Thèse de doctorat

Université de Limoges

École Doctorale Sciences et Ingénierie pour l'Information, Mathématiques (ED 521)

Laboratoire Xlim

Thèse pour obtenir le grade de

Docteur de l'Université de Limoges

Electronique des Hautes Fréquences, Photoniques et Systèmes

Présentée et soutenue par

LI Ao

Le 18 décembre 2017

Performances des codes correcteurs d'erreur LDPC appliqués au lien Fronthaul optique haut-débit pour l'architecture C-RAN du réseau 5G : conception et implantation sur FPGA

Thèse dirigée par Christelle Aupetit-Berthelemot, Jean-Pierre Cances et Vahid Meghdadi

JURY :

Président du jury :
M. Antoine Berthet, Professeur, Université Paris Saclay, Centrale Supélec

Rapporteurs
M. Catherine Douillard, Professeur, Institut Mines Telecom Atlantique
M. Iyad Dayoub, Professeur, Université de Valenciennes et du Hainaut-Cambresis

Examinateurs
Mme. Christelle Aupetit-Berthelemot, Professeur, Université de Limoges
M. Jean-Pierre Cances, Professeur, Université de Limoges
M. Vahid Meghdadi, Professeur, Université de Limoges

A mes parents,

à mes grands parents,

à la mémoire de mon oncle

à toute ma grande famille

Ao LI

Remerciements

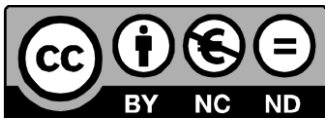
Je remercie très chaleureusement mes directeurs de thèse, Christelle AUPETIT-BERTHELEMOT, Vahid MEGHDADI et Jean-Pierre CANCES, qui ont su diriger mes travaux avec beaucoup de disponibilité, de tact et d'intérêt. Je les remercie pour leur patience et bienveillance tout au long de ces années. Ils n'ont ménagé ni leurs commentaires, toujours judicieux et rigoureux, ni leurs encouragements. Qu'ils trouvent ici l'expression de ma profonde gratitude.

J'exprime aussi mes remerciements aux membres du jury pour avoir accepter d'évaluer mes travaux de thèse.

Je remercie également mes nombreux collègues de bureau qui ont partagé mon quotidien durant cette thèse, au niveau du laboratoire : Jordan, Asma, Karim, Yosra, Imad, Abraham, Fréjus, Thomas, …Je les remercie tous pour leur bonne humeur, leur sympathie et les nombreux échanges qu'on a eu qui ont contribué de quelque manière que ce soit au bon déroulement de ma thèse. Je tiens aussi à remercier mes amis : Yongfei, Xiang, Yujin, Guo qiang, Qifeng... pour le soutien qu'ils ont montré à mon égard dans les moments difficiles.

Mes derniers remerciements, et non des moindres, s'adressent à mes parents qui m'ont toujours accompagné et soutenu dans mes choix.

Droits d’auteurs



Sommaire

Introduction

Les progrès de la connectivité Internet entraînent des changements socio-économiques, avec l'apparition de nombreux nouveaux services haut-débit, comme la télévision à la demande, la santé électronique, les voitures autonomes, la réalité augmentée, les jeux en lignes, les maisons, usines et les villes intelligentes (M2M, IoT, etc…).

Nombreux sont les articles qui donnent les prévisions des évolutions de ce nouveau monde connecté. On se propose ici d'en reporter quelques éléments significatifs qui illustrent le contexte croissant de besoin de débit, mobilité, connectivité et ubiquité auxquels les futurs réseaux de télécommunication devront apporter des réponses :

« En 2020, on comptera 420 millions d'automobiles connectées, correspondant à un taux de croissance annuel moyen de 34% pour un total de 74 millions en 2014. Les systèmes embarqués vont dominer le marché en 2020 » [1].

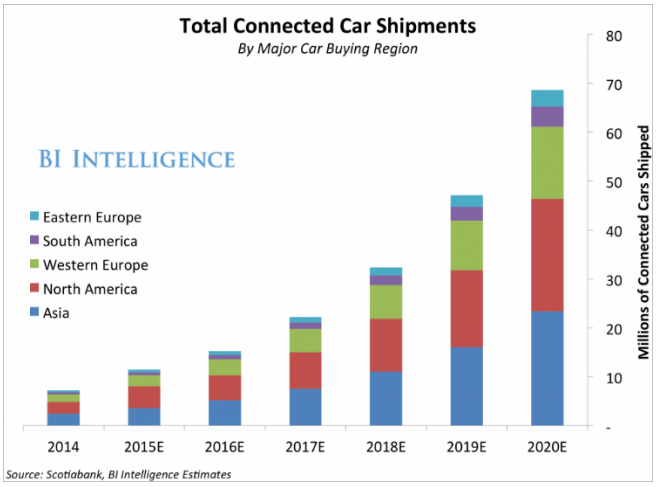


Figure 0-1 : Estimation du nombre de voitures connectées

« L'e-sport, en passe de devenir un véritable phénomène de société, est un marché émergent dont les revenus, encore limités aujourd'hui, sont en forte progression : ils pourraient atteindre 3 milliards d'euros d'ici 2021 représentant ainsi 4% du marché du jeu vidéo et dépasseront les 10,5 milliards d'euros en 2030 avec une croissance annuelle moyenne sur la période de 37,6%. Son audience passera de 240 à 410 millions de personnes en 2021 » : Laurent Michaud, 2017 [2].

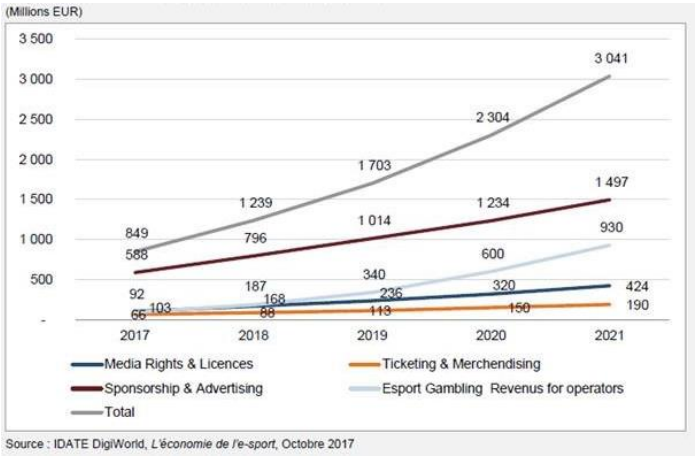


Figure 0-2 : Evolution de l'e-sport et de ses composantes

« D'ici à 2020, 60% de tous les dispositifs technologiques/électroniques seront «intelligents» (« smart ») et 25% seront connectés, ceci représente une différence significative par rapport à la situation actuelle où seulement 1% sont intelligents et 57% sont connectés » [3].

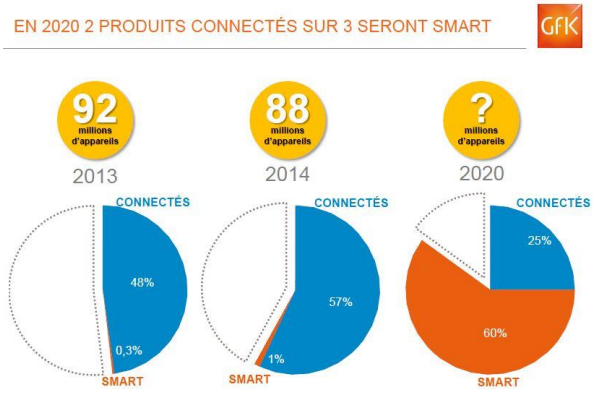


Figure 0-3 : Estimation de l'évolution des produits connectés

Pour résumé, la Figure 0-4 propose une illustration intéressante de l'évolution sociétale des comportements au regard des technologiques mobiles en proposant un exemple pertinent sur la façon dont les téléphones intelligents sont branchés et comment les gens sont devenus habitués à vivre leur vie en les ayant toujours à la main.

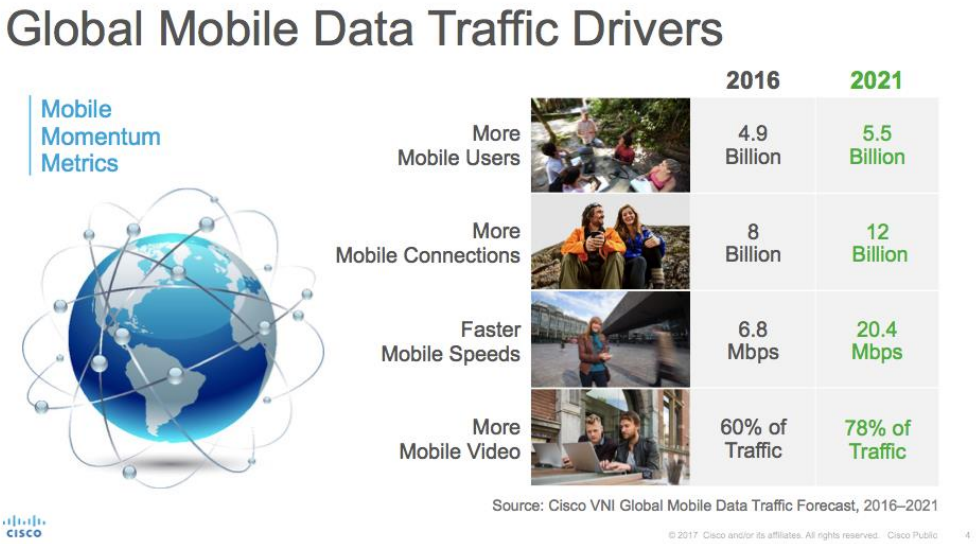


Figure 0-4: Evolution globale des trafics de données mobiles mondiaux.

Pour répondre à l'ensemble des besoins des usagers, les opérateurs et acteurs du développement des réseaux n'ont de cesse d'accroitre le très haut débit fixe et le très haut débit mobile. C'est pourquoi on a pu voir l'émergence d'une nouvelle génération de radiotéléphonie tous les 10 ans, dès 1980 avec la 1G jusqu'à aujourd'hui où la 4G est en plein déploiement et la 5G au cœur de toutes les réflexions. En effet, des activités de recherche, de normalisation et de développement sont en cours sur la prochaine génération mobile (5G). Il est prévu que cette génération traitera des volumes de données plus élevés, beaucoup plus d'appareils connectés avec des exigences de service diverses qui nécessitent le besoin de systèmes évolués avec des capacités étendues. La centralisation et la virtualisation du réseau d'accès radio (RAN), aussi appelées Cloud-Radio Access Network (C-RAN), ont été identifiées comme un facteur clé de la flexibilité, de l'évolutivité et de la gestion orientée vers les services attendues dans ces futurs systèmes. Le C-RAN consiste en la centralisation de l'intelligence des stations de base (BBU : Base Band Unit) en les déplaçant des sites antennaires vers les CO (Central Office), puis la virtualisation des fonctions radio centralisées. De cette manière, seuls les RRH (Remote Radio Head) restent présents sur les sites antennaires facilitant le déploiement et la maintenance tout en garantissant une couverture et un débit maximal. Un nouveau segment optique, appelé le **fronthaul**, et l'exploitation de la D-RoF (Digital-Radio over Fiber) permet ce déport. Ce lien transporte ainsi le signal radio numérique à un débit binaire élevé en utilisant le protocole CPRI (Common Public Radio Interface) [4], qui résulte d'une numérisation à haute résolution des signaux radioélectriques donnant des signaux à haute sensibilité au débit. Pour pallier la solution coûteuse qui consiste à utiliser un réseau de distribution point-à-point entre les RRH et les BBU, le WDM (Wavelength Division Multiplexing) constitue une solution intéressante. Néanmoins, son déploiement dans les réseaux d'accès présente encore quelques difficultés, comme, le manque de solutions de surveillance de l'infrastructure optique et aussi la difficulté de gestion des longueurs d'onde introduite par l'utilisation d'émetteurs-récepteurs optiques colorés. C'est pourquoi, le projet Lampion, dans lequel s'inscrivent ces travaux de thèse, a proposé l'utilisation prometteuse d'émetteurs achromatiques, notamment des RSOA (Reflective Semiconductor Optical Amplifier) en configuration self-seeded. Ce type d'émetteur, bon marché et peu encombrant, serait identique

pour tous les RRH et permettrait une allocation en longueur d'onde intrinsèque. Le Fronthaul fibré permettrait d'atteindre un débit entre 2,5 Gb/s et 10 Gb/s. La performance de cette topologie est toujours à l'étude car comme le CPRI évolue avec les caractéristiques radios utilisées et le système d'antenne, de très forts débits binaires peuvent être rapidement requis en 5G, où des technologies telles que Massive MIMO et mmWaves sont attendues.

Pour assurer la qualité de service et combattre les imperfections du lien fronthaul et les pertes introduites par le désaccord entre MUX et DeMUX, l'ajout d'un codage correcteur d'erreurs FEC est considéré comme nécessaire pour assurer la performance demandée. L'objectif de cette thèse est de déterminer des FEC optimisés afin de les appliquer et de les implémenter dans le contexte du C-RAN.

Le premier chapitre de ce manuscrit présente le contexte de l'étude et l'état de l'art des technologies mobiles radio jusqu'à la description détaillée du C-RAN pour la 5G. Les contraintes liées à cette topologie sont ensuite décrites. Le chapitre se conclut sur la potentialité et la faisabilité de cette technologie en combinaison avec un codage FEC.

Le second chapitre présente le modèle du canal optique considéré dans la liaison par fibre. L'étude s'appuie principalement sur le critère de la probabilité de blocage. Les différents codes qui sont très populaires et utilisés dans les standards de télécommunication sont analysés et comparés. Finalement les codes LDPC sont choisis pour le contexte C-RAN 5G.

Le troisième chapitre détermine les performances des algorithmes à décisions dures pour les codes LDPC. Leurs performances sont comparées et nous montrons finalement que l'algorithme GDBF (Gradient Descent Bit Flipping) représente un excellent compromis complexité-performances.

Pour s'approcher du contexte réel d'implémentation, un convertisseur CAN 2 bits est ajouté dans le chaine de décodage. Les paramètres (valeurs optimisées pour la quantification ainsi que le nombre minimum des itérations) sont déterminés. Finalement, la méthode BWGDBF (Balanced Weighted GDBF) est proposée pour atteindre la meilleure performance.

Les études sur les différentes méthodes d'implémentation sont menées dans le dernier chapitre avec certaines propositions pour réduire la latence et augmenter le débit pour la méthode BWGDBF.

Le mémoire se termine par une conclusion sur les résultats obtenus durant cette thèse et par les perspectives.

Chapitre I

Contexte de l'étude et architecture C-RAN

Chapitre I. Contexte de l'étude et architecture C-RAN

I.1. Introduction

Les technologies radiomobiles ont connu une évolution considérable durant ces dix dernières années pour répondre aux besoins des usagers. En effet, les habitudes ont évolué ultra rapidement et tout un chacun souhaite aujourd'hui communiquer partout, à n'importe quel moment et sur tout type de support. Selon la 11ème édition annuelle de Cisco Visual. Networking Index ™ (VNI) [1], il y aura plus de personnes, parmi la population mondiale, qui utilisera les téléphones mobiles (5,5 milliards) que les comptes bancaires (5,4 milliards), l'eau courante (5,3 milliards) ou les téléphones fixes (2,9 milliards). La forte croissance des connexions des utilisateurs mobiles, des smartphones et de l'Internet des objets (IoT) ainsi que l'amélioration de la vitesse du réseau et la consommation de vidéos mobiles devraient multiplier par sept le trafic de données mobiles au cours des cinq prochaines années. Le trafic de données mobiles progresserait à un taux de croissance annuel composé (TCAC ou CAGR : compound annual growth rate) de 47% entre 2016 et 2021, pour atteindre 49,0 exaoctets par mois d'ici 2021 (Figure I-1).

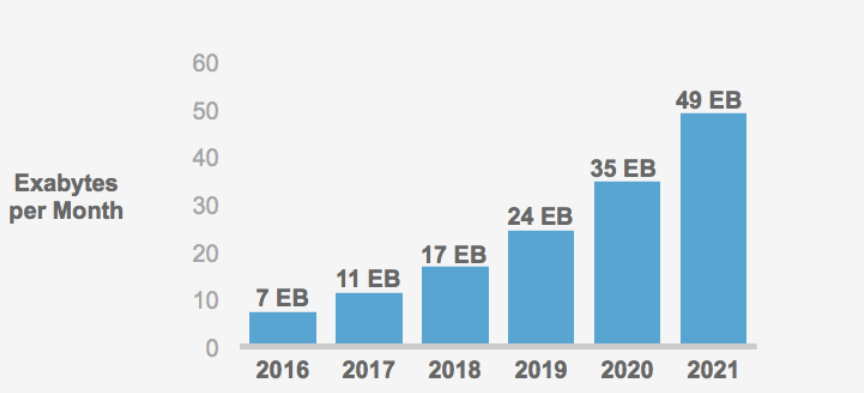


Figure I-1: Prévision Cisco du trafic de données mobiles pour la période 2016-2021 [1].

Pour répondre à cette évolution et à l'attente de tous, quatre générations de technologies radio mobile ont été développées et déployées et la cinquième (5G) est en cours de standardisation et est espérée en déploiement d'ici 2020. Les opérateurs mobiles auront besoin de fonctionnalités innovantes et rapides, de latence réduite et d'approvisionnement dynamique que le réseau 5G devrait pouvoir fournir pour répondre non seulement aux demandes des abonnés, mais aussi aux nouvelles tendances des services dans les marchés mobiles, résidentiels et commerciaux. Cisco prévoit que la 5G représentera 1,5% du trafic total de données mobiles d'ici 2021 et générera 4,7 fois plus de trafic qu'une connexion 4G moyenne et 10,7 fois plus de trafic qu'une connexion 3G moyenne. Ce développement de la 5G se fera en considérant les différents défis qui se sont présentés au fur et à mesure que les opérateurs ont déployé et étendu leurs réseaux 4G :

(i) Un nombre insuffisant de chambres d'équipement au regard d'un nombre d'antennes plus important pour couvrir une même zone (car la fréquence de fonctionnement du LTE est plus importante que pour la 2G ou 3G) engendre un CAPEX important lors du déploiement,

(ii) Dans une société au sein de laquelle est prônée l'économie d'énergie et de la protection de l'environnement, la consommation d'énergie est devenue une préoccupation majeure pour les opérateurs. La consommation de puissance est la plus importante au niveau du réseau d'accès mobile et des sites antennaires. China Mobile estime que 72% de la puissance totale est consommée à ce niveau [6]. Les couts d'exploitation du réseau (OPEX) diminueront inévitablement si cet élément est considéré dans la 5G.

(iii) La gestion des interférences induites par le nombre croissant de cellules et une distance inter-cellules diminuée, en particulier si on parle de « small-cells ».

(iv) Pour terminer sur un point de vue économique, les opérateurs de réseau mobile doivent couvrir les frais de construction, d'exploitation, de maintenance et de mise à niveau alors même que le revenu moyen par utilisateur (ARPU) reste stable ou même diminue au fil du temps, car l'utilisateur lambda est de plus en plus avide de données, mais s'attend à payer de moins en moins.

Pour adresser tous ces challenges de nombreuses études ont été menées au cours de la dernière décennie pour définir de potentielles solutions parmi lesquelles on pourrait citer le Cloud-RAN qui s'apparente bien aux termes de « Centralized, Clean, Cloud, Collaborative » RAN.

Ce chapitre propose une brève description de l'évolution du réseau radio mobile, du réseau fixe et du réseau d'amenée, (appelé communément et dans cette thèse le backhaul) qui permettra de donner une vision globale du contexte dans lequel est né le C-RAN. S'en suivra une description de l'architecture C-RAN et d'un nouveau tronçon qui en découle : le lien fronthaul. Pour terminer, nous présenterons brièvement le projet collaboratif ANR LAMPION dans lequel s'inscrivent complètement ces travaux et les contraintes qui y sont afférentes. Ce chapitre va permettre au lecteur de bien appréhender le contexte et les challenges de cette thèse.

I.2. Evolution des Réseaux d'accès Radio

Une approche généraliste autorise à diviser l'architecture du Réseau Mobile en trois parties que sont :

(i) Le Réseau d'Accès Radio (ou Radio Access Network : RAN),

(ii) Le réseau d'amenée (Backhaul),

(iii) Le réseau Cœur (Core Network : CN),

Le lien backhaul permet le lien entre le CN et le RAN [7][8]. Il peut être en fibres optiques, en cuivre ou supporté par des faisceaux hertziens. Le débit disponible doit être suffisant et adapté au débit des données issues des boucles locales fixes ou mobiles pour ne pas générer d'étranglement. Le RAN comporte les stations de base (BS), équipement qui gère la liaison radio entre les utilisateurs finaux, et le réseau et le contrôleur de stations de Base (BSC). Il permet la connexion radio entre un terminal (téléphone mobile, un ordinateur ou tout autre équipement sans fil) et le reste du réseau comme schématisé en Figure I-2. C'est une partie majeure des réseaux modernes de télécommunication.

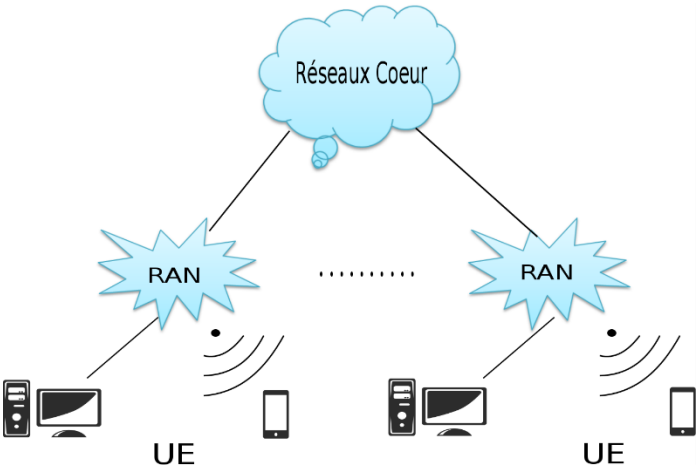


Figure I-2: Schématique simplifiée du Réseau Mobile.

Pour faire face aux besoins des utilisateurs présentés en introduction de ce chapitre, les opérateurs ont dû faire évoluer les architectures RAN en prenant bien en compte les paramètres suivants :

- Permettre des transmissions efficaces et fiables
- Supporter les standards présents et futurs
- Réduire les CAPEX et OPEX
- Réduire les consommations d'énergie

Nous présentons dans la suite de ce paragraphe les différentes générations de RAN.

I.2.1. GERAN

Le GSM Edge Radio Access Network (GERAN) est la partie "accès réseau" du réseau de téléphonie mobile GSM EDGE (Global System for Mobile communications/Enhanced Data GSM Evolution) de deuxième génération (2G) qui est apparu dans les années 90. Le GSM offrait un service vocal avec mobilité, un service de messages courts (SMS Short Message Service) et un transfert de données à bas débit (9 kbit/s). Il diffère de la première génération 1G car il utilise des technologies numériques et des méthodes de transmission par accès multiple par répartition dans le temps (TDMA). Le GSM fonctionne dans les bandes 900 MHz et 1,8 GHz en Europe et dans les bandes 1,9 GHz et 850 MHz aux États-Unis. Avec l'arrivée du GPRS (General Packet Radio Service) dès les années 2000, des transmissions à des débits jusqu'à 40 kbit/s étaient possible puis 473,6 kbit/s avec l'EDGE. Pour résumer, le réseau d'accès 2G supporte 3 technologies radio, GSM pour les appels téléphoniques et le SMS, et GPRS/EDGE pour les services de transmission de données.

Il est composé d'antennes, de stations de base (BS), et de contrôleurs de réseau (BSC). Les BS sont reliées aux BSC par l'interface "Abis" eux-mêmes connectés par l'interface A au réseau cœur qui comporte un système de commutation de réseau (NSS Network Switching System) [9][10], comme illustré sur la Figure I-3. Le lien backhaul est en cuivre. Les BS comportent les modules émetteurs et récepteurs radio (TRX) et permettent les opérations de traitement du signal. Les BSC gèrent la puissance d'émission et agrègent le trafic avant communication avec le NSS.

Un opérateur peut avoir plusieurs GERAN en fonction de la largeur de couverture souhaitée. Un avantage majeur du réseau 2G est sa capacité d'itinérance internationale (Roaming),

permettant aux utilisateurs d'accéder aux mêmes services lorsqu'ils voyagent à l'étranger. Cela donne aux consommateurs une connectivité transparente et identique dans plus de 210 pays. L'itinérance par satellite GSM a également étendu l'accès aux services dans les zones où la couverture terrestre n'est pas disponible.

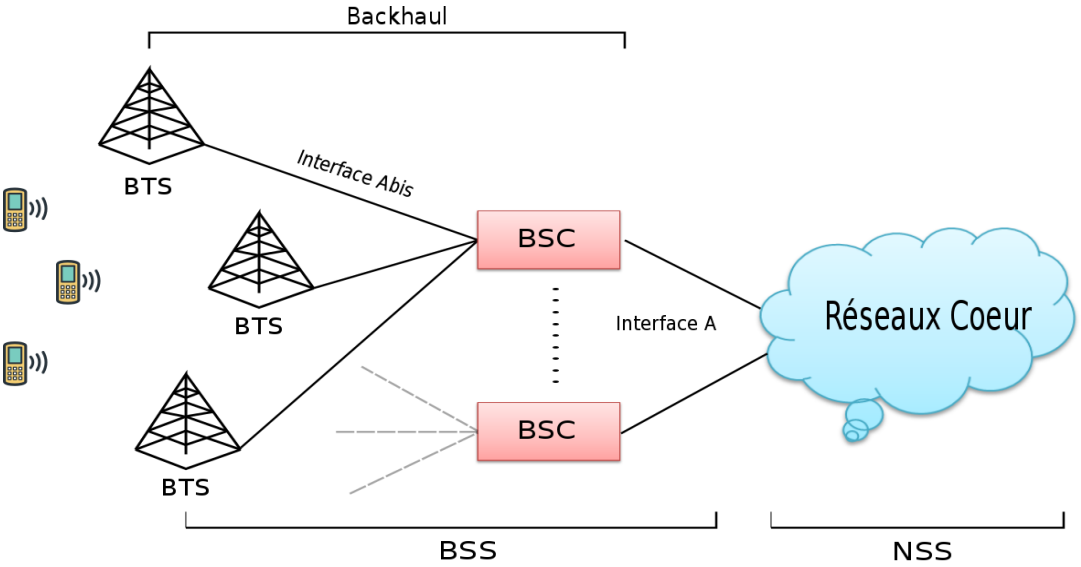


Figure I-3: Représentation simplifiée de l'architecture GSM comportant le GERAN

I.2.2. UTRAN

Le réseau d'accès 3G est appelé UTRAN (UMTS Terrestrial Radio Access Network) [11], domaine spécifique dédié à la radio. L'UMTS (Universal Mobile Telecommunications System) est la technologie retenue comme norme mondiale [12][13], sous la poussée des industriels et opérateurs télécoms (consortium appelé 3GPP : 3rd Generation Partnership Project), pour les systèmes de télécommunications mobile dits de troisième génération (3G), qui ont succédé progressivement au GSM.

Les améliorations attendues par rapport au GSM étaient de :

- Permettre un accès plus rapide à Internet depuis les téléphones portables en augmentant significativement les débits,
- Améliorer les qualités des communications : services vocaux, mais surtout les services de données,
- Répondre au problème croissant de saturation des réseaux GSM, notamment en zone urbaine dense,
- Permettre l'itinérance entre les pays et diminuer le coût des téléphones et des équipements de réseau grâce aux économies d'échelle.

Pour répondre à ces challenges, l'UMTS est basée sur la technologie W-CDMA (Wideband Code Division Multiple Access) qui utilise des bandes, l'une pour le sens montant (uplink), l'autre pour le sens descendant (downlink), de 3,84 Mhz transmises sur une porteuse de 5 Mhz. Le débit maximal supporté par un seul code est de 384 kbit/s. Chaque transmission est associée à un code spécifique connu par la station de base et le terminal mobile ce qui permet de distinguer les appels se produisant dans la même porteuse. La station de base dans cette génération mobile est appelée Node B. Plusieurs stations de base sont contrôlées par le contrôleur de réseau radio (RNC) via l'interface "Iub". Un RNC gère une centaine de Node B. Avec cette technologie, UMTS offre 384 Kbit/s, son architecture est illustrée sur la Figure I-4.

Des débits de 2 Mbit/s peuvent être obtenus en allouant plusieurs codes ou plusieurs intervalles de temps à un utilisateur, mais des raisons techniques limitent le bon fonctionnement de ce système aux bâtiments ou aux petites cellules. Le standard définit donc que le débit doit au minimum atteindre 2 Mbit/s pour le point fixe (indoor), 384 kbit/s pour des vitesses de déplacement faibles et 144 kbit/s pour des mouvements rapides.

L'interface "Iur" est utilisée pour la communication entre les RNC afin de gérer entre autres les transferts et l'interface "Lu" connecte les RNC au réseau cœur (NSS), domaine chargé de la commutation, du routage et du contrôle des services.

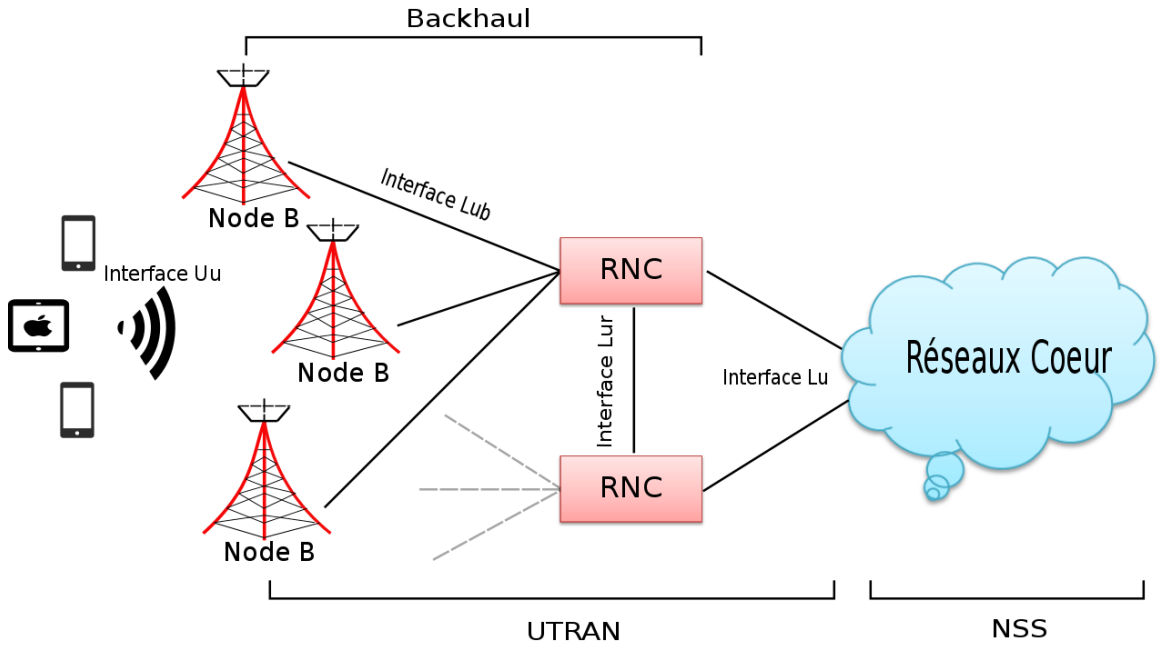


Figure I-4: Représentation simplifiée de l'architecture 3G comportant le UTRAN

Deux évolutions de l'UMTS, que sont le HSPA ou 3G+ (High Speed Packet Access) et HSPA+, décrites dans le 3GPP Version 5 [14] [15][16] ont permis l'augmentation du débit et la réduction de la latence. Le débit montant maximum disponible est alors de 5,75 Mbit/s et le descendant maximum de 14,4 Mbit/s. La modulation utilisée est sélectionnée dynamiquement en fonction de la qualité du canal radio (on a vu apparaitre la 16QAM en liaison descendant en plus de QPSK initialement utilisée dans l'UMTS). L'évolution HSPA+, qui a permis d'introduire la 64QAM en liaison descendante et la 16QAM montante) et l'utilisation de plusieurs porteuses (dual, triple ou quadruple carrier) a permis d'offrir un débit descendant de 42 Mbit/s et montant de 11.5 Mbit/s.

Les performances atteintes (débit, capacité) ont permis à l'UMTS et à ses deux évolutions d'atteindre un succès universel avec un fort déploiement mondial. Grâce à cela, de nouveaux services ont vu le jour (vidéo, TV, jeux, applications mobiles ...) et de plus en plus de terminaux ont été proposés sur le marché (smartphones, tablettes, 3G + dongles ...). Cela a conduit à une augmentation très rapide du nombre de clients mobiles et du trafic dans les réseaux, ce qui a engendré le besoin d'une nouvelle génération de réseaux mobiles pour offrir plus de capacité et de débit avec une latence réduite [11][17].

I.2.3. eUTRAN

La quatrième génération de réseau mobile, LTE (Long Term Evolution), a été suggérée par la 3GPP dès 2004 comme devant être une "évolution long-terme" de l'UMTS pour offrir une plus grande capacité par cellule (zone dans laquelle un émetteur de téléphonie mobile peut entrer en relation avec des terminaux), un meilleur débit, de plus grandes distances, une latence réduite et davantage d'agilité en fréquence. La partie accès radio de ce réseau est l'eUTRAN (Evolved UTRAN), qui assure la connexion entre les terminaux mobiles et le réseau cœur de l'opérateur mobile. Le standard définit aussi que le débit théorique peut atteindre 100 Mbit/s pour le lien descendant et 50 Mbit/s pour le lien montant. L'eUTRAN introduit une nouvelle conception de RAN complètement différent de celui des architectures précédentes (GERAN et UTRAN). Comme illustré sur la Figure I-5, le LTE introduit une nouveauté dans l'architecture de réseau : le contrôleur radio a été supprimé réduisant ainsi la latence de bout en bout. La station de base, appelée "eNodeB", est directement connectée au réseau central appelé "Evolved Packet Core" (EPC) via une interface logique appelée "S1" [18]. Les eNodeB se composent de deux parties : RU (Radio Unit) aussi parfois appelé RRH (Remote Radio Head) et DU (Digital Unit) ou BBU (Base Band Unit) dédié au traitement numérique. Les deux entités gèrent la conversion des données numériques et des données modulées ou non, l'amplification, etc… Les fonctions de contrôle du RNC utilisées dans l'UMTS ont été réparties entre l'eNodeB et les entités du réseau cœur. La coordination entre les eNodeBs adjacents est réalisée via l'interface "X2". Les deux interfaces S1 et X2 constituent le backhaul mobile [19].

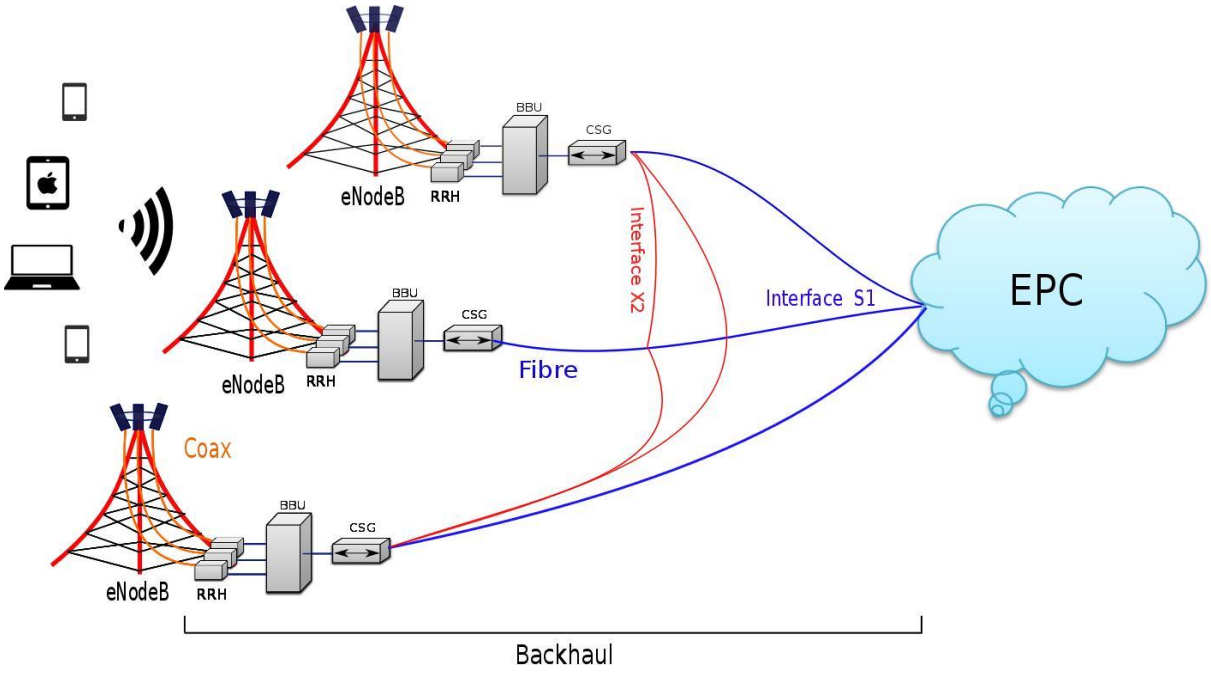


Figure I-5: Représentation simplifiée de l'architecture 4G

L'autre nouveauté est que ce réseau utilise le multiplexage par répartition orthogonale de fréquence (OFDM : Orthogonal Frequency Division Multiplexing). Cela a permis une meilleure utilisation du spectre avec une variété de largeurs de bande possibles (1,4 MHz, 3 MHz, 5 MHz, 10 MHz, 15 MHz et 20 MHz) et une robustesse accrue aux interférences des canaux radio [19].

Une version améliorée du LTE, appelée LTE-A (LTE Avanced) a été proposée dans la version 10 du 3GPP et définit un débit théorique de 1 Gbit/s en liaison descendante et de 500 Mbit/s en liaison montante. Pour atteindre de tels débits, des nouvelles fonctionnalités radio ont été proposées, telles que l'agrégation de porteuses (CA : Carrier Aggregation) permettant la transmission sur plusieurs porteuses radio simultanément et le multipoint coordonné (CoMP Coordinated Multipoint) autorisant la coordination dynamique de la transmission et de la réception sur une variété de stations de base différentes. L'objectif est d'améliorer la qualité globale pour l'utilisateur ainsi que l'utilisation du réseau. On peut observer (Figure I-5) que le backhaul est composé à la fois de support cuivre et fibre.

Les réseaux LTE sont commercialisés depuis 2011. Fin 2012, le nombre d'abonnés au LTE dépassait les 60 millions, tandis que le nombre total d'appareils mobiles s'élevait à 7 milliards [22], dépassant quasi la population mondiale. Le nombre d'abonnés devrait encore augmenter, atteignant 9 milliards en 2021 [23], notamment avec la popularité croissante de la communication Machine to Machine (M2M).

I.2.4. C-RAN

Jusque-là, l'évolution des réseaux d'accès radio portait principalement sur l'augmentation des débits pour les clients mobiles alors que la cinquième génération doit prendre en compte la prolifération des appareils mobiles et portables ainsi que les habitudes de consommation des usagers. Pour ce faire elle doit présenter une plus grande portabilité et interopérabilité des services et permettre à tous ces appareils d'être connectés de manière transparente, avec le réseau fournissant des moyens de calcul très performants et des vidéos ou données multimédias en temps réel.

D'ici 2021, une nouvelle génération de réseaux mobiles sera normalisée [24] – la 5G - pour satisfaire une demande de trafic toujours croissante en offrant une vitesse accrue et des délais plus courts pour permettre un Internet tactile. Il devra également être capable d'offrir une très haute fiabilité, et de connecter un grand nombre d'appareils différents, comme par exemple des voitures et des capteurs à Internet, formant ainsi un Internet of Things (IoT) [26]. Les besoins futurs qui ont été identifiés pour 2020 peuvent être retrouvés dans les livres blancs de Cisco [1] et ngmn [26]. La Figure I-6 présente 8 groupes de cas d'usage de la 5G [26] et le Tableau I-1 le trafic et la diversité des données mobiles [1].

Cette nouvelle évolution du réseau devrait permettre une société entièrement mobile et connectée et de nombreuses transformations socio-économiques, y compris en matière de productivité, de durabilité et de bien-être [26][28].

