, pp. 9–15, 2009.

[32] J. Choi and C. Lee, “Maximum a posteriori (MAP)-based tag estimation method for dynamic framed-slotted aloha (DFSA) in RFID systems,” EURASIP Journal on Wireless Communications and Networking, vol. 2012, no. 1, p. 268, 2012.

[33] D. K. Klair, K. Chin, and R. Raad, “A survey and tutorial of RFID anti-collision protocols,” IEEE Communications Surveys Tutorials, vol. 12, no. 3, pp. 400–421, 2010

[34] D. K. Klair, K. Chin, and R. Raad, “An investigation into the energy efficiency of pure and slotted aloha-based REID anti-collision protocols,” in 2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2007, pp. 1–4.

[35] M. Bolic, D. Simplot-Ryl, and I. Stojmenovic, RFID systems: research trends and challenges. John Wiley & Sons, 2010.

[36] Jae-Dong Shin, Sang-Soo Yeo, Tai-Hoon Kim, and Sung Kwon Kim “Hybrid Tag Anti-collision Algorithms in RFID Systems”,2006

[37] P. Pupunwiwat and B. Stantic, “Dynamic framed-slot ALOHA anti-collision using precise tag estimation scheme,” in Proceedings of the Twenty-First Australasian Conference on Database Technologies-Volume 104, 2010, pp. 19–28.

[38] H. Wu and Y. Zeng, “Bayesian tag estimate and optimal frame length for anti-collision aloha RFID system,” IEEE Transactions on Automation Science and Engineering, vol. 7, no. 4, pp. 963–969, 2010.

[38] B. Knerr, M. Holzer, C. Angerer, and M. Rupp, “Slot-wise maximum likelihood estimation of the tag population size in FSA protocols,” IEEE Transactions on Communications, vol. 58, no. 2, pp. 578–585, 2010. 121

[39] W. Chen, “Optimal frame length analysis and an efficient anti-collision algorithm with early adjustment of frame length for RFID systems,” IEEE Transactions on Vehicular Technology, vol. 65, no. 5, pp. 3342–3348, 2016.

[40] G. One, EPC™ radio-frequency identity protocols generation-2 UHF RFID standard specification for RFID air interface protocol for communications at 860 MHz-960 MHz, 2018. [Online]. Available: https://www.gs1.org/sites/default/files/docs/epc/ gs1-epc-gen2v2-uhf-airinterface_i21_r_2018-09-04.pdf.