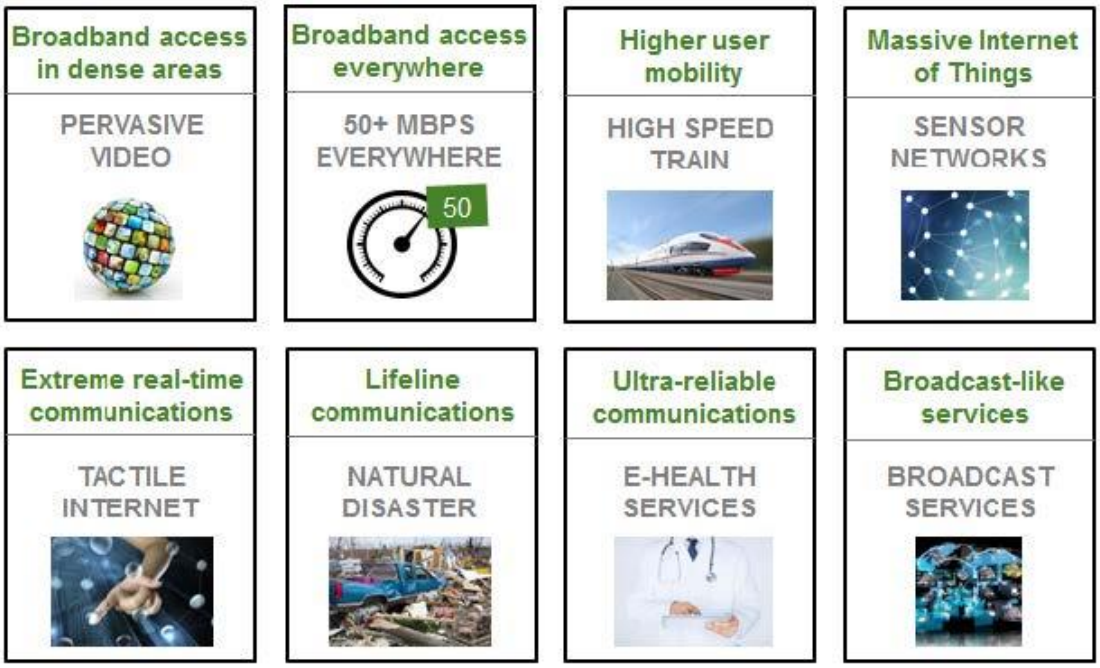


Figure I-6: Cas d'usage de la 5G classés en 8 groupes [26]

Tableau I-1 : Prévision du trafic de données mobiles selon Cisco [1]. Cette prévision inclut uniquement le trafic cellulaire et exclut le trafic déchargé sur le Wi-Fi et les petites cellules des dispositifs dual-modes. La catégorie «other portable devices» comprend les lecteurs, consoles de jeux portables et autres dispositifs avec connectivité cellulaire intégrée. Les "Wearables" sont inclus dans la catégorie "M2M"

	2016	2017	2018	2019	2020	2021	CAGR 2016–2021
By Application Category (TB per Month)							
Web, data, and VoIP	2,153,676	2,938,884	3,779,988	4,674,801	5,538,615	6,434,681	24%
Video	4,375,000	7,225,123	11,415,329	17,564,661	26,067,686	38,148,326	54%
Audio streaming	559,999	843,394	1,193,711	1,620,662	2,103,876	2,674,183	37%
File sharing	151,874	258,617	403,273	592,352	820,954	1,102,867	49%
By Device Type (TB per Month)							
Nonsmartphones	109,505	137,852	169,955	199,173	236,257	269,189	20%
Smartphones	5,887,078	9,328,403	14,076,023	20,710,278	29,484,004	42,017,358	48%
Tablets and PCs	1,085,059	1,514,749	2,040,640	2,681,672	3,457,800	4,439,720	33%
M2M	157,998	284,415	505,292	861,025	1,409,949	2,224,543	70%
Other portable devices	910	599	391	328	432	659	-6%
By Region (TB per Month)							
North America	1,411,021	2,000,301	2,776,564	3,753,177	4,838,494	6,397,092	35%
Western Europe	736,377	1,084,396	1,534,120	2,167,831	3,019,843	4,189,615	42%
Asia Pacific	3,109,117	4,900,007	7,434,743	11,048,030	15,911,056	22,845,908	49%
Latin America	449,944	688,890	1,023,408	1,475,498	2,078,670	2,898,651	45%
Central and Eastern Europe	923,803	1,396,079	2,013,989	2,836,076	3,886,561	5,252,334	42%
Middle East and Africa	610,286	1,196,346	2,009,476	3,171,864	4,853,817	7,367,869	65%
Total (TB per Month)							
Total Mobile Data Traffic	7,240,550	11,266,018	16,792,300	24,452,476	34,588,442	48,951,469	47%

Les architectures de réseaux d'accès précédents auront du mal à répondre aux exigences accrues des utilisateurs. De plus, l'idée d'installer d'autres stations de base n'est plus

acceptable d'un point de vue coût de construction, consommation d'énergie, interférences entre canaux... Aussi, pour répondre aux besoins de capacité, plus de cellules peuvent être déployées ou bien la capacité de chaque cellule doit être augmentée. Les petites cellules peuvent être déployées dans des endroits où l'activité de l'utilisateur final est importante. Dans les déploiements denses (multiplicité de cellules), la gestion des interférences est un défi qui peut être relevé en mettant en œuvre des techniques telles que la coordination inter-cellulaire (eICIC) et multipoint coordonné (CoMP). Pour augmenter la capacité de chaque cellule, il est possible d'utiliser plus d'antennes et la technologie massive-MIMO (massive multi-input multiple output). Comme le spectre est rare dans les bandes actuellement explorées, les bandes de fréquences supérieures, y compris les ondes millimétriques, sont explorées. Enfin, la virtualisation est devenue courante dans le secteur des Technologies de l'Information, elle est maintenant considérée comme applicable aux réseaux mobiles. De plus, l'optimisation de la virtualisation des fonctions réseau (NFV : Network Function Virtualisation) et du réseau (SDN : Software Defined Networking) offre des avantages potentiels en termes de déploiement rentable et flexible des stations centrales et des stations de base [26]. Par conséquent, la nouvelle innovation du réseau mobile d'accès haut débit, pour laquelle les techniques NFV et SDN sont des catalyseurs, est proposée et nommé C-RAN (Cloud-RAN) [27]. Elle est considérée comme une base technologique majeure pour les réseaux 5G. Le concept principal de C-RAN est de regrouper les BBU de plusieurs stations de base dans un pool à BBU centralisé et virtualisé, tout en laissant les RRH et antennes sur les sites cellulaires. Le C-RAN permet un fonctionnement du réseau efficace en termes de consommation d'énergie (jusqu'à 90% d'économie) et d'économies potentielles sur les ressources en bande de base [28]. En outre, il améliore la capacité du réseau en effectuant un équilibrage de charge et un traitement coopératif des signaux provenant de plusieurs stations de base. Par conséquent, les stations de base avec une charge élevée et une densité de déploiement élevée peuvent être déployées dans l'architecture C-RAN. En revanche, pour des zones peu denses (ex les zones rurales) ; le C-RAN n'est pas la solution la plus rentable.

Le débit de données maximum attendu pour la 5G est de 10 Gbit/s, cependant, il est mentionné que 20 Gbit/s pourraient être supportés dans certaines conditions. La 5G prendrait également en charge différents débits de données selon l'environnement. Par exemple, un débit de données de 100 Mbit/s devrait être proposé pour une couverture étendue et pour les hotspots, un débit de données pouvant atteindre 1 Gbit/s serait attendu. Pour ce faire, les opérateurs auront besoin de manière rentable de combiner plusieurs normes (GSM, CDMA, WCDMA, LTE entre autres).

I.2.4.1. Evolution des stations de Base

Comme mentionné au-dessus, le C-RAN est une architecture dans laquelle les BBU sont centralisées dans un pool et peuvent être partagées par plusieurs BS. Dans cette partie, nous allons expliquer l'évolution des stations de base pour aboutir au concept du C-RAN.

Dans l'architecture traditionnelle (Figure I-7), les fonctionnalités radio (RRH) et de traitement en bande de base (BBU) sont intégrées dans les stations de base et le C-RAN tire profit de leur séparation. La BBU effectue un traitement numérique des signaux de bande de base de la couche 1 (L1) avec toutes les fonctions des couches supérieures, et des interfaces avec le réseau backhaul. La RRH s'interface avec les antennes et effectue les fonctions L1 restantes, c'est-à-dire la conversion numérique-analogique/analogique-numérique des signaux en bande de base, la conversion fréquence/puissance et l'amplification de puissance. Le module antennaire et le module radio RRH sont alors reliés par des câbles cuivre co-axiaux [7]. Cette

architecture est très connue pour le déploiement de la 1G et 2G, mais est consommatrice d'énergie.

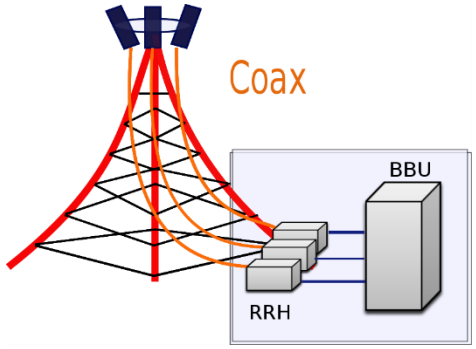


Figure I-7: Architecture traditionnelle

Une première étape d'amélioration a consisté à déplacer les RRH (qui contiennent les amplificateurs de puissance, les duplexeurs, les amplificateurs bas bruit…) au plus près des modules antennaires (distributed base station) [7][29] pour réduire les longueurs de câbles coaxiaux (Figure I-8). Ainsi, on peut gagner 3dB par rapport l'architecture précédente. La BBU est toujours dans le CS (cell site) cabinet. Les RRH et BBU sont connectés en utilisant la fibre optique monomode qui porte un signal radio numérisé (technologie D-RoF : Digital Radio over Fiber) [30]. Pour ces signaux, certaines interfaces de transport ouvertes ont déjà été spécifiées [31] (CPRI : Common Public Radio Interface [4][32], OBSAI : Open Base Station Architecture Initiative [33] [34], ORI : Open Radio equipment Interface [35]), qui définissent également comment les signaux en bande de base sont numérisés (échantillonnés et quantifiés), comment ils sont multiplexés et mappés pour une transmission de données unique mais aussi les protocoles de contrôle / gestion spécifiques utilisés. Le standard CPRI est le plus utilisé sur le marché, les informations utiles dans le standard CPRI sont plus importantes que dans l'OBSAI (93.75% contre 80%). Le CSG (Cell Site Gateway) a pour rôle d'encapsuler les données pour les transmettre dans le réseau backhaul (Figure I-8). Toutes ces interfaces transportent 3 types d'informations : le signal radio numérisé, le management de canal et la synchronisation.

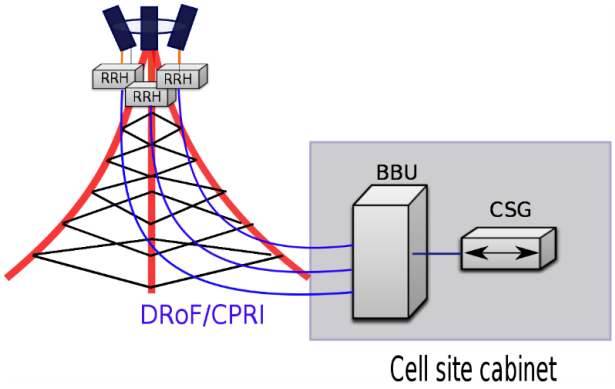


Figure I-8 : C-RAN : stations de Base distribuées

La dernière étape consiste à déplacer les BBU vers le « core central office » (CCO). Ceci est permis du fait que les RRH et BBU sont connectées en fibre optique monomode. Cette migration fait apparaitre un nouveau tronçon fibre plus long appelé Fronthaul. Il est à noter que

l'on peut intégrer dans le réseau des centres offices intermédiaires (ICO : intermediate Central Office). Après s'être focalisé principalement sur les stations de base, nous proposons de synthétiser, dans le paragraphe suivant l'impact de ces évolutions sur l'architecture plus étendue du réseau et les différentes configurations envisageables.

I.2.4.2. Les liens "Backhaul" et "Fronthaul"

La division des stations de base en BBU et RRH et la définition d'interfaces publiques ouvertes (CPRI, OBSAI, ORI) pour le fronthaul présentent de très nombreux avantages. En effet, cela va faciliter la conception et mise en œuvre de fonctionnalités avancées et permettre l'interopérabilité entre les différents fabricants mais aussi ouvrir le marché à de nouveaux arrivants. De facto, des économies substantielles sur les coûts peuvent être réalisées par les fabricants et les opérateurs de réseau.

Il est à noter que, l'innovation la plus importante est quand même qu'à ce stade, les BBU et les RRH ne sont plus considérées comme des parties différentes de la même BS, mais comme des nœuds de réseau distincts. En fait, ils peuvent être placés indépendamment sur l'infrastructure.

Partant du fait que le réseau mobile typique peut être divisé en trois parties : réseau d'accès radio (RAN), réseau backhaul et réseau cœur, nous proposons de reporter ici une vision plus complète du réseau selon les évolutions du RAN vue précédemment.

Pour rappel, le RAN comprend tous les systèmes assurant des fonctions liées à l'accès radio, c'est-à-dire gérant directement la transmission et la réception radio vers/depuis des dispositifs mobiles et uniquement ceux-là. Le réseau backhaul effectue l'agrégation et le transport du trafic entre le RAN et le réseau central et son architecture et sa mise en œuvre peuvent être presque indépendantes des architectures RAN et réseau cœur. Enfin, le réseau cœur est responsable de toutes les autres fonctions non liées à l'accès radio et sert de passerelle vers tous les autres réseaux mobiles et fixes.

Pour résumer, à ce stade et selon les évolutions du RAN précédemment décrites, il existe 4 possibilités de positionnement des BBUs et RRHs telles que listées ci-dessous [29]. Les deux premières n'intègrent pas d'hôtels à BBU et peuvent être considérées comme des RAN avec seulement le backhaul et les deux dernières considèrent les hôtels à BBU amenant dans le réseau une strate supplémentaire : le fronthaul.

- A-BS : tout dans la même Station de Base (Figure I-9)

Les BBUs et RRHs sont inclus dans le même cabinet (CS) et sont reliés par de très courts tronçons de fibre D-RoF (en bleu sur la Figure I-9). Les signaux radio analogiques sont transmis par des câbles coaxiaux, dont les longueurs peuvent atteindre 10 mètres en fonction de la typologie des sites. Le trafic du backhaul généré par chaque BBU dans le site cellulaire est agrégé par la passerelle CSG (SDH/SONET ou Ethernet), et envoyé vers le cœur, à travers la partie restante du RAN.

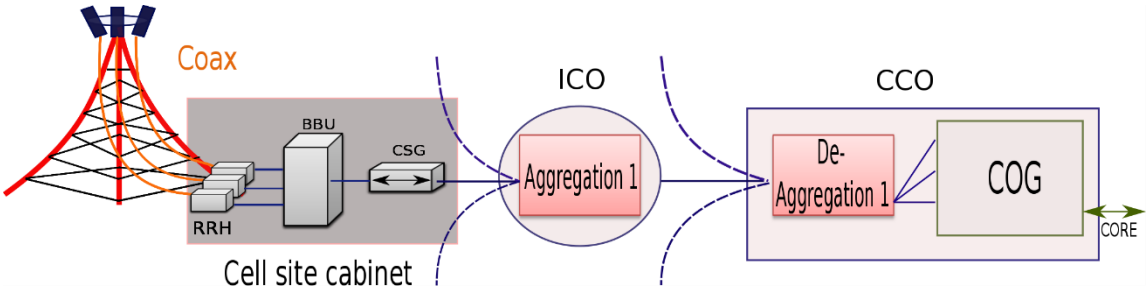


Figure I-9 : BBU et RRH dans la station de Base

- D-BS : stations de base distribuées (Figure I-10)

Les BBUs restent dans le CS, mais les RRH sont installés à côté de module d'antennes, au plus près pour réduire les pertes. Ces dernières deviennent des têtes radio distantes, périphériques autonomes intégrant leurs propres sous-systèmes d'alimentation et de refroidissement. Le CPRI est utilisé via de très courtes fibres dédiées. C'est ce qui va permettre l'apparition des futurs hôtels à BBU.

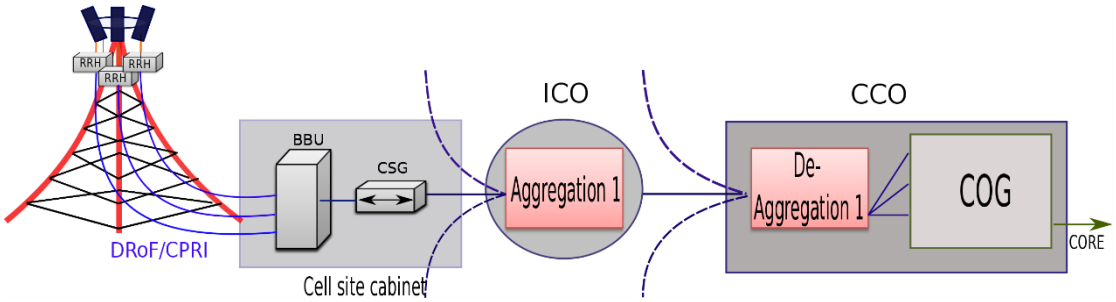


Figure I-10 : Les stations de Base distribuées

- FH-I : les premiers hôtels à BBU

Les BBUs sont concentrées dans le central intermédiaire avec apparition du lien fronthaul (Figure I-11). Le lien backhaul assure la liaison vers le CCO. Un avantage de cette architecture est la réduction de la taille des armoires du site cellulaire, tandis que le CCO peut gérer plus efficacement un grand nombre d'hôtels à BBU (réduction de la consommation d'énergie et économies de coûts).

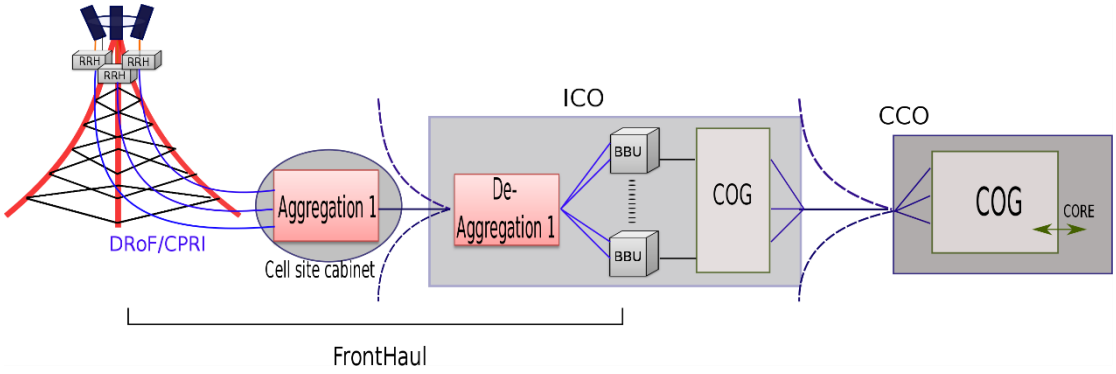


Figure I-11 : Les premiers hôtels à BBU dans les centraux intermédiaires

- FH-C : le lien Fronthaul jusqu'au central

C'est le central qui héberge les BBUs permettant d'effectuer l'agrégation à des niveaux plus élevés et sur un seul site qui collecte les données backhaul ou de fronthaul (Figure I-12).Dans cette configuration, cela permet de mettre en œuvre des CO intermédiaires comme des nœuds complètement passifs, avec une grande réduction d'énergie et de coûts.

En général, cette architecture permet une plus grande concentration des BBUs dans les hôtels individuels et des possibilités de mise en commun potentiellement plus élevés. Ceci entraîne des économies d'énergie et de coûts par la réduction du nombre de sites hôteliers.

Quand on considère le concept de convergence fixe-mobile (CFM), cette dernière solution présente une grande pertinence car la réduction du nombre d'hôtels peut être considérée comme faisant partie de la réduction plus générale du nombre de nœuds actifs.

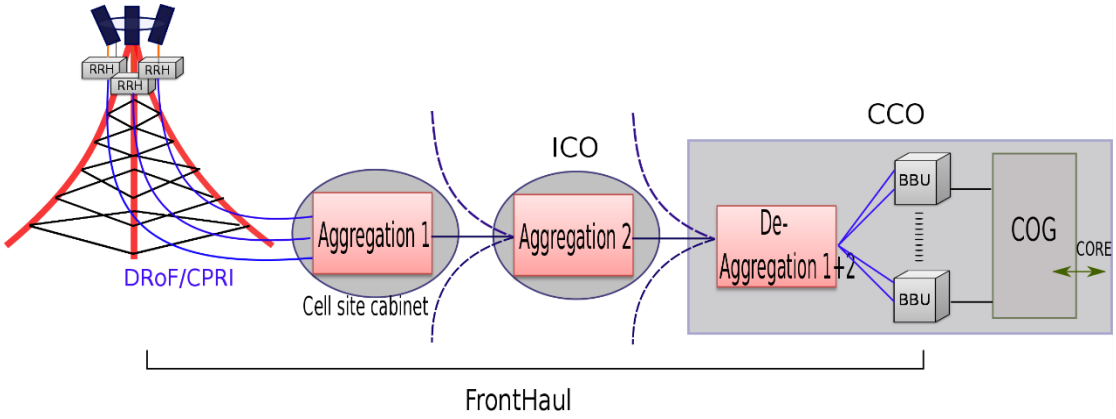


Figure I-12 : Architecture avec BBU au CCO

Pour cette dernière configuration, le lien fronthaul est beaucoup plus long et des contraintes supplémentaires apparaissent, telles que la gestion de la latence qui joue un rôle critique. C'est en partie cela qui peut rendre le C-RAN comme non optimal pour les zones peu denses telles que les zones rurales ou les centraux sont éloignés des sites cellulaires.

Si on considère le management des BBUs dans le backhaul, deux approches différentes sont possibles :

- Les BBUs de différentes stations de base sont situées dans le même central office et sont connectées aux RRH à l'aide d'une interface standard telle que le CPRI (Common Public Radio Interface). Généralement, il existe un site BBU par cellule. La communication des BBUs se fait au sein de l'hôtel par l'interface X2 standardisée et via l'interface S1 avec les MASG *(*Mobile Access Site Gateway) (Figure I-13)

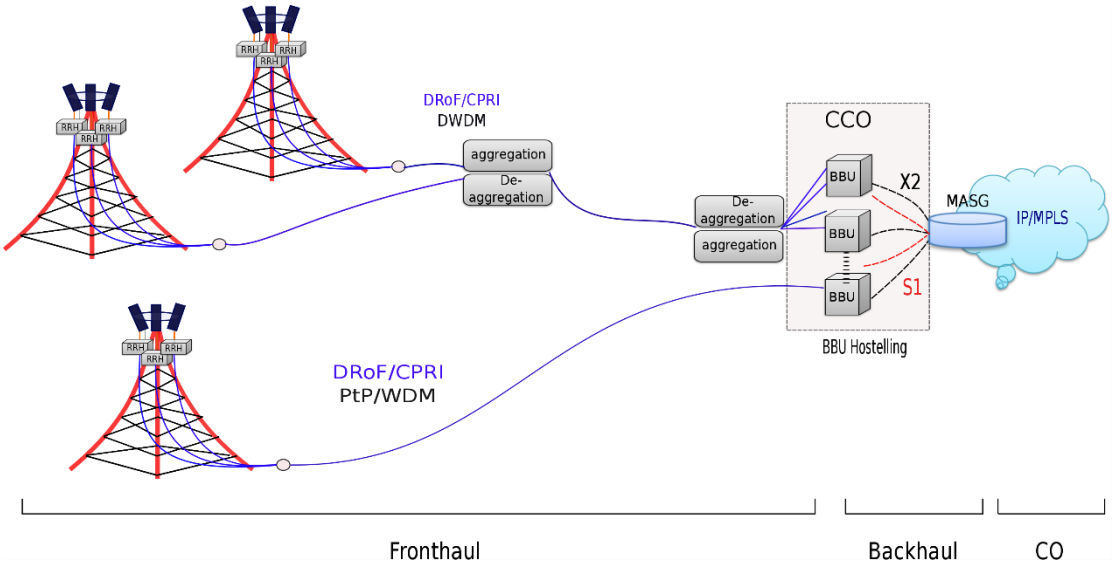


Figure I-13 : Architecture BBU Hostelling

- BBU pooling appelé aussi « BBU virtualisation » (Figure I-14) permet un management virtuel d'un grand nombre de RRHs situés sur différents sites d'antennes (software) et une réduction du nombre d'interfaces S1 et X2. Ce modèle est en cours d'étude et d'évaluation par les opérateurs.

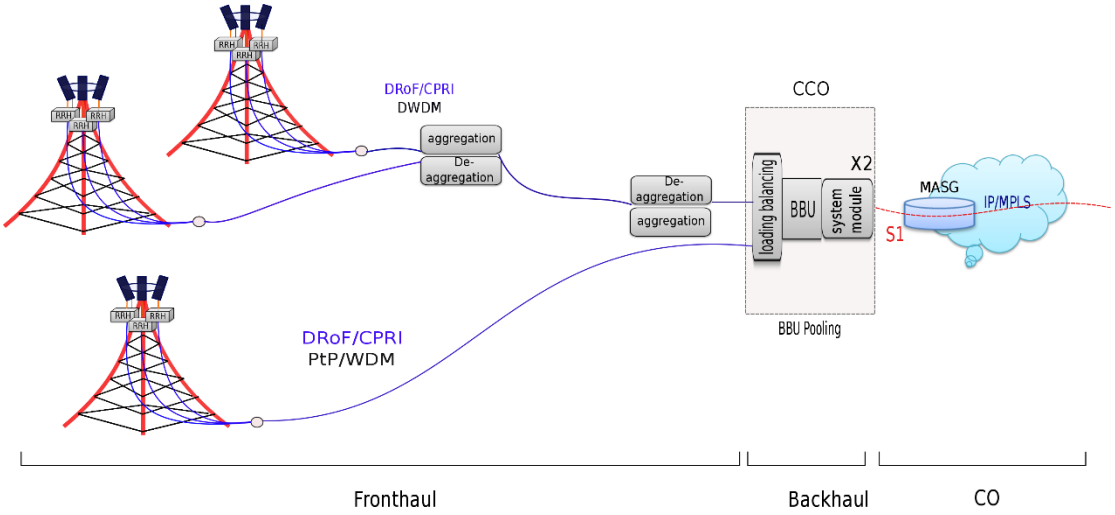


Figure I-14 : Architecture BBU Pooling

I.2.4.3. Les contraintes du lien Fronthaul

Pour mettre en œuvre un segment de fronthaul, il est obligatoire de respecter certaines exigences interdépendantes, qu'elles soient au niveau des aspects techniques, de la réglementation et des contraintes de gestion des opérations.

I.2.4.3.1. En termes de débit

Le débit de l'interface CPRI est constant et s'étale de 614,4 Mbit/s (CPRI 1) à 24,33 Gbit/s selon la technologie d'accès radio mobile (RAT), la bande passante de la porteuse radio et l'option de MIMO mise en œuvre.

Le débit de données CPRI est calculé en utilisant la formule suivante :

$$Bitrate = M \times Sr \times N \times 2(IQ) \times C_w \times C \quad \text{(I-1)}$$

Avec :

- M le nombre d'antennes par secteur (implémentation MIMO)
- S_r est le taux d'échantillonnage utilisé pour la numérisation du signal (échantillons/s/porteuse)
- N est le nombre de bit/échantillon (bits/sample)
- 2(IQ) est un facteur multiplicatif correspondant aux données I/Q (I en phase et Q en quadrature)
- C_w représente le facteur de contrôle du mot CPRI
- C'est un facteur de codage, il peut être égal à 10/8 pour le codage 8B/10B ou 66/64 pour le 64B/66B.

Considérant cela, le Tableau I-1 résume les débits CPRI obtenus avec l'expression (1).

Tableau I- 1 : Exemple de débits CPRI

RAT	GSM 1T1R	GSM 1T2R	UMTS 1T1R	UMTS 1T2R	LTE 10 MHz 2x2	LTE 10 MHz 2x2	LTE 20 MHz 2x2	LTE 20 MHz 4x4
CPRI bit-rate	12.304 Mbit/s	24.608 Mbit/s	307.2 Mbit/s	614.4 Mbit/s	1228.8 Mbit/s	2457.6 Mbit/s	2457.6 Mbit/s	4915.2 Mbit/s

I.2.4.3.2. En termes de latence

La condition la plus stricte est imposée par la future génération de technologie radio mobile LTE–Advanced [19][20] qui préconise un temps de propagation aller-retour maximum de l'ordre de 400 µs à 500 µs afin de pouvoir implémenter le CoMP (Coordonate MultiPoint). Cela inclut les retards induits par le transport dans les fibres (à hauteur typiquement de 5µs/km) et les équipements présents sur le lien de fronthaul. Il est démontré dans [36], pour être en accord avec ces contraintes de latence du C-RAN que la longueur du lien fronthaul doit être entre 15 et 20 km, en considérant qu'il n'y a pas d'équipement actif dans le lien et selon l'implémentation RAN. Une étude est également reportée sur les implantations des sites d'Orange dans les zones de forte ou moyenne densité. Il en ressort que leur réseau est en adéquation avec les exigences du C-RAN en termes de Latence :

- Distance entre les antennes et le central :
    - 75% des liaisons ont une longueur <4km
    - 95% des liaisons ont une longueur <8km
    - 99% des liaisons ont une longueur <20km
- Nombre d'antennes potentiellement connectées au central office :
    - 20% des centraux seraient connectés à plus de 8 sites antennaires
    - 45% des centraux offices seraient connectés à plus de 4 sites antennaires
    - 70% des centraux seraient connectés à plus d'un site antennaire
    - 30% des centraux seraient connectés à un seul site antennaire

I.2.4.3.3. En termes de synchronisation et de gigue

Les spécifications de la technologie LTE exigent une précision de la fréquence radio sur l'interface air de ± 50 ppb (parties par milliard). Cette fréquence est transportée par le lien CPRI et l'impact du lien de fronthaul est limité à une déviation de fréquence max de ± 2 ppb. Par ailleurs, des contraintes supplémentaires sur la synchronisation en temps et en phase seront définies prochainement par les organismes de normalisation (5GPP).

La gigue (jitter) en réception est également spécifiée pour une interface CPRI en se basant sur les interfaces 10GEth (standard IEEE802.3. av) : La gigue totale (TJ) maximum est limitée à 0.65UI (Unit Interval) ; la gigue déterministe (DJ) à 0.35UI et la combinaison du DJ et de la gigue aléatoire (RJ) est limitée à 0.55UI.

I.2.4.3.4. En termes de configuration des sites mobiles

Les configurations typiques observées dans les zones urbaines font état de 3 secteurs pour chaque technologie mobile du type macro-cell (un secteur couvrant un angle de 120°). Totalisant chaque technologie radio déployée en France (2G, 3G, 4G x fréquences radio), cela implique jusqu'à 15 RRH par site macro-cell, et sans compter l'arrivée des small-cells (Micro, Pico & Femto-Cells, et points d'accès WiFi) qui viendront en support aux macro-cells pour améliorer la couverture d'une zone[36]. On parle alors typiquement d'un déploiement à long terme de 10 small-cells par site antenne macro. On aurait alors sur un site antenne : 15 RRH macro + 10 smalls cells soit 25 liens de fronthaul à créer.

Le lien fronthaul nécessite une connectivité point à point CPRI, il y aura donc autant de liens de fronthaul que d'antenne déportées. Si ces liens sont réalisés en fibre optique, il est possible d'en diminuer le nombre en utilisant un multiplexage en longueur d'onde qui permet de maintenir la connectivité point à point.

I.2.4.4. Les différents types de lien Fronthaul et techniques associées

Comme nous l'avons expliqué dans le paragraphe 1.2.4.2, la centralisation des BBUs a engendré la création d'un nouveau segment, le fronthaul, dans le réseau et a impliqué l'apparition de nouveaux points de démarcation (DP) entre opérateurs fixe et mobile (Figure I-15). De plus, pour réduire les coûts d'intervention et faciliter le management des sites antennaires, il est recommandé que l'opérateur mobile puisse communiquer des informations de supervision vers l'opérateur fixe et vice versa (tels que puissance optique émise/reçue, coupure de fibre, etc)... la conséquence est que pour permettre cela à tous les opérateurs présents sur les sites antennaires, il est impératif d'offrir une offre de fibre optique supervisée pour ce lien fronthaul [38].

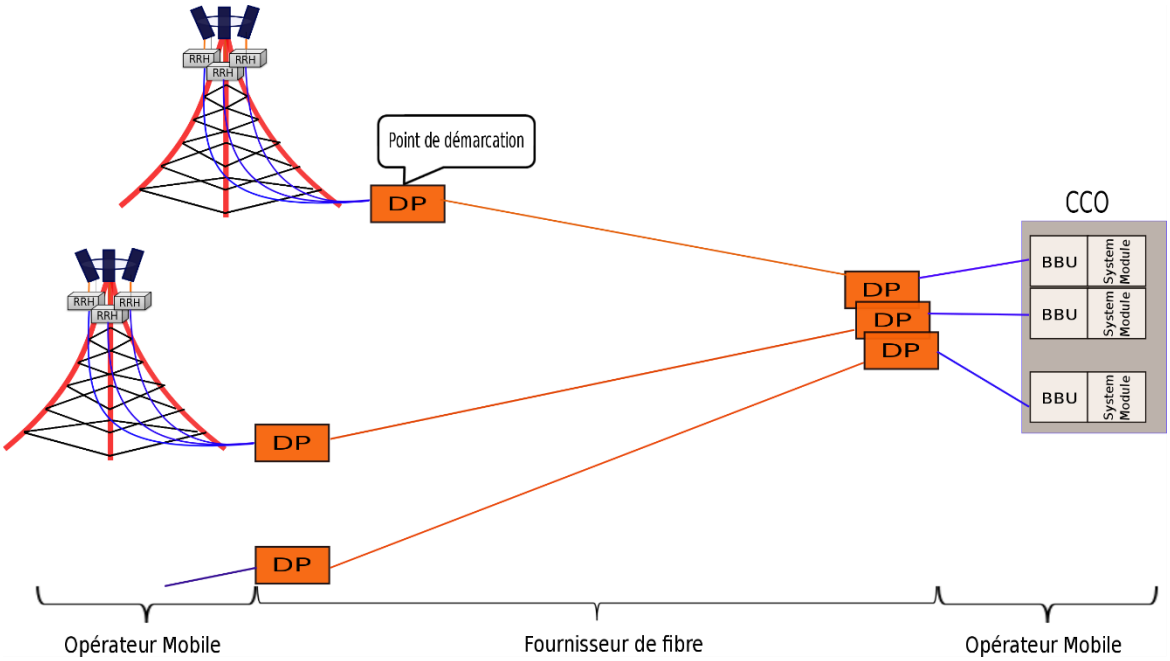


Figure I-15 : Architecture de fronthaul avec point de démarcation entre opérateurs fixe/mobile

Si on considère le nombre de RRHs (micro-site : 5 RRHs, macro-site : 15 RRHs) à raccorder à un seul central, il apparait indispensable de maintenir un nombre minimum de fibres à déployer pour le fronthaul en utilisant des techniques de multiplexage temporel ou en longueur d'onde. Une alternative possible est d'utiliser des technologies micro-ondes sans fil (micro-wave technologies).

I.2.4.4.1. Déport sans fil micro-wave.

Cette solution peut être mise en œuvre seulement pour réaliser un déport de très courte distance (quelques centaines de mètres) quand la pose ou l'utilisation de fibre optique est impossible. Cette solution est facilement déployable permettant des réductions de coût d'OPEX et de CAPEX. Ce type de lien Figure I-16 est constitué de Wireless Fronthaul Modules (WFM) qui supporte le protocole CPRI, connectés d'un côté de la liaison aux RRHs et de l'autre aux BBUs à l'aide de tronçons de fibres optiques.

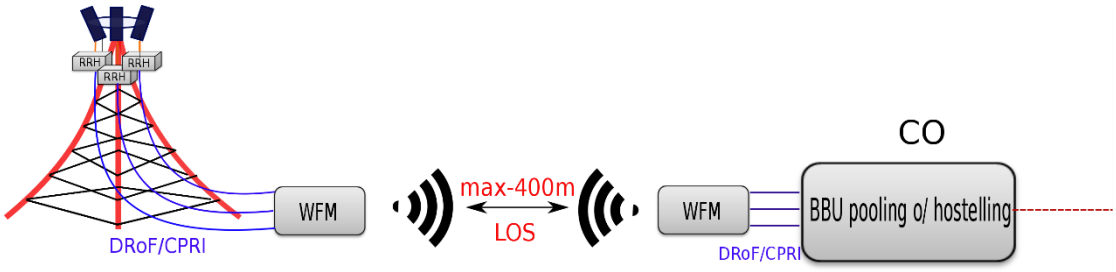


Figure I-16 : Architecture de fronthaul wireless

Il a été montré dans [39] que ces technologies micro-wave permettent des débits de 7,5 Gbit/s (débit de 3 liens CPRI3) avec une porteuse à 5,8 GHz et une bande passante de 70 MHz. Elles sont toujours à l'étude car c'est un choix qui pourrait convenir pour le fronthaul.

I.2.4.4.2. Fronthaul fibré : Multiplexage TDM sur WDM

L'OTN (Optical Transport Network) est une solution basée sur le standard ITU-T G.709 [40]-[45] qui consiste à encapsuler un signal numérique (par exemple de type CPRI) dans des containers spécifiques et à envoyer le tout via le réseau optique. Il présente un certain nombre d'avantages, en prenant en compte la nécessité d'ajouts d'éléments actifs au central et sur les sites antennaires :

- Le système est standardisé et permet la surveillance, la gestion et la maintenance du lien optique
- L'utilisation de codes correcteurs d'erreurs (FEC : Forward Error Coding) pour améliorer les performances
- On peut identifier très facilement les points de démarcation entre l'opérateur mobile et le fournisseur de services

Il permet de réaliser un multiplexage temporel de plusieurs trafics de données, typiquement 4×2457.6Mbit/s CPRI agrégés sur un débit en ligne de 10Gbit/s.

Pour la gestion du nombre de fibre entre les RRHs et le central, il est possible d'y associer un multiplexage en longueur d'onde.

Ainsi, pour cette solution, plusieurs flux fronthaul sont mappés de bout en bout en longueurs d'onde, par multiplexage temporel (TDM), sur WDM en utilisant l'OTN. Pour ce faire, des MUX/DEMUX OTN ("wrippers") sont positionnés à la fois au niveau des sites antennaires et du central pour permettre de mapper des signaux CPRI dans des conteneurs OTN (couche basse) qui sont multiplexés (couche haute) et transmis sur des longueurs d'onde différentes.

I.2.4.4.3. Multiplexage passif basé en longueur d'onde (WDM-PON)

Les flux Fronthaul sont transmis sur des canaux de longueur d'onde séparés, au moyen d'émetteurs-récepteurs fonctionnant selon la technologie WDM (Wavelength Division Multiplexing). Ceci permet de multiplexer plusieurs longueurs d'onde sur quelques fibres, via un multiplexeur WDM passif placé dans chaque site cellulaire. Sur le site de central, les longueurs d'onde entrantes sont séparées par des démultiplexeurs passifs et envoyées à des ports CPRI séparés de BBU, équipés d'émetteurs-récepteurs WDM.

Le principal intérêt de cette solution est que l'infrastructure reste passive et basée sur un multiplexeur (MUX) et un dé-multiplexeur (DMUX) comme indiqué sur la Figure I-17. Le choix de la grille de multiplexage est conditionné au le nombre de liens CPRI et aux paramètres environnementaux (typiquement la température des antennes extérieures) des éléments passifs (MUX/DMUX) et actifs (transceivers).

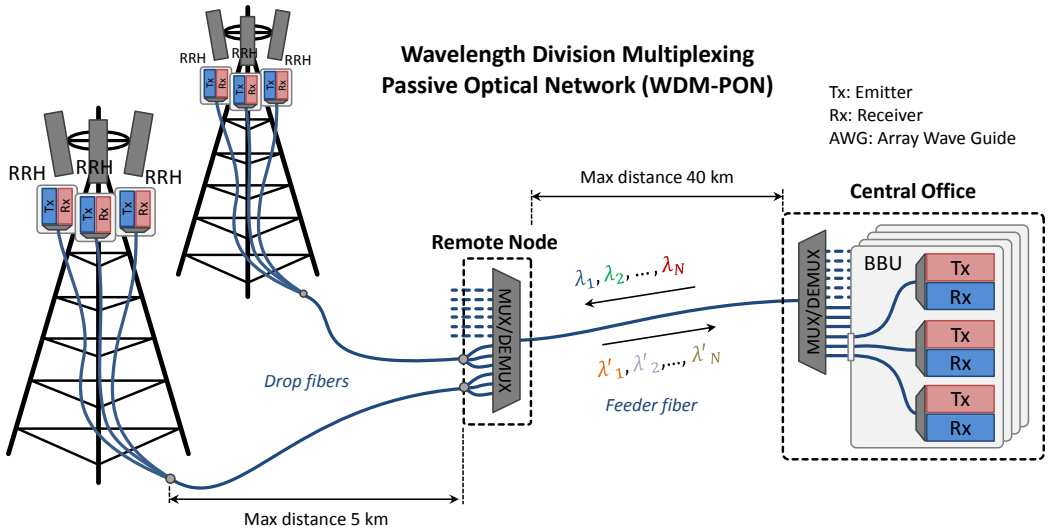


Figure I-17 : Schéma de principe du WDM-PON dans le contexte du C-RAN

- Le CWDM

Parmi les différentes technologies WDM, le Coarse WDM (CWDM) est considéré comme le plus pratique, car il est le mieux adapté pour les équipements extérieurs (il ne nécessite pas de contrôle de température). On peut multiplexer jusqu'à 16 longueurs d'onde, espacées de 20 nm (canaux normalisés de 1270 nm à 1610 nm) sur une seule fibre. Au maximum, un site antennaire macro comporte 15 RRH donc il serait possible d'agréger le fronthaul du site cellulaire entier en une seule fibre. Un inconvénient est qu'une gestion appropriée est nécessaire pour aligner les longueurs d'onde des émetteurs-récepteurs des points d'extrémité pour chaque flux de fronthaul. A titre d'illustration, la Figure I-18 propose une architecture basée sur un transport bi-fibre montant/descendant : 15 transceivers CWDM (disponibles commercialement pour transporter du CPRI (jusqu'au CPRI3) et fonctionnant à des températures -40 ; +85°C) sont insérés dans les RRH et 15 autres dans les BBU. Le 16^{ième} canal CWDM est mis à disposition pour réaliser la supervision du lien fronthaul de manière semi-passive : du côté du site antenne, un élément passif de bouclage (fonction miroir) est inséré sur les ports communs des multiplexeurs ; du coté central, un canal CWDM est dédié à la supervision (typiquement un canal en bande U) et inséré sur le lien de fronthaul à l'aide de coupleurs dissymétriques. L'aller-retour de ce canal dans le lien bi-fibre permet d'obtenir les relevés de puissances (Px1 et Px2) et de budget optique [31].

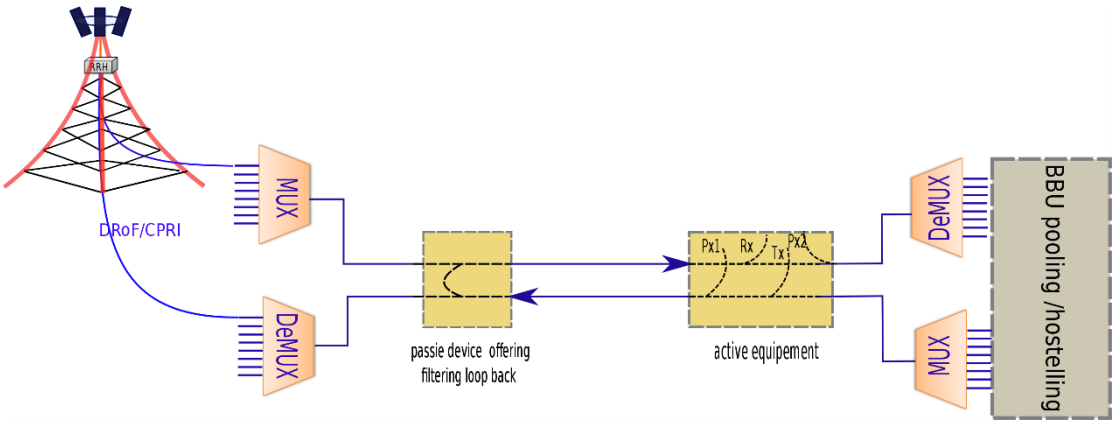


Figure I-18 : Exemple de fronthaul mobile CWDM sur 2 fibres avec insertion d'outils de monitoring

S'il y a moins de 9 RRHs, une configuration monofibre serait plus appropriée Elle permettrait de n'utiliser que deux multiplexeurs, des transceivers CWDM (dual fiber SFP : Small Form Factor double fibre) seraient nécessaires.

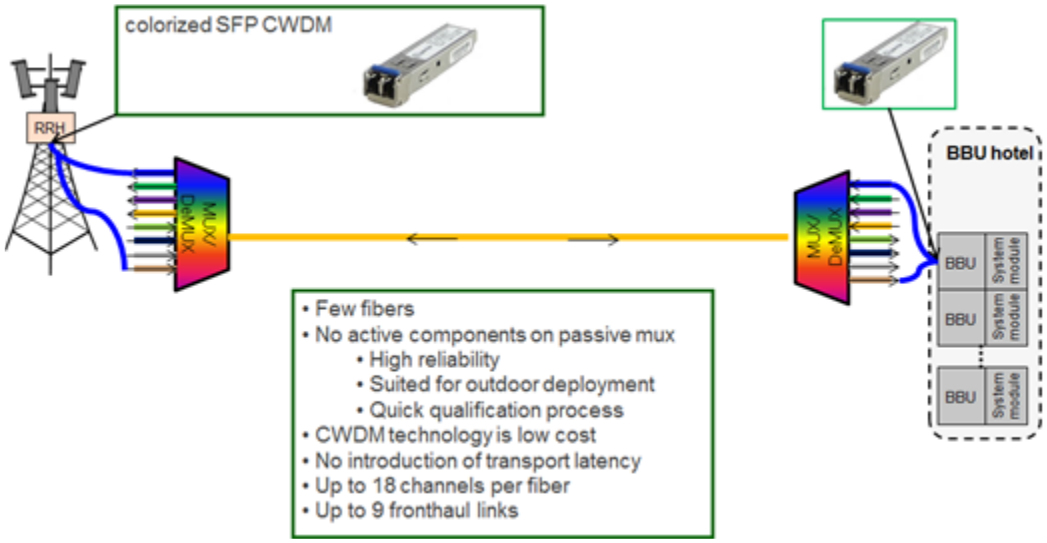


Figure I-19 : Exemple de fronthaul mobile CWDM monofibre [31]

Une solution de supervision (monitoring) resterait toujours à définir (problèmes de rétrodiffusion de Rayleigh à résoudre) au travers d'une solution semi-active.

- Dense-WDM (DWDM)

Les réseaux métropolitains DWDM sont actuellement déployés avec des distances allant de quelques 10 km à plusieurs centaines de km. L'objectif du DWDM est de réduire l'espacement des canaux (12,5 GHz, 25 GHz, 50 GHz, 100 GHz ou 200 GHz) par rapport au CWDM afin d'augmenter le nombre de canaux. La grille standardisée des canaux DWDM propose jusqu'à 128 longueurs d'ondes et les transceivers actuels permettent de transporter des données jusqu'à 10Gbit/s. Les canaux sont normalisés en bande C de 1520 nm à 1570 nm [46] et en bande L de 1570 nm à 1624 nm [47]. Actuellement, dans le réseau d'accès optique (NG-PON2) on utilise 40 canaux DWDM, transportant chacun un trafic allant de 1 Gbit/s à 10 Gbit/s. L'intérêt du multiplexage en longueur d'onde est donc de permettre l'augmentation du débit total tout en conservant des débits par canaux relativement faibles. L'électronique associée et les bandes passantes des composants optiques restent ainsi plus attractives comparées à une liaison monocanale pour le même débit. Cependant, le DWDM engendre d'autres problèmes, notamment le surcoût dû aux équipements de multiplexage (MUX/DEMUX) pour lesquels il faudra utiliser des filtres AWG (Array Waveguide Gratings) Gaussian ou flat top, la nécessité de la stabilité des longueurs d'onde des émetteurs, la diaphonie entre les canaux, etc… Un autre facteur important à prendre en compte dans le DWDM est la gestion des longueurs d'onde dans le réseau. Il est possible de discerner trois grandes catégories d'émetteurs, les émetteurs (colorisés) utilisant une longueur d'onde fixe (lasers DFB), les émetteurs accordables en longueur d'onde (lasers accordables) et les émetteurs achromatiques (modulateurs réflectifs)

Etant donné que le nombre de canaux est important dans le DWDM, il est plus approprié d'utiliser la configuration monofibre que la configuration à double fibre pour le déploiement de fronthaul. Dans le déploiement massif de DWDM ou CWDM utilisant des SFP colorisés sur le fronthaul, il sera nécessaire de connaitre parfaitement quel type de SFP a été utilisé sur le site de l'antenne et le BBU correspondant dans le CO. Dans le cas où le SFP colorisé est en panne, il doit être changé par le même SFP colorisé, ce qui implique une gestion de stock onéreuse. Dans le cas d'émetteur colorisé fixe, une liaison DWDM avec 40 canaux nécessitera 40 émetteurs verrouillés sur une longueur d'onde différente. Utiliser des sources accordables en longueur d'onde nécessitera un seul type d'émetteur, mais il faudra gérer son accordabilité sur le canal à transmettre et ils sont plus chers. Une alternative serait d'utiliser des émetteurs achromatiques, dont les performances sont un peu moindres mais qui simplifierait grandement la gestion du réseau.

Parmi les technologies d'émetteur-récepteur DWDM disponibles commercialement aujourd'hui, il n'existe pas de produits compatibles avec la gamme de température -40 ; +85°C nécessaire en environnement extérieur. Le projet ANR LAMPION, dans lequel s'est inscrite cette thèse, avait pour but de répondre à ce besoin de multiplexage DWDM sur le lien fronthaul en utilisant des transceivers achromatiques bas-coût, basés sur des RSOAs (Reflective Semi-conductor Optical Amplifiers) self-seeded. Leur principal avantage est que la longueur s'établit automatiquement par l'infrastructure, par une simple connexion du transceiver à un port du MUX/DMUX. La Figure I-20 représente une architecture implémentant cette technologie pour le transport de liens CPRI dans le sens montant.

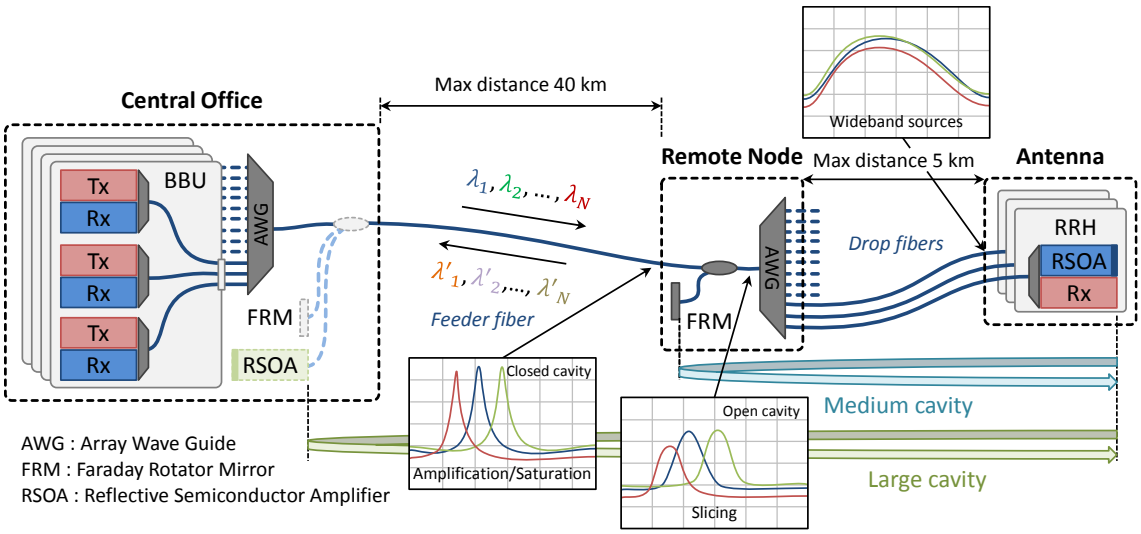


Figure I-20 : Exemple d'architecture RSOA Self-seeded DWDM-PON en sens montant dans le contexte du C-RAN

Chaque RSOA émet un spectre ASE (Amplified Spontaneous Emission) large bande qui est ensuite découpé par le canal de l'AWG (Array Wave Guide) correspondant au port sur lequel il est connecté. Un diviseur placé après l'AWG envoie une partie du flux optique vers la fibre en direction du CO (Central Office) et l'autre partie vers un miroir qui se charge d'effectuer une contreréaction optique pour tous les RSOA, permettant ainsi de créer une cavité optique longue de quelques mètres à quelques kilomètres. Les flux optiques de tous les RSOA effectuent plusieurs allers-retours dans la cavité, subissant une amplification jusqu'à la saturation et entrainant ainsi l'amincissement des raies des sources dans le spectre optique.

Le gain des RSOA pouvant être très sensible à la polarisation du champ optique, le miroir utilisé est souvent un FRM (Faraday Rotator Mirror). Ce miroir peut être placé soit au RN (Remote Node), soit au CO. Dans ce dernier cas, il peut être judicieux d'utiliser un RSOA à la place du miroir en régime de saturation afin d'une part de compenser les pertes importantes de la cavité, et d'autre part d'effacer les données afin de produire un retour optique nettoyé [50]. L'architecture RSOA self-seeded WDM-PON présente l'avantage d'être très flexible. Un seul type d'émetteur est nécessaire, le RSOA. L'allocation en longueur d'onde se fait intrinsèquement en fonction du port de l'AWG sur lequel il est connecté, ce qui représente un formidable atout quant au déploiement et au maintien du réseau. Cette technologie RSOA self–seeded avait été proposée comme candidate pour le réseau d'accès DWDM afin d'atteindre 2,5 Gbit/s en bande C [48].

I.2.4.5. Le projet ANR Lampion

Le projet LAMPION avait pour objectif de démontrer la faisabilité de l'élaboration d'un système de télécommunication basé sur le multiplexage en longueur d'onde (Wavelength Division Multiplexing Passive Optical Network (PON WDM)), dans le but de l'utiliser pour réaliser les liens de fronthaul mobile (contexte du C-RAN) et pour les réseaux métropolitains DWDM. Le PON WDM offre la possibilité d'associer une longueur d'onde à un seul utilisateur qui peut être de n'importe quel type (client entreprise, réseau point à point, terminaison de réseau longue distance, terminaison de réseau mobile, terminaison de réseau d'accès). Des systèmes commerciaux sont disponibles avec un débit de transmission de 1,25 Gbit/s, mais le PON WDM n'a pas atteint un vaste marché pour le moment. En effet, le coût des technologies mises au point jusqu'à présent réduit l'utilisation du PON WDM dans les réseaux d'accès. Par ailleurs, le PON WDM n'a jamais été normalisé par les institutions. LAMPION proposait d'étendre les performances d'un système PON DWDM en termes de débit (jusqu'à 10 Gbit/s) et de portée (> 80 km). La technologie WDM choisie dans le projet LAMPION est basé sur le « self seeded RSOA » Reflective Semiconductor Optical Amplifier, car elle apporte des avantages en termes de coût en utilisant des dispositifs achromatiques et identiques pour chaque terminaisons du réseau et de plus l'établissement de la longueur d'onde est simplifiée : la longueur d'onde est réglée automatiquement et seulement en connectant le RSOA à un canal d'un multiplexeur associé à un miroir, tous deux placés dans l'infrastructure du réseau. Cette opération créée une source à cavité externe dont la longueur varie selon la distance entre le RSOA et le miroir Figure I-21. Cette technologie RSOA self–seeded avait été proposée comme candidate pour le réseau d'accès DWDM afin d'atteindre 2,5 Gbit/s en bande C [48].

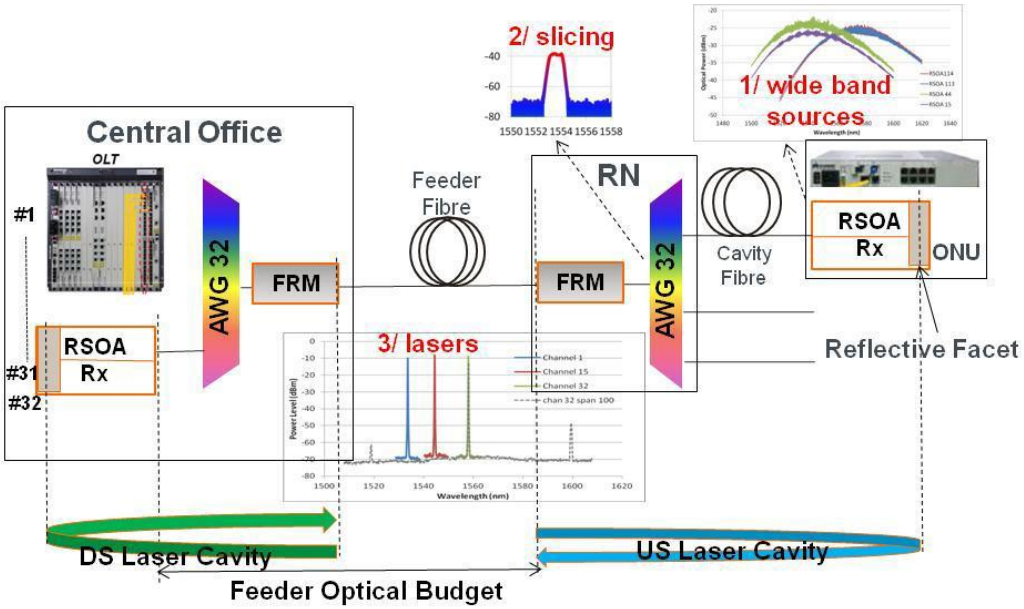


Figure I-21 : Exemple d'utilisation de WDM-PON avec des émetteurs achromatiques dans le contexte C-RAN [49]

Le projet LAMPION proposait en particulier d'étudier la qualité de la source créée par le self seeded RSOA. Cette étude devait conduire à déterminer des moyens spécifiques pour améliorer les performances en transmission du système. En outre, plusieurs runs de composants RSOA ont développés et une évaluation complète de leurs caractéristiques effectuée.
Pour améliorer les performances de transmission, l'amplification optique a été testée expérimentalement et des codes correcteurs d'erreur développés spécifiquement pour cette technologie. Enfin, dans le but de démontrer la faisabilité du système, les RSOA ont été intégrés dans des SFP avec leurs cartes de service associés et FEC embarqué.

Le projet Lampion regroupait cinq partenaires : Orange Labs, Ekimops, III-V Lab, l'université de Rennes/laboratoire FOTON et l'Université de Limoges/laboratoire Xlim.

Les travaux de recherche reportés dans ce manuscrit ont été menés sur le développement des codes correcteurs d'erreurs de type LDPC et leur implémentation sur FPGA en tenant compte des contraintes demandées dans le cadre du projet LAMPION. Plus précisement, une latence circuit strictement inférieure à $5\mu s$ a été exigée afin de ne pas dépasser la limite de temps de transmission sur la fibre optique.

I.2.4.6. Résumé

Ce chapitre a présenté les évolutions des réseaux d'accès mobile qu'il faut envisager pour répondre au besoin croissant de débit inhérent aux changements de comportement des usagers et à l'utilisation croissante et ubiquitaire de terminaux mobiles. Un inventaire des évolutions de RANs (GERAN, UTRAN, EUTRAN, C-RAN) liés aux générations successives des réseaux mobiles (2G, 3G, 4G, 5G). Les raisons de ces évolutions et les performances ont été reportées à chaque fois.

L’accent est mis sur la dernière/future version, le C-RAN pour lequel on a amené les RRH au plus près des antennes (réduction des longueurs de câbles coaxiaux et des pertes) puis centralisé les BBU dans des hôtels au sein des centraux.

En quelques points essentiels, le C-RAN présente plusieurs avantages :

- La réduction de coût d’infrastructure
- La réduction des énergies consommées
- La réduction de coût de CAPEX et OPEX
- L’amélioration des implémentations des techniques de transmission radio
- Des mécanismes de sécurité pour empêcher l'accès non autorisé et les attaques malveillantes
- La gestion à distance, y compris la mise à jour software. Un port d'accès local protégé pour le fonctionnement sur site et des indications d'état local pour le dépannage sont requis
- La collecte d'alarmes et d'événements pour la prévention/localisation de pannes à distance
- Une empreinte minimale et émetteur-récepteur de facteur de forme standard

Un nouveau tronçon appelé fronthaul est apparu, tronçon pour lequel plusieurs solutions possibles de réalisation ont été présentées. Le WDM-PON a été retenu comme la plus en adéquation avec les contraintes imposées. Pour répondre aux exigences imposées par les normes RAT (Radio Access Technologies : norme qui définit l'architecture réseau et spécifie les fonctions, les interfaces et les protocoles assignés aux nœuds RAN et aux dispositifs mobiles ; par exemple, WCDMA/HSPA, LTE, WiMax) sur la précision et la stabilité de l'interface radio, les BBU et les RRH doivent être synchronisées précisément les unes avec les autres. En conséquence, le Fronthaul présente des exigences de QoS (latence) strictes, en termes de taux d'erreur binaire (TEB), de fréquence et de synchronisation de phase. Par d'exemple, le CPRI spécifie que le TEB ne doit pas être supérieur à 10^{-12} et que la gigue introduite par le lien CPRI ne peut pas dépasser 2 parties par milliard (ppb) au budget de précision de fréquence de la station de base.

Pour terminer, nous avons introduit le projet ANR LAMPION au sein duquel ses travaux de thèse ont été effectués, qui propose une solution à base de technologie RSOA self-seeded pour atteindre 2,5 Gbit/s sur le lien fronthaul. Les codes correcteurs d’erreur étant au cœur de ces travaux de thèse pour garantir le TEB sur le lien fronthaul tel que décrit dans le chapitre, nous proposons dans le second chapitre d’en faire un état de l’art.

Chapitre II

Les Codes Correcteurs d'erreurs

Chapitre II. Les Codes Correcteurs d'erreurs

II.1. Introduction

Dans le contexte du C-RAN (Cloud radio access network), pour assurer la qualité de service en tenant compte de l'instabilité de l'amplificateur optique, un code correcteur d'erreurs, dit FEC (Forward Error Correction) est proposé. Globalement, le codage correcteur d'erreur est une technique pour contrôler et corriger les erreurs de transmission de données introduites par des canaux de propagation bruités. Un mathématicien américain Richard-Hamming a proposé à partir des années 1940 à 1950 [51] une idée importante pour le codage FEC qui consiste à rajouter au code source un message redondant en utilisant un ECC (Error Correcting Code ou bien Code de Correction d'Erreurs) comme illustré sur la Figure II-1.

Figure II-1 : Utilisation des FEC dans une transmission

Les messages redondants ou les redondances se calculent en fonction des données (messages) ou des bits utiles, ce code permet de sécuriser les informations transmises mais également de détecter et certaines fois de corriger un nombre limité d'erreurs dans les messages reçus sans demander de retransmission de messages (mécanisme appelé ARQ : Automatic Repeat Request). Par exemple, pour un code C constitué de N bits, un message de ce code contient K bits d'informations utiles et $(N-K)$ bits de redondance. Le code est toujours noté $C(N,K)$ et sa performance dépend du nombre de bits de redondance mais aussi des règles (algorithmes) qui génèrent ces redondances. Dans le cas normal, plus il y a des bits de redondance, meilleures sont les performances que nous pouvons obtenir. En revanche, le taux de codage ou plutôt le rendement de codage qui est calculé par (K/N), et qui correspond au rapport des bits d'information utiles sur le nombre total de bits utilisés pour coder un message, doit rester élevé dans le contexte C-RAN pour permettre d'obtenir des débits de transmission très élevés. En conséquence, nous devrons faire non seulement un compromis entre le rendement de code et les performances, mais nous devrons aussi améliorer l'efficacité des processus de codage et de décodage pour les différents algorithmes.

II.2. Chaîne de communication

Il est important de connaitre la capacité du canal avant la transmission. Dans l'article de Shannon paru en 1948 [53], Shannon donne le modèle simplifié d'une communication sans fil. Ce modèle est composé de 6 parties : S (source), CS (codage de source), CC (codage de canal), DC (décodage canal), DS (décodage source), D (destination). La Figure II-2 montre le modèle utilisé par Shannon pour la description de la communication. Evidemment, le FEC a un impact considérable sur la performance de la chaîne complète d'émission-réception.

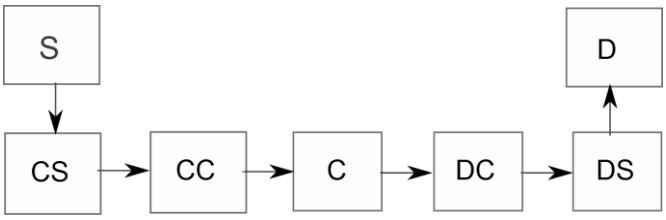


Figure II-2 : Paradigme de Shannon

II.3. Codage de source

Des bits de redondance sont rajoutés avant que le message ne soit émis par la source S. Le codage de source a pour but d'éliminer la redondance de la source (compression). Il faut pour cela répartir équitablement l'information sur l'ensemble des symboles x_i. On définit alors l'entropie d'une source comme étant l'information moyenne émise par la source [54] :

$$H(X) = E\left(log_2\left(\frac{1}{p_i}\right)\right) = \sum_i p_i log_2(1/p_i) \qquad \text{(II-1)}$$

où p_i est la probabilité d'émettre le i[ème] symbole. Pour une source binaire avec $p_\mathrm{i} = x$, on peut alors déduire la fonction d'entropie binaire :

$$H_2(X) = xlog_2(1/x) + (1-x)log_2(1/(1-x)) \qquad \text{(II-2)}$$

Si l'entropie de la source est inférieure à l'entropie maximale possible, alors la source est qualifiée de redondante et la taille (longueur) du message émis peut être réduite par un codeur de source.

Concernant le codage Source, le premier théorème de Shannon indique qu'il existe un procédé de codage déchiffrable où la longueur moyenne des mots par symbole de la source est aussi voisine que l'on veut de sa borne inférieure $H/log2(q)$ où q est la taille de l'alphabet considéré.

II.4. Codage de canal

Le codage de canal a pour but de protéger le message émis par la source normalisée (sans redondance) contre les bruits et les perturbations introduits par le canal de propagation. Pour assurer un transfert sur un canal bruité, il est nécessaire d'introduire de la redondance dans le message transmis.

En réception, le décodeur reçoit le message émis par le codeur perturbé à cause de bruit du canal. L'information mutuelle $\mathrm{I(X;Y)}$ en Figure II-3 entre l'entrée et la sortie du canal peut être exprimée sous la forme [54] :

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \qquad \textit{(II-3)}$$

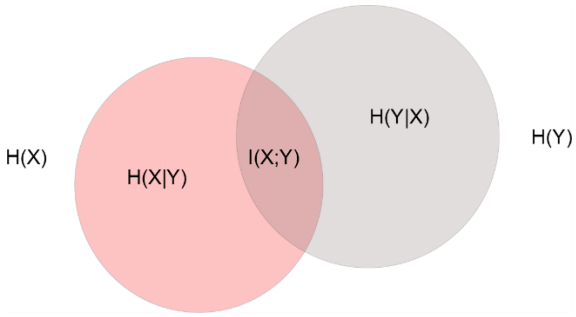


Figure II-3 : Information mutuelle

Où $H(X)$ est l'entropie de la source et $H(X|Y)$ est le niveau d'incertitude sur la variable X connaissant Y. Le théorème concernant le codage de canal est le résultat le plus important de la théorie de l'information (Figure II-3). Ce théorème est connu sous le nom de second théorème de Shannon (Noisy-Channel Coding Theorem).

II.5. Second théorème de Shannon

Ce théorème peut être décomposé en trois parties [55] :

1. Pour tous les canaux discrets sans mémoire, la capacité du canal C a les propriétés suivantes. Pour tout $\varepsilon > 0$ et $R < C$, pour N assez grand, il existe un code de longueur N et de rendement R muni d'un algorithme de décodage tel que la probabilité d'erreur bloc maximale soit $< \varepsilon$.

$$C = \max_{px}(X;Y) \qquad (II\text{-}4)$$

2. Si une probabilité d'erreur binaire p_b est acceptable, le débit $R(p_b)$ peut être atteint, avec :

$$R(p_b) = \frac{C}{1-H_2(p_b)} \qquad (II\text{-}5)$$

3. Pour tout p_b, les débits plus grands que $R(p_b)$ ne peuvent pas être atteints.

Selon le théorème de Shannon, on doit respecter ces critères pour atteindre le meilleur résultat pour un rendement R donné du code.

Dans le lien fronthaul, les mesures expérimentales prises par Ekinops ont montré que le bruit est gausien et que les effets non linéaires de la fibre ne sont pas prépondérants. Donc le canal est considéré comme un canal AWGN (Additive White Gaussian Noise) d'après les données fournies par l'entreprise Orange Labs. Le canal AWGN et le canal BSC (canal binaire symétrique dont nous utiliserons le modèle au cours de ce mémoire) sont présentés dans la partie suivante.

II.6. Canal AWGN

Le canal AWGN possède une entrée et une sortie continue en amplitude (le canal reste discret en temps). La perturbation (bruit) introduite par le canal est l'addition d'un bruit blanc de densité spectrale de puissance constante et de distribution gaussienne en amplitude. Le signal reçu peut s'exprimer par : $Y = X + E$ où E suit une loi normale de moyenne nulle et de variance σ^2 (Figure II-4). La perturbation ou le bruit est caractérisé par cette variance qu'on obtient à partir

de la somme des variances des différents bruits rencontrés sur la transmission supposée tous gaussiens et indépendants (bruit d'émission spontanée, bruit thermique, bruit de battement entre le signal et le bruit, bruit de battement entre le bruit et lui-même...) [56].

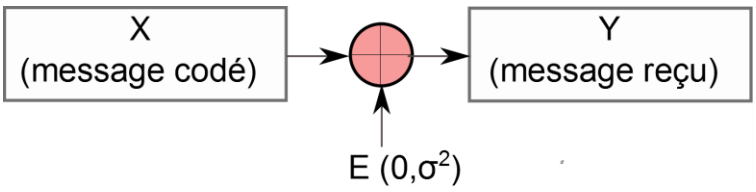


Figure II-4 : Canal AWGN

II.6.1. Caractéristique mathématique

La densité de probabilité d'une variable E à statistique gaussienne, de moyenne nulle et de variance σ^2 s'exprime selon la formule suivante :

$$P_e(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}} \qquad (II\text{-}6)$$

Si la moyenne m de la variable gaussienne E de variance σ^2 est non nulle, c'est à dire la distribution est normale non-centrée. La densité de probabilité de E est donnée par :

$$P_e(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-m)^2}{2\sigma^2}} \qquad (II\text{-}7)$$

Pour faciliter les analyses des résultats, on utilise la modulation BPSK qui est représentée sur la Figure II-5 ci-dessous.

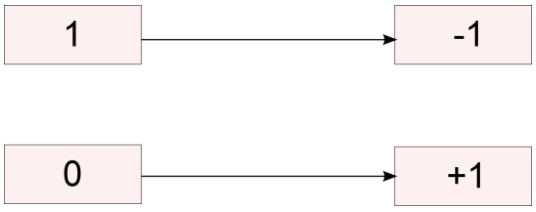


Figure II-5 : BPSK modulation

La Figure II-6représente un exemple de densité de probabilité d'une variable suivant la loi gaussienne centrée et non-centrée ($m = \pm 1$).

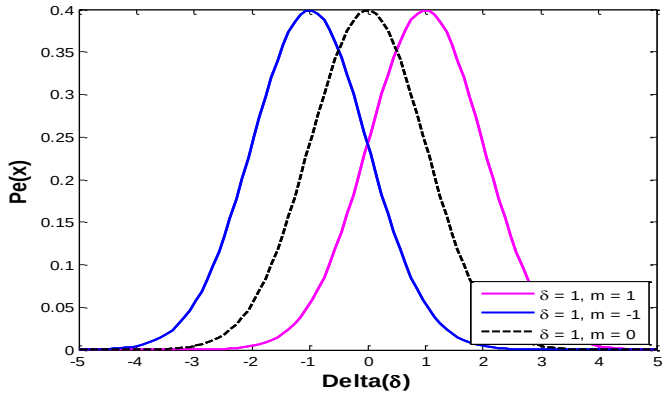


Figure II-6 : Densité de probabilité $P_e(x)$ d'une loi gaussienne

II.6.2. Capacité du canal AWGN

Supposns une modulation binaire (BPSK), pour un signal avec une bande-passante W, une puissance S, une puissance de bruit N traversant la canal AWGN, la capacité du canal C s'écrit sous la forme :

$$C = W log_2(1 + {}^{S}/_{N}) \qquad (II\text{-}8)$$

Pour une transmission bianire, si le taux de transmission (débit d'information) est égal à *R*, E_b exprime l'energie par bit, on écrit alors :

$${}^{C}/_{W} = log_2\left(1 + \left(\frac{R * E_b}{N_0 * W}\right)\right) \qquad (II\text{-}9)$$

Donc quand W tend vers l'infini, la valeur asymptotique s'exprime par :

$$SNR = \frac{E_b}{N_0} \geq \lim_{W \to \infty} \frac{2^{\frac{C}{W}} - 1}{\frac{R}{W}} = ln\, 2 = -1.59\ dB \qquad (II\text{-}10)$$

Cette formule nous indique que, dans le cas idéal, en considérant une bande-passante infinie, pour transferer des informations sans erreur, il faut que $SNR_{min} \geq -1.59\ dB$.

Supposons aussi connue la capacité du canal C (bit/symbole), le taux de transmission ou bien rendement du code R (bit/symbole), l'energie par bit E_b, l'energie par symbole E_s, on a $E_s = R * E_b$. Selon les équations précedentes et d'après l'égalité : $C = f\left({}^{E_s}/_{N_0}\right) = f\left({}^{R * E_b}/_{N_0}\right)$, on obtient :

$$R \leq C = f\left({}^{E_s}/_{N_0}\right) = f\left({}^{R * E_b}/_{N_0}\right) \qquad (II\text{-}11)$$

Quand $R = C$, on exploite au maximum la capacité de notre canal. En supposant que la probabilité d'erreur est équal à $P_b(e)$, on cherche alors la valeur de $({}^{E_b}/_{N_0})_{min}$ quand $P_b(e) = 0$. On peut obtenir aussi $R' \leq C$ Quand $P_b(e) \neq 0$:

$$R' = R\left(1 + P_b(e)log_2 P_b(e) + \left(1 - P_b(e)\right)log_2\left(1 - P_b(e)\right)\right) \qquad (II\text{-}12)$$

Finalement on peut avoir :

$$\left(^{E_b}/_{N_0}\right)_{min} = f^{-1}(R')/R \qquad (II\text{-}13)$$

A partir de ces équations, on peut calculer les limites de Shanon avec ou sans erreurs.

II.7. Canal BSC

Le canal binaire symétrique (BSC : Binary Symmetric Channel) est illustré sur la Figure II-7. Il possède 2 entrées et 2 sorties (0, 1). Il transmet chaque bit correctement avec une probabilité $1 - f$ et inverse l'entrée avec une probabilitéf.

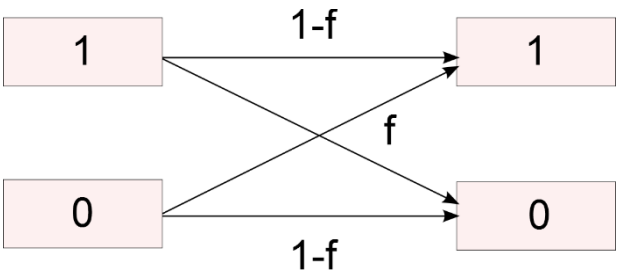


Figure II-7 : Canal BSC

Au niveau du récepteur, avant le décodage, il existe un seuil S qui permet au récepteur de décider de la valeur de la donnée reçue, comme l'illustre la Figure II-8.

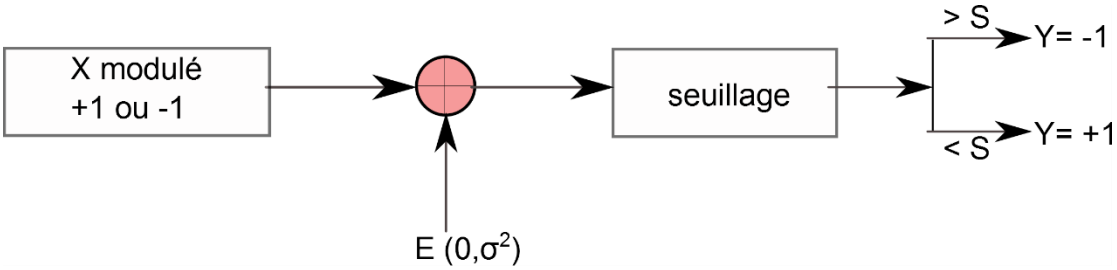


Figure II-8 : Chaine équivalente émission/réception pour un canal BSC

II.7.1. Capacité du canal BSC

La capacité du canal binaire symétrique est obtenue pour une distribution uniforme (i.e. $\Pr(x = 0) = \Pr(x = 1) = \frac{1}{2}$) et est égale à [57] :

$$C_{BSC} = 1 - H_2(f) \qquad (II\text{-}14)$$

$$ou\ \ H_2(f) = xlog_2\left(^1/_x\right) + (1 - x)log_2\left(^1/_{1\,-\,x}\right) \qquad (II\text{-}15)$$

$H_2(f)$ est une fonction d'entropie binaire donnée. Si on suppose ici un seuil S égal à 0, on peut obtenir la probabilité d'erreur P_b :

$$P_b = \frac{1}{2}\int_{-\infty}^{0} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x+1)^2}{2\sigma^2}} dx + \frac{1}{2}\int_{0}^{\infty} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-1)^2}{2\sigma^2}} dx = Q\left(\frac{1}{\sigma}\right) \qquad (II\text{-}16)$$

Avec : $\sigma^2 = N_0/2$

Donc on peut obtenir deux critères :

- La limite de Shannon sans erreur :

$$R = C = 1 + P_b log_2(P_b) + (1 - P_b) log_2(1 - P_b) \qquad (II\text{-}17)$$

- La limite de Shannon avec erreurs :

$$C = R(1 + P_b(e) log_2\big(P_b(e)\big) + \big(1 - P_b(e)\big) log_2(1 - P_b(e))) \qquad (II\text{-}18)$$

Les courbes de la Figure II-9 donnent la capacité sans erreur pour les canaux AWGN et BSC.

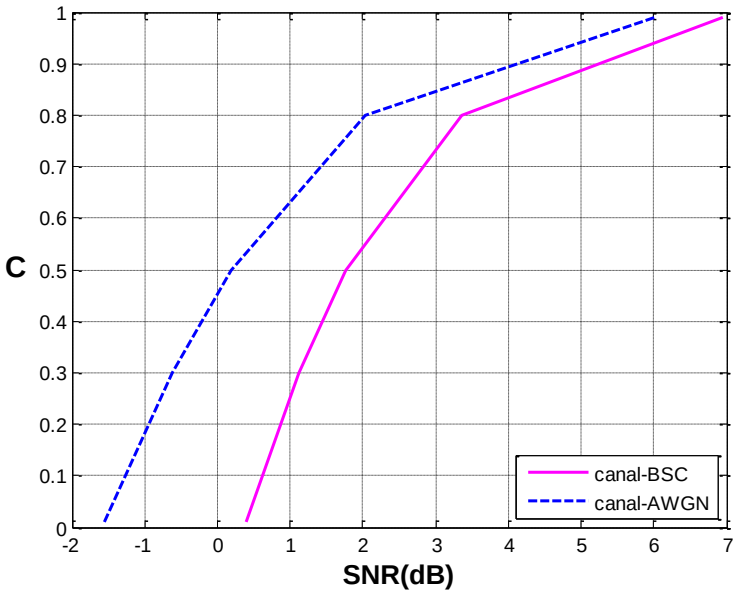


Figure II-9 : Capacité des canaux BSC et AWGN pour BPSK

On peut conclure que quand le SNR tend vers l'infini, le débit d'information tend vers 1 mais n'atteint jamais 1.

II.8. Conclusion

Nous avons rappelé les calculs de capacité avec la modulation binaire pour des canaux de type AWGN, BSC et montré l'importance du paramètre *R*. Dans le contexte de notre étude où le rendement du code est toujours supérieur à 0.8, on se concentrera plus particulièrement sur la zone où $SNR \geq 3\ dB$ en respectant la limite de Shannon.

II.9. Codage FEC

II.9.1. Introduction

La probabilité de blocage obtenue dans la partie précédente correspond à une borne qui donne accès au rendement de code maximum. Il n'est cependant pas garanti que la performance des codes correcteurs d'erreurs atteigne cette limite dans tous les cas. Dans ce chapitre on introduit les principaux codes qui approchent au mieux la limite de Shannon et qui ont les meilleurs performances dans notre contexte comme les codes LDPC (low density parity check code) [66], les codes RS (Reed-Solomon code) [63] ou les codes BCH (Bose, Ray-Chaudhuri et Hocquenghem) [64] et les codes polaires [67]. De plus, nous justifierons notre choix du code LDPC comme un excellent compromis performances/complexité pour mener à bien les travaux entrepris avec Orange dans le contexte du C-RAN. Pour finir, nous allons donner les résultats de simulation et les améliorations apportées à la partie décodage.

II.9.2. Théorie des codes

Dans l'article de Shannon 1948 [53], Shannon a prouvé qu'il était possible de transmettre un débit R de maniere fiable sur un canal non-fiable de capacité C tant que $R < C$, par contre pour le cas inverse, on ne peut pas garantir la fiablité du message reçu à travers le canal [68]. Dans ce papier, Shannon aussi introduit la notion de "codes" comme un ensemble fini de vecteurs dans l'alphabet d'entrée. Si on suppose que tous les vecteurs possèdent une longueur identique N et que N est égal à K/R (qui peut être codé sur K bits), il faut alors N utilisations du canal pour transmettre K bits. Le débit (rendement du code) est donc égal à : K/N bits par utilisation du canal. On a tout simplement :

$$N = K + \left(1 - \frac{K}{N}\right) N = K + M$$

Avec *K* qui représente la taille en bits du message à coder et *M* qui représente la redondance apportée par le codage de canal.

En réception, c'est le décodeur qui, à partir du vecteur y reçu et de l'alphabet de sortie, va en déduire le mot de code C envoyé par l'encodeur. Si le canal introduit des erreurs, alors on ne peut pas identifier le mot de code transmis de manière certaine. On peut cependant trouver le mot de code le plus probable en maximisant la probabilité $p(y|c)$. Le décodage alors effectué est dit décodage à Maximum de Vraisemblance, Maximum Likelihood (ML), et suit la règle :

$$x = argmax_{c \in C} p(y|c) \tag{II-19}$$

Où C est l'ensemble des mots de code, aussi noté :

$$p(y|x) = \prod_{n=1}^{N} p(y_i|x_i) \tag{II-20}$$

La Figure II-10 illustre comment la règle du Maximum de Vraisemblance (ML) permet de détecter le message transmis en minimisant la distance entre le mot décodé et tous les mots de code dans l'alphabet. C'est ainsi que la distance minimale de Hamming notée d_{min} et qui est définie comme la plus petite des distances de Hamming entre deux mots de code distincts, constitue un paramètre très important dans la conception d'un code efficace. Si le mot de code qui contient N bits codés à partir de K bits avec une redondance de M bits est noté (N, K, M)

alors le pouvoir de détection du code, correspondant au nombre maximal d'erreurs détectables dans un mot dépend directement de la distance de Hamming minimale. Il est déterminé par $t = d_{min} - 1$.

Le pouvoir de correction, défini comme le nombre maximal d'erreurs corrigibles dans un mot est donné par [69] :

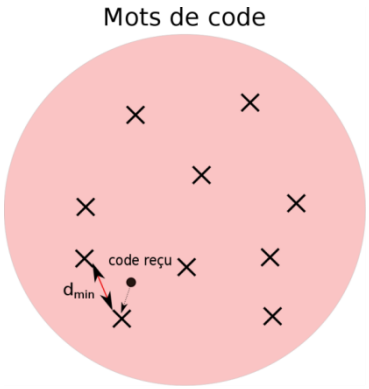


Figure II-10 : Distance de Hamming

$$\left[\frac{t}{2}\right] = \left[\frac{d_{min} - 1}{2}\right] \qquad \textit{(II-21)}$$

Shannon a prouvé qu'il existe des codes de rendement arbitrairement proche de la capacité pour lesquels la probabilité d'erreur du décodeur ML tend vers 0 quand la longueur du code tend vers l'infini. Ce théorème ne donne cependant pas la méthode permettant de construire ces codes. De plus, même si un code (aléatoire) atteignant la capacité pour un certain rendement était connu, la réalisation de l'encodage ainsi que du décodage par des algorithmes efficaces (polynomiaux) reste un problème ouvert.

Depuis longtemps, les scientifiques ont cherché à trouver des codes qui atteignent ou s'approchent de la capacité de canal. Pour une efficacité de codage et de décodage, Elias et Golay introduisent le concept de code linéaire en définissant des codes comme un sous-espace vectoriel d'un espace vectoriel de dimension finie, c'est à dire, au lieu de stocker une liste de 2^K mots de code dans l'encodeur, on fait une simple multiplication matricelle [70]. Ils ont ainsi montré que la détection à Maximum de Vraisemblance (ML) pour un canal binaire symétrique (BSC) était un problème NP complet.

Dans les années 1960s, les codes RS [63] ont été inventés et ils peuvent être décodés par un algorithme qui est appelé algorithme de Berlekamp Massey. La complexité de ces algorithmes limite la longueur des codes (toujours fixée à $(255, K)$) à quelques centaines de symboles entrainant un écart entre les performances des codes et la limite donnée par Shannon largement supérieur à 3 dB. En 1993, les turbo codes [71] sont présentés par C. Berrou et provoquent un émoi dans la communauté scientifique concernée puisqu'ils permettent de s'approcher très près (moins de 0.5 dB) de la borne de Shannon sur un canal AWGN tout en présentant une complexité de décodage raisonnable. Le décodage s'opère de façon itérative en échangeant à travers des entrelaceurs les informations extrinsèques issues des décodeurs de canal à sorties pondérées. Enfin en 1996, MacKay et Neal redécouvrent les codes LDPC qui avaient été inventés pour la première fois avant eux par Gallager en 1962, et généralisent

l'emploi des graphes de Tanner pour modéliser les échanges d'informations extrinsèques entre nœuds de variable et nœuds de parité.

II.9.3. Représentations polynomiales et matricielles

Les messages sont généralement codés (encoded message) de manière polynomiale ou matricielle. La représentation polynomiale consiste à écrire K bits avec un polynome $M(X)$ de degré $K-1$ en X :

$$M(X) = \sum_{i=0}^{K-1} m_i X^i \qquad \textit{(II-22)}$$

Où les m_i correspondent aux valeurs des différents symboles représentant le mot à coder : $M = [m_0 m_1 \ldots\ldots m_{K-1}]$. On définit également un polynôme générateur $G(X)$ de degré $N-k$ qui va multiplier le polynôme $M(X)$ pour obtenir le mot de code généré :

La multiplication est donnée par :

$$C(X) = M(X) * G(X) = \left(\sum_{i=0}^{K-1} m_i X^i\right) * \sum_{i=0}^{N-K} g_i X^i \qquad \textit{(II-23)}$$

Les g_i sont les coefficients associés au polynôme générateur $G(X)$.

Donc finalement, le mot $M(X)$ peut s'écrire sous la forme :

$$C(X) = \sum_{i=0}^{N-1} c_i X^i \qquad \textit{(II-24)}$$

Selon les équations précédentes, le message en sortie de l'encodeur est donc le résultat de la convolution de deux vecteurs.

Les mots de code en sortie de l'encodeur peuvent aussi s'exprimer de manière matricielle. Le mot de code *N* bits est obtenue par la multiplication des bits utiles (message) K bits avec la matrice d'encodage G de taille $K \times N$. Il peut ainsi s'écrire :

$$C_{Nbits} = [m_0 m_1 \ldots m_{K-1}]_K * G_{K \times N} \qquad \textit{(II-25)}$$

Si g_{ij} présente les élements de matrice G, chaque élément c_j peut s'exprimer comme une combinaison linéaire des m_i :

$$c_j = \sum_{i=0}^{K-1} m_i * g_{ij} \qquad \textit{(II-26)}$$

Cette représentation est identique à la représentation polynomiale mais permet parfois de visualiser plus facilement les schémas de codage.

II.9.4. Les deux grandes catégories de FEC

Deux grandes familles de FEC (codes correcteurs d'erreurs) sont à distinguer : les codes convolutifs et les codes en blocs. Par définition, les codes convolutifs calculent la redondance de manière continue à mesure que le flot de données arrive alors que les codes en blocs génèrent la redondance par blocs de données. Quelle que soit la nature du code, il est dit systématique si l'on retrouve les K bits d'information utiles M dans le mot codé C. Cela se traduit dans la représentation matricielle par le fait qu'une partie de la matrice génératrice est composée de la matrice identité. Cette représentation est appelée la forme systématique de la matrice génératrice.

II.9.4.1. Codes convolutifs

Les codes convolutifs, aussi appelés codes convolutionnels ont été initialement inventés en 1954 par Elias [72]. Le principe des codes convolutifs est de considérer le message à émettre comme une séquence semi-infinie de symboles. Il consiste à passer le message utile à coder dans une succession de registre à décalage. Il y a 3 paramètres des codes convolutifs : K (bits informations utiles), B (le nombre de regitres à décalage du code), N (bits codés après le codeur). On définit aussi la longeur de contrainte du code qui est égale à : $B + 1$. Pour un codeur convolutif illustré en Figure II-11, on utilise le plus souvent un schéma de codage de rendement *R* égal à 0.5. Parmi les codes convolutifs, il existe deux catégories : les codes non-systématiques (NSC : Non Systematic convolutional), et les codes systématiques récursifs (ou RSC : Recursive Systematic Convolutional codes). Les performances de ces deux types de code sont très comparables, cependant l'architecture des codeurs RSC se prête mieux à une implantation de type turbo code. On peut illustrer sur un exemple assez simple les principales différences entre les deux types de code. Prenons l'exemple d'un code convolutif NSC [5,7], on peut écrire les polynômes générateurs de ce code sous la forme :

$$G = [1 + D + D^2, 1 + D^2] \qquad \textit{(II-27)}$$

On peut obtenir deux versions du codeur RSC équivalent de la façon suivante :

Première version :

$$G_1 = [1 + D + D^2, 1 + D^2] \qquad \textit{(II-28)}$$

Deuxième version :

$$G_2 = (1 + D + D^2)\left[1, \frac{1 + D^2}{1 + D + D^2}\right] \qquad \textit{(II-29)}$$

Le schéma ci-dessous illustre la version NSC du code [5,7] tandis que le schéma d'après (Figure II-12) illustre la version G2 du codeur [5,7] en version RSC.

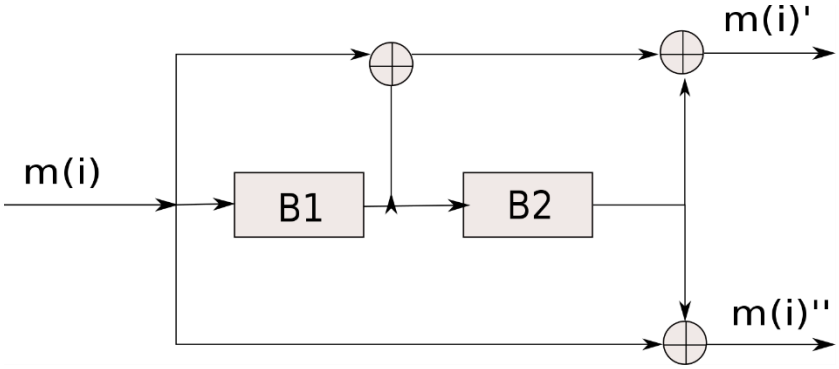


Figure II-11 : Codeur convolutif NSC

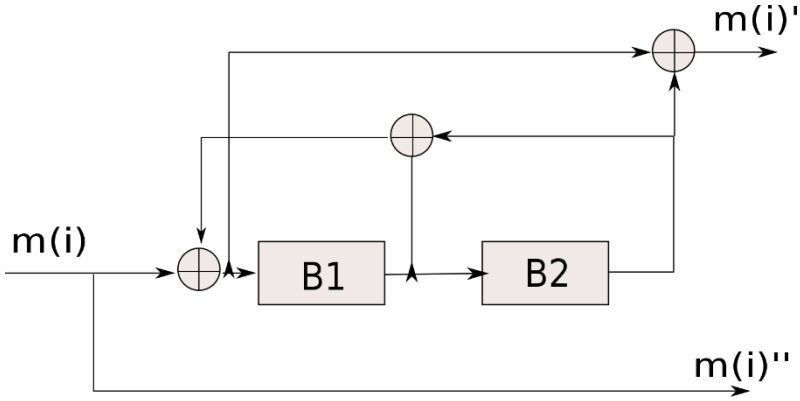


Figure II-12 : Codeur convolutif récursif systématique RSC

Même si les performances entre codes NSC et codes RSC sont très proches, il est montré dans [58] qu'un codeur récursif systématique présente de meilleures performances à faibles SNRs qu'un code convolutif non systématique, cependant les différences sont minimes. Concernant le décodage des codes convolutifs (NSC ou RSC), on utilise l'algorithme de Viterbi [59] qui approxime au mieux le décodage par maximum de vraisemblance et qui utilise un diagramme en treillis comme celui illustré sur la Figure II-13. Les états sont représentés par l'état des registres à décalage du codeur (B1 et B2 sur les Figure II-11 et Figure II-12). Les états communiquent entre eux en fonction de leurs états courant et de la nouvelle entrée (0 ou 1) qui arrive. La suite des états parcourus tout au long du codage du message à transmettre associée à la suite des bits obtenus en sortie forme le treillis du code. L'algorithme de Viterbi va alors chercher parmi tous les chemins possibles à travers le treillis celui qui minimise la distance entre la séquence reçue et les différentes séquences reconstruites à travers le treillis. Le nombre d'états d'un codeur convolutif est égal à 2^B où B désigne le nombre de registres à décalage du code.

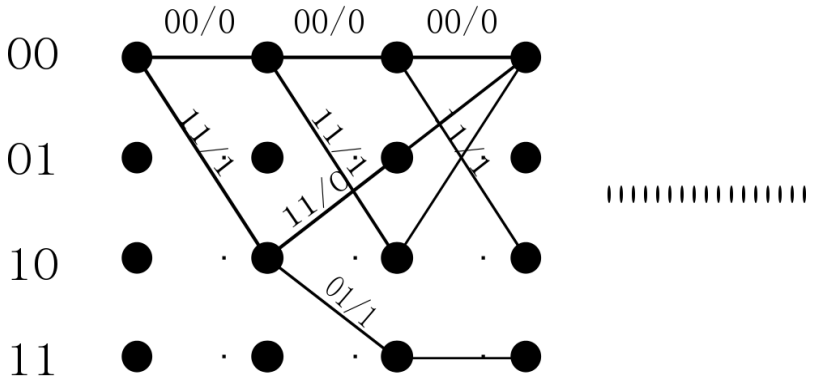


Figure II-13 : Exemple de diagramme en treillis

Une des principales difficultés pour l'emploi des codes convolutifs dans les liaisons optiques concerne le rendement obtenu. En effet, il est possible en utilisant des matrices de poinçonnage d'améliorer le rendement du code mais on dégrade rapidement les performances en termes de taux d'erreur[65].

L'utilisation des codes convolutifs récursifs systématiques en concaténation parallèle a donné lieu à la naissance des turbo-codes convolutifs [60].

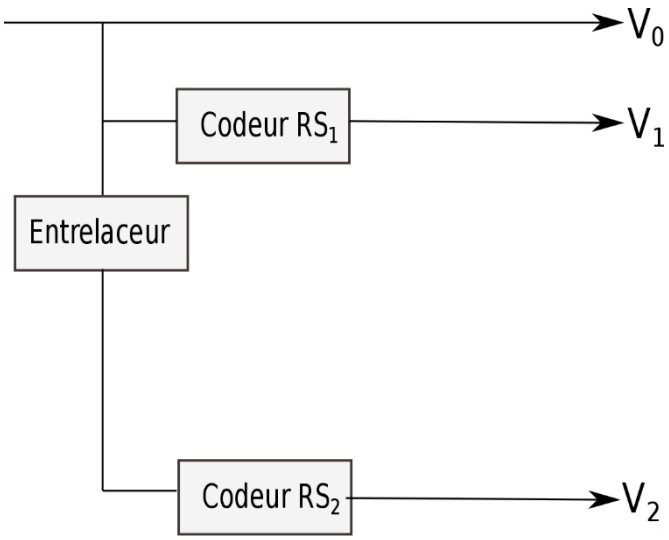


Figure II-14 : Exemple de codeur pour turbo-codes

Comme on peut le voir sur la Figure II-14, le codeur possède à l'origine un rendement égal à 1/3. On peut améliorer facilement cette valeur jusqu'à 1/2 en poinçonnant alternativement les sorties V1 et V2. D'autres motifs de poinçonnage plus sophistiqués permettent d'obtenir des rendements plus élevés.

En ce qui concerne le décodage l'élément fondamental qui constitue la brique du décodeur est l'algorithme de pondération ou de calcul des LLR's (Log-Likelihood Ratios) des bits d'information. Plusieurs algorithmes de pondération ont été utilisés depuis le plus sophistiqué appelé l'algorithme BCJR (Bahl, Cocke, Jelinek and Raviv) [61] jusqu'aux algorithmes simplifiés de type Max log-MAP ou SOVA (Soft Output Viterbi Algorithm) [62]. Le schéma de

décodage correspondant au codeur précédant est illustré sur Figure II-14. Le principe consiste à échanger itérativement des informations extrinsèques entre les deux décodeurs.

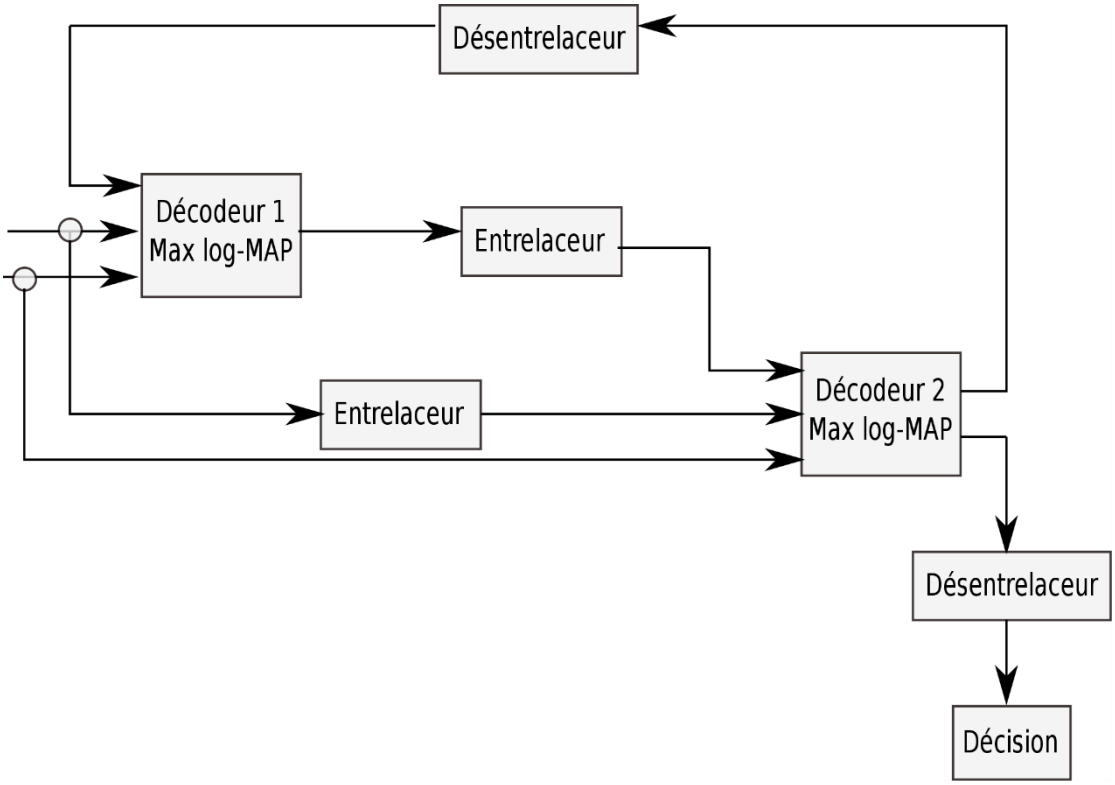


Figure II-15 : Exemple de décodeur pour turbo-codes

Du point de vue des performances, il est admis à l'heure d'aujourd'hui que les turbo-codes sont aussi bons que les codes LDPC. Le gros souci consiste, comme pour les codes LDPC à éviter l'effet plancher à forts SNR's, ceci nécessite une conception optimisée de l'entrelaceur [52].

II.9.4.2. Codes en bloc

Le codage en bloc consiste à découper les informations en blocs de *K* bits avec un message codé en sortie qui contient *N* bits. Ainsi le message codé est souvent exprimé sous forme (N, K), la redondance est donc égale à $N - K$ bits comme montré en Figure II-16. Si on utilise la représentation matricielle, en appelant *G* la matrice génératrice du code, on obtient l'écriture :

$$Kbits(message\ utile) * G_{K \times N} = N\ bits(message\ codé) \quad \text{(II-30)}$$

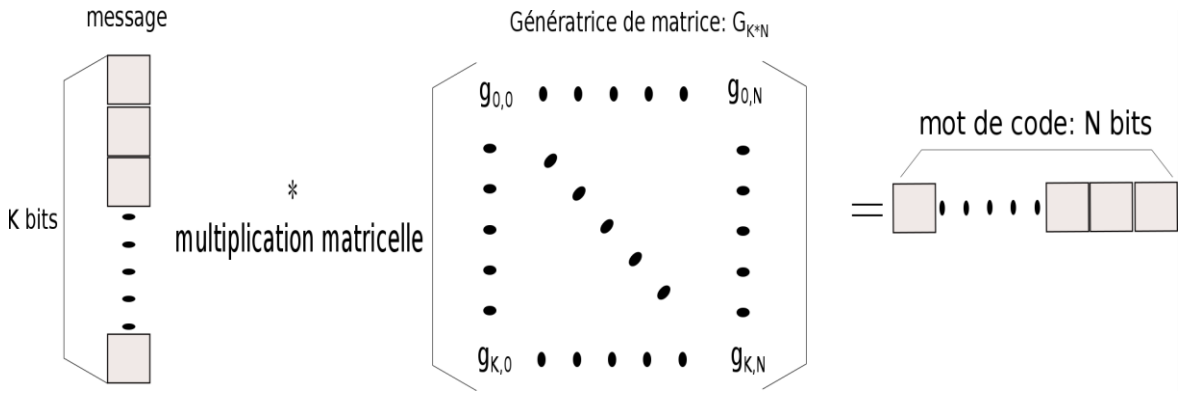


Figure II-16 : Codage en bloc

Cette écriture mathématique très simple montre bien qu'il est beaucoup plus facile d'ajuster le rendement d'un code en bloc comparé à celui d'un code convolutif. En fait, historiquement, on s'aperçoit que la plupart des codes correcteurs d'erreurs sont des codes linéaires. Ces codes linéaires admettent une représentation systématique, donc ce sont donc des codes systématiques. Un code systématique est aussi appelé un code polynomial.

Parmi les codes linéaires, un code est dit cyclique si toute permutation circulaire d'un mot de code est aussi un mot de code. Les codes RS (Reed-Solomon) [63] qui sont très utilisés dans de nombreux contextes sont des codes cycliques. Parmi les autres codes cycliques, on peut citer les codes de Hamming [51], les codes CRC (cyclic redundancy check) [75] et les codes BCH (Bose-Chaudhuri-Hocquenghem)[76][77].

Les codes LDPC (Low Density Parity Check) qui font l'objet d'une grande partie des travaux présentés dans ce mémoire sont des codes en blocs linéaires mais ne sont pas cycliques.

Concernant le décodage, il se construit à partir de la matrice de parité (Parity Check matrix). La matrice de parité est définie de façon à satisfaire la relation :

$$G * H^T_{(N-K,N)} = 0(module\ 2) \qquad (II\text{-}31)$$

Dans cette définition, $H^T_{(N-K,N)}$ correspond à la matrice transposée de la matrice de parité H de taille $(N-K, K)$. Si un mot de code s'exprime sous la forme :

$$C = Message * G \qquad (II\text{-}32)$$

Alors en multipliant à droite l'égalité précédente par la transposée de *H* on peut obtenir :

$$C * H^T = Message * G * H^T = 0\ (modulo\ 2) \qquad (II\text{-}33)$$

Si le produit $C * H^T \neq 0$, c'est qu'il existe des erreurs dans les bits réçus (il y a des chances qu'on ne puisse detecter a ucune erreur alors qu'il y en a plusieurs erreurs dans la séquence mais que, malgré cela, le message apparaît bon dans l'équation de parité), il est important de noter que plus la distance entre deux mots de code est élevée, meilleurs la performance qu'on peut atteindre.

Le résultat de $C * H^T$ s'appelle syndrome, les algorithmes de décodage des codes en bloc sont généralement basés sur le calcul du syndrome. les décodages par syndrome les plus connus

sont l'algorithme d'Euclide [79] ainsi que l'algorithme de Berlekamp Massey [80][81]. Cependant dans le cas des codes LDPC, on utilise un algorithme de décodage itératif basé sur le critère du maximum vraisemblance.

II.9.4.3. Conclusion

Dans cette partie, nous avons introduit les deux principaux types de code FEC (codes en bloc et code convolutif). Le gros intérêt d'un code en bloc par rapport au code convolutif est la possibilité de régler beaucoup plus facilement le rendement du code. Dans le cas d'un code convolutif de rendement 80% ou 90% on remarque une dégradation importante des performances. Ceci explique que dans la suite de ce mémoire nous aborderons les codes en blocs de type RS et LDPC.

II.9.5. Les applications des codes correcteurs d'erreurs

Dans cette partie, nous allons présenter trois types de codes : les codes RS, LDPC et les codes polaires. Les contraintes d'implémentation seront abordées pour chacun d'entre eux.

II.9.5.1. Les codes RS

Les codes RS sont des codes efficaces qui existent depuis des années, ils sont déjà appliqués pour les systèmes longues distances normalisés (ITU-G975, ITU-G709) [83]. Pour évaluer la performance des codes étudiés, nous utiliserons comme critère le gain de codage (NCG : Net Coding Gain). En fait, pour augmenter le gain de codage, deux solutions sont possibles : soit on augmente le SNR, soit on diminue le débit utile en concaténant plusieurs encodeurs [85][84], par exemple, RS (239,223) +RS(255,239).

Les codes de Reed-Solomon (non-binaires) sont obtenus par extension des codes BCH (binaires), les symboles binaires des codes BCH sont transformés en symboles m-aires pour obtenir les codes RS. Ils sont basés sur la théorie des corps de Galois [86], en fait les codes RS sont basés sur le $GF(2^m|m \geq 3)$ et les codes BCH sur $GF(2)$. De plus, ils sont tous des codes linéaires cycliques. Les codes RS sont toujours écrits sous forme (N, K) ou N est relié à la classe du code : $N = 2^m - 1$. Le nombre maximum d'erreurs corrigibles est exprimé par t ou $2t = N - K$.

II.9.5.1.1. Encodage des codes RS

Les codes RS sont des codes en bloc, donc si on les exprime avec une matrice génératrice sous forme polynômiale (G), on obtient :

$$G(x) = (x - \alpha)(x - \alpha^2) \cdots (x - \alpha^{N-K}) \qquad \textit{(II-34)}$$

Où α est une racine primitive du corps $GF(2^m)$.
R étant défini comme le reste de la division euclidienne de la quantité : $message \times X^{2t}$ par G, on peut écrire :

$$C = message \times X^{2t} + R \qquad \textit{(II-35)}$$

Donc la propriété pour ce mot de code est de s'annuler pour $X = \alpha^i$ avec $1 \leq i \leq \alpha$.

II.9.5.1.2. Décodage des codes RS

L'algorithme le plus répandu pour le décodage RS est l'algorithme de Berlekamp-Massey [81][86][80]. Il est basé sur des décisions dures. L'algorithme de décodage est appliqué si le syndrome du code n'est pas nul. Le principe est de trouver parmi tous les mots de code valides celui qui est à la plus petite distance du mot erroné détecté. En effet, cet algorithme procède en quatre étapes (Figure II-17) :

- Calcul du syndrome en prenant $X = \alpha^i$.
- Calcul des polynômes de localisation des erreurs et de l'amplitude
- Calcul des racines et évaluation des deux polynômes
- Sommation du polynôme constitué et du polynôme reçu pour reconstituer l'information de départ sans erreur.

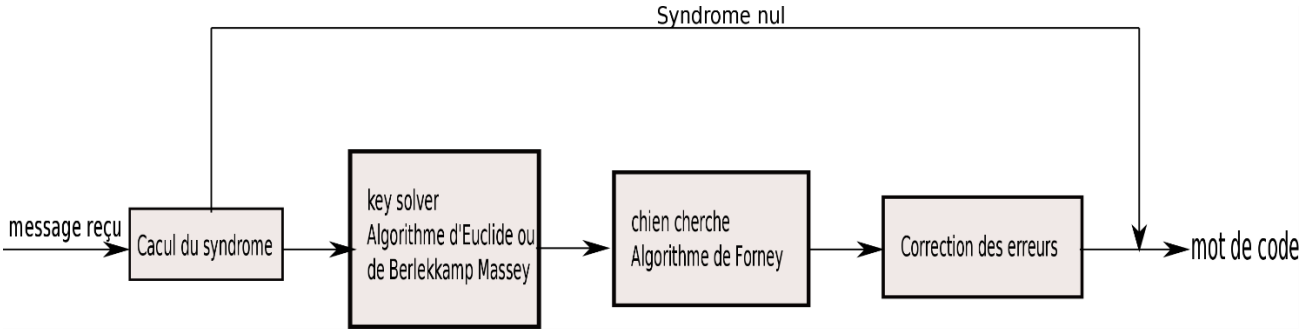


Figure II-17 : Schéma du décodage RS

II.9.5.1.3. Performance

Les codes RS sont des codes linéaires, non-itératifs, et cycliques. On peut obtenir le résultat par simulation ou par calculs mathématiques.

Pour chaque bit, si on définit la probabilité d'erreur corrigible P_{ec} :

$$P_{symbole} = 1 - (1 - P_{ec})^m \qquad (II\text{-}36)$$

Donc la probabilité de non-détection P_{endc} par symbole est déterminée par :

$$P_{endc} = \sum_{i=1}^{N} A_i P_{symbole}{}^{i} (1 - P_{symbole})^{N-i} \qquad (II\text{-}37)$$

Où $\begin{cases} A_0 = 1 \\ A_j = 0, \text{quand } 1 \le j \le N - K \\ A_j = \binom{N}{j} \sum_{h=0}^{j-1-(N-K)} (-1)^h \binom{j}{h} \left[(N+1)^{j-h-(N-K)} - 1\right], \text{quand } (N-K)+1 \le j \le N \end{cases}$

La probabilité d'erreur symbole P_{es} pour le décodage s'exprime :

$$P_{es} \leq \sum_{i=t+1}^{N} \binom{N}{i} P_{symbole}{}^{i} (1 - P_{ec})^{N-i} \qquad (II\text{-}38)$$

Donc la probabilité totale d'erreur par symbole s'écrit :

$$P_{total} = P_{endc} + P_{es} \cong P_{es} = \frac{1}{N} \sum_{j=t+1}^{N} j \binom{N}{j} P_{symbole}{}^{j} (1 - P_{ec})^{N-j} \qquad (II\text{-}39)$$

Si l'on fait une approximation :

$$P_{total} = \frac{t+1}{N} \binom{N}{t+1} P_{symbole}{}^{t+1} (1 - P_{ec})^{N-t-1} \qquad (II\text{-}40)$$

Dans la Figure II-18, on peut tracer les courbes pour les codes (255, K), (240,K).

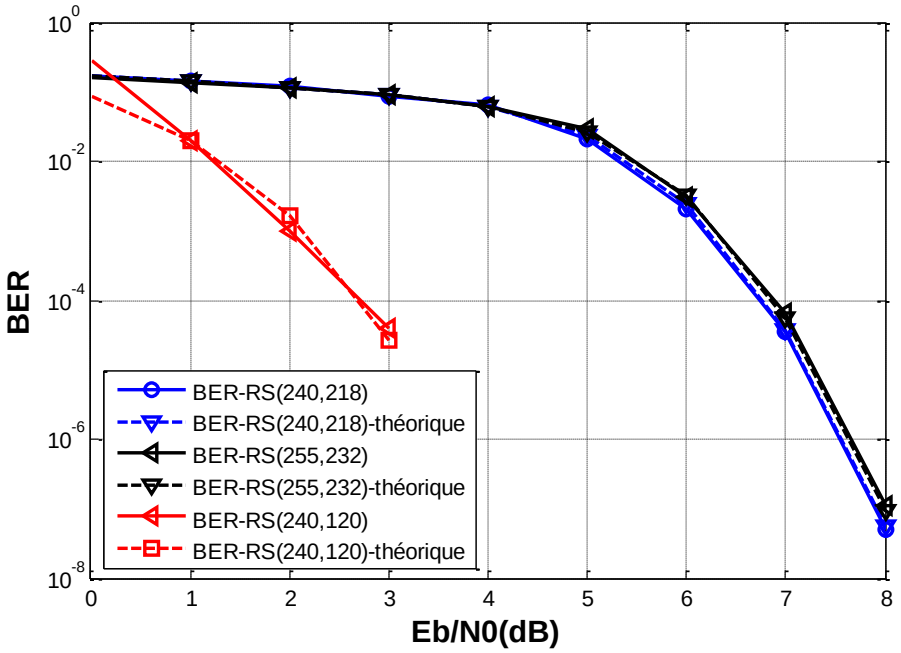


Figure II-18 : Les performances pour différents codes RS

II.9.5.1.4. Conclusion

On peut observer sur la Figure II-18 que, si l'on fixe le rendement du code à 0.9, quand on augmente la taille du bloc, la performance se dégrade. Plus on rajoute des redondances, meilleures sont les performances que l'on peut atteindre, par exemple pour atteindre un TEB de 10^{-7}, le code RS (240, 218) a besoin d'un SNR de 7.8 dB, en revanche, pour RS (255, 232), on a besoin d'un SNR d'environ 8.0 dB. Les codes RS sont déja normalisés et appliqués dans certains domaines télécom (sous-marin). Dans notre contexte, en prenant en compte de contraintes comme la latence, l'optimisation du circuit que nous avons réalisé, et le CAN (convertisseur analogique numérique) que nous allons aborder dans les prochaines parties, le choix de ce code reste à justifier et ce code doit être comparé aux autres en terme de complexité [89][90].

II.9.5.2. Codes polaires

Les codes polaires [67] ont été découverts pour la première fois en 2008 par Arikan, ils sont considérés comme des codes qui atteignent la capacité du canal pour tous les types de canaux connus. Leurs constructions sont basées sur un phénomène de polarisation de canal. Par ailleurs, les opérations de décodage sont implémentées avec une complexité $O(NlogN)$ comparé avec celui des codes RS $O(N^3)$ [87][88].

Il existe quatre paramètres principaux relatifs à l'architecture d'un décodeur : la surface, la latence, le débit et les performances de décodage. A performance et débits équivalents, il semble que la complexité de décodage inférieure présentée par ce type de code en fait un candidat redoutable pour certaines applications de la 5G[91]. Cependant il existe dans la littérature peu d'architectures complètes proposées pour les codes polaires [92][93][94]. Les analyses de ces architectures de décodage montrent que la capacité de mémoire est un point fort contraignant pour l'implémentation quand on augmente la taille des messages. Pour pallier à ce problème, des recherches [95][96][97][98] sont été réalisées pour réduire l'utilisation des ressources, réduire la latence et augmenter le thoughput.

II.9.5.2.1. Encodage des codes polaires

Les codes polaires sont des codes linéaires, ils peuvent être systématiques [99] ou non-systèmatiques. On suppose un code polaire, CP (N, K) , composé de deux parties : $(N-K)$ "0" bits fiables appelés bits gelés (frozen bits) et K bits d'information. Il peut aussi s'exprimer sous forme matricielle : $G_N = F_N = F_2^{\otimes log_2 N}$, d'où $F_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ et $\otimes$ est la puissance de Kronecker.

Si l'on considère le schéma CP[2,2] (Figure II-19) suivant classique :

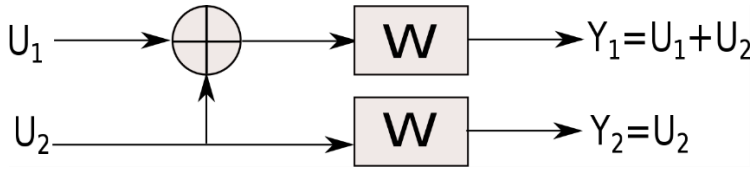


Figure II-19 : Matrice génératrice pour CP[2,2]

Pour la variable U_1, on part des variables d'observation Y_1 et Y_2, on a l'équation : $U_1 = X_1 + X_2$, alors que pour la variable U_2 on a les équations : $U_2 = X_2, U_2 = X_1 + U_1$. On a les représentations :

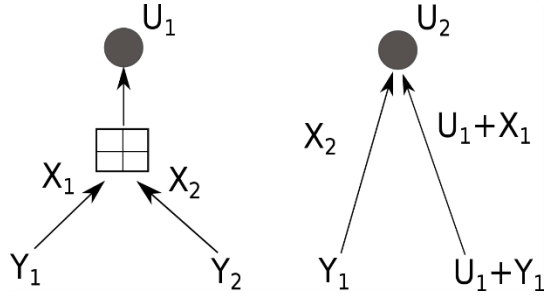


Figure II-20 : représentations des variables U1 et U2

De la même façon, pour CP [4,4] (Figure II-21), la matrice génératrice et le graphe de codage sont donnés par :

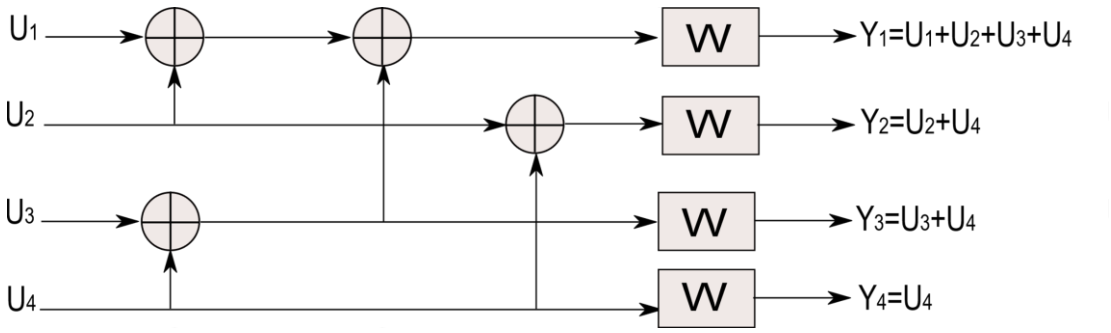


Figure II-21 : Matrice génératrice pour CP [4,4]

Si l'on exprime le résultat sous forme matricielle :

$$G_{CP[4,4]} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \qquad (II\text{-}41)$$

Nous pouvons observer que la matrice génératrice est récursive. Si l'on ajoute de la redondance dans le message, par exemple pour CP [8,4], la matrice génératrice peut s'exprimer comme suit :

$$G_{CP[8,4]} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad (II\text{-}42)$$

Le message donné en entrée s'exprime sous la forme : $Message = [0, 0, 0, U_4, 0, U_6, U_7, U_8]$. Donc $[Y_1 \cdots Y_8]$ peut s'exprimer comme suit :

$$[Y_1 \cdots Y_8]^T = Message \times G_{CP[8,4]} = \begin{bmatrix} U_4 + U_6 + U_7 + U_8 \\ U_4 + U_6 + U_8 \\ U_4 + U_7 + U_8 \\ U_4 + U_8 \\ U_6 + U_7 + U_8 \\ U_6 + U_8 \\ U_7 + U_8 \\ U_8 \end{bmatrix} \qquad (II\text{-}43)$$

Les zéros dans le message sont des 'frozen bits', ces valeurs sont déjà connues par le décodeur dans le récepteur.

On peut aussi remarquer qu'un code de taille $2N$ utilise deux fois la structure d'un code de taille N, par exemple $G_{CP[4,4]} = \begin{bmatrix} G_{\mathrm{CP}[2,2]} & 0 \\ G_{\mathrm{CP}[2,2]} & G_{\mathrm{CP}[2,2]} \end{bmatrix}$. Donc on peut conclure que :

$$U_n = [U_{n/2}, U'_{n/2}] \qquad \textit{(II-44)}$$

$$U_n \times G_n = \left[U_{\frac{n}{2}}, U'_{\frac{n}{2}}\right] \times G_n \qquad \textit{(II-45)}$$

$$= \left[U_{\frac{n}{2}}, U'_{\frac{n}{2}}\right] \times \begin{bmatrix} G_{n/2} & 0 \\ G_{n/2} & G_{n/2} \end{bmatrix}$$

$$= \left[\left[U_{\frac{n}{2}}, U'_{\frac{n}{2}}\right] \times \begin{bmatrix} G_{n/2} \\ G_{n/2} \end{bmatrix}, U'_{\frac{n}{2}} \times G_{n/2}\right]$$

$$= \left[[U_n] \times \begin{bmatrix} G_{n/2} \\ G_{n/2} \end{bmatrix}, U'_{\frac{n}{2}} \times G_{n/2}\right]$$

Avec :

$$\begin{cases} U_{n/2}, les\ n/2\ premiers\ bits\ de\ U_n \\ U'_{n/2}, les\ n/2\ dernieres\ bits\ de\ U_n \end{cases}$$

Plus généralement, on suppose un code polaire N de rendement 0,5, la taille N est composé de n étages de $N/2$ noeuds de parité et $N/2$ noeuds de variable avec degré 3 qui représentent les connections avec les autres nœuds. Si l'on veut augmenter les rendements des codes polaires, il suffit alors de diminuer le nombre des bits gelés (frozen bits).

II.9.5.2.2. Décodage des codes polaires

Le premier algorithme de décodage pour ces codes est à décision dur, appelé aussi annulation successive (SC), cette méthode est la méthode la plus utilisée et nous allons l'introduire dans ce qui suit.

Le deuxième algorithme de décodage est à décision souple appelé SCAN, c'est à dire, au lieu d'exprimer les bits en 0 ou 1, les bits sont introduits en valeurs flottantes. C'est évident que ce décodage consomme beaucoup de ressources dans le circuit car il génère un grand nombre de variables intermédiaires. Cependant, à partir d'une certaine taille de message, cette méthode de décodage des codes polaires s'avère plus performante que l'algorithme de propagation de croyance (Belief Propagation : BP) des codes LDPC.

Une fois qu'on a reçu le message $Y = Y_1^N = [Y_1, \cdots, Y_N]$ à partir du mot de code $X = X_1^N = [X_1, \cdots, X_N]$, le principe de décodage SC est de reconstituer le vecteur U en utilisant la fonction de Y (logarithme de rapport de vraisemblance noté LLR). En gros, le principe de ce décodage

est d'estimer un bit U_i à partir de l'observation du canal en connaissant les bits précédemment estimés.

Tout d'abord, rappelons les deux lemmes de base :

Lemme 1 : soit le schéma de la Figure II-22 :

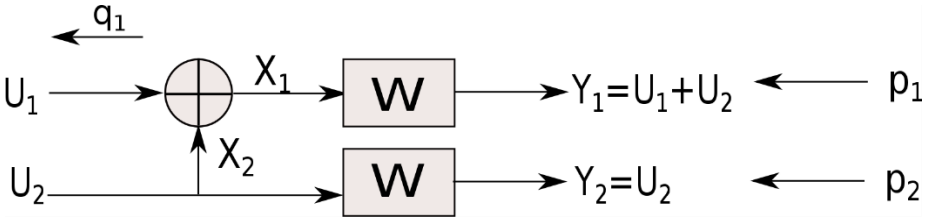


Figure II-22 : Représentation du lemme 1

Si on dispose des deux probabilités suivantes :

$$p_1 = Proba(X_1 = +1|Y_1) \qquad (II\text{-}46)$$

$$p_2 = Proba((X_2 = +1|Y_2) \qquad (II\text{-}47)$$

Alors on a :

$$q_1 = proba(U_1 = +1|Y_1, Y_2) = p_1 + p_2 - 2 * p_1 p_2 \qquad (II\text{-}48)$$

En effet :

$$q_1 = proba(U_1 = +1|Y_1, Y_2) = proba(X_1 \oplus X_2|Y_1, Y_2)$$

$$= Proba(X_1 = 1\ et\ X_2 = 0|\ Y_1, Y_2) + Proba(X_1 = 0\ et\ X_2 = 1|\ Y_1, Y_2) = p_1(1 - p_2) + p_2(1 - p_1)$$

$$= p_1 + p_2 - 2 * p_1 p_2$$

Lemme 2 : soit le schéma de la Figure II-23 :

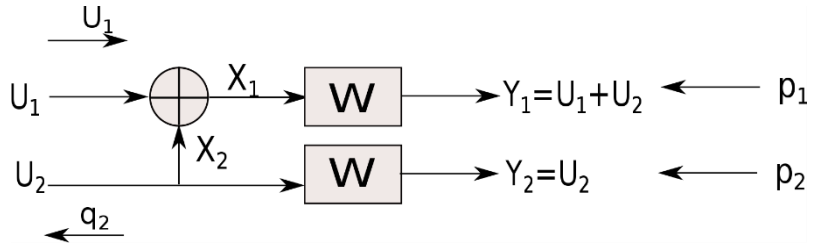


Figure II-23 : Probabilité des entrées en fonction des sorties

Si on dispose des probabilités suivantes :

$$p_1 = Proba(X_1 = +1|Y_1) \quad (II\text{-}49)$$

$$p_2 = Proba((X_2 = +1|Y_2) \quad (II\text{-}50)$$

En connaissant la valeur U_1 on a alors :

$$\begin{aligned} q_2 &= Proba(U_2 = +1|U_1, Y_1, Y_2) \\ &= \frac{p_1 p_2}{p_1 p_2 + (1-p_1)(1-p_2)} \; si \; U_1 = 0 \\ &= \frac{p_1 p_2}{(1-p_1)p_2 + p_1(1-p_2)} \; si \; U_1 = 1 \end{aligned} \quad (II\text{-}51)$$

Ceci permet d'utiliser par exemple la propriété de .

Au niveau des LLR, on a alors deux types de calcul qui peuvent se produire :

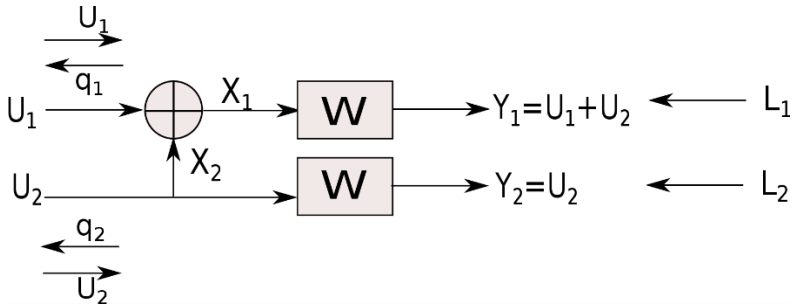


Figure II-24 : les équations exprimées en LLR

$$p_1 = Proba(X_1 = +1|Y_1) = {e^{L_1}}/{1 + e^{L_1}} \quad (II\text{-}52)$$

$$et \; p_1 = Proba(X_2 = +1|Y_2) = {e^{L_2}}/{1 + e^{L_2}}$$

On veut alors calculer :

$$\begin{aligned} L(q_1) &= log\left[\frac{Proba(U_1 = +1|Y_1, Y_2)}{Proba(U_1 = 0|Y_1, Y_2)}\right] \\ &= log\left[\frac{p_1 + p_2 - 2 * p_1 p_2}{1 - p_1 - p_2 + 2 * p_1 p_2}\right] \\ &= log[\frac{e^{L_1} + e^{L_2}}{1 + e^{L_2 + L_1}}] \end{aligned} \quad (II\text{-}53)$$

Les LLRs L_1et L_1sont supposés connus et U_1 est un bit gelé parfaitement connu égal à 0 par exemple. On veut calculer :

$$L(q_2) = log\left[\frac{Proba(U_2 = +1|Y_1, Y_2)}{Proba(U_2 = 0|Y_1, Y_2)}\right]$$

$$= log\left[\frac{e^{L_1} + e^{L_2}}{1 + e^{L_2+L_1}} \Big/ \frac{1}{1 + e^{L_2+L_1}}\right] = L_1 + L_2 \ si \ U_1 = 0 \qquad (II\text{-}54)$$

$$L(q_2) = log\left[\frac{Proba(U_2 = +1|Y_1, Y_2)}{Proba(U_2 = 0|Y_1, Y_2)}\right]$$

$$= log\left[\frac{e^{L_2}}{e^{L_1} + e^{L_2}} \Big/ \frac{e^{L_1}}{e^{L_1} + e^{L_2}}\right] = L_2 - L_1 \ si \ U_1 = 1$$

Donc nous allons maintenant appliquer ces calculs au cas du code à CP (4,2) dessiné dans la Figure II-25.

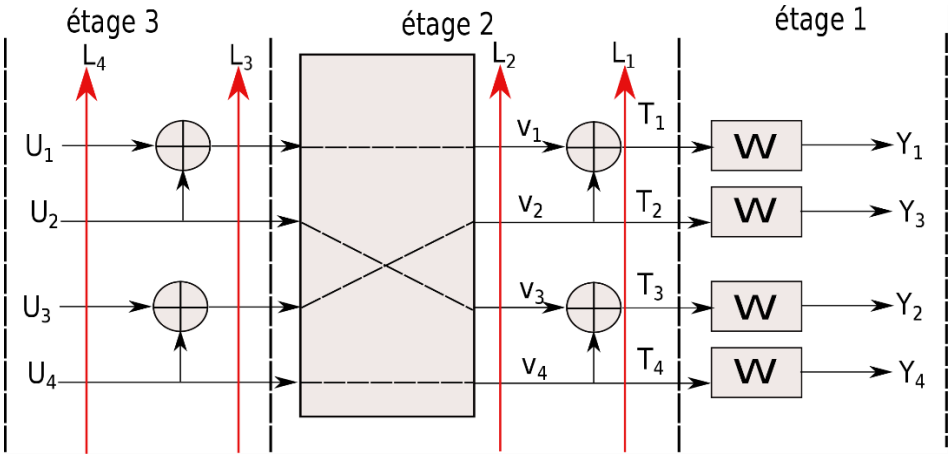


Figure II-25 : Schéma CP(4,2)

Le décodage se compose en trois étages de la droite vers la gauche (étage 1 à étage 3). Les bits gelés sont U_1 et U_3 tous deux égaux à 0. Donc pour les LLRs, on a :

$$L_1^{(n)}(T_{n,n\in N}) = log\left[\frac{Proba(T_n = +1|Y_1, Y_2)}{Proba(T_n = 0|Y_1, Y_2)}\right] \qquad (II\text{-}55)$$

Si on considère que le canal est un canal AWGN, alors :

$$L_1^n = log\left(\frac{1}{\sigma\sqrt{2\pi}} exp^{\frac{-(Y-1)^2}{2\sigma^2}} \Big/ \frac{1}{\sigma\sqrt{2\pi}} exp^{\frac{-(Y+1)^2}{2\sigma^2}}\right) = {2Y}/{\sigma^2} \qquad (II\text{-}56)$$

On peut donc déduire :

$$L_2^{(1)}(v_1) = log[\frac{exp\left(L_1^{(1)}(T_1)\right) + exp\left(L_1^{(3)}(T_3)\right)}{1 + exp\left(L_1^{(1)}(T_1)\right) + exp\left(L_1^{(3)}(T_3)\right)}] \quad \textit{(II-57)}$$

$$L_2^{(2)}(v_2) = log[\frac{exp\left(L_1^{(2)}(T_2)\right) + exp\left(L_1^{(4)}(T_4)\right)}{1 + exp\left(L_1^{(2)}(T_2)\right) + exp\left(L_1^{(4)}(T_4)\right)}] \quad \textit{(II-58)}$$

Puis :

$$L_3^{(2)}(U_2) = L_2^{(1)}(v_1) + L_2^{(2)}(v_2) \quad \textit{(II-59)}$$

$$L_3^{(4)}(U_2) = L_2^{(3)}(v_3) + L_2^{(4)}(v_4) \quad \textit{(II-60)}$$

Avec : $L_2^{(3)} = L_1^{(3)}(T_3)$ et $L_2^{(4)} = L_1^{(4)}(T_4)$. Le décodage de U_2 est possible et donne alors une décision : $\hat{U}_2$. Lorsque l'on a cette décision deux possibilités :

Soit $\hat{U}_2 = 0$ et dans ce cas :

$$L_3^{(3)}(v_3) = L_1^{(1)}(T_1) + L_1^{(3)}(T_3) \quad \textit{(II-61)}$$

$$L_4^{(4)}(U_4) = L_3^{(3)}(v_3) + L_3^{(4)}(v_4) \quad \textit{(II-62)}$$

$$L_3^{(4)}(v_4) = L_1^{(2)}(T_2) + L_1^{(4)}(T_4) \quad \textit{(II-63)}$$

D'où :

$$L_4^{(4)}(v_4) = L_3^{(3)}(v_3) + L_3^{(4)}(v_4) = L_1^{(1)}(T_1) + L_1^{(3)}(T_3) + L_1^{(2)}(T_2) + L_1^{(4)}(T_4) \quad \textit{(II-64)}$$

Soit $\hat{U}_2 = 1$ et dans ce cas :

$$L_3^{(3)}(v_3) = -L_1^{(1)}(T_1) + L_1^{(3)}(T_3) \quad \textit{(II-65)}$$

$$L_4^{(4)}(U_4) = L_3^{(3)}(v_3) + L_3^{(4)}(v_4) \quad \textit{(II-66)}$$

$$L_3^{(4)}(v_4) = -L_1^{(2)}(T_2) + L_1^{(4)}(T_4) \quad \textit{(II-67)}$$

D'où :

$$L_4^{(4)}(U_4) = L_3^{(3)}(v_3) + L_3^{(4)}(v_4) = -L_1^{(1)}(T_1) + L_1^{(3)}(T_3) - L_1^{(2)}(T_2) + L_1^{(4)}(T_4)$$

On peut aussi remarquer ici que, pour déterminer le bit U_4, il faut bien décider la valeur de U_2, si un bit n'est pas bien décidé (erreur de décodage), l'erreur peut se propager dans tous les sens en décodage ce qui peut rendre les résultats erronés.

Le décodage peut alors se représenter sous la forme :

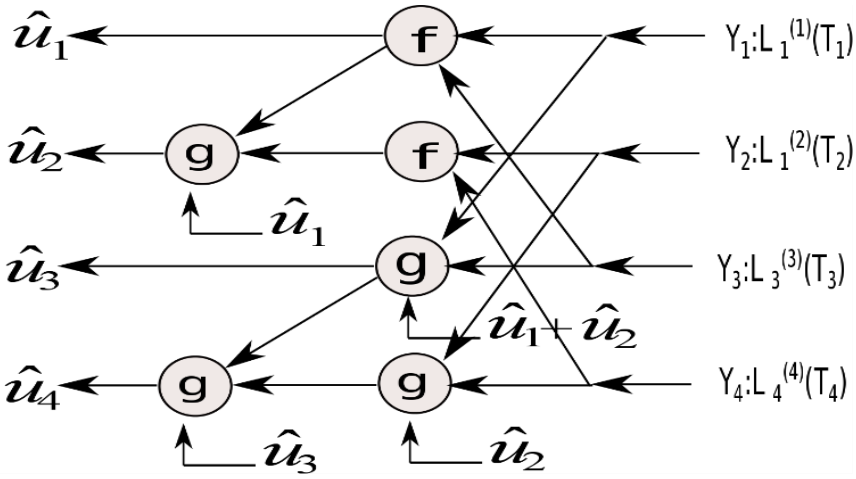


Figure II-26 : g et f fonctions de code polaire

L'opérateur dans la Figure II-26 f correspond à l'opération : $f(a,b) = \log\left(\frac{a+b}{1+ab}\right)$ et l'opérateur g correspond à $g_{\hat{u}_s}(a,b) = (1-2\hat{u}_s)\log(a) + \log(b)$.

Lorsque le décodage met à jours un LLR en dernière étage par exemple à l'étage trois dans la Figure II-25. Le décodeur peut prendre la décision du bit $\hat{U}_n$ tel que :

$$\hat{U}_n = \begin{cases} 1, si\ \ L_n^n > 0 \\ 0, sinon. \end{cases}$$

En plus, les bits gelés sont égaux à 0 peu importe la valeur de ses LLRs dans le dernier étage. Ce décodage (SC) peut être généralisé dans tous les cas, c'est à dire que n'importe quelle taille de code polaire peut être réduite et simplifiée afin d'arriver à l'exprimer comme sur la Figure II-24.

II.9.5.2.3. Performance

Selon [87], les codes polaires peuvent atteindre une très bonne performance à partir d'une taille de paquet de 10000, dans cette partie, les codes polaires $N = 1024, 4096, 8192$ avec rendement 0,5 sont présentés.

Dans la Figure II-27, on peut observer que pour atteindre un TEB (taux d'erreur bit ou BER : bit error rate) de 10^{-3}, le code polaire CP (8196,4096) a besoin de 1,6 dB de SNR (rapport de signal à bruit), à contrario le CP (1024,512) a besoin d'environ 2.3 dB de SNR, il y a donc au moins 0,7 dB de décalage entre ces deux codes CP.

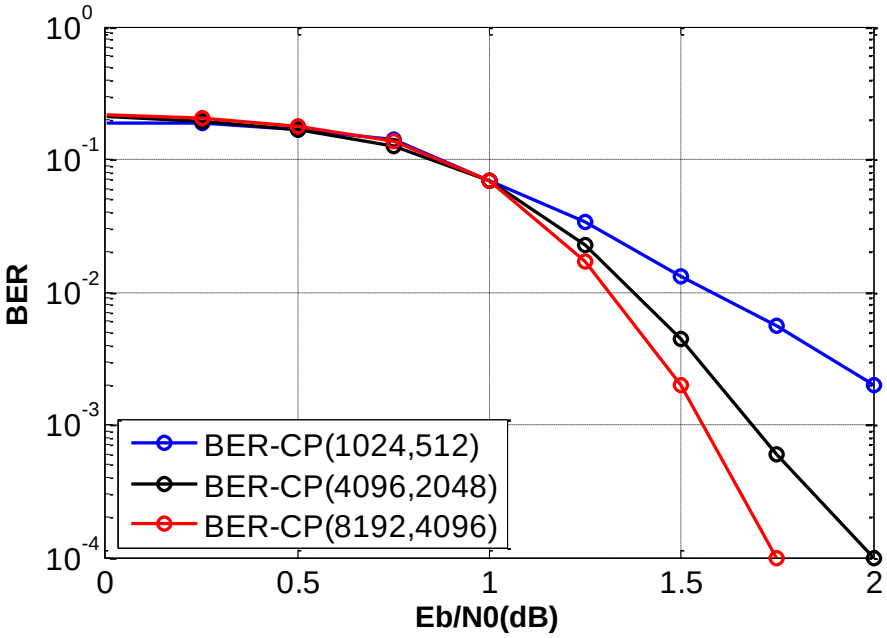


Figure II-27 : Performance des codes polaires avec rendement 0.5

II.9.5.3. LDPC codes

Les codes LDPC (low density parity check) et les principes du décodage itératif par propagation de croyance ont été trouvés et introduits par Gallager [66] en 1962. Ces codes étaient plus ou moins tombés dans l'oubli jusqu'à ce que Mackey et Neal les réhabilitent à la fin des années 90 [50]. Ce sont des codes en bloc linéaires qui permettent de s'approcher des bornes de Shannon de façon aussi efficace que les turbo-codes. Parmi les codes LDPC on peut distinguer de nombreuses sous-familles en particulier les codes binaires et les codes *q*-aires. Normalement, le code LDPC est plutôt présenté sous forme matricielle $[G, H]$ où G est la matrice génératrice du code et H est la matrice de parité qui est creuse (contient beaucoup de zéros et très peu d'un). Ces codes peuvent être des codes sysmatiques à condition de prendre :

$$G = [I_K \;\; P_{N-K}] \qquad (II\text{-}68)$$

$$H = [-P^T{}_K \; I_{N-K}] \qquad (II\text{-}69)$$

Où I sont des matrices identités et *G* et *H* vérifient l'équation : $G \times H \; modulo \; 2 = 0$. Donc si on tranmet un message M avec K bits en utilisant G et H, on a :

$$[M(Kbits) \times G_{K\times N}] \times H^{T}_{(N-K)\times N} modulo\ 2 \qquad (II\text{-}70)$$
$$= M(Kbits) \times [G_{K\times N} \times H^{T}_{(N-K)\times N}] mod\ 2 = 0$$

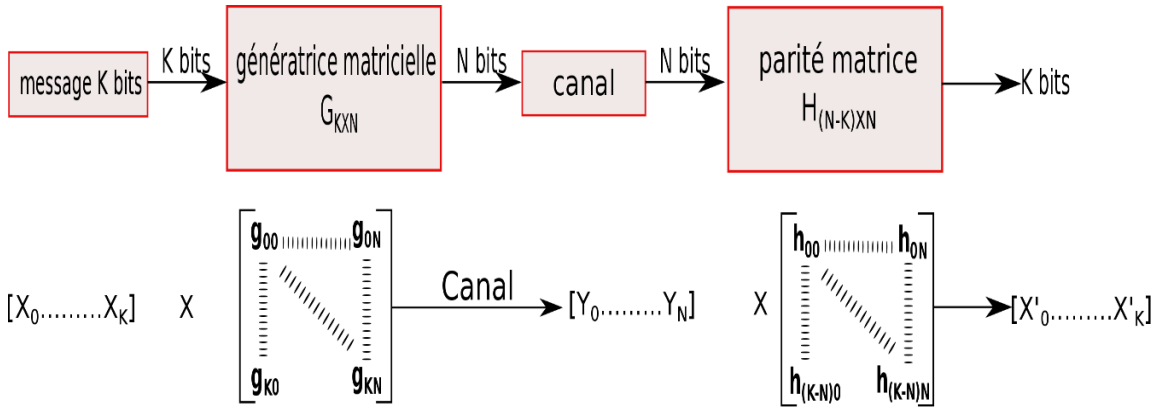


Figure II-28 : Chaîne de transmission pour les codes LDPC

En 1981, Tanner utilise un graphe bipartite pour mieux présenter les relations entre les nœuds de variables et les nœuds de parité, ce graphe est appelé graphe de Tanner. Un graphe de Tanner peut représenter n'importe quel code linéaire. Pour les codes LDPC, on utilise souvent un graphe de Tanner pour représenter la matrice de parité. Par exemple, pour un code $H(3,6)$:

$$H = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Son graphe de Tanner peut-être présenté sous la forme suivante:

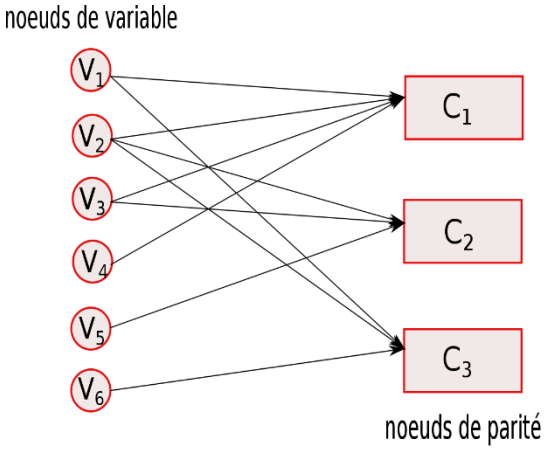


Figure II-29 : Graphe de Tanner

On peut conclure que le nombre des nœuds de variable est égal au nombre de N bits reçus (aussi égal au nombre de colonnes de la matrice H), le nombre des noeuds de parité est égal au nombre des lignes $(N-K)$ de matrice H, si les messages reçus ont des codes de mot, les relations entre les noeuds de variable et les nœuds de parité suivent les équations suivantes en prenant le même exemple en Figure II-29 :

$$\begin{cases} (v_1 + v_2 + v_3 + v_4) \text{ modulo } 2 = 0 = C_1 \\ v_2 + v_3 + v_5 = 0 = C_2 \\ v_1 + v_2 + v_6 = 0 = C_3 \end{cases}$$

Si les équations de parité ne sont pas satisfaites (égales à 0 modulo 2), il existe des erreurs dans les mots reçus, donc il faut le décodage pour les détecter et les corriger.

Aujourd'hui il est reconnu que les codes LDPC sont des codes très performants qui arrivent à corriger un nombre conséquent d'erreurs au cours des itérations. L'élément le plus important dans la réalisation d'un bon code LDPC c'est la matrice *H* qui va conditionner la qualité du décodage itératif. Il est à noter que lorsque la matrice de parité *H* est déterminé on peut en déduire la matrice *G* par la résolution d'un système linéaire d'équations de la forme : $G * H^T = 0$. Il y a cinq paramètres importants à respecter pour obtenir une bonne matrice de parité *H*.

- Le rendement
- La génération de cycles courts (girth)
- La taille
- Le poids des colonnes et des lignes
- L'algorithme de construction de la matrice

II.9.5.3.1. Rendement et taille de la matrice parité

On peut augmenter le rendement d'un code LDPC en ajoutant des 1s dans les lignes de la matrice H. Il faut distinguer le cas des matrices *H* régulières où le poids des lignes et des colonnes est constant et le cas des matrices *H* irrégulières où le poids des lignes et des colonnes varie. Il est montré dans la littérature qu'un code LDPC irrégulier optimisé présente de meilleures performances qu'un code LDPC régulier, cependant sa construction et son optimisation sont assez délicates à mettre en œuvre. Il faut pour cela optimiser les distributions du poids des lignes et des colonnes du code en utilisant l'outil de density evolution (DE) [103][104][105][106][107]. Un exemple d'optimisation peut consister en la recherche de distributions du poids des lignes et des colonnes qui vont maximiser le rendement du code pour un seuil de fonctionnement donné (waterfall region) en termes de rapport signal à bruit. Le gros inconvénient des codes LDPC irréguliers consiste alors en la génération au niveau circuit de la matrice finale *H*. Par contraste, même si les performances sont moindres il est beaucoup plus facile au niveau circuit de générer des matrices *H* régulières et c'est cette approche que nous retiendrons par la suite. Si la matrice de parité est régulière, elle peut être aussi notée (N, d_v, d_c), où d_c répresente le poids d'une ligne et d_v réprésente le poids d'une colonne. En prennant l'example en Figure II-30, H est aussi noté (12,3,6).

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Figure II-30 : Matrice H

Le rendement peut se calculer en fonction de d_v et d_c :

$$R = 1 - {}^{d_v}/_{d_c} \qquad \textit{(II-71)}$$

Pour un code irrégulier on peut utiliser les polynômes générateurs de la distribution des poids des lignes et des colonnes pour calculer le rendement. Si on suppose que la distribution du poids des nœuds de variables se met sous la forme : $\lambda(x) = \sum_{i=1}^{M} \lambda_i x^i$ avec $\sum_{i=1}^{M} \lambda_i = 1$ puisqu'en fait λ_i représente la probabilité d'avoir un nœud de variable de poids *i* et que la distribution du poids des nœuds de parité se met sous la forme : $\rho(x) = \sum_{j=1}^{N} \rho_j x^j$ avec $\sum_{j=1}^{N} \rho_j = 1$ alors on peut montrer que le rendement du code LDPC irrégulier se met sous la forme :

$$R = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} = 1 - \frac{\sum_{j=1}^{N} \frac{\rho_j}{j+1}}{\sum_{i=1}^{M} \frac{\lambda_i}{i+1}} \qquad \textit{(II-72)}$$

Si on se limite au cas des codes LDPC réguliers il est montré qu'un poids de trois par colonne constitue le meilleur choix pour obtenir les performances les plus élevées pour le code.

II.9.5.3.2. Cycle (Girth), trapping set et stopping set

Même si le poids des lignes et des colonnes est fixé il y a encore de nombreux paramètres à régler pour obtenir une matrice *H* performante. Un des principaux critères de construction à respecter consiste à éviter la génération de cycles courts dans le graphe de Tanner équivalent du code. Un cycle représente depuis un nœud de variable donné l'ensemble des nœuds de parité et de variable qui vont lui être connectés jusqu'à ce que l'on retombe sur le nœud de variable de départ. Ainsi sur la Figure II-31 ci-dessous on a représenté un cycle d'ordre 4 dans le graphe de Tanner et sur la Figure II-32 on voit comment ce cycle d'ordre 4 apparaît dans la matrice de parité correspondante. On voit bien sur la Figure II-31 que si l'on part de V_1 on revient à ce même nœud de variable après quatre passages ($V_1 \rightarrow C_2 \rightarrow V_3 \rightarrow C_3 \rightarrow V_1$). Les cycles courts sont très pénalisants car ils font intervenir peu de nœuds intermédiaires et de ce fait les informations extrinsèques qu'ils génèrent au cours du décodage deviennent rapidement fortement corrélées. On appelle girth la longueur de cycle minimale que l'on peut rencontrer dans un graphe de Tanner. Il est évident que plus la valeur du girth sera élevée meilleures seront les performances de décodage du code ainsi constitué. Cependant si l'élimination des cycles d'ordre 4 voire 6 est assez aisée, l'élimination de cycles d'ordre plus élevé est très difficile à programmer.

D'autre part, dans l'optimisation de la matrice *H* d'un code LDPC le critère d'élimination des cycles courts n'est pas le seul à prendre en compte. Il faut également s'occuper d'éliminer le plus possible les configurations correspondants à des trapping set ou à des stopping set [109][110][111][112][113][114][115].

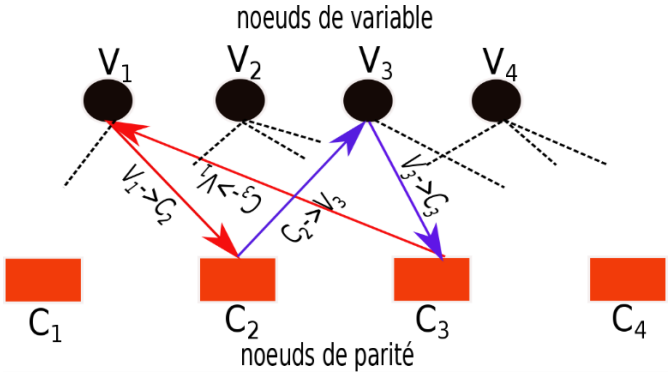


Figure II-31 : présence d'un cycle d'ordre 4 dans un graphe de tanner

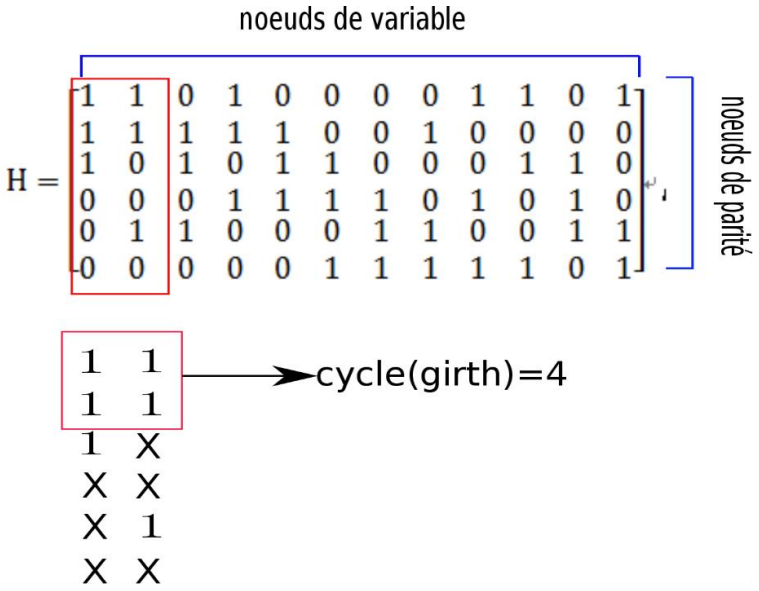


Figure II-32 : cycle 4 dans la matrix H

Sur la Figure II-33 nous montrons comment un cycle court d'ordre 4 peut pénaliser l'algorithme de décodage par propagation de croyance. On peut observer que même si le premier et deuxième nœud de variable sont erronés, avec l'équation de parité (calcul des syndromes) : $Y \times H^T$, on obtient que les valeurs de parité C_1, C_2 sont quand même égaux à 0, en conséquence, on ne peut ni détecter ces erreurs ni corriger ces erreurs.

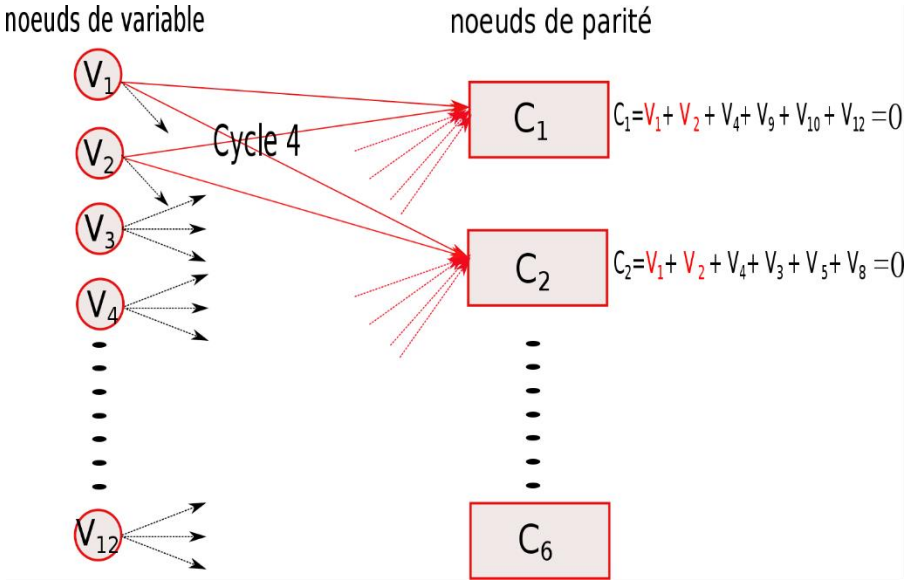


Figure II-33 : cycle dans la Tanner graph

En ce qui concerne les cycles d'ordre 6, nous montrons un exemple sur la Figure II-34 et la Figure II-35.

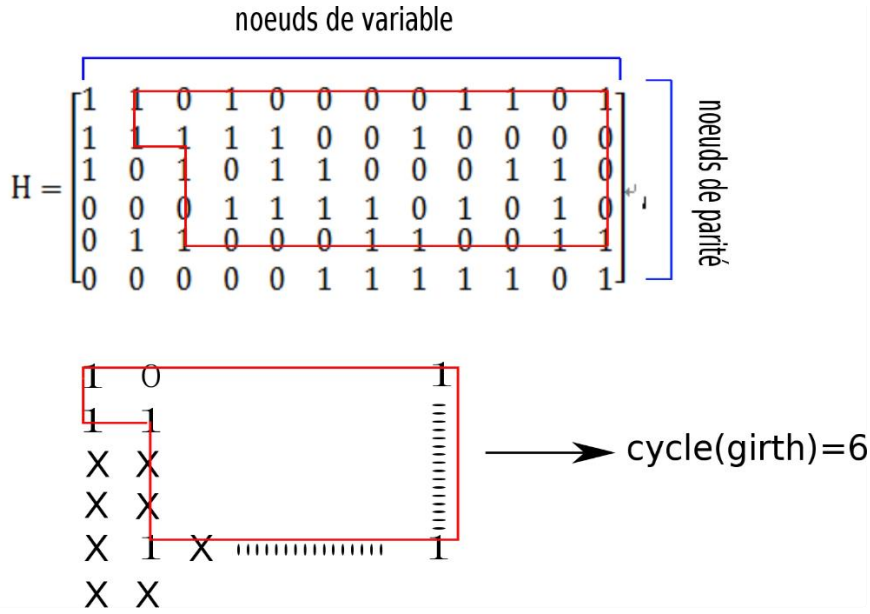


Figure II-34 : Cycle 6 dans la matrice H

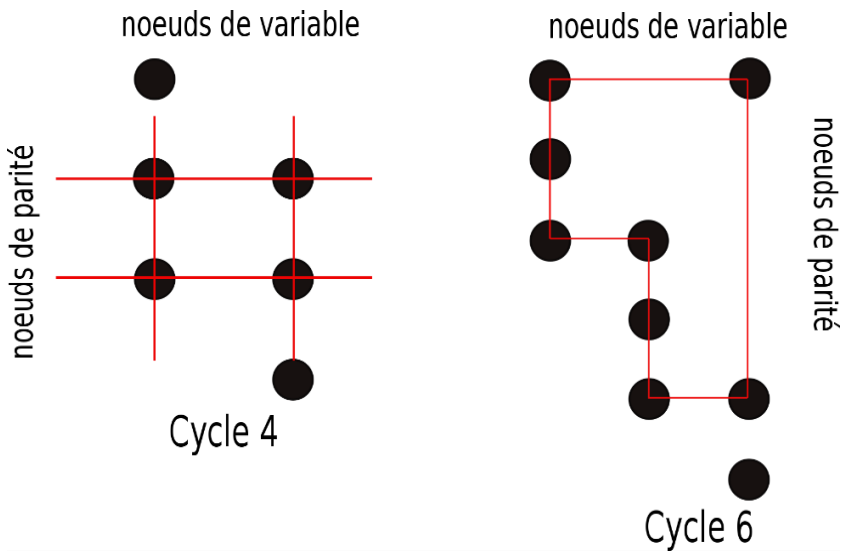


Figure II-35 : Cycle 4 et Cycle 6

Les trapping set et stopping set sont des configurations particulières des nœuds de variable et de parité pour lesquelles l'algorithme de propagation de croyance n'arrive pas à converger et tourne en boucle infinie sans apporter d'information supplémentaire au niveau des informations extrinsèques.

- Trapping set [110]

Lorsque le décodage d'un code LDPC échoue, on appelle un ensemble d'erreurs *T* dans le graphe de Tanner qui n'ont pas convergé vers les bonnes valeurs. On dit que *T* est un trapping set (*a*,*b*) lorsqu'il y a *a* nœuds de variable dans *T* connectés à un nombre *b* de nœuds de parité avec un nombre de connexions impair. Intuitivement, un trapping set est un ensemble de nœuds de variable qui sont mal connectés au reste du graphe dans le sens où cela empêche une bonne convergence de l'algorithme de propagation de croyance.

- Stopping set [116]

Un stopping set *S* est un sous ensemble de l'ensemble des nœuds de variable où tous les voisins des nœuds de variable dans *S* sont connectés à *S* au moins deux fois.

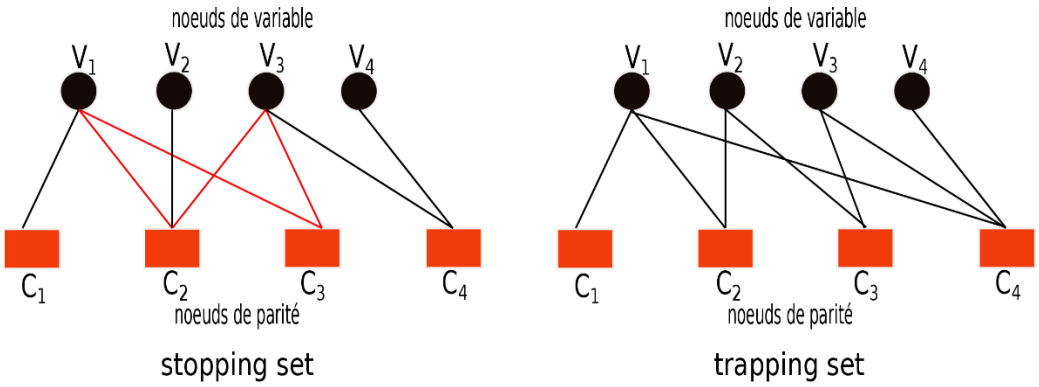


Figure II-36 : Stopping set et trapping set dans le graphe de Tanner

Deux exemples de stopping set et de trapping set sont illustrés sur la Figure II-36 ci-dessus. Le stopping set illustré correspond à un cycle d'ordre 4 (V_1, C_2, V_3, C_3). Pour la figure illustrant le cas du trapping set on peut se rendre compte que si V_2, V_3 et V_4 sont erronés alors les équations de parité seront satisfaites en C_1, C_4 et C_3. Seule l'équation de parité C_2 ne sera pas satisfaite, on a donc un trapping set (3,1) : (V_2, V_3, V_4, C_4). De façon générale, la présence de stopping set et de trapping set se traduit par l'apparition de taux d'erreur plancher à forts SNR's. Il faut donc éliminer le plus possible ces configurations dans la construction de la matrice de parité. En particulier, il est fondamental d'éliminer les cycles courts d'ordre 4 pour obtenir de bonnes performances de décodage.

II.9.5.3.3. Construction de la matrice H

Pour construire la matrice de parité régulière, il existe principalement deux méthodes : la méthode aléatoire et la méthode déterministe. Les méthodes aléatoires les plus connues dans la littérature pour la construction de la matrice *H* sont la méthode de Gallager et la méthode PEG (Progressive Edge Growth). Pour la méthode déterministe, la plus utilisée est la méthode QC-cyclique.

Méthode de Gallager : Gallager a proposé une méthode aléatoire pour construire la matrice H dans [66]. La matrice H se décompose en plusieurs sous-matrices $[H^1 \cdots H^{d_v}]$ avec le nombre de sous-matrice égal à d_c, chaque ligne a $(N - d_c)/d_v$ élements. De plus, chaque sous-matrice a un seul élément égal à 1 dans chaque colonne. Les 1s dans la matrice H^1 forment une figure semblable à celle d'un escalier comme illustré sur la Figure II-37. Les autres sous-matrice sont crées par permutations aléatoires. MacKay et Davey ont aussi montré que la meilleure performance est obtenue quand le poids de chaque colonne est égal à trois.

$$H^1 = \begin{bmatrix} 1\,1\,1\,1 & & & O \\ & 1\,1\,1\,1 & & \\ & & 1\,1\,1\,1 & \\ O & & & \ddots \end{bmatrix}$$

Figure II-37 : méthode de construction de Gallager

Méthode PEG (Progressive Edge Growth): une méthode très efficace pour construire un graphe de Tanner ayant une large girth est donné en utilisant l'algorithme PEG (Progressive Edge-Growth) [117][118][119]. L'algorithme PEG est introduit par Hu et al [117] et permet la création de graphes avec de larges girths. Pour un code de paramètres donnés ci-dessous :

– *N*, le nombre de nœuds de variable.

– *M*, le nombre de nœuds de contrôle.

– $D_{\mathrm{v}} = \left\{ d_{\mathrm{v}_0}, d_{\mathrm{v}_1}, ..., d_{\mathrm{v}_{N-1}} \right\}$, l'ensemble des degrés des nœuds de variable où d_{v_n} est le degré du (n + 1)ème nœud de variable. La construction du graphe de Tanner se fait de manière progressive ; nœud de variable par nœud de variable tout en maximisant le girth localement. Pour chaque nouveau nœud de variable v_n, la création d'une nouvelle branche $E^i_{\mathrm{v}_n}$ se base sur l'expansion du sous graphe de v_n jusqu'à la profondeur *k* où le cardinal de l'ensemble des

nœuds de contrôle présents sur le sous graphe n'évolue plus. Si à la profondeur k tous les nœuds de contrôle figurent dans le sous graphe étendu alors la nouvelle branche est créée par une connexion entre v_n et un nœud de contrôle c_m qui apparaît pour la première fois à cette profondeur k. Ainsi un ou plusieurs cycle(s) de longueur 2 (k + 1) sont créés (Figure II-21). Tandis que, si tous les nœuds de contrôle ne sont pas présents dans le sous graphe étendu alors la nouvelle branche est construite entre v_n et un de ces nœuds non présents. Dans ce cas il n'y a pas de nouveau 2 (k + 1) -cycle créé.

Le sous graphe étendu est appelé arbre des voisins ou encore arbre de calcul et est noté T_{v_n}. L'arbre a 2 (k + 1) niveaux indexés entre [0, 2k + 1], où le 0eme niveau contient uniquement le nœud racine v_n, les niveaux pairs contiennent uniquement des nœuds de variable représentés par ● tandis que les niveaux impairs contiennent uniquement des nœuds de contrôle représentés par ■. L'ensemble des nœuds de contrôle présents aux niveaux $2k_i$ + 1 dans un arbre est noté par $M_{v_n}^{k}$ où $k_i \leq k$. L'ensemble complémentaire à $M_{v_n}^{k}$ est noté $\overline{M}_{v_n}^{k}$. Ainsi l'algorithme original PEG peut être décrit comme suit :

Input : N, M, D_v

Output : G

for n = 0 to N − 1 do

  for i = 0 to $d_{v_n} - 1$ do

    if i = 0 then

    $E_{v_n}^{0} \leftarrow$ branche (c_m,v_n), où $E_{v_n}^{0}$ est la première branche incidente à v_n et c_m est le nœud de contrôle qui a le plus faible degré courant du sous graphe avec les branches existantes : $E_{v_0} \cup E_{v_1} \cup E_{v_2} \cup ... \cup E_{v_{n-1}}$

    end

    else

    étendre, avec les branches existantes, le sous graphe depuis le nœud de variable v_n jusqu'à la profondeur k pour laquelle le cardinal de $M_{v_n}^{k}$ n'augmente plus mais est inférieur à M ou bien $\overline{M}_{v_n}^{k} \neq \phi$ mais $\overline{M}_{v_n}^{k+1} = \phi$. $E_{v_n}^{i} \leftarrow$ branche (c_m,v_n), où $E_{v_n}^{i}$ est la $i^{\text{ème}}$ branche incidente de v_n et c_m est un noeud de contrôle choisi aléatoirement dans l'ensemble des noeuds de contrôle appartenant à $\overline{M}_{v_n}^{k}$ et ayant le plus faible degré.

    end

  end

end

Cette version originale est appelée "greedy" du fait de l'expansion jusqu'à la profondeur k pour laquelle $\overline{M}_{v_n}^{k} \neq \phi$ mais $\overline{M}_{v_n}^{k+1} = \phi$. Hu et al proposent dans [60] une version "non-greedy" qui

consiste à tronquer l'arbre à la profondeur $k_{max} = g_t/2-2$, où g_t est le girth ciblé. Une des particularités de l'algorithme PEG est sa flexibilité. L'algorithme peut être utilisé pour construire des codes réguliers ($d_{v_n} = d_v$) et des codes irréguliers avec des tailles arbitraires. Il y a beaucoup de dérivées du PEG en particulier pour les codes irréguliers dont l'objectif principal est la gestion du profil d'irrégularité afin d'obtenir un code de large girth avec un faible plancher d'erreurs.

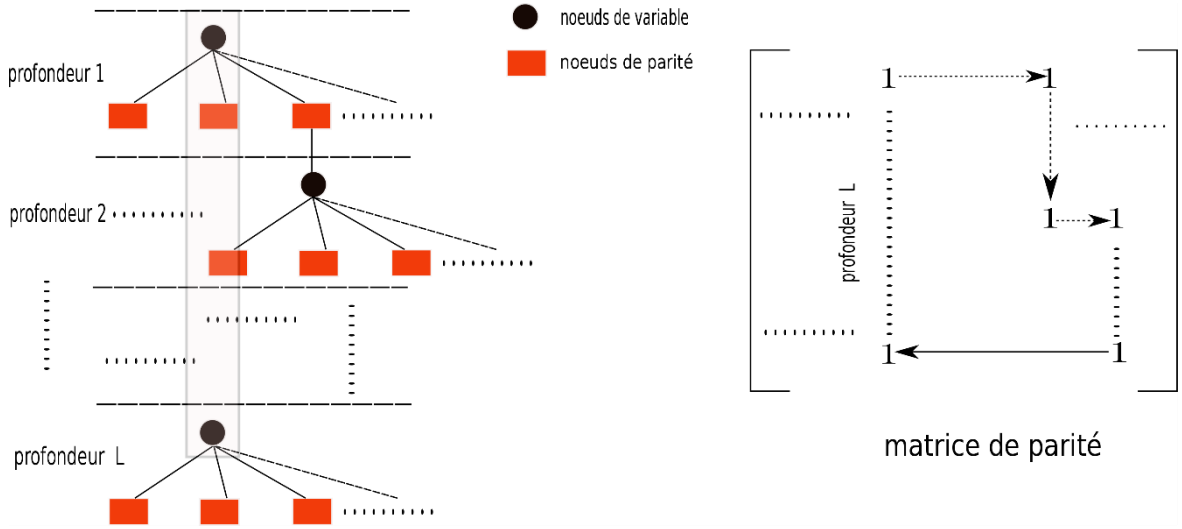


Figure II-38 : PEG algorithme

En utilisant la méthode PEG, les auteurs ont aussi déterminé la borne supérieure atteignable pour le girth quand cette matrice de parité est régulière :

$$g_{sup} \leq 4 * ([t] + 2) \qquad (II\text{-}73)$$

Avec : $t = \frac{\log\left[(M-1)\left(1-\frac{d_v}{d_c(d_v-1)}\right)+1\right]}{\log(d_v-1)(d_c-1)}$ et $M = N - K$.

A partir de cette borne pour le girth on peut également obtenir des bornes pour la distance minimale du code réalisé, on arrive alors à la formule suivante :

$$d_{min} \geq \begin{cases} 1 + \frac{d_s[(d_s-1)^{\frac{g-2}{4}} - 1]}{d_s - 2} \; si \frac{girth}{2} est\ impair \\ 1 + \frac{d_s[(d_s-1)^{\frac{g-2}{4}} - 1]}{d_s - 2} + (d_s-1)^{\frac{g-2}{4}} \; si \frac{girth}{2} est\ pair \end{cases} \qquad (II\text{-}74)$$

La formule illustre bien l'intérêt d'avoir un girth élevé pour obtenir des codes performants.

Méthode QC (quasi-cyclique) : les matrices de parité aléatoire de grande taille permettent de synthétiser des codes dont les performances sont proches des bornes de Shannon. Cependant des matrices aléatoires de grande taille se prêtent mal à une implantation matérielle avec une complexité raisonnable. C'est pour cela que d'autres méthodes basées sur des structures plus régulières ont vu le jour, en particulier les méthodes de génération de codes LDPC quasi-cycliques ont rencontré un vif succès. C'est ainsi que des méthodes de

construction de codes quasi-cycliques font partie de standards de systèmes de télécommunication tels que (802.11, DVB-S2, 802.16e) [120] [121] [122].

Un exemple de matrice de code LDPC quasi-cyclique (QC-LDPC) régulière (J, L) [123] est présenté en Figure II-39, où $1 \leq j \leq J-1, 1 \leq l \leq L-1$, et la taille de paquet est égale à N. $I(p_{j,l})$ represente une matrice de permutation de taille $p \times p$, elle est obtenue par décalage des colonnes de la quantité $p_{j,l}$ de la matrice identité $I(0)$, le paramètre p quant à lui est égal à : $p = N/L$.

$$H = \begin{bmatrix} I(0) & I(0) & I(0) & I(0) \\ I(0) & I(p_{1,1}) & \dots & I(p_{1,L-1}) \\ \vdots & \vdots & \vdots & \vdots \\ I(0) & I(p_{J-1,1}) & \dots & I(p_{J-1,L-1}) \end{bmatrix}$$

Figure II-39 : Matrice d'un code QC-LDPC

Il est évidemment d'autant plus facile d'obtenir une matrice avec un girth élevé quand cette dernière est très creuse. Si on augmente le rendement du code sans changer le poids des colonnes (le poids de la colonne reste toujours égal à 3), selon les formules (2-70), (2-71) on augmente juste le poids de la ligne. Malheureusement, cela aura pour conséquence de rendre la matrice moins creuse, donc la probabilité de faire apparaître des cycles courts augmente (trapping set ou stopping set) et il va falloir trouver un moyen d'éliminer ces cycles. On utilise donc une méthode CE d'élimination de cycles [123]. Le principe de cette méthode se divise en 2 étapes :

Choisir aléatoirement un nœud de variable, on part depuis ce nœud de variable et à la fin on revient sur ce nœud, on examine toutes les possibilités de génération de cycles courts d'ordre 4 ou 6.

Une fois qu'on a identifié les cycles courts il faut les éliminer. Pour cela on va remplacer la matrice circulant $I(p_{j,l})$ qui contient le noeud de variable impliqué dans un cycle court par $I(p_{j,l}+1)$ et puis revenir à l'étape précédente.

La Figure II-40 illustre ces étapes d'élimination des cycles courts.

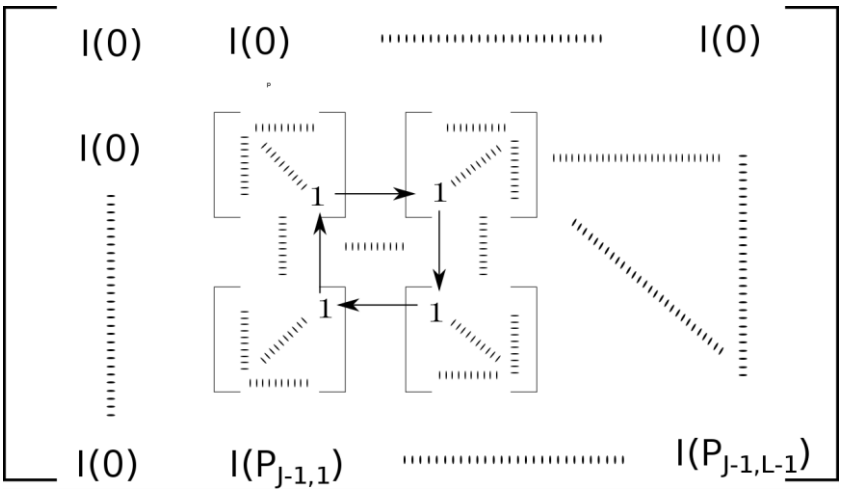


Figure II-40 : Elimination des cycles courts

II.9.5.3.4. Conclusion

Nous avons introduit les paramètres importants qui interviennent dans la conception d'un code LDPC performant. En particulier nous avons montré qu'il était très important d'éliminer les cycles courts. Nous avons alors proposé principalement deux méthodes de construction pour les matrices génératrices de code : la méthode PEG (Progressive Edge Growth) qui permet de construire des matrices aléatoires sans cycle court et les méthodes déterministes telles que la méthode QC-LDPC qui utilise une construction déterministe à partir de matrices de permutation de la matrice identité. Avec ces méthodes, on peut construire la matrice avec des cycles d'ordre 6 (le plus petit cycle ou girth) pour une taille de paquet égale à 1000.

II.9.5.3.5. Décodage souple pour LDPC codes

Globalement, les décodages LDPC se basent sur 2 méthodes : algorithmes à décisions souples (soft decision) qui approximent la solution à Maximum de Vraisemblance nous les qualifierons d'algorithmes optimaux ou bien algorithmes à décisions hard (hard decision) ou algorithmes sous optimaux, dans cette partie nous allons d'abord décrire les algorithmes à décisions souples.

Décodage souple : Cette méthode est basée sur l'échange d'informations probabilistes (informations extrinsèques) entre les nœuds de variable et les nœuds de parité. Elle tente d'approximer au mieux la solution de décodage à maximum de vraisemblance ou ML (Maximum Likelihood). Elle porte plusieurs noms dans la littérature comme algorithme de propagation de croyance (BP : belief propagation) ou algorithme MPA (Message Passing Algorithm). Plusieurs variantes existent pour l'application concrète de l'algorithme selon les simplifications opérées [120][121][122] pour obtenir une implémentation à complexité tolérable. Pour la présentation qui suit on considère un canal AWGN et une modulation de type BPSK.

II.9.5.3.5.1. Notation

Soit *H* la matrice de parité, de taille $m \times n$ à entrées binaires, où $n > m \geq 1$. Le code binaire C est défini par $C \triangleq \{c \in F_2^n : C \times H^T = 0\}$, ou F_2 représente le corps de Galois binaire. Par convention, nous introduisons le code bipolaire $\tilde{C}$ correpondant à C de la manière suivante :

$$\tilde{C} \triangleq \{(1-2c_1, 1-2c_2, \cdots 1-2c_n) : c \in C\}$$

Un canal AWGN à entrée binaire est considéré, il est défini par $y = c + z$ avec $c \in \tilde{C}$. Le vecteur $z = (z_1, z_2, \cdots z_n)$ est un vecteur d'échantillons de bruit blanc gaussien de moyenne nulle et de variance σ^2.

On pose $N(i)$ et $M(j)$ avec $i \in [1,m]$ et $j \in [1,n]$ les ensembles d'indices suivant : $N(i) \triangleq \{j \in [1,n] : h_{ij} = 1\}$ et $M(j) \triangleq \{i \in [1,m] : h_{ij} = 1\}$ où h_{ij} est le ij-ème élément de la matrice de parité H. En utilisant cette notation nous pouvons écrire la condition de parité ainsi : $syndrome = \prod_{j \in N(i)} x_j = 1 \ \forall i \in [1,m]$ ce qui est équivalent à $(x_1, x_1, \cdots, x_n) \in \tilde{C}$.

Remarque : sans modulation, la condition de parité s'écrit : $S(syndrome) = \sum_{j \in N(i)} x_j = 0$.

II.9.5.3.5.2. Algorithme Sum-product (SPA) en log domaine

L'algorithme sum-product algorithme se décompose en 6 parties comme il est montré sur la Figure II-41 : vérification des syndrome, initialisation des valeurs pour les nœuds de variable

et les nœuds de parité, renouvellement des nœuds de variable, renouvellement des nœuds de parité, prise de décision sur les bits, vérification des syndromes.

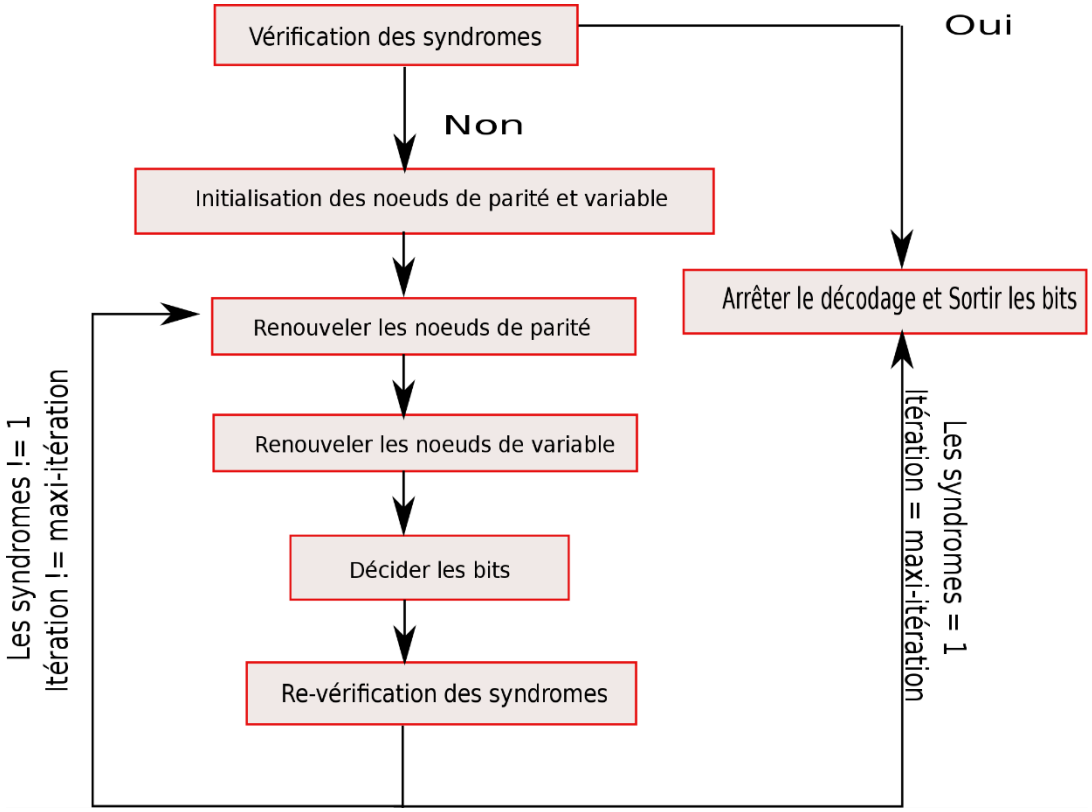


Figure II-41 : Algorithme de décodage BP

Etape 1 : Vérification et Initialisation

Cette étape peut accélérer le décodage, en prenant la décision des messages reçus $y(y_1 \cdots y_n)$), (le seuil de décision vaut toujours 0), on peut traduire ces messages en binaire $x(x_1 \cdots x_n)$. On calcule les sydromes par l'équation $\prod_{j \in N(i)} x_j$, si les syndromes sont équaux à 1, il n'y a pas d'erreurs dans les bits reçus, on arrête le décodage et on obtient le mot de code $C(c_1 \cdots c_n)$, sinon on commence l'étape 2.

Etape 2 : Mise à jour des nœuds de variable

Les nœuds de variable sont initialisés avec les LLRs du canal comme ils ont déjà été calculés pour les codes polaires :

$$V_{n \to m} = L(x|y)_{AWGN} = log \frac{Pr(x=1|y)}{Pr(x=0|y)} = \frac{2y_n}{\sigma^2} \qquad (II\text{-}75)$$

Les nœuds de parité sont initialisés avec 0 :

$$C_n = 0$$

Etape 3 : Mise à jour des nœuds de parité

Les nœuds de parité sont renouvelés en prenant en compte les probabilités issues des nœuds de variable :

$$E_{m\to n}{}^{BP}(x_n) = 2tanh^{-1}\left\{\prod_{n'\in N(m)\backslash n} tanh\left[\frac{V_{n'\to m}(x_{n'})}{2}\right]\right\} \quad \textit{(II-76)}$$

Avec $n' \in N(m)\backslash n$ qui représente l'ensemble des bits (noeuds de variable) liés au noeud de parité m sauf le n^{ème} bit. Les nœuds de variable sont mis à jour selon la formule :

$$V_{n\to m}(x_n) = \frac{2y_n}{\sigma^2} + \sum_{m'\in M(n)\backslash m} E_{m'\to n}{}^{BP}(x_n) \quad \textit{(II-77)}$$

Où $m' \in M(n)\backslash m$ représente l'ensemble des noeuds de parité qui sont liés au noeud de variable n sauf le m^{ème} noeud de parité.

Etape 4 : Décision des bits

On prend une décision dure sur les bits $\hat{x}_n(\hat{x}_1, \cdots, \hat{x}_n)$ selon la règle :

$$\hat{x}_i = \begin{cases} 0, si\ V_i(x_i) \leq 0\ avec\ i \in \{1, \cdots, n\} \\ 1, si\ V_i(x_i) \geq 0\ avec\ i \in \{1, \cdots, n\} \end{cases} \quad \textit{(II-78)}$$

D'où : $V_i(x_i) = \frac{2y_n}{\sigma^2} + \sum_{m\in M(n)} E_{m\to n}{}^{BP}(x_n)$.

Etape 5 : Re-vérification

On recalcule les syndromes par l'équation $\hat{x}_n \times H^T$, si les syndromes sont tous égaux à 1, on arrête le décodage et on sort le mot de code $\hat{x}_n$, sinon on continue l'étape 2. Le décodage va s'arrêter lorsque les syndromes sont tous égaux à 1 ou lorsque le maximum nombre d'itérations est ateint.

Il existe de nombreuses méthodes qui visent à obtenir une implémentation matérielle de l'algorithme BP avec une complexité raisonnable [127][128] mais il semble très difficile de pouvoir l'implémenter sur des architectures matériels vue les opérations faites dans la métrique.

II.9.5.3.5.3. Algorithme Min-sum (MS)

Dans le but d'implémenter l'algorithme BP pour une utilisation circuit, Fossorier a introduit une méthode MS pour approximer le BP algorithme. Comparant l'algorithme MS et BP, on change juste l'étape 3:

Etape 3 : au lieu de calculer $tanh^{-1}$ et $tanh$, on approxime la fonction par :

$$E_{m\to n}{}^{MS}(x_n) = \gamma_m * \mu_{m,min} \quad \textit{(II-79)}$$

Avec : $\gamma_m = XOR\{signe(V_{n'\to m})|\ n' \in N(m)\backslash n\}$

et $\mu_{m,min} = \min\{abs(V_{n'\to m})|\ n' \in N(m)\backslash n\}$

En utilisant l'algorithme MS, on peut mieux comprendre comment les algorithmes à décisions souples permettent de corriger des erreurs de transmission.

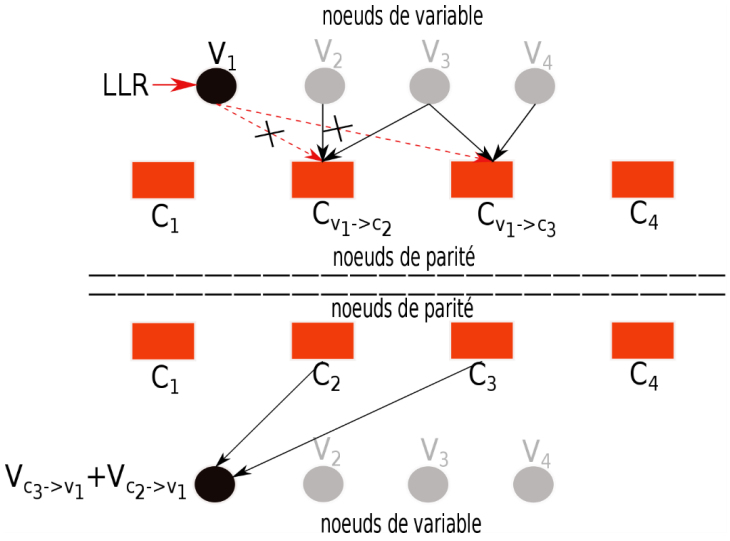


Figure II-42 : Algorithme BP ou message passing

Comme il est montré en Figure II-42, les algorithmes à décodage souple (BP) utilisent les bits plus fiables pour corriger les bits moins fiables, si V_1 est erroné, il suffit d'utiliser les bits corrects V_2 et V_3 pour le corriger, si V_1 et V_2 sont tous erronés, dans ce cas, il faut d'abord utiliser les autres bits corrects pour corriger V_2, et ensuite on peut corriger V_1. Cependant, quand il y a des cycles courts dans la matrice, ils empêchent la convergence. Comme on a montré en Figure II-43, l'algorithme converge toujours à partir des nœuds de parité et de variable plus fiables vers les nœuds de parité et de variable moins fiables.

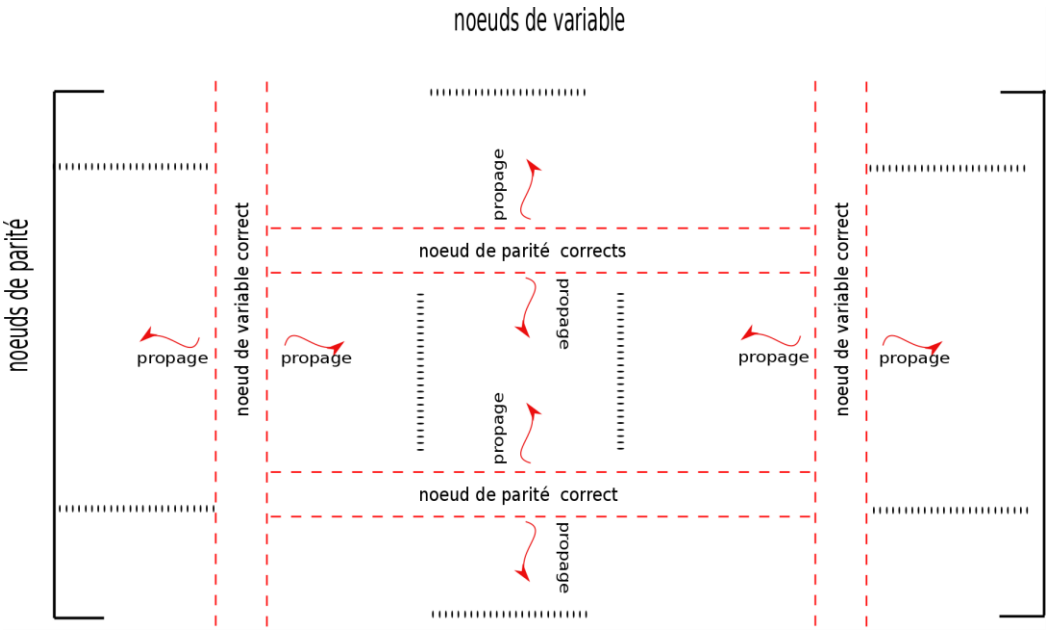


Figure II-43 : Propagation des noeuds corrects

En appelant, ${E_{m\to n}}^{MS}(x_n)$ la vitesse de convergence de l'algorithme Min-Sum, on peut montrer que : ${E_{m\to n}}^{MS}(x_n)$ est toujours inférieur à ${E_{m\to n}}^{BP}(x_n)$. Donc avec cette approximation, la performance de MS est dégradée par rapport à celle de l'algorithme BP.

Pour faire en sorte que la valeur de ${E_{m\to n}}^{MS}(x_n)$ se rapproche de celle de ${E_{m\to n}}^{BP}(x_n)$, il existe beaucoup de méthodes comme l'algorithme MinSum normalisé (NMS) ; MS avec offset (OMS) [129] ; Modifié MS avec offset (MOMS) [130][131] ; self-corrected MS(SCMS) [132] etc...

Par exemple, pour NMS et OMS, on a défini le facteur normalisé $1/\alpha$ et le facteur d'offset β :

$$ {E_{m\to n}}^{NMS} = \alpha * {E_{m\to n}}^{MS} \qquad (II\text{-}80) $$

$$ {E_{m\to n}}^{OMS} = max\{abs({E_{m\to n}}^{MS}) - \beta, 0\} \qquad (II\text{-}81) $$

Avec :

$$ \alpha = E[{E_{m\to n}}^{BP}]/E[{E_{m\to n}}^{MS}] \qquad (II\text{-}82) $$

$$ \beta = E[{E_{m\to n}}^{BP}] - E[{E_{m\to n}}^{MS}] \qquad (II\text{-}83) $$

Où $E[{E_{m\to n}}^{BP/MS}]$ est l'espérance des valeurs ${E_{m\to n}}^{BP/MS}$.

D'après les articles, la meilleure variante de l'algorithme MS permet de s'approcher très près de la performance BP.

II.9.5.3.5.4. Performances

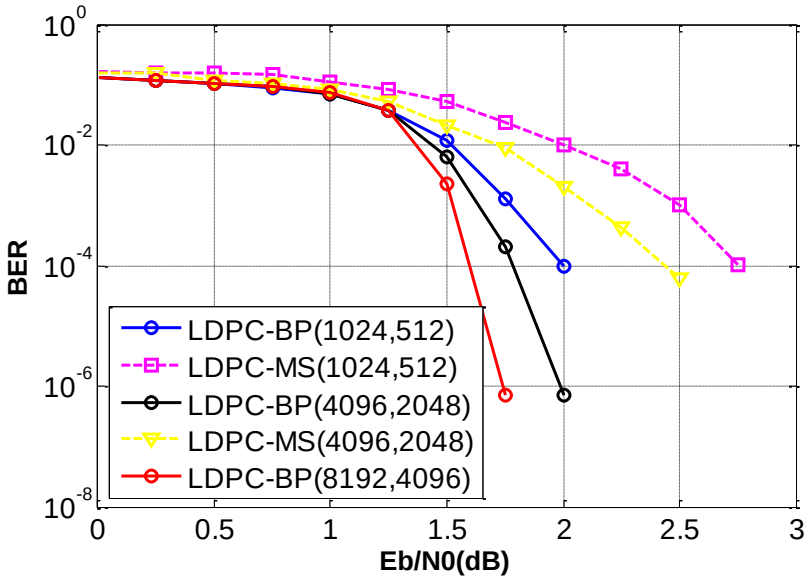


Figure II-44 : Performances des algorithmes BP et MS avec différents tailles de matrices

Nous illustrons les performances des codes LDPC sur la Figure II-44. Sur chaque courbe le nombre d'itérations est fixé à 100. Les matrices synthétisées ont chacune un girth ≥ 6. On voit évidemment l'influence de la taille des matrices, les performances s'améliorent quand la taille des matrices augmente. Par exemple, pour atteindre un taux d'erreur TEB 10^{-4}, le code $(8192, 4096)$ a besoin de 1.6 dB de SNR, le code $(4096, 2048)$ a besoin de 1.8 dB de SNR, et le code $(1024, 512)$ a besoin de 2 dB de SNR, il y a donc à chaque fois 0.2 dB d'écart entre eux. Selon la Figure II-44 on peut aussi observer la dégradation de la méthode MS par rapport à BP, pour atteindre TEB 10^{-4}, l'algorithme MS avec la taille $(4096, 2048)$ a besoin de

2.4 dB de SNR, tandis que le même TEB est atteint pour un SNR de 1.8 dB avec l'algorithme BP. On constate toujours à peu près un écart compris entre 0.6 et 0.8 dB entre les algorithmes MS et BP.

Sur la Figure II-45 nous avons reporté à titre de comparaison les performances des codes LDPC-BP et nous les avons comparés à celles des codes polaires (CP) pour un rendement de 0.5. On peut constater que même si les performances sont très proches (moins de 0.2 dB d'écart) pour les codes (8192,4096) le code LDPC-BP l'emporte à chaque fois et cet écart grimpe à 0.4 dB pour les codes (4096,2048) toujours pour un TEB cible égal à 10^{-4}.

Sur la Figure II-46, nous nous sommes placés dans un contexte de transmission optique (C-RAN) avec un rendement de code fixé à 0.9, une taille de paquets de l'ordre de 1000 bits et nous utilisons l'algorithme de décodage min-sum (MS). Nous rajoutons aussi sur la courbe les performances d'un code de Reed-Solomon (240-218). On voit que la meilleure performance est obtenue par le code LDPC qui affiche un gain de 0.7 dB par rapport au code polaire et un gain supérieur à 2 dB par rapport au code RS (240-218).

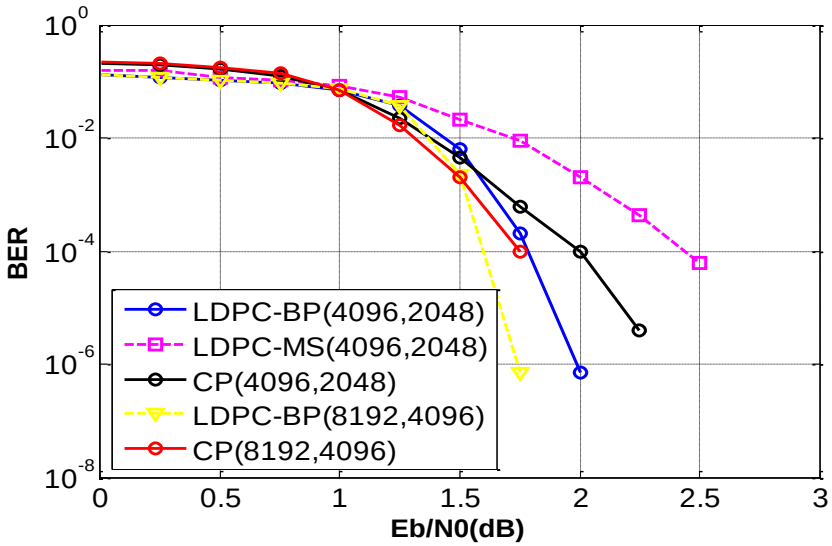


Figure II-45 : Comparaison des performances entre codes LDPC et codes polaires

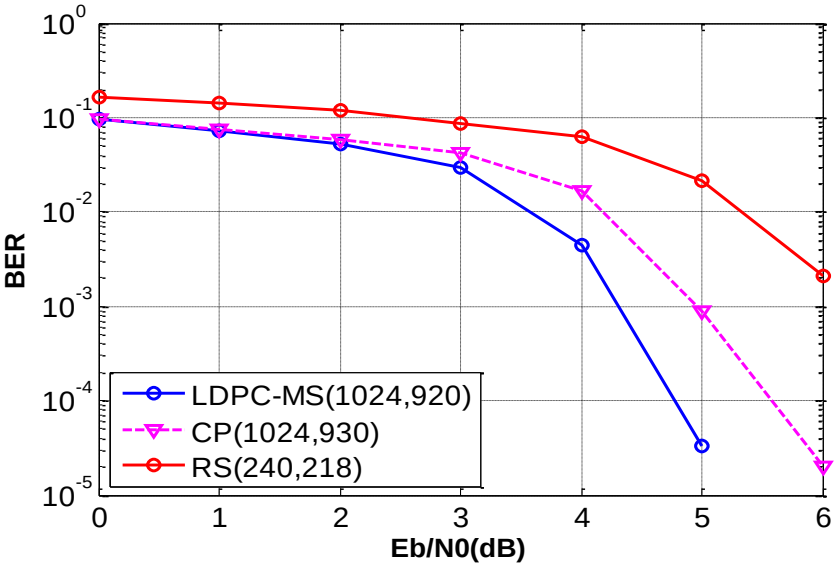


Figure II-46 : Comparaison des performances entre MS, CP, et RS

II.10. Conclusion

Dans ce chapitre, nous avons introduit les concepts de base en communications numériques : capacité d'un canal de communication et description des principaux codes correcteurs d'erreur qui permettent de s'approcher de la capacité d'un canal. Le contexte traité pour la capacité du canal est celui du C-RAN, nous avons calculé le rendement maximum que l'on peut atteindre pour un rapport signal à bruit fixé. En ce qui concerne la description des codes, l'accent a été mis sur trois types de code : les codes polaires, les codes de Reed-Solomon et les codes LDPC. Du point de vue performances, on constate que pour des paquets de grande taille les codes polaires l'emportent sur les codes LDPC tandis qu'à faible et moyenne tailles ce sont les codes LDPC qui passent devant. De plus, d'un point de vue implémentation circuit, nous avons montré l'intérêt d'utiliser l'algorithme Min-Sum pour simplifier l'algorithme de propagation de croyance développé à l'origine. Ce dernier algorithme Min-Sum ou MS présente une excellente performance. Pour la suite d'étude, on va essayer de trouver une méthode à décision dure pour approcher la performance de l'algorithme MS afin de l'implémenter plus facilement sur une plateforme FPGA.

Chapitre III

Algorithmes à décisions dures pour codes LDPC

Chapitre III. Algorithmes à décisions dures pour codes LDPC

III.1. Introduction

Comme on l'a montré dans le chapitre 2, les codes LDPC présentent des pouvoirs de correction importants qui les rendent très attractifs pour une utilisation sur des canaux fortement perturbés. Il existe deux grandes familles de décodage pour les codes LDPC : décodage à décisions dures et souples (Figure III-1). La méthode qui donne les meilleures performances est la méthode de décodage de type à décisions souples, elle est basée sur le calcul des LLR's et sur l'échange d'informations extrinsèques entre nœuds de variable et nœuds de parité. Dans la pratique et en vue d'une implémentation on doit considérer un nombre de bits de quantification fini ce qui va introduire une perte dans les performances. On montre [133] que si l'on ne veut pas trop dégrader la performance, le nombre minimum de bits pour enregistrer un message des nœuds de variable ou des nœuds de parité doit être supérieur à 4. Dans notre contexte de transmissions à très hauts débits où la latence joue un rôle important le décodage itératif à décisions souples peut introduire des délais prohibitifs, c'est pourquoi nous allons nous orienter dans ce chapitre vers les algorithmes à décisions dures. Cependant nous essaierons de définir des algorithmes novateurs capables de s'approcher des performances des algorithmes à décisions souples.

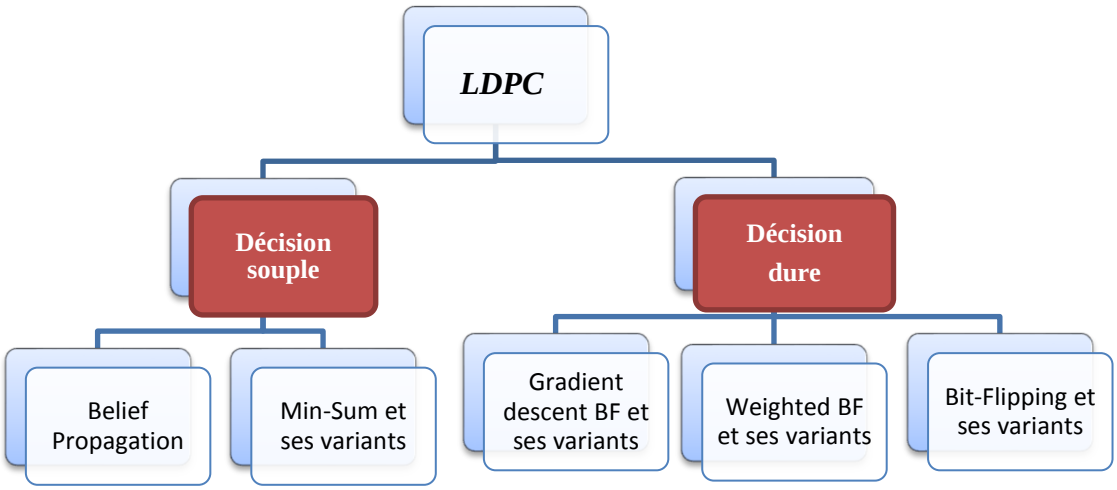


Figure III-1 : Familles de code LDPC

III.2. Algorithmes à décisions dures

Comme on a montré dans le chapitre précédent, les algorithmes à décisions souples utilisent les LLRs pour évaluer les fiablités des messages reçus. Pourtant, Gallager a proposé un algorithme BF (bit-flipping) dans sa thèse de doctorat qui travaille en décisions dures. On appele la fonction qui décide les inversions des bits la fonction d'inversion noté Δk. L'équation, Δk pour la méthode d'inversion de Gallager s'exprime par :

$$\Delta k^{BF} \triangleq \sum_{i \in M(n)} \prod_{j \in N(i)} x_j \qquad \textit{(III-1)}$$

Cette méthode a beaucoup simplifié les procédures par raport aux algorithmes souples. Le problème pour cette méthode est l'existence des cycles dans la matrice de parité H, S'il n'y a pas de cycle dans la matrice de parité H(cycle-free), l'algorithme BF est suffisant pour corriger toutes les erreurs. Au contraire, s'il y a des cycles dans la matrice H, Δk^{BF}n'arrive pas de detecter et corriger toutes les erreurs et les performances de cet algorithme sont très loin de celles obtenues par les méthodes souples itératives. Une première idée pour résoudre ce problème est d'ajouter des pondérations sur les syndromes ou signaux reçus dans la formule de Δk^{BF} afin de casser les cycles et de bien corriger les bits incorrects. Ainsi, comme nous l'avons illustré en Figure III-2, les bits x_1, x_2, x_3 sont des bits dans un cycle de 6, utilisant la formule de Δk^{BF}, on n'arrive pas détecter ces bits. Si on ajoute une pondération sur les syndromes $\sum_{i \in M(n)} \prod_{j \in N(i)} x_j$ ou les signaux reçus x_n, y_n, on peut probablement détecter et inverser le bit x_1 dans la prémière itération, celui-ci casse le cycle 6 et l'indication de la formule $\sum_{i \in M(n)} \prod_{j \in N(i)} x_j$ va détecter et corriger x_2, x_3 en deux itérations. Ces méthodes [134]-[155] sont connues pour éliminer l'impact des cycles afin d'améliorer la performance de décodage.

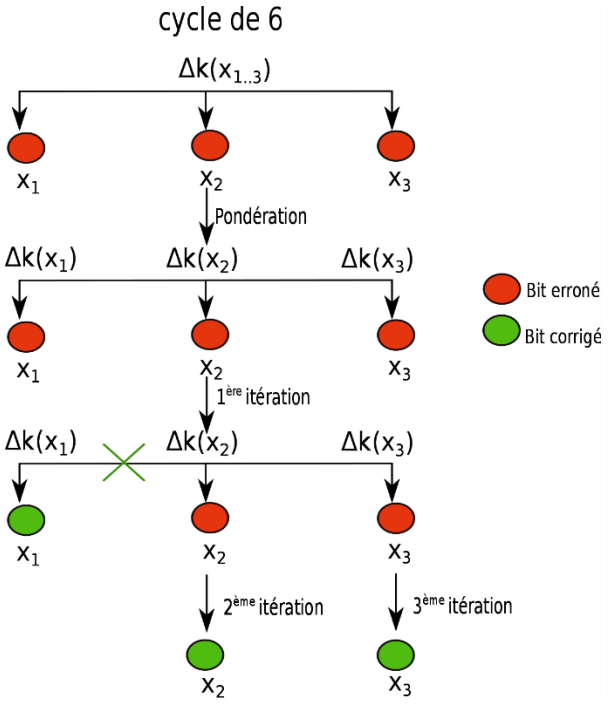


Figure III-2 : Exemple d'élimination des cycles 6

On conclut que la fonction d'inversion remplace les calculs de LLR dans les algorithmes durs pour évaluer les fiabilités des messages reçus. La fonction d'inversion peut aussi s'écrire sous la forme :

$$\Delta k^{général}(x) \triangleq \alpha * \emptyset(x_n, y_n) + \sum_{i \in M(n)} \omega_i * \prod_{j \in N(i)} x_j \qquad \textit{(III-2)}$$

Où $\emptyset(x_n, y_n)$ représente la métrique (fiabilité) des messages reçus y_k, x_k repréente la valeur après décision sur y_k, α et ω sont des valeurs pour ajuster les valeurs entre $\emptyset(x_n, y_n)$ et le syndrome $\sum_{i \in M(n)} * \prod_{j \in N(i)} x_j$.

La fonction d'inversion peut s'exprimer à l'aide du graphe de Tanner. Comme montré sur la Figure III-3, d'abord, les nœuds de variable x_k envoient des valeurs vers les noeuds de parité, ensuite ces valeurs des noeuds de parité vont deviner quel sont les noeuds de variables qui sont les plus probables d'être erronés en effectuant les calculs de syndromes. Pour cela on regarde quels sont les bits qui apparaissent le plus souvent dans les équations de parité erronées. Par exemple, ici C_2 et C_3 sont des noeuds de parité erronée, et on trouve que x_1 et x_3 sont les plus probables d'être erronés. La fonction d'inversion se fait par la sommation de $\emptyset(x_n, y_n)$ et du syndrome afin de considérer quels est le bit qu'il faut inverser.

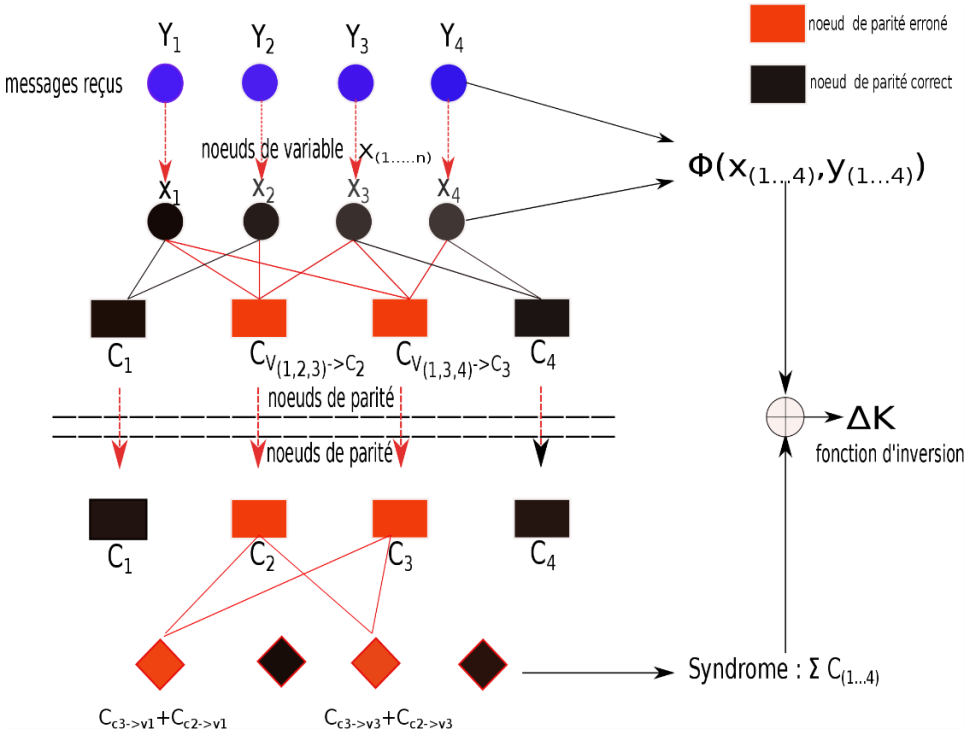


Figure III-3 : Fonction d'inversion

Pour réduire la complexité et le délai dans l'implémentation sur FPGA, au lieu d'inverser un seul bit qui a la valeur minimale de Δk dans une itération, on peut inverser plusieurs bits pour converger vers un décodage plus rapide. Donc il faut définir un seuil (threshold) θ tel que :

$$\begin{cases} si\ \Delta k^{XBF} < \theta, x_k = -x_k \\ sinon, x_k = x_k \end{cases}$$

L'impact du seuil pour la fonction d'inversion doit être discuté aussi. Figure III-4 montre un exemple des valeurs reçues qui sont calculées par fonction d'inversion, si le seuil (threshold) est pris à 0, les bits qui ont les valeurs négatives sont considérées comme des bits érronés, si le seuil est pris à 1, il y aurait cinq bits qui peuvent être inversés dans une itération.

L'inconvénient pour ceci est que l'on peut créer des erreurs imprévisibles qui vont faire diverger l'algorithme, surtout si le nombre de bits à inverser en une seule fois est assez important.

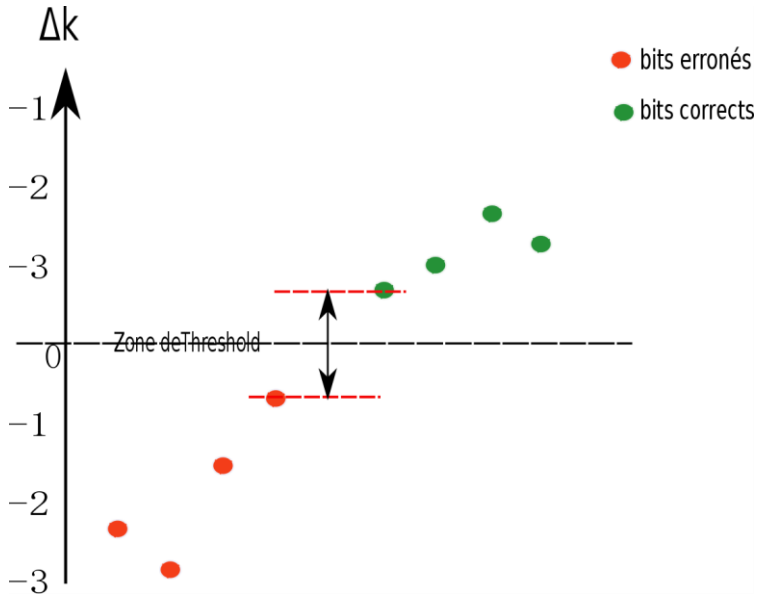


Figure III-4 : Exemple des valeurs reçus obtenus par la fonction d'inversion

On peut détecter et corriger ces bits avec un nombre d'itérations assez faible en utilisant cette méthode. En plus, parfois, l'algorithme d'inversions multiples peut éliminer l'impact des cycles dans la matrice de parité aussi.

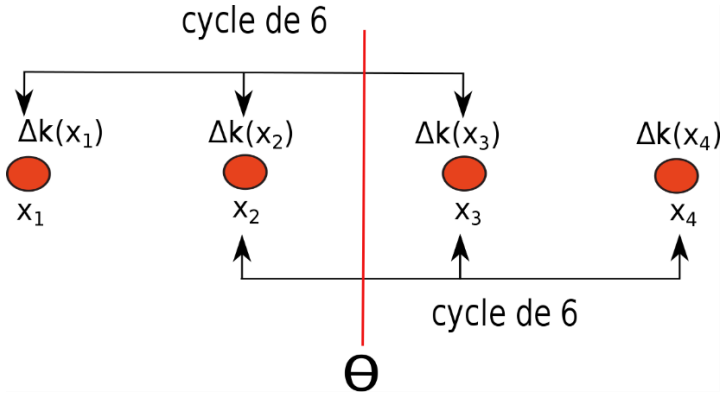


Figure III-5 : Exemple d'élimination d'un cycle de longueur 6 en utilisant la méthode d'inversions multiples

Un exemple est montré en Figure III-5, dans la zone des faibles SNR. Si on corrige juste un bit parmi les 4 bits x_1, x_2, x_3, x_4, ceci peut générer les autres erreurs qui vont probablement empêcher le décodage. Si on arrive à fixer un seuil entre x_3 et x_2, c'est à dire x_1 et x_2 vont être corrigés dans un premier temps et x_3, x_4 vont être détectés par l'indication de $\sum S_i$ et corrigés lors des prochaines itérations. Ce seuil a permis d'éviter la divergence du décodage, et a réduit le temps de décodage.

De même que pour les algorithmes souples, tous ces algorithmes peuvent être représentés sous forme graphique comme le montre le schéma en Figure III-6. Dans la prochaine partie,

on va présenter trois algorithmes principaux (BF, WBF, surtout GDBF) et ses variantes pour les algorithmes à décisions duress.

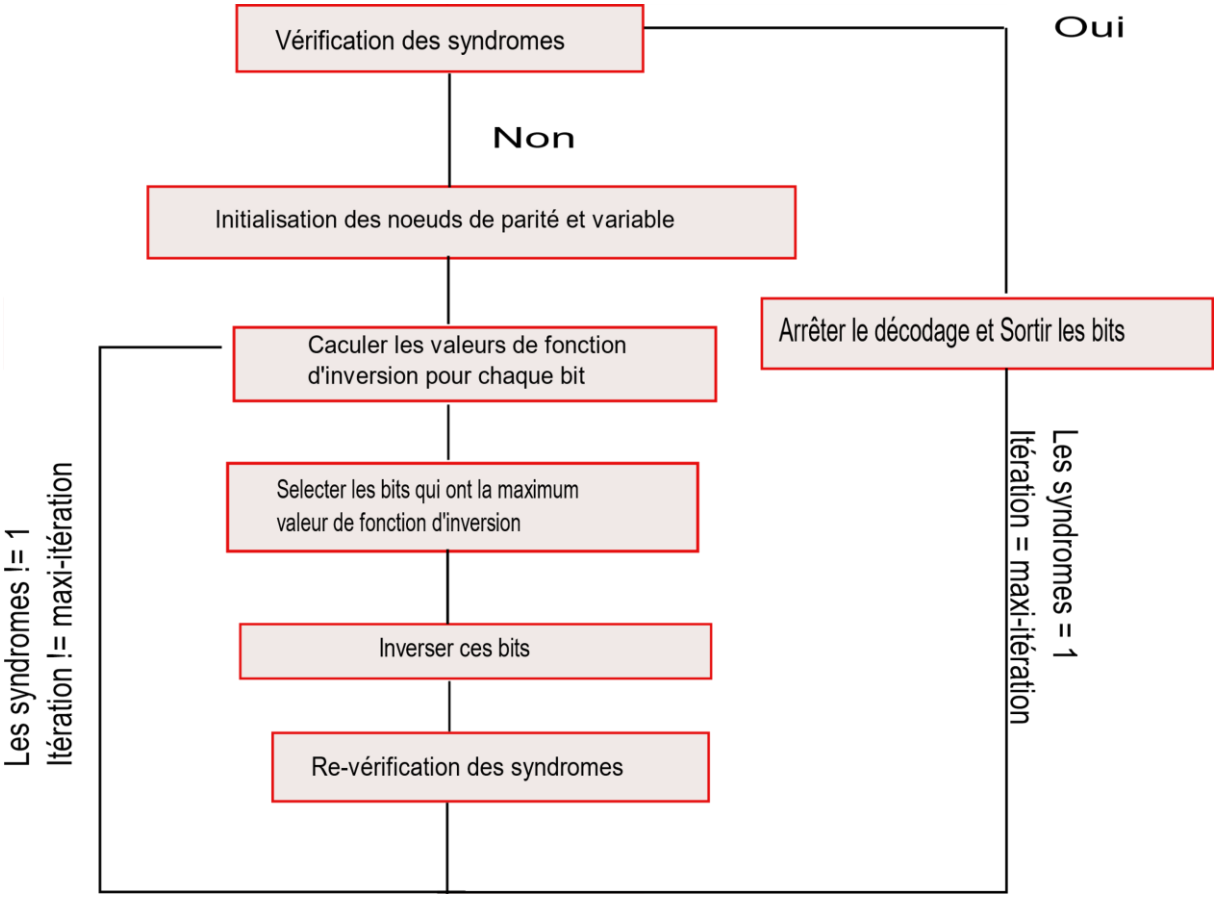


Figure III-6 : Algorithmes à décisions dures pour les codes LDPC

III.2.1. L'algorithme Bit flipping

Comme on a présenté dans la dernière partie, l'algorithme Bit-Flippling est proposé dans la thèse de Gallager, il utilise seulement les valeurs des syndromes. Sa fonction d'inversion s'exprime par :

$$\Delta k^{BF}(x) \triangleq \sum_{i \in M(n)} \prod_{j \in N(i)} x_j \qquad (III\text{-}3)$$

où $\alpha * \emptyset(x_n, y_n) = 0$ et $\omega = 1$ par rapport à l'équation générale donnée en (III-2) $\Delta k^{générale}$.

Prenons un simple exemple pour cet algorithme, soit :

$$H = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}$$

Figure III-7 : Exemple de matrice H

On suppose la séquence émise :

$C = [1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1]$,

Avec une modulation BPSK, on obtient :

$\tilde{C} = 1 - 2C = [-1 \quad 1 \quad 1 \quad -1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad 1 \quad -1]$

On superpose un bruit Gaussien AWGN, on obtient en réception :

$y_k = \tilde{C} + bruit \quad = [-0.8 \quad 0.9 \quad 0.6 \quad -1.2 \quad 1.3 \quad -1.1 \quad -1.1 \quad 0.8 \quad -0.7 \quad 1.6 \quad 1.3 \quad 0.2]$

Avant le décodage, on prend la décision des messages reçus avec un seuil (threshold) égal à 0. Donc on peut obtenir :

$x_k = [-1 \quad 1 \quad 1 \quad -1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad 1 \quad 1]$

- **Etape 1** : Détection des erreurs

Le syndrome peut être calculé comme :

$$S_j = \prod_{j \in N(i)} x_j \qquad \textit{(III-4)}$$

On fait la multiplication des éléments non-nuls dans chaque ligne de la matrice parité H en prennant la valeur x_k.

$$\begin{cases} S_j = 1, correct \\ S_j = -1, incorrect \end{cases}$$

Par example, ici la multiplication des éléments non-nuls dans la matrice de parité H Figure III-7 conduit au résultat suivant :

$$\begin{bmatrix} S_1 \\ S_2 \\ S_3 \\ S_4 \\ S_5 \\ S_6 \end{bmatrix} = \sum \begin{bmatrix} -1 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 \\ -1 & 1 & 1 & -1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 1 & -1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 & -1 & -1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 & -1 & 1 & 0 & 1 \end{bmatrix}$$

Donc le syndrome pour la première ligne donne :

$$S_1 = \prod_{j \in N(i)} x_j = -1 * 1 * -1 * -1 * 1 * 1 = -1$$

De la même façon :

$$[S_2, S_3, S_4, S_5, S_6,] = [1 \quad 1 \quad 1 \quad -1 \quad -1]$$

Pour simplifier les calculs, on fait la sommation des syndromes, si $\sum S_j = M$ (taille des colonnes de la matrice de parité), il n'y a pas d'erreur dans la séquence reçue et le décodage s'arrête. Sinon, on doit calculer la fonction d'inversion pour identifier quels sont les bits à inverser. Dans cet exemple, M vaut 6, on peut bien conclure $\sum S_j = 0 \neq M$, donc on doit continuer le décodage.

- **Etape2 :** cacul des fonctions d'inversion

La fonction inversion peut être calculée par la somme des éléments non-nuls de chaque colonne dans la matrice de parité H Figure III-7 en prenant les valeurs des syndromes :

$\Delta k^{BF}(1) = S_1 + S_2 + S_3 = 1$

De la même façon :

$$\Delta k^{BF}(1 \to 12) = [1 \quad -1 \quad 1 \quad 1 \quad 3 \quad 1 \quad -1 \quad -1 \quad -1 \quad -1 \quad 1 \quad -3]$$

- **Etape 3 :** Inversion des bits détectés

Comme on a montré en Figure III-6, le bit qui a la valeur minimale de Δk^{BF} doit être inversé, c'est à dire le $12^{ème}$ bit doit être inversé, après la correction, on continue la prochaine itération, on obtient cette fois ci $\sum S_j = M$, donc on arrête le décodage et on obtient la séquence décodée. Si ce n'est pas le cas, on continue le décodage jusqu'à ce que toutes les équations de parité soient satisfaites ou que le nombre maximum d'itérations soit atteint.

III.2.1.1. Amélioration de l'algorithme BF

L'algorithme BF conduit à la plus simple implémentation parmi toutes les méthodes d'inversion. Cependant une des faiblesses importantes de cet algorithme réside dans son comportement dans la zone des faibles SNR's. Le fait d'inverser plusieurs valeurs de bits en même temps peut conduire à la génération de nouvelles erreurs. Ce problème est illustré sur la Figure III-8. Comme montré en Figure III-8 , x_2 , x_3 sont des bits erronés, on peut observer qu'après le calcul de la fonction inversion Δk^{BF} , x_1, x_3 x_4 sont considérés ici maintenant comme des bits erronés. Pourtant, x_1, x_4 sont des bits corrects.

Le phénomène est d'autant plus marqué que les séquences reçues comportent un nombre d'erreurs assez élevé et que le rendement du code est important. En effet dans le cas de hauts rendements, si on a inversé trop de bits dans une itération, cela peut générer des cycles dans la matrice ce qui va entraîner la divergence de notre algorithme de décodage.

En conséquence, il y a deux méthodes principales pour éliminer ce problème, soit on fixe un nombre maximum de bits à inverser pour chaque itération : MBF (Multiple BF), soit on fixe une probabilité pour décider quel bit erroné doit être inversé[136][137][138][139] , par exemple en utilisant l'algorithme PBF : Probabilistic Bit Flipping algorithm.

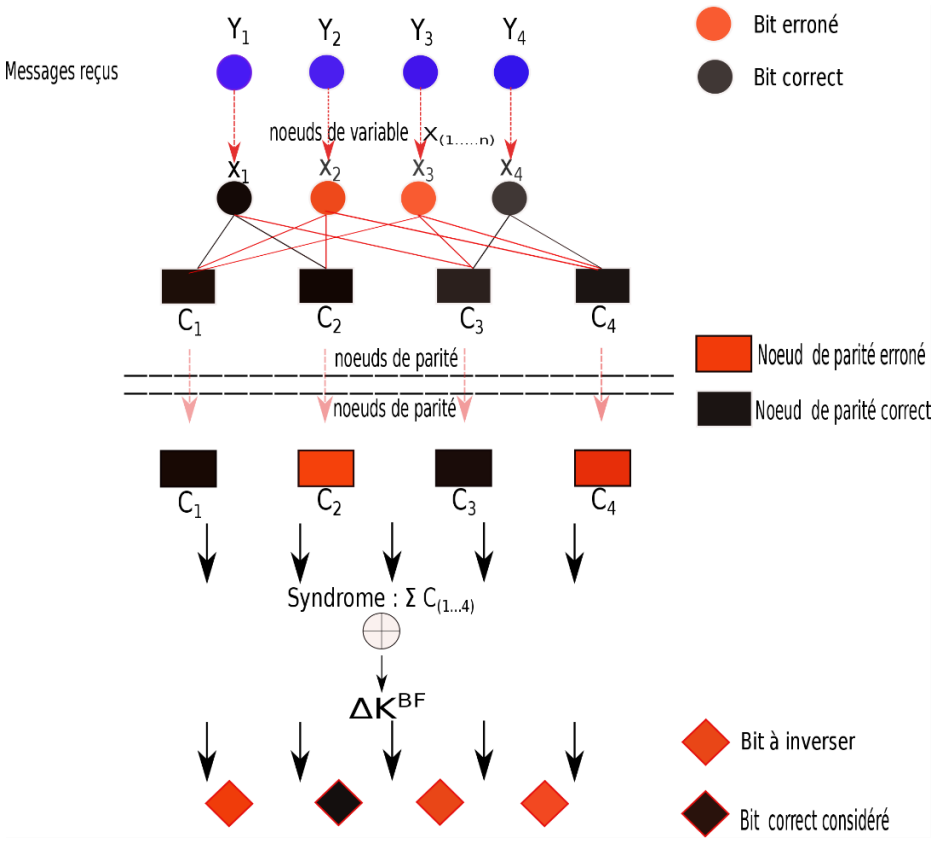


Figure III-8 : Exemple de calcul pour l'algorithme BF

III.2.1.1.1. MBF (Multiple BF) :

On reprend la même formule que la fonction d'inversion de l'algorithme BF:

$$\Delta k^{BF}(x) \triangleq \sum_{i \in M(n)} \prod_{j \in N(i)} x_j \qquad (III\text{-}5)$$

On Suppose la présence d'un ensemble de bits B qui ont des valeurs minimums de Δk^{BF} :

$$B = \{n | \Delta k^{BF} = \Delta k^{BF}{}_{min}\} \qquad (III\text{-}6)$$

Parmi ces bits, on fixe un nombre maximum de bits Δ_{FB} à inverser lors de chaque itération.

III.2.1.1.2. PBF (probabilistic BF)

Au lieu d'inverser tous les bits qui ont des minimum valeur de Δk^{BF} :

$$B = \{n | \Delta k^{BF} = \Delta k^{BF}{}_{min}\} \qquad (III\text{-}7)$$

On fixe une probabilité p_f et on génére une probabilité $p(x)$ aléatoirement pour chaque bit :

$$\begin{cases} p(x_k) \geq p_f, & x_k = -x_k \\ p(x_k) < p_f, & x_k = x_k \end{cases}$$

III.2.1.2. Conclusion

Les complexités des algorithmes à décisions dures sont dépendantes des fonctions d'inversion. Dans le Tableau III-1, on peut observer que les performances de ces algorithmes d'inversion malgré les améliorations proposées[136], restent assez faibles.

Tableau III-1 : Performances et complexité de l'algorithme BF et de ses variantes

Algorithme	Performance	Complexité
BF	Pauvre	Très Simple
MBF	Moins Bien	Simple
PBF	Moins Bien	Simple

III.2.2. Weighted bit-flipping (WBF)

Les performances des algorithmes dérivés du Bit flipping ne sont pas suffisantes pour satisfaire les exigences de performances du système optique proposé. C'est pourquoi l'on s'oriente vers d'autres solutions telles que l'algorithme WBF (Weighted Bit Flipping) [82]. Le principe de cet algorithme est d'ajouter une pondération pour les syndromes calculés. En réception, plus la valeur absolue de l'échantillon reçu $abs(y_n)$ sera faible et moins fiable sera le bit décidé. A partir de la formule générale de $\Delta k^{générale}$ de la partie précédente, la fonction d'inversion de WBF s'exprime par :

$$\Delta k^{WBF} \triangleq -\sum_{i\in M(n)} \beta_i \prod_{j\in N(i)} x_j \qquad \textit{(III-8)}$$

D'où $\alpha * \emptyset(x_k, y_k) = 0$ et $\omega = B_i$, $B_i \triangleq \min_{j\in N(i)} abs(y_j)$.

Un exemple est démontré pour cet algorithme ci-dessous, ici on prend les mêmes paramètres (H, x_n, y_n) que pour l'algorithme BF décrit précédemment.

- **Etape 1** : Détection des erreurs

On examine les syndromes comme on l'a fait pour la méthode BF, si $\sum S_j = M$, on arrête le décodage, sinon on passe à l'etape 2 pour calculer la fonction d'inversion Δk^{WBF}, ici $\sum S_j = 0 \neq M$, il y a des erreurs dans la séquence reçue, donc on passe à l'étape 2.

- **Etape 2** : Calcul des fonctions d'inversion

Dans cette étape, on calcule d'abord B_i. On cherche les positions où $h_{ij} = 1$, et puis on garde le minimum de la valeur absolue pour la même position dans le vecteur y_k. Pour la première ligne dans la matrice parité H, on trouve $h_{11}, h_{12}, h_{14}, h_{19}, h_{1,10}, h_{1,12} = 1$, donc on obtient la valeur minimale dans le vecteur $[y_1, y_{2,} y_4, y_9, y_{10}, y_{12}]$. Pour le prémier syndrome S_1, $B_i = abs(y_{12}) = 0.2$.

De la même façon :

$$\begin{bmatrix}\beta_2 = 0.6\\ \beta_3 = 0.6\\ \beta_4 = 0.7\\ \beta_5 = 0.2\\ \beta_6 = 0.2\end{bmatrix}$$

Avec les syndromes on a calculé dans la partie précédente :

$$S = [S_1, S_2, S_3, S_4, S_5, S_6,] = [-1 \quad 1 \quad 1 \quad 1 \quad -1 \quad -1]$$

La fonction d'inversion de la méthode WBF se calcule :

$$\Delta k^{WBF} = -\sum_{i \in M(n)} \beta_i S_i$$

$$= [1.4 \quad 1.0 \quad 1.4 \quad 1.5 \quad 1.9 \quad 1.5 \quad 1.1 \quad 1.0 \quad 1.1 \quad 1.0 \quad 1.5 \quad 0.6]$$

Etape 3 : Inversion des bits détectés

On cherche et on inverse le bit qui a la valeur minimale (la moins fiable) de Δk^{WBF}, ici le $12^{ème}$ bit doit être inversé. Après l'inversion des bits, on ré-examine les syndromes, si $\sum S_j = M$ on arrête le décodage et on sort la séquence corrigée, sinon on continue les itérations jusqu'à ce que toutes les conditions soient satisfaites.

III.2.2.1. Amélioration de l'algorithme WBF

L'algorithme WBF permet d'améliorer les performances de l'algorithme BF. Basé sur cette méthode, on peut ajouter une métrique pour fiabiliser plus sa fonction d'inversion, plusieurs méthodes sont proposées dans la littérature scientifique :

III.2.2.1.1. MWBF (Modified WBF) [141] :

$$\Delta k^{MWBF} \triangleq abs(y_n) - \sum_{i \in M(n)} \beta_i S_i \qquad \textit{(III-9)}$$

D'où $\emptyset(x_n, y_n) = abs(y_n)$. En mode single-flip, dans chaque itération, le bit qui a la valeur minimale de Δk^{MWBF} doit être inversé.

III.2.2.1.2. IMWBF (Improved modified WBF) [142] :

$$\Delta k^{IMWBF} \triangleq \alpha * abs(y_n) - \sum_{i \in M(n)} \beta_i' \prod_{j \in N(i)} x_j \qquad \textit{(III-10)}$$

D'où $\alpha * \emptyset(x_n, y_n) = \alpha * abs(y_n)$, $\omega = B_i'$, α est optimisé par simulations. $B_i' \triangleq \min_{j' \in N(i) \setminus j} abs(y_{j'})$, le principe de cette méthode est d'éviter l'impact sur le bit soi-même comme pour les algorithmes soft-décision (MS ou BP). De la même façon comme pour l'algorithme MWBF, en mode single-flip, dans chaque itération, le bit qui a minimum valeur de Δk^{IMWBF} doit être inversé.

III.2.2.1.3. RRWBF (Reliability based ratio WBF) [143] :

$$\Delta k^{RRWBF} \triangleq -\sum_{i\in M(n)} \frac{1}{R_{mn}} \prod_{j\in N(i)} x_j \qquad \textit{(III-11)}$$

$$R_{mn} = \beta \frac{abs(y_k)}{max(abs((y_i^{1\cdots n}))|h_{ij}=1} \qquad \textit{(III-12)}$$

Où $\alpha * \emptyset(x_n, y_n) = 0$, $\omega = \frac{1}{R_{mn}}$, β représente le facteur normalisé qui satisfait l'équation : $\sum_{n\in M(m)} R_{mn} = 1$. $max(abs((y_i^{1\cdots n}))|h_{ij}=1$ décrit le maximum d'amplitude des noeuds de variable liés au i^{th} noeud de parité. Le bit qui a la valeur minimale de Δk^{RRWBF} doit être inversé aussi.

Les autres méthodes comme PWBF (Parallel WBF) [144], IPWBF (Improved Parallel WBF) [147], LPWBF (Liu-pados WBF), SLBWBF (Simplified Loop Break WBF), WSWBF (Weighted-sum WBF) [142] DWBF (Dynamic WBF) [148], AMWBF (Adaptive WBF) [145], MIWBF(Multiple improved WBF) [146] sont basées sur les méthodes précédentes qui manipulent plutôt le facteur de fiabilité sur les valeurs de la fonction d'inversion afin d'éliminer les erreurs supplémentaires générées par les vraies erreurs.

III.2.2.2. Conclusion

Tableau III-2 : Performance et Complexité des algorithmes WBF et de leurs variantes

Algorithme	Performance	Complexité
WBF	Bien	Très simple
MWBF	Très Bien	Simple
IMWBF	Excellent	Moyen
RRWBF	Très Bien	Moyen
PWBF	Très Bien	Complexe
IPWBF	Très Bien	Complexe
LPWBF/SLBWBF	Très Bien	Moyen
WSWBF	Très Bien	Moyen
AMWBF	Très Bien	Moyen
DWBF	Excellent	Très Complexe

Comme le montre sur le Tableau III-2, certains algorithmes comme l'algorithme (DWBF)[148], atteignent des performances très proches de l'algorithme MS mais leur complexité d'implémentation est importante. Les algorithmes WBF et IMWBF présentent un très bon compromis performances/complexité.

III.2.3. L'algorithme GDBF (gradient descent BF)

L'algorithme GDBF est une méthode qui donne la meilleure performance parmi tous les algorithmes à décisions dures, il a été proposé par Sundararajan *et al*.[149]. La problématique de maximum de vraisemblance (LLR) est équivalente à trouver un mot de code (bipolaire) dans la séquence reçu $\tilde{C}$, qui donne la valeur maximale de correlation. Il s'écrit : $\tilde{x} = argmax_{x\in\tilde{C}} \sum_{j=1}^{n} x_j y_j$. Basé sur la règle de corrélation, la fonction à optimiser est définie comme suit [150] :

$$f(x) \triangleq \sum_{i=1}^{n} x_j y_j + \sum_{i=1}^{m} \prod_{j\in N_{(i)}} x_j \qquad \textit{(III-13)}$$

La première partie représente la corrélation entre le mot de code bipolaire x_k et le mot de code reçu y_k, la deuxième partie représente la sommation du syndrome bipolaire de x. Si et seulement si $x \in \tilde{C}$, $f(x)$ atteint la valeur maximale avec $\sum_{i=1}^{m} \prod_{j\in N_{(i)}} x_j = M$.

Si on calcule la dérivée de $f(x)$ par rapport à x_k, on obtient :

$$\frac{\partial}{\partial x_k} f(x) = y_k + \sum_{i \in M(k)} \prod_{j \in N(i) \backslash k} x_j \qquad \textit{(III-14)}$$

Pour une très petite valeur réelle s, on peut obtenir une approximation du 1^{er} ordre, l'équation s'exprime :

$$f(x_1 \cdots, x_k + s \cdots, x_n) \cong f(x) + s \frac{\partial}{\partial x_k} f(x) \qquad \textit{(III-15)}$$

Si on veut que $f(x) + s \frac{\partial}{\partial x_k} f(x)$ est supérieur à $f(x)$ afin que la fonction d'optimisation atteigne la valeur maximale, s doit toujours être supérieur à 0, sinon l'inversion de $k^{ème}$bit ($x_k := -x_k$) pourrait augmenter la valeur de la fonction d'optimisation. En conséquence, en considérant le produit de x_k et la dérivation de x_k dans x. La fonction d'inversion de GDBF peut s'exprimer par :

$$\Delta k^{GDBF} = x_k y_k + \sum_{i \in M(k)} \prod_{j \in N(i)} x_j \qquad \textit{(III-16)}$$

Normalement la valeur minimale de $\Delta \mathrm{k}^{\mathrm{GDBF}}$ doit être inversée. Dans ce qui suit, on donne un example pour cette méthode en utilisant les mêmes valeurs des paramètres (H, x_k, y_k) d'entrée comme la méthode BF.

- **Etape 1** : Détection des erreurs

On utilise la même équation $\sum S_j = \sum_{i \in M(k)} \prod_{j \in N(i) \backslash k} x_j$ pour justifier s'il y a des erreurs dans la séquence reçue, pour cet example, $\sum S_j = 0$, donc on passe à étape 2.

- **Etape 2** : Calcul des fonctions d'inversion

On calcule la fonction d'inversion de GDBF :

$$\Delta k^{GDBF}(1 \cdots 12) = \begin{bmatrix} 1.8 \\ -0.1 \\ 1.6 \\ 2.2 \\ 4.3 \\ 2.1 \\ 0.1 \\ -0.2 \\ -0.3 \\ 0.6 \\ 2.3 \\ -2.8 \end{bmatrix}$$

- **Etape 3** : Inversions des bits détectés

On inverse le bit qui a la valeur minimale de Δk^{GDBF}, dans cet example, le $12^{ème}$ bit doit être inversé, et puis on ré-examine les syndromes pour décider si on va arrêter le processus de décodage ou on continue à la prochaine itération.

III.2.3.1. Amélioration de l'algorithme GDBF

Le principe pour l'algorithme GDBF est d'atteindre la valeur maximale pour la fonction à optimiser $f(x)$, s'il existe des cycles dans la matrice de parité H, il y a un phénomène de maximum local qui apparaît.

Un exemple est montré sur la Figure III-9, on suppose ici trois bits erronés (x_1, x_2, x_3) qui sont tombés dans un cycle 6 avec une matrice de parité avec poids de colonne et poids de ligne $(3,6)$. Pour la prémière itération, l'indication de $\sum_{i\in M(k)} \prod_{j\in N(i)} x_j$ conduit au résultat : S_2, S_4 et S_6 sont détectés comme des nœuds de parité erronés, donc les erreurs se trouvent dans les nœuds adjacents de ces nœuds erronés. En utilisant la fonction d'inversion de GDBF Δk^{GDBF}, on considère que x_4 doit être inversé car c'est le bit qui participe le plus aux équations de parité (S_2, S_4). Dans la deuxième itération, après le calcul de Δk^{GDBF}, le bit x_4 se retrouve comme un bit erroné, on tourne dans une boucle d'inversion, c'est à dire, le bit x_4 sera inversé dans les itérations restantes.

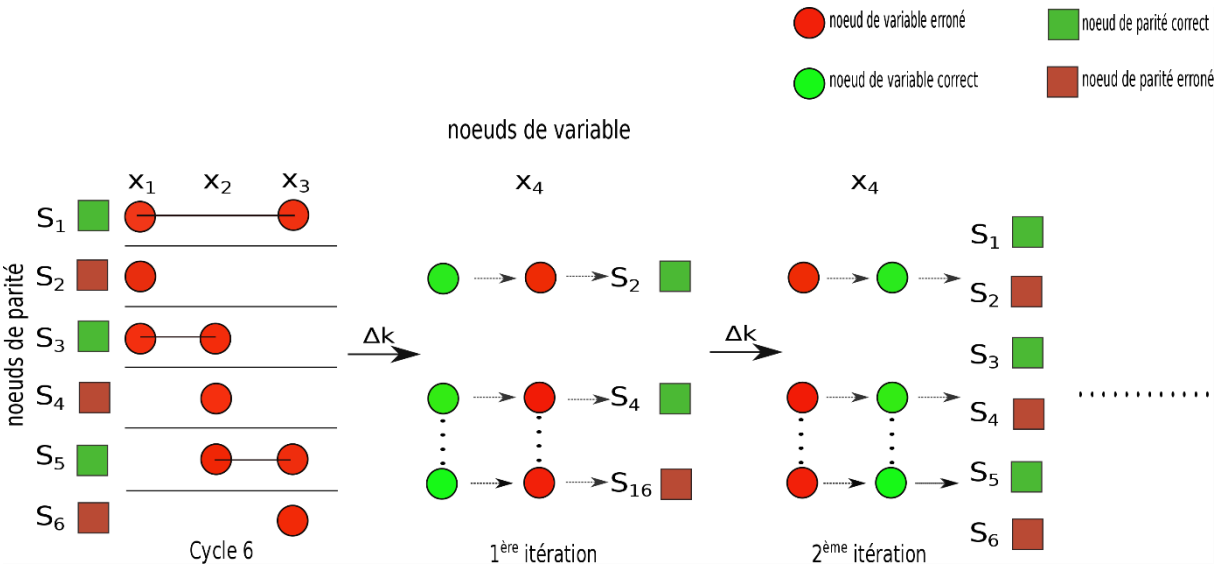


Figure III-9 : Exemple de maximum local (1)

Si on regarde la fonction à optimiser dans ce cas, la valeur de $f(x)^{GDBF}$ est stabilisée mais il n'a pas encore atteint sa valeur maximale, on l'appelle maximum local ${f(x)_{LM}}^{GDBF}$.

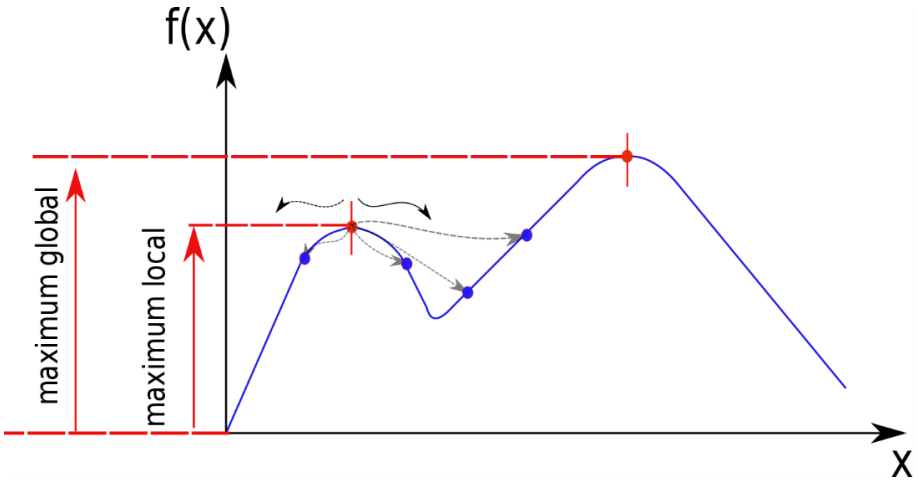


Figure III-10 : Maximum local

Le principe des améliorations de l'algorithme GDBF algorithme est de créer des possibilités de casser les cycles de matrice de parité H afin d'éviter le maximum local.

III.2.3.1.1. HGDBF (Hybrid GDBF) :

Dans la papier [149], Wadayama a aussi proposé un algorithme basé sur la fonction de coût ou fonction d'objectif ou encore fonction à optimiser, on remplace les étapes 2 et 3 par les procedures suivantes :

Evaluer la valeur de la fonction d'objectif en supposant :

$$f_1 \coloneqq f(x), \mu = 0$$

Avec :

$$\begin{cases} \mu = 1, mode\ single\ inversion \\ \mu = 0, mode\ multiple\ inversion \end{cases}$$

- Multiple mode (MGDBF):

L'ensemble des bits à inverser s'exprime sous la forme :

$$\Delta k^{HGDBF} = \Delta k^{GDBF} \qquad (III\text{-}17)$$

$$B = \{n | \Delta k^{HGDBF}(x) \leq \theta\} \qquad (III\text{-}18)$$

Où θ est estimé par la simulation, il varie avec les paramètres de la matrice de parité H (girth, rendement, taille de matrice).

- Single mode (SGDBF)

Il se manipule comme les alogrithmes BF normaux, on inverse le bit qui a la valeur minimale pour la fonction d'inversion :

$$x_k = -x_k | \Delta k^{HGDBF}(x_k) \triangleq arg\ min_{k \in [1,n]} \Delta k^{HGDBF}(x) \qquad (III\text{-}19)$$

- Choix du mode :

Au départ de l'algorithme, on utilise d'abord le mode multiple en supposant $\mu = 0$, et on a toujours $f_1 < f_2$, avec les itérations on continue, tandis que dès que : $f_1 \geq f_2$, on change et on commute en single mode ($\mu = 1$). Cette méthode permet à la fonction de coût de s'approcher d'un maximum local. Il ne s'agit pas d'un algorithme optimal. Dans son papier Wadayama a aussi proposé une méthode pour quitter le maximum local: comme il est montré en Figure III-11, en supposant 2 seuils (threshold) θ_1 et θ_2, d'abord on inverse un ensemble de bits en utilisant θ_1 pour faire converger le décodage, et puis quand la fonction d'objectif se stabilise dans la zone du maximum local, la fonction d'inversion peut atteindre le maximum global avec threshold θ_2 utilisant l'équation (III-18) d'où : $\theta = \theta_2$.

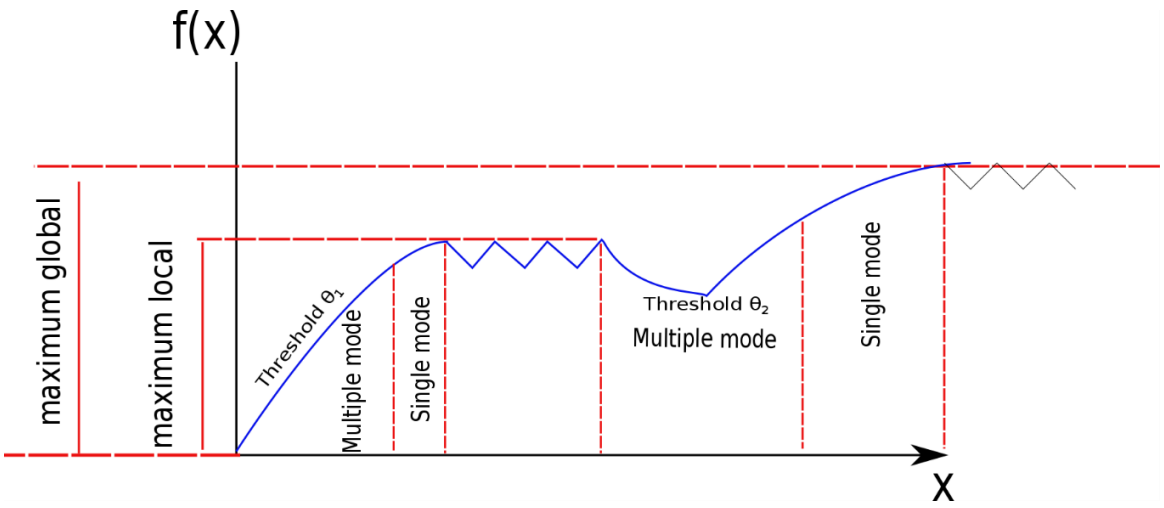


Figure III-11 : mode de fonctionnement de l'algorithme HGDBF

HGDBF atteint la meilleure performance parmi les algorithmes GDBF et sa performance est très comparable à la performance de l'algorithme MS. En revanche, il a besoin d'un plus grand nombre d'itérations pour évaluer la fonction d'objectif, en conséquence, il est peu réalisable en vue d'une implémentation circuit à cause des délais et de la complexité. En plus, θ_1 et θ_2 sont estimés par simulation, donc la convergence de cet algorithme n'est pas toujours stable.

III.2.3.1.2. AT-GDBF (Adaptive Threshold GDBF) [155]

Pour résoudre le problème d'instablilté de l'algorithme HGDBF, l'algorithme AT-GDBF algorithme n'utilise plus le mode single ou multiple en proposant un seuil λ_k adaptatif.

Dans l'étape 2 et l'étape 3, après les calculs de la fonction d'inversion :

$$\Delta k^{ATGDBF} = \Delta k^{GDBF} \qquad (III\text{-}20)$$

$$\begin{cases} si\ \Delta k^{ATGDBF}(x_k) < \lambda_k, x_k = -x_k \\ sinon,\ \ \lambda_k = \theta\ \lambda_k \end{cases} \qquad (III\text{-}21)$$

θ et λ_k s'exprime:

$$\lambda_k = \lambda_0 2^z \qquad (III\text{-}22)$$

$$\lambda_0 = 2^d$$

Avec :

$$z = -(-log_2\left(\emptyset_{SNR}/\lambda_0\right) + mode(-log_2\left(\emptyset_{SNR}/\lambda_0\right), d)) \qquad (III\text{-}23)$$

Où la fonction $\emptyset_{SNR}$ présente une équation polynômiale et a est la valeur du SNR en dB :

$$\emptyset_{SNR} = -0.005a^4 + 0.089a^3 - 0.666a^3 + 2.258a - 2.897 \qquad (III\text{-}24)$$

Dans le papier [155] un algorithme ES-ATGDBF (early stop ATGDBF) est proposé afin de réduire le nombre d'itérations et l'énergie consommée. La performance est dégradée par

rapport à HGDBF algorithme mais elle reste comparable à celle des algorithmes MWBF et WBF.

III.2.3.1.3. RRWGDBF (Reliability-Ratio weighted GDBF) [152] :

A partir de la méthode RRWBF présentée dans la partie précédente, on peut aussi pondérer par rapport aux calculs des syndromes, on obtient un nouvel algorithme baptisé RRWGDBF et qui est proposé en [152], la fonction d'inversion s'exprime suivant :

$$\Delta k^{RRWGDBF}(x_k) = x_k y_k + \sum_{i \in M(k)} \beta_{ik} \prod_{j \in N(i)} x_j \qquad \textit{(III-25)}$$

D'où :

$$\beta_{ik} = \frac{1}{|y_k|} \sum_{j \in N(i)} |y_j| \qquad \textit{(III-26)}$$

Le papier [94] introduit aussi trois méthodes :

- Méthode Single flip :

Le bit qui a la valeur minimale de la fonction d'inversion $\Delta k^{RRWGDBF}$ va être inversé :

$$x_j = -x_j | \, j = arg \min_{k \in (1 \cdots n)} \Delta k^{RRWGDBF}(x) \qquad \textit{(III-27)}$$

- Méthode hybride :

Le principe est le même que pour l'algorithme HGDBF.

Début de l'algorithme :

$$B = \{n | \, \Delta k^{RRWGDBF}(x) \leq \theta\} \qquad \textit{(III-28)}$$

Quand $f(x)$ commence à se stabiliser :

$$x_j = -x_j | \, j = arg \min_{k \in (1 \cdots n)} \Delta k^{RRWGDBF}(x) \qquad \textit{(III-29)}$$

- Méthode Adaptive Threshold :

Pour cette méthode, on n'utilise que le mode multiple en remplaçant le seuil (threshold) θ avec le threshold adaptive $\theta_{RRWGDBF}$:

$$\theta_{RRWGDBF} = \frac{(\sigma_y^2)^2}{2} \frac{1}{N_{neg}} \sum_{k \in (1 \cdots N_{neg})} \Delta k^{RRWGDBF}(x) \qquad \textit{(III-30)}$$

Où N_{neg} est nombre d'inversions négatives et σ_y est la variance des bits reçus, σ_y s'exprime par:

$$\sigma_y = \frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2 \qquad \textit{(III-31)}$$

Cette méthode peut largement réduire la compléxité de circuit et sa performance est comparable à celle des méthodes IMWBF et SGDBF.

III.2.3.1.4. IHGDBF (Improve Hybrid GDBF) [153] :

Dans les méthodes précédentes, on pondère sur la partie syndrome $\sum_{i\in M(k)}\prod_{j\in N(i)} x_j$ et on définit les thresholds pour améliorer les performances, mais il faut bien reconnaître que ces méthodes restent encore complexes. Pourtant, la correlation entre x_k et y_k ($\emptyset(x_k, y_k)$) permet d'identifier également la fiabilité des bits reçus (plus la valeur de la corrélation entre x_k et y_k est grande et plus les bits reçus sont fiables). En général, la conversion bipolaire binaire est exécutée avant transmission (BPSK modulation), donc on a toujours $i.e.\{0,1\}^n \rightarrow \{1,-1\}^n$, une erreur est générée quand $\{\tilde{C} = -1 | y_j \geq 0, x_j = 1\}$ ou $\{\tilde{C} = 1 | y_j < 0, x_j = -1\}$, on peut observer que si la valeur de bruit est supérieure à 1.0, alors l'erreur apparaît. Autrement dit, le nombre des erreurs est égale au nombre des valeurs reçues supérieures à 2.0 $(y_j \geq 2.0)$. Donc on peut inverser les bits sans fixer les threshholds. Cette méthode a été proposée pour la première fois dans [153] afin de réduire la complexité et la latence.

Dans l'étape d'initialisation, on définit un nouveau paramètre B_{max} et μ:

$$B_{max} = \left|\{j | (|y_j| > 2.0), j \in [1,n]\}\right| \; et \; \mu = 0 \qquad \textit{(III-32)}$$

D'où $\begin{cases}\mu = 0, multiple\ inversion \\ \mu = 1,\ single\ inversion\end{cases}$

Dans l'étape d'inversion des bits, on obtient 2 cas comme suit :

$$\begin{cases}\mu = 0, B_l = max[B_{l-1}/2, 1] \\ \mu = 1, \quad B_l = 1\end{cases} \qquad \textit{(III-33)}$$

Où l représente l'indice de l'itération courante et l_{max} représente le nombre maximum des itérations. L'ensemble des bits à inverser B_l est inclus dans l'ensemble des bits qui ont des valeurs minimales de fonction d'inversion $\Delta k^{IHGDBF} | \Delta k^{IHGDBF} = \Delta k^{GDBF}$.

III.2.3.1.5. NGDBF (Noisy GDBF):

Dans le papier [151] les auteurs proposent un nouvel algorithme qui introduit une perturbation aléatoire dans la fonction d'inversion. L'étape de calcul de la fonction d'inversion peut être modifiée comme suit :

$$\Delta k^{NGDBF} = x_k y_k + \omega \sum_{i\in M(k)} s_i + q_k \qquad \textit{(III-34)}$$

Où $\omega \in R^+$ représente la pondération des syndromes, q_k est une distribution aléatoire Gaussienne avec variance $\sigma^2 = (\frac{\vartheta N_0}{2}) | 0 < \vartheta < 1$, et une moyenne égale à zéro. Tous les q_k sont distribués aléatoirements et independents.

Les modes SNGDBF (single flippe) et MNGDBF (multiple flippe) sont aussi introduits en fixant un threshold θ_k qui est obtenu par simulations expérimentales.

Pour stabiliser les résultats, NGDBF réduit le nombre d'itérations ($\cong 100$) par rapport à HGDBF. Cependant, Vue leur complexité, les algorithmes stochastiques (SNGDBF, MNGDBF) ne sont pas présentés dans la partie simulation.

III.2.3.2. Conclusion

Tableau III-3 : Performance et complexité pour les méthodes XGDBF

Algorithmes	Performance	Complexité
SGDBF	Bien	Moyen
MGDBF	Très Bien	Moyen
HGDBF	Excellent	Complexe
ATGDBF	Bien	Moyen
RRWGDBF	Excellent	Complexe
SNGDBF	Excellent	Complexe
MNGDBF	Excellent	Complexe

Le Tableau III-3 montre un résumé des performances des différents algorithmes d'inversion évoqués [149][151][152][153][155][154]. Pour l'application visé, à savoir le C-RAN optique, il nous faut choisir le meilleur compromis performances/complexité parmi les algorithmes présentés.

III.2.4. Performance

Dans cette partie, on va donner les résultats des simulations pour différents algorithmes basés sur les méthodes d'inversion : algorithmes dures basiques : BF, WBF, GDBF mais aussi les algorithmes : IWMBF, RRWGDBF en faisant varier les paramètres (rendement, nombre d'itérations, la taille) afin de faire un choix définitif pour la suite des études.

En vue d'une implémentation circuit, les matrices construites sont des matrices quasi-cycliques et, de plus, les cycles d'ordre 4 sont éliminés à l'aide de la méthode présentée dans [125][109]. Les rendements des matrices de parité H varient avec les valeurs suivantes : $0.5, 0.7$ et 0.9. Ces valeurs sont choisies pour s'adapter au contexte C-RAN. La taille des paquets est fixée à 1000 bits. On obtient donc les tailles suivantes des matrices de parité : H (500, 1000), (700, 1000) et (900, 1000) avec les poids de colonne et de ligne correspondants : $(3,6), (3,10)$ et $(3,30)$. De plus, pour préserver des temps de latence assez faibles [149][151][154] le nombre d'itérations est choisi assez faible de l'ordre d'une vingtaine d'itérations.

Sur la Figure III-12 sont illustrées les performances des algorithmes BF, WBF, GDBF pour la matrice $(500, 1000)$ avec le rendement 0.5 . Pour être conforme aux références [141][142][149][151], nous illustrons également les performances avec un nombre d'itérations égal à 100. Les courbes pointillées représentent les performances des algorithmes avec 100 itérations. Les courbes continues représentent les performances des algorithmes avec 20 itérations. La courbe 'Canal' correspond à la performance sans décodage. On peut bien observer que les algorithmes à décisions dures améliorent la performance par rapport au système non-codé. Parmi tous ces algorithmes c'est l'algorithme SGDBF qui atteint la

meilleure performance. En revanche, il y a $2.5\ dB$ de marge entre l'algorithme souple MS et GDBF pour un TEB (Taux Erreur Binaire aussi appelé TEB : Bit Error Rate dans cette thèse) égal à 10^{-4}. En plus, on peut aussi observer que 20 itérations sont suffisantes pour atteindre la performance optimale comparé à 100 itérations pour l'algorithme MS. Mais par contre, pour les algorithmes durs, 20 itérations ne sont pas suffisantes pour stabiliser la performance.

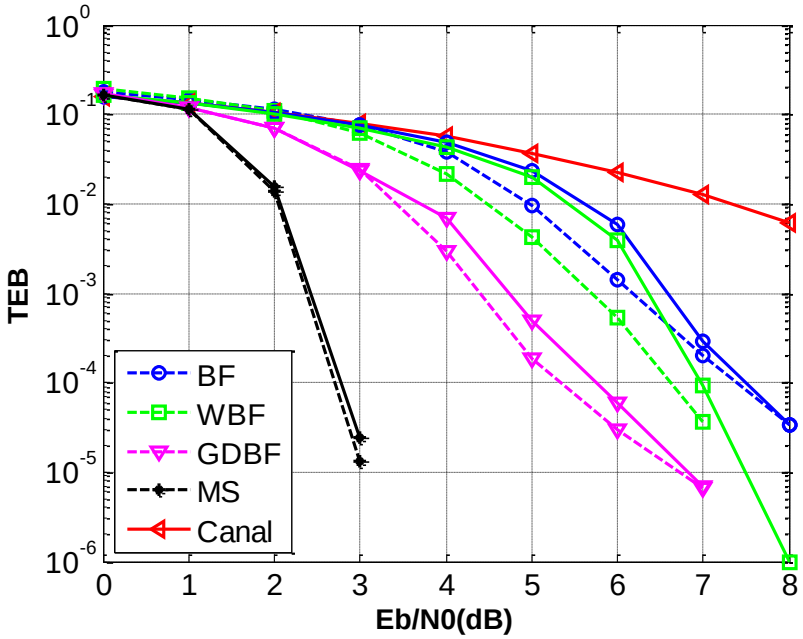


Figure III-12 : Comparaison des performances entre BF WBF et GDBF pour la matrice (3, 6) avec une taille (500, 1000) et un rendement 0.5

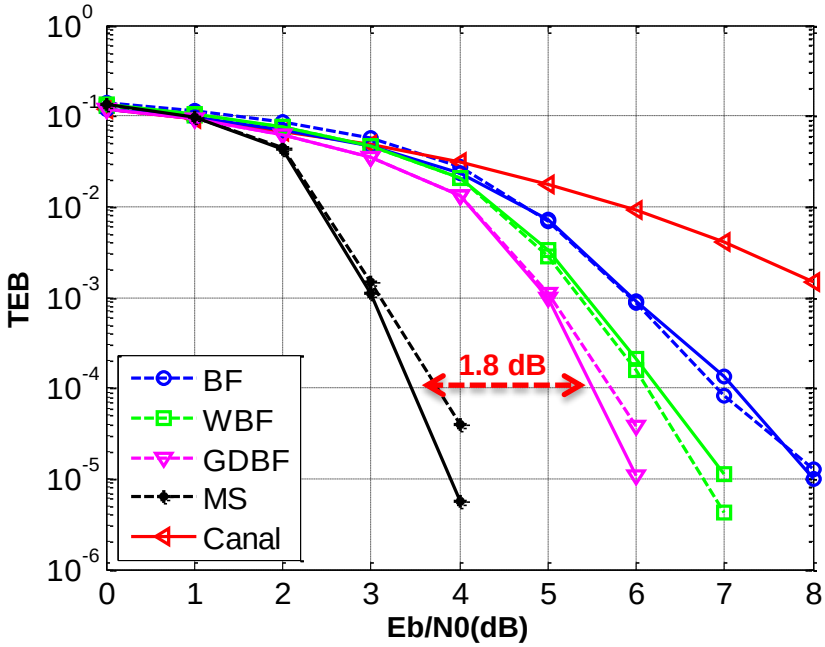


Figure III-13 : Comparaison des performances entre BF WBF et SGDBF pour la matrice (3, 10) avec une taille (700, 1000) et un rendement 0.7

Si le rendement de matrice augmente jusqu'à 0.7, comme montré sur la Figure III-13, on observe que les performances des algorithmes se dégradent par rapport au cas du rendement 0.5 illustré sur la Figure III-12. En revanche, l'écart de performance entre les algorithmes MS et SGDBF est réduit à 1.8 dB pour un TEB égal à 10^{-4}. On voit aussi que la différence des performances entre 20 itérations et 100 itérations est négligeable.

Pour répondre aux exigences des hauts débits dans le contexte du C-RAN, le code avec rendement 0.9 est enfin simulé. Comme illustré sur la Figure III-14, quand on augmente le rendement jusqu'à 0.9, on observe que les performances se dégradent comparées aux courbes correspondant à des rendements 0.7 et 0.5. Mais les performances des algorithmes MS et SGDBF se rapprochent encore avec un écart de seulement 0.8 dB pour un TEB égal à 10^{-4}. De plus, vingt itérations sont désormais suffisantes pour obtenir une bonne stabilité dans la convergence des algorithmes. Par la suite c'est ce nombre de vingt itérations qui sera retenu.

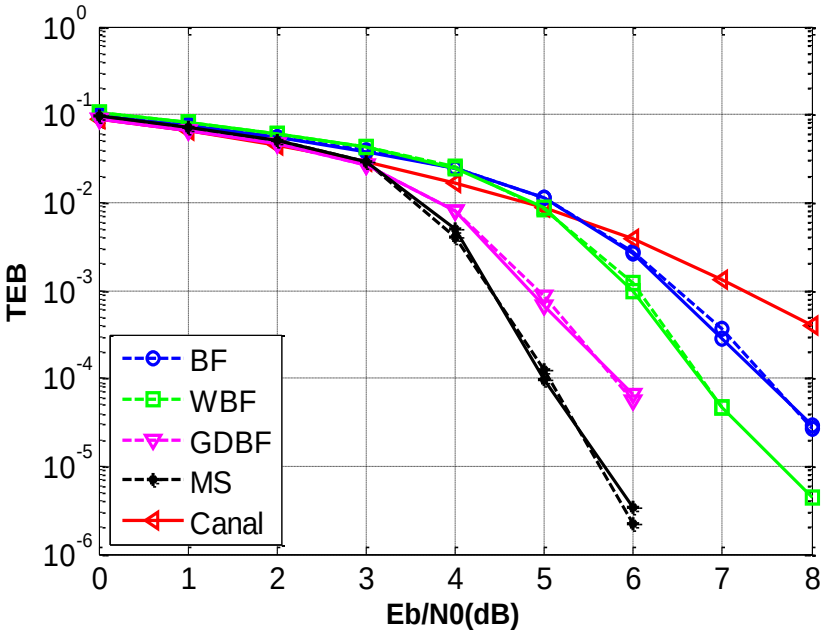


Figure III-14 : Comparaison des performances entre BF WBF et SGDBF pour la matrice (3, 30) avec une taille (900, 1000) et un rendement 0.9

Sur la Figure III-15, on a illustré les performances de l'algorithme à décisions dures le plus performant à savoir l'algorithme GDBF pour les trois rendements étudiés. A priori il semblerait qu'à chaque fois que le rendement augmente de 0.2, on est confronté à une dégradation des performances de l'ordre de 0.4 dB.

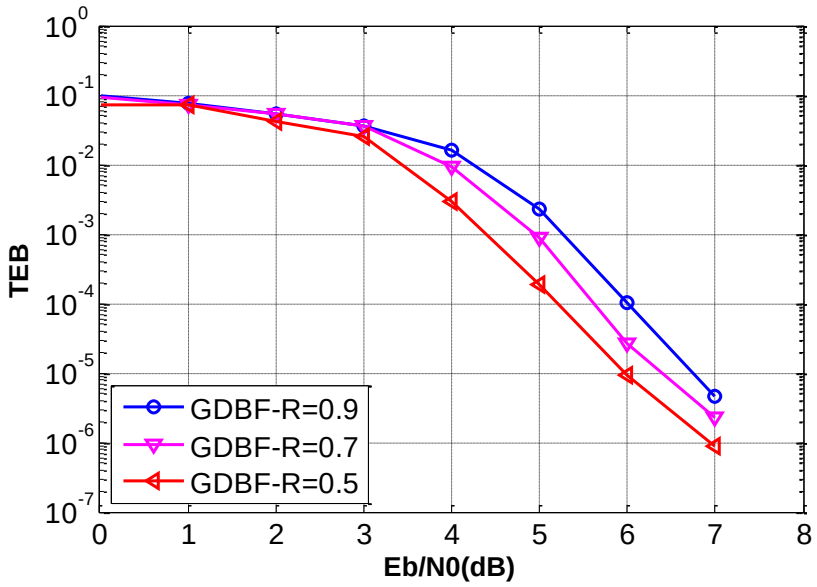


Figure III-15 : Dégradation des performances en fonction du rendement du code

Sur la Figure III-16 sont illustrées les performances des algorithmes IMWBF, RRWGDBF par rapport aux algorithmes WBF et GDBF dans le contexte des rendements les plus élevés : 0.7, 0.9. Pour IMWBF, la valeur α est prise égale à 0.2 pour les rendements 0.7 et 0.9 car nous avons trouvé que c'était la meilleure valeur (cf Figure III-18). On peut aussi observer que pour l'algorithme RRWGDBF, il converge plus rapidement et atteint de meilleures performances que l'algorithme GDBF pour un rendement de 0.7. En revanche, quand on augmente le rendement à 0.9 (Figure III-17), l'algorithme RRWGDBF converge plus rapidement au départ, mais il apparaît le phénomène d'erreur plancher quand la valeur de Eb/N_0 est située entre $6\,dB$ et $8\,dB$. Ceci est dû à la pondération β_i (III-26) dans la fonction d'inversion $\Delta k^{RRWGDBF}$, en effet la fonction d'inversion devient inefficace quand la matrice de parité n'est pas très creuse avec en plus la présence de trapping set et stopping set dans la matrice.

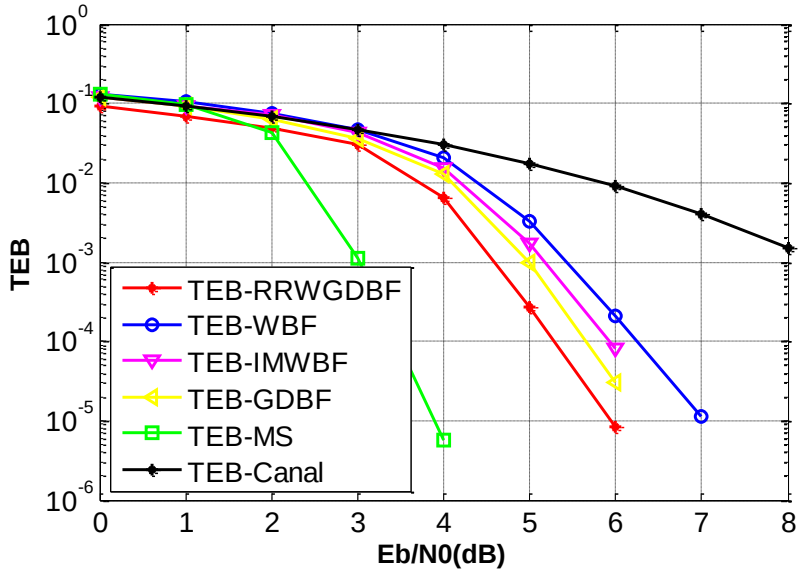


Figure III-16 : Performances des algorithmes avec rendement du code 0.7

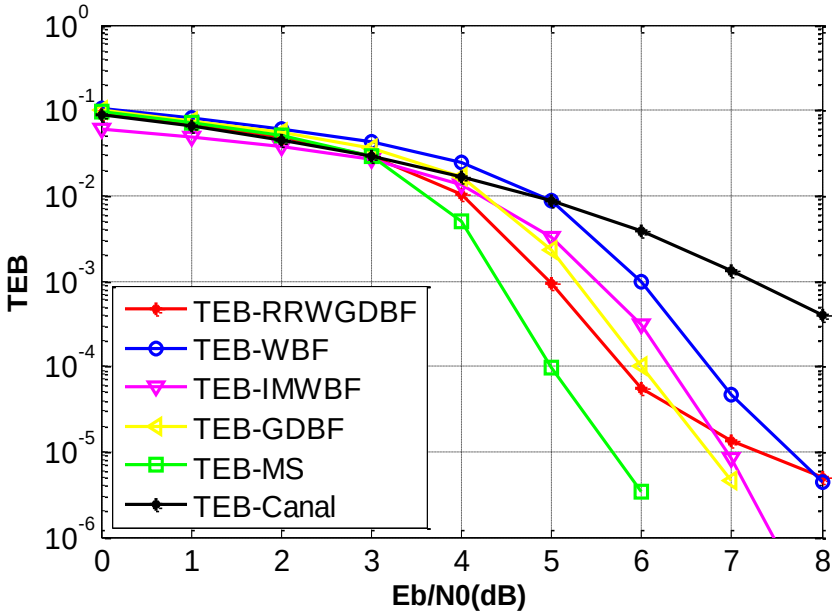


Figure III-17 : Performance des algorithmes avec rendement du code 0.9

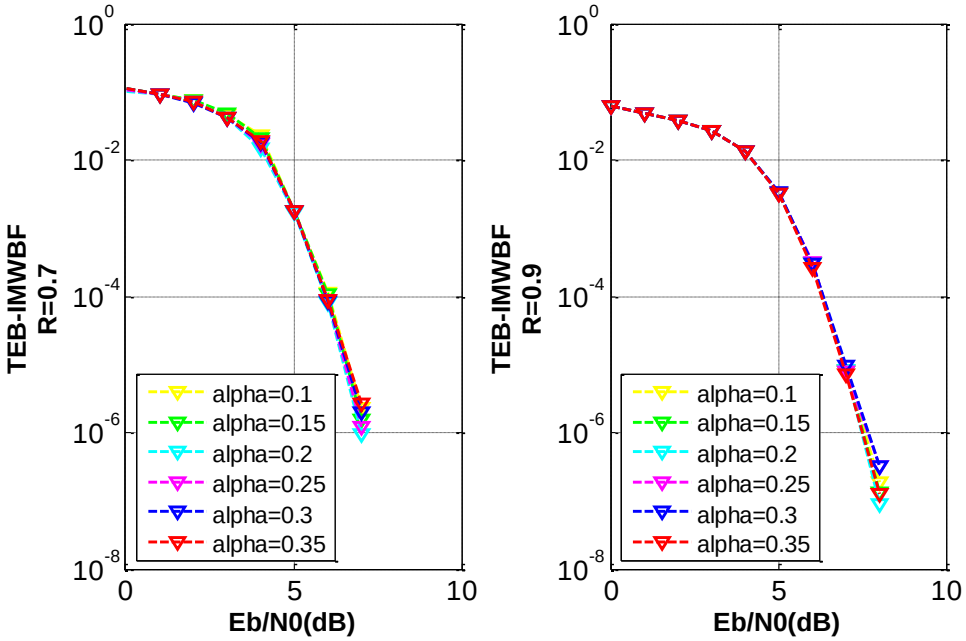


Figure III-18 : Performances de l'algorithme IMWBF en fonction de α

La Figure III-19 montre les performances de TEB en sortie en fonction de TEB en entrée. Il apparaît sur ces courbes que ce sont les performances de l'algorithme MS qui se dégradent le plus en fonction du rendement. Quand le TEB en entrée vaut 10^{-3}, le TEB en sortie de l'algorithme MS atteint 10^{-12} avec rendement du code 0.7, mais il atteint 10^{-9} avec le rendement du code égale à 0.9. Pour ces deux rendements différents, les performances de l'algorithme IMWBF s'appochent des performances de l'algorithme GDBF, on voit bien aussi l'apparition du plancher d'erreur pour l'algorithme RRWGDBF pour un rendement de 0.9.

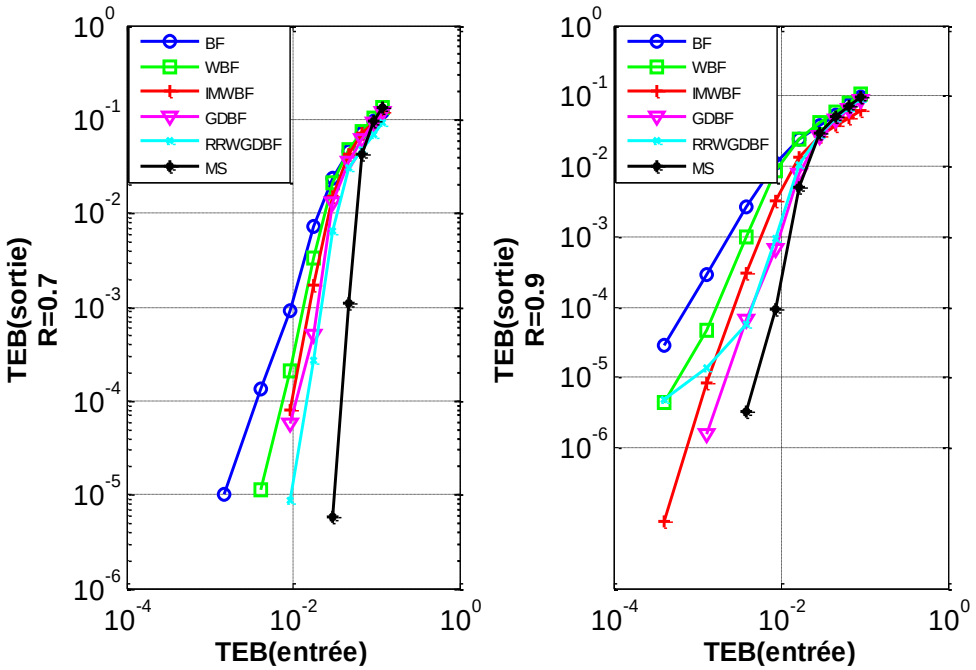


Figure III-19 : Performance de TEB entrée en fonction de TEB sortie avec les différents rendements du code

Tableau III-4 : Comparaison des différents algorithmes quand TEB en entrée vaut 10^{-3} avec rendement du code 0.7

Algorithme(R=0.7)	$TEB_S(TEB_e(10^{-3}))$
BF	$10^{-4} \rightarrow 10^{-5}$
WBF	$10^{-5} \rightarrow 10^{-6}$
IMWBF	$10^{-6} \rightarrow 10^{-8}$
GDBF	$10^{-7} \rightarrow 10^{-8}$
RRWGDBF	$10^{-8} \rightarrow 10^{-10}$
MS	$10^{-10} \rightarrow 10^{-12}$

Tableau III-5 : Comparaison des différents algorithmes quand TEB en entrée vaut 10^{-3} avec rendement du code 0.9

Algorithem(R=0.9)	$TEB_S(TEB_e(10^{-3}))$
BF	$10^{-3} \rightarrow 10^{-4}$
WBF	$10^{-4} \rightarrow 10^{-5}$
IMWBF	$10^{-5} \rightarrow 10^{-6}$
GDBF	$10^{-6} \rightarrow 10^{-7}$
RRWGDBF	10^{-5}
MS	$10^{-7} \rightarrow 10^{-9}$

III.2.5. Conclusion

Dans ce chapitre nous avons introduit les différents algorithmes à décisions dures comme BF, WBF GDBF et ses améliorations PBF, MWBF, IMWBF, IHGDBF, RRWGDBF etc... Les algorithmes sont basés sur différentes fonctions d'inversion, et ce sont ces différentes fonctions d'inversion qui influent grandement sur les performances. Généralement, la fonction d'inversion est conçue pour indiquer les bits probablement erronés, elle est basée sur les calculs de syndrome : $\sum_{i\in M(k)} \prod_{j\in N(i)} x_j = \sum S_i$. Si la matrice n'est plus très creuse, c'est à dire quand on augmente le rendement du code jusqu'à des valeurs de 0.9, il apparaît des cycles dans la matrice de parité qui vont dégrader la performance du code, donc l'indication $\sum S_i$ n'est plus fiable pour détecter les bits erronés. Dans l'équation $\Delta k^{général}$, la partie $\emptyset(x_k, y_k)$ est là pour évaluer la fiabilité des bits reçus afin d'augmenter le taux de détection des bits erronés. Précisément pour réduire l'impact des cycles dans la matrice de parité, il y a en gros deux méthodes :

- Ajouter et définir des pondérations ω et α [141]-[147][152] pour évaluer les fiabilités des syndromes calculés et des bits reçus, ces paramètres ont pour effet d'élargir les distances entre les bits erronés et les bits corrects (Figure III-4). Par contre, il faut bien justifier le rapport entre indication des fiabilités des bits $\emptyset(x_k, y_k)$ et indications des syndromes $\sum S_i$, sinon la fonction d'inversion ne peut pas bien justifier les bits erronés dans les zones à forts Eb/N_0, il va probablement apparaître le phénomène d'erreur plancher (error-floor) comme pour la performance de l'algorithme RRWGDBF montrée en Figure III-17.
- Définir un seuil (threshold) pour inverser un ensemble de bits dans une itération, cette méthode noté M-XBF peut réduire les impacts des cycles et permettre une convergence plus rapide pour le décodage [148][151]. Le seuil peut être défini une seule fois au départ du décodage ou bien peut être renouvelé à chaque itération.

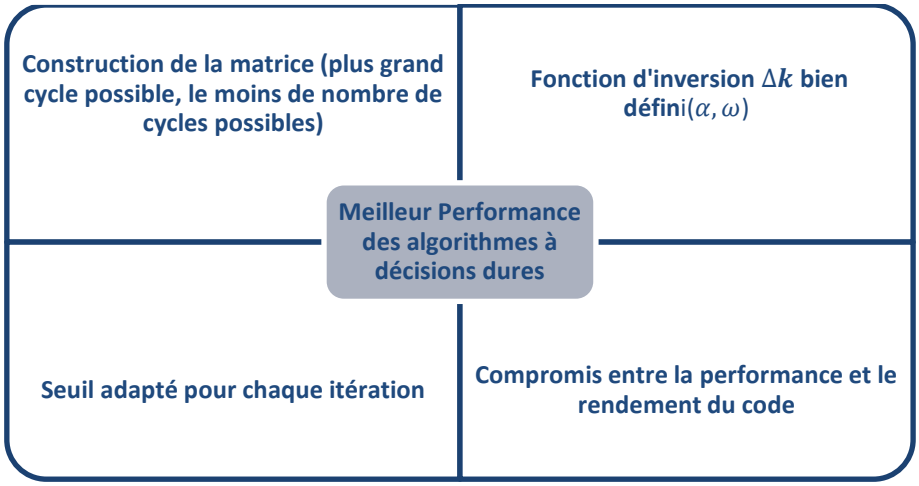


Figure III-20 : Eléments clés pour atteindre la meilleure performance

Sur la Figure III-20 on a montré tous les éléments pour améliorer les performances des algorithmes à décisions dures. Dans ce chapitre, on a aussi donné les performances de ces méthodes, et nous avons montré pourquoi l'algorithme GDBF est finalement choisi pour le reste de nos études dans le contexte C-RAN en tant que meilleur compromis performances/complexité d'implémentation.

Chapitre IV

Application avec CAN 2-bit

Chapitre IV. Application avec CAN 2-bits

IV.1. Introduction

Dans tous les systèmes de télécommunications réels, les messages sont transmis de façon analogique ou de façon numérique. Ces valeurs flottantes (messages transmis qui sont impactés par les bruits) sont enregistrées avec un nombre de bits qui est limité par le matériel ou bien par le périphérique. Typiquement, dans le cadre de cette thèse, on s'intéresse à la partie récepteur/décodeur. Donc un CAN (Convertisseur Analogique Numérique) est introduit dans notre chaîne de décodage.

La résolution du CAN est un paramètre très important qui décide grandement de la performance des algorithmes de décodage. Dans ce chapitre on va faire les études sur l'algorithme GDBF et ses variantes en utilisant une conversion analogique/numérique (CAN) sur deux bits. En plus, on proposera un algorithme BWGDBF (Balance Weighted GDBF) pour approcher la performance de l'algorithme GDBF sans CAN et pour réduire le nombre d'itérations utilisées dans le décodage afin de réduire la latence dans le circuit.

IV.2. CAN et modélisation du canal

Le CAN est un dispositif électronique permettant la conversion d'un signal analogique en un signal numérique.

Pour la suite des études, le canal AWGN (Additive White Gaussian Noise) est remplacé par un canal discret. Etant donné la faible performance des algorithmes à décisions dures, nous utiliserons un CAN pour améliorer ces performances. Il est constaté par la simulation qu'avec 2 bits une amélioration remarquable est obtenue. D'un point de vue d'implémentation, il est facile de combiner 3 comparateurs afin d'obtenir 4 symboles à partir du signal reçus y_k. Donc la résolution du convertisseur CAN est définie sur 2 bits.

Par conséquent, on peut avoir quatre symboles : 00, 01, 10, 11 par rapport aux signaux analogique reçus comme montré en Figure IV-1.

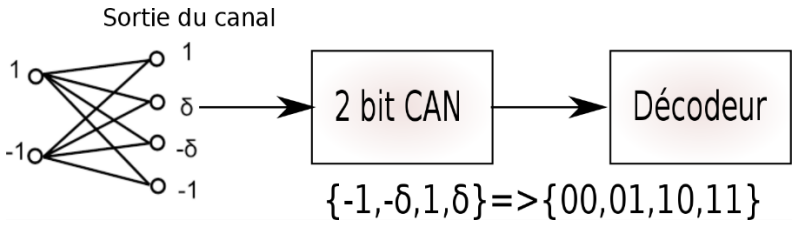


Figure IV-1 : 2 bit CAN

Au lieu de présenter le TEB en fonction de Eb/N_0, on peut exprimer le TEB en fonction de la probabilité d'erreur P_b sur le canal AWGN :

$$P_b = \int_{-\infty}^{0} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-1)^2}{2\sigma^2}} dx = \int_{1}^{\infty} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}} dx \qquad (IV\text{-}1)$$

A partir de cette équation, on peut calculer différentes probabilités d'erreur pour différentes valeurs de seuil delta (δ). En utilisant la fonction de répartition de la queue d'une Gaussienne, on peut déduire la valeur de variance du bruit(σ) par l'inversion de la Q-fonction :

$$\sigma = \frac{1}{Q^{-1}(P_b)} \qquad (IV\text{-}2)$$

Selon la distribution de probabilité montrée en Figure IV-2, on peut obtenir les régions de quantification sous la forme :

$$\begin{cases} y \in (\delta, +\infty), symbole = 1 \\ y \in (0, \delta), symbole = \delta \\ y \in (-\delta, 0), symbole = -\delta \\ y \in (-\infty, -\delta), symbole = -1 \end{cases} \qquad (IV\text{-}3)$$

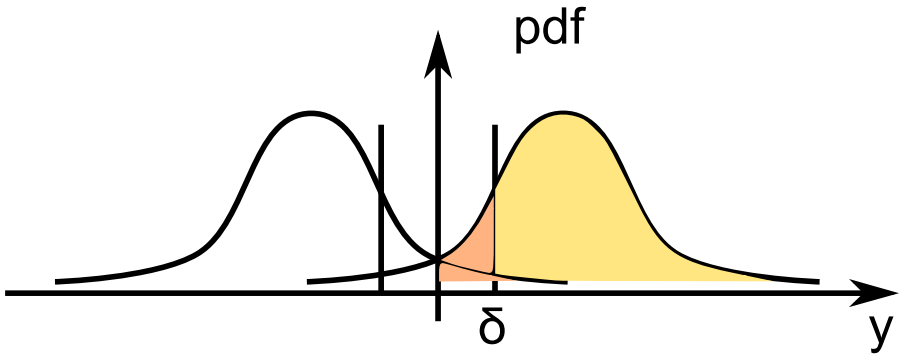


Figure IV-2 : Distribution de probabilité en fonction de δ

Donc avec ces règles, on peut déduire les probabilités d'erreur quand on reçoit les symboles 1 ou -1. Le bit émis vaut 1 ou -1 après la modulation BPSK :

$$P_{1\to1} = P_{-1\to-1} = \int_{\delta}^{\infty} \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-1)^2}{2\sigma^2}} dx = Q\left(\frac{\delta-1}{\sigma}\right)$$

$$P_{1\to\delta} = P_{-1\to-\delta} = Q\left(-\frac{1}{\sigma}\right) - Q\left(\frac{\delta-1}{\sigma}\right) \qquad (IV\text{-}4)$$

$$P_{1\to-\delta} = P_{-1\to\delta} = Q\left(\frac{-\delta-1}{\sigma}\right) - Q\left(-\frac{1}{\sigma}\right)$$

$$P_{1\to-1} = P_{-1\to1} = Q\left(\frac{-\delta-1}{\sigma}\right)$$

Il est important d'étudier l'impact de la valeur de δ sur les performances des algorithmes, donc dans la prochaine partie on va présenter les performances en fonction de δ.

IV.2.1. Performances de l'algorithme GDBF en fonction de delta

Dans cette partie on va montrer les performances de GDBF en fonction de delta pour la matrice de parité H avec les différents rendements du code : $0.5, 0.7, 0.9$.

Les performances de l'algorithme GDBF pour les différents delta δ sont illustrées en Figure IV-3, on peut voir que pour un rendement du code égal à 0.5, la méthode GDBF atteint la meilleure performance tandis que delta est entre 0.3 et 0.4. En plus, les performances avec CAN se degradent nettement par rapport à la performance analogique (la performance normale sans CAN).

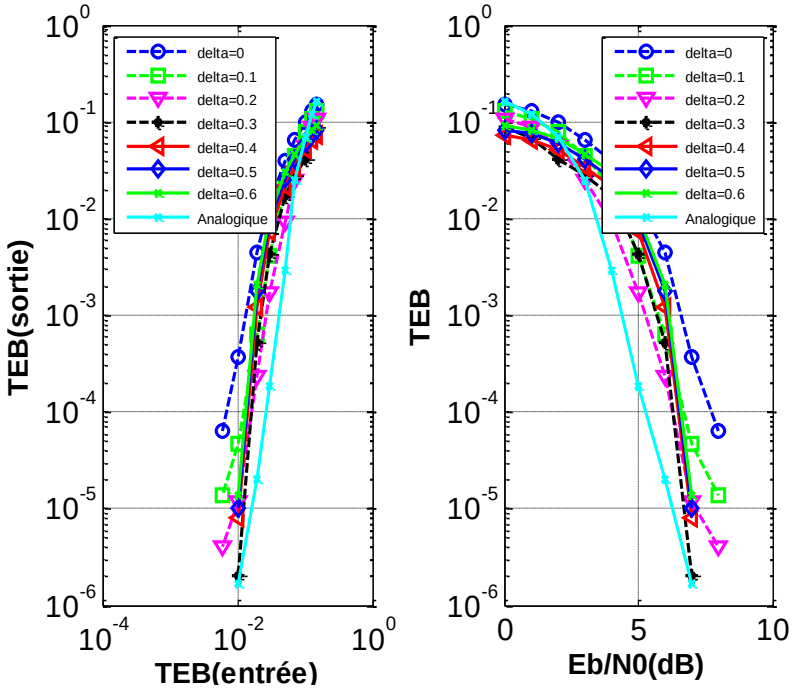


Figure IV-3 : Les performances en fonction de delta avec rendement du code 0.5 et 60 itérations

La Figure IV-4 montre les performances de l'algorithme GDBF avec différents δ pour le rendement du code 0.7. On a observé la dégradation entre les courbes avec CAN et la courbe sans CAN. Pour ce rendement du code, la méthode GDBF conduit à la meilleure performance quand delta est égal à 0.3.

La Figure IV-5 montre les performances de l'algorithme GDBF avec différents δ pour le rendement du code égal à 0.9. Les dégradations des performances peuvent être négligées. En plus, l'algorithme GDBF atteint la meilleure performance quand delta est situé entre : 0.2 et 0.3.

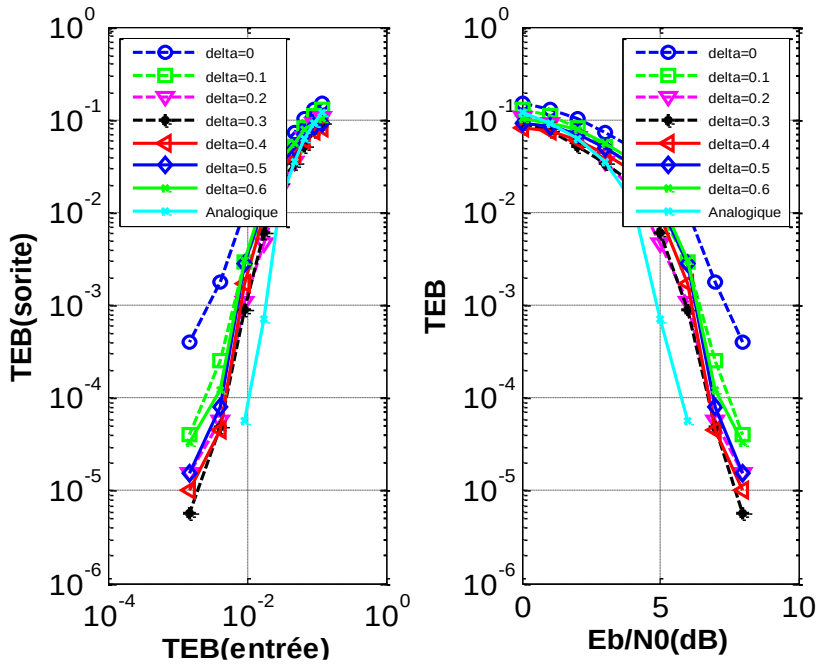


Figure IV-4 : Les performances en fonction de delta avec rendement du code 0.7 et 30 itérations

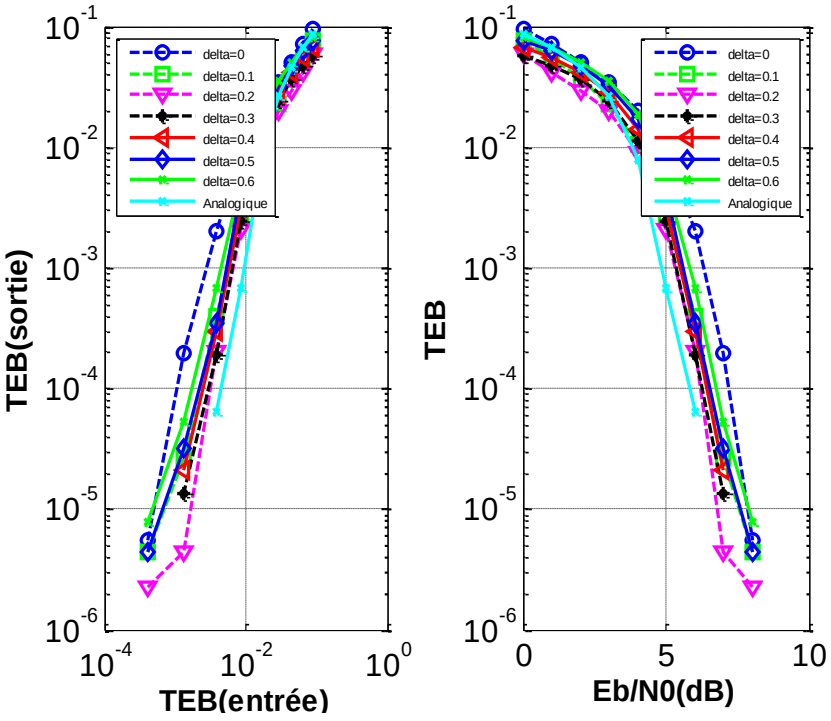


Figure IV-5 : Les performances en fonction de delta avec rendement du code 0.9 et 20 itérations

IV.3. Amélioration des algorithmes

Dans le cas normal (sans CAN), la méthode GDBF choisit le bit qui a pour valeur le minimum de la fonction d'inversion Δk^{GDBF} à inverser. En revanche, plusieurs bits peuvent avoir la valeur minimale de Δk^{GDBF} ce qui peut conduire en fonction de l'ensemble des bits à inverser à la création de trapping set ou stopping set qui vont empêcher la convergence de l'algorithme.

Comme on l'a représenté dans la partie précédente, le papier [149] a proposé un algorithme HGDBF qui inverse d'abord en mode single et puis en mode multiple en évaluant la valeur de la fonction d'objectif ou en fixant un seuil. Cette méthode peut améliorer un peu la performance en créant une opportunité de quitter un maximum local afin d'atteindre le maximum global et il peut aussi réduire la latence du décodage. Donc en suivant cette méthode, on va proposer un algorithme qui utilise le mode single et le mode multiple. Au lieu d'utiliser un seuil optimisé θ_{op} par simulation pour le mode multiple (il dépend des différents rendements du code, différentes tailles des matrices de parité, du poids des colonnes et des lignes et du nombre des cycles dans la matrice), on utilise le nombre des bits maximum à inverser dans Figure II-35 chaque itération.

Etant donné que le cycle minimum dans notre matrice de parité est de 6, pour éviter de créer un nouveau cycle 6 dans les bits reçus, le nombre maximal des bits à inverser doit être inférieur ou égal à 3 :

$$B = \{n_1, n_2, n_3 |\ \Delta k^{GDBF}(x) = \Delta k_{min}^{GDBF}(x)\} \qquad \textit{(IV-5)}$$

D'où n_1, n_2, n_3 sont choisis aléatoirement parmi les bits détectés.

Quand on augmente le rendement du code, par exemple 0.7 et 0.9, si on inverse au maximum 3 bits au départ de l'algorithme, cela peut aussi parfois créer des trapping set ou stopping set, parce que la matrice de parité n'est pas très creuse. Donc on décompose notre algorithme en deux méthodes :

SM mode : Single mode puis multiple mode, noté SMGDBF.

MS mode : Multiple mode puis single mode, noté MSGDBF.

Il est très important de connaitre à quel moment on change le mode (Les résultats suivants sont basés sur les codes avec rendements du code égaux à 0.7 et 0.9). Au lieu d'utiliser la fonction d'objectif pour évaluer si les corrections des bits sont efficaces (la valeur de fonction d'objectif s'accroît quand le nombre d'itération augmente), on peut utiliser la valeur $\sum S_i$. Comme on a illustré en partie des décodages durs, S_i vaut 1 si les bits participant à ce noeud de parité sont corrects, si tous les bits sont corrects (l'algorithme atteint la meilleure performance), $\sum S_i$ vaut M (nombre des noeuds de parité dans la matrice), donc si on a bien corrigé un bit, la valeur $\sum S_i$ augmente. Pour l'algorithme GDBF et ses variantes, quand la fonction d'inversion arrive à un local maximum, on constate l'oscillation de la valeur de $\sum S_i$.

La Figure IV-6Figure IV-5 et la Figure IV-7 montrent la valeur de la sommation des syndromes $\sum S_i$ en fonction du nombre d'itérations pour les différents rendements du code quand TEB (entrée) vaut 10^{-2} (où on commence à améliorer la performance : zone error-floor), les courbes pointilles représente les cas moins fréquents (les résultats des simulations sont faits par la méthode GDBF avec 2 bits CAN).

Selon la Figure IV-6 , on peut obtenir que pour le rendement du code 0.7, $\sum S_i$ peut atteindre la valeur maximum M $(285 \leq M \leq 300)$ si le nombre d'itérations vaut 30 en utilisant le single

mode et environ 20 itérations en utilisant le multiple mode. En plus, le mode multiple et le mode single peuvent corriger tous les bits. On change le mode quand la pente des courbes commence à réduire. Donc on change single mode à multiple mode après 15 iterations pour la méthode SMGDBF, et changement du mode après 10 itérations pour la méthode MSGDBF.

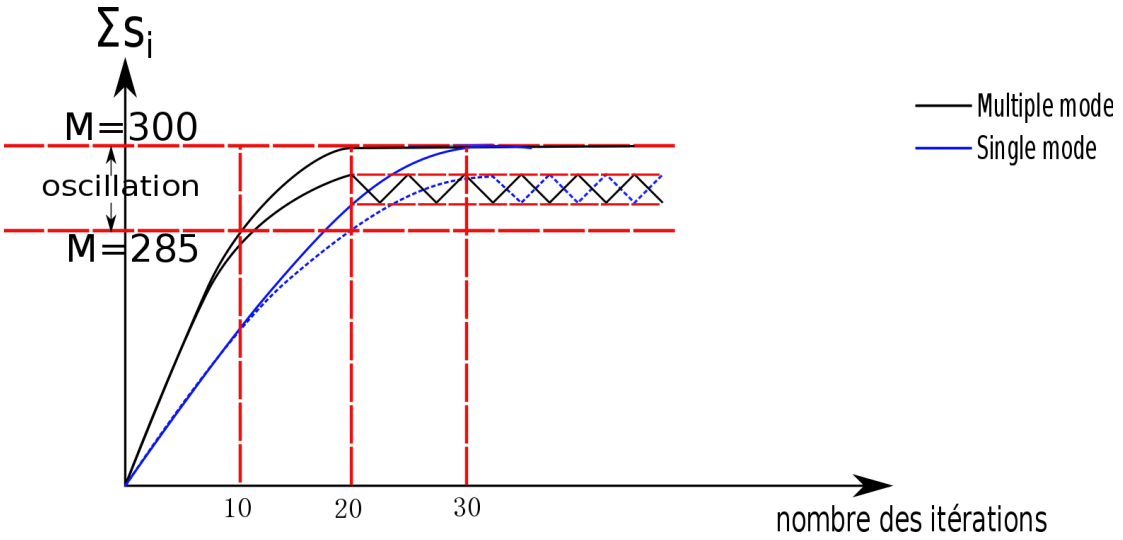


Figure IV-6 : $\sum S_i$ en fonction de nombre d'itérations pour les différents modes avec rendement du code 0.7 quand TEB(entrée) vaut 10^{-2}

Selon les simulations, on donne aussi Tableau IV-1 suivant pour caractériser la méthode en fonction des différentes itérations avec différents TEB (entrée) :

Tableau IV-1 : Saturations des méthodes SMGDBF et MSGDBF en fonction des différents TEB(entrée) et itérations pour rendement du code 0.7

TEB(entrée)	Méthode			
	SMGDBF ($i^{ème}$ $itération$)	MSGDBF ($i^{ème}$ $itération$)	Nombre d'itérations total	
			SMGDBF	MSGDBF
10^{-2}	$15^{ème}$	$10^{ème}$	22	18
$5 * 10^{-3}$	$11^{ème}$	$8^{ème}$	18	15
10^{-3}	$9^{ème}$	$6^{ème}$	14	12

On peut observer que la convergence s'accélère (la convergence de MSGDBF est toujours meilleur que la convergence de SMGDBF) quand on réduit le TEB (entrée). Dans ce cas, le nombre d'itérations diminue également pour atteindre un maximum local ou global.

Pour le rendement du code 0.9 avec 10^{-2} de TEB (entrée) affiché en Figure IV-7, il est très difficile d'atteindre la valeur maximum de $\sum S_i$ (global maximum quand $M = 100$) en utilisant un seul mode. Comparé avec la Figure IV-6, pour le mode SMGDBF, on change le mode après la $10^{ème}$ itération, pour le mode MSGDBF, on change le mode après la $6^{ème}$ itération.

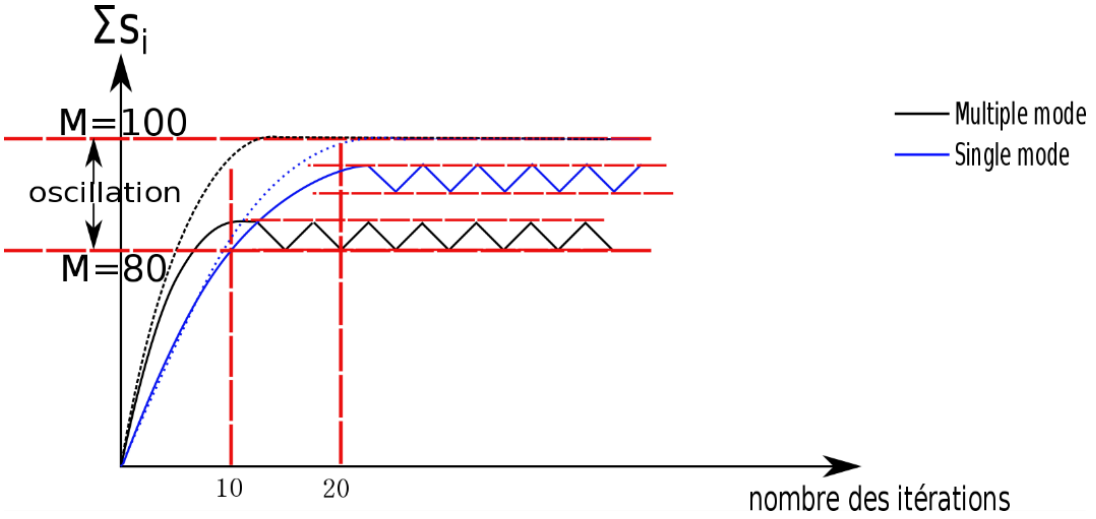


Figure IV-7 : $\sum S_i$ en fonction de nombre d'itérations pour les différents modes avec rendement du code 0.9 quand TEB (entrée) vaut 10^{-2}

Un tableau est donné ci-dessous afin d'étudier la variation du nombre d'itérations total en fonction des différents TEB(entrée).

Tableau IV-2 : Saturations des méthodes SMGDBF et MSGDBF en fonction des différents TEB (entrée) et itérations pour rendement du code 0.9

TEB(entrée)	Méthode			
	SMGDBF ($i^{ème}$ $itération$)	MSGDBF ($i^{ème}$ $itération$)	Nombre d'itérations total	
			SMGDBF	MSGDBF
10^{-2}	$10^{ème}$	$6^{ème}$	15	13
$5 * 10^{-3}$	$6^{ème}$	$4^{ème}$	11	10
10^{-3}	$4^{ème}$	$3^{ème}$	9	9

Comparé avec le Tableau IV-2, on peut obtenir qu'ils aient besoin d'un nombre moindre d'itérations pour arriver à un maximum local (zone d'oscillation), on peut aussi observer que quand on réduit la valeur de TEB(entrée), les nombres totaux d'itérations de la méthode SMGDBF et de la méthode MSGDBF sont quasiment pareils.

On peut conclure que la méthode SMGDBF et la méthode MSGDBF peuvent converger plus rapidement pour le décodage (elles utilisent moins d'itérations pour atteindre les maxima locaux ou globaux). Ici on prend le pire cas (TEB(entrée) vaut 10^{-2}) afin de généraliser les algorithmes SMGDBF et MSGDBF pour tous les TEB(entrée) montrés en Figure IV-8 et Figure IV-9. Le maximal nombre des bits B à inverser dans chaque itération est pris égal à 3 quand on est en mode multiple.

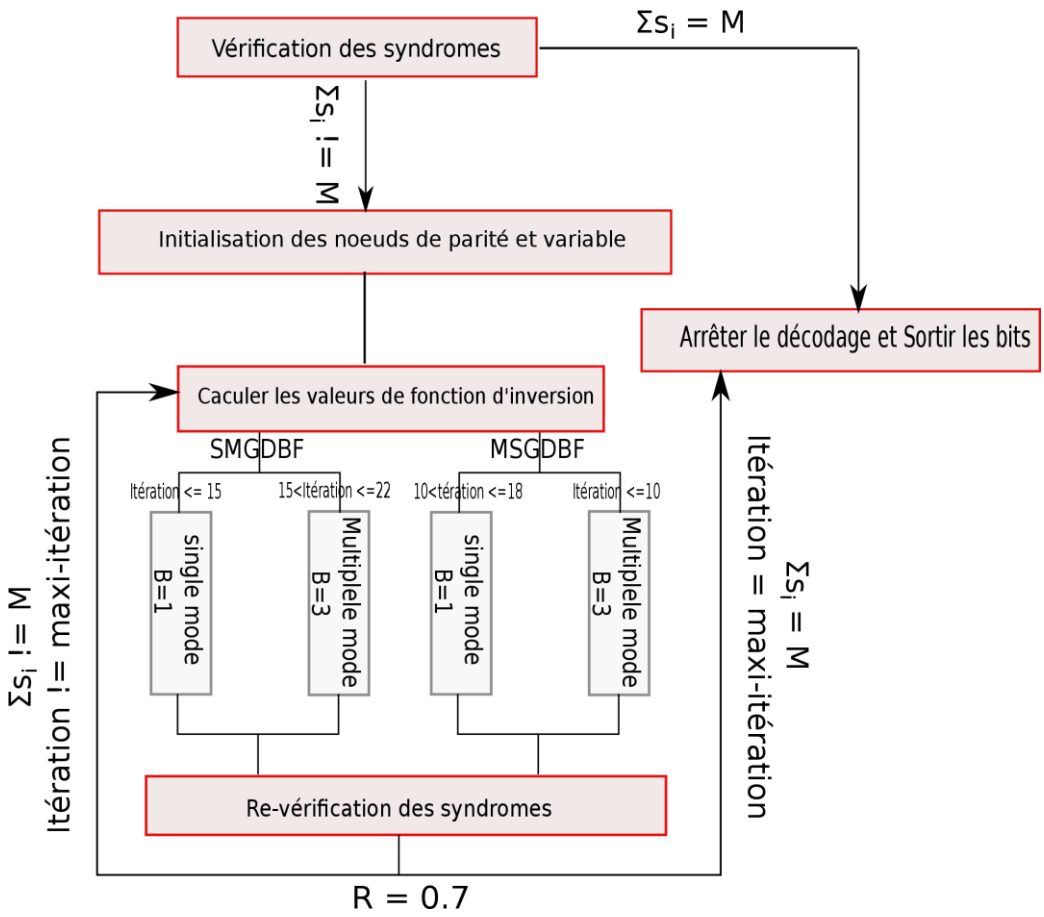


Figure IV-8 : Algorithmes SMGDBF et MSGDBF pour un rendement de code égal à 0.7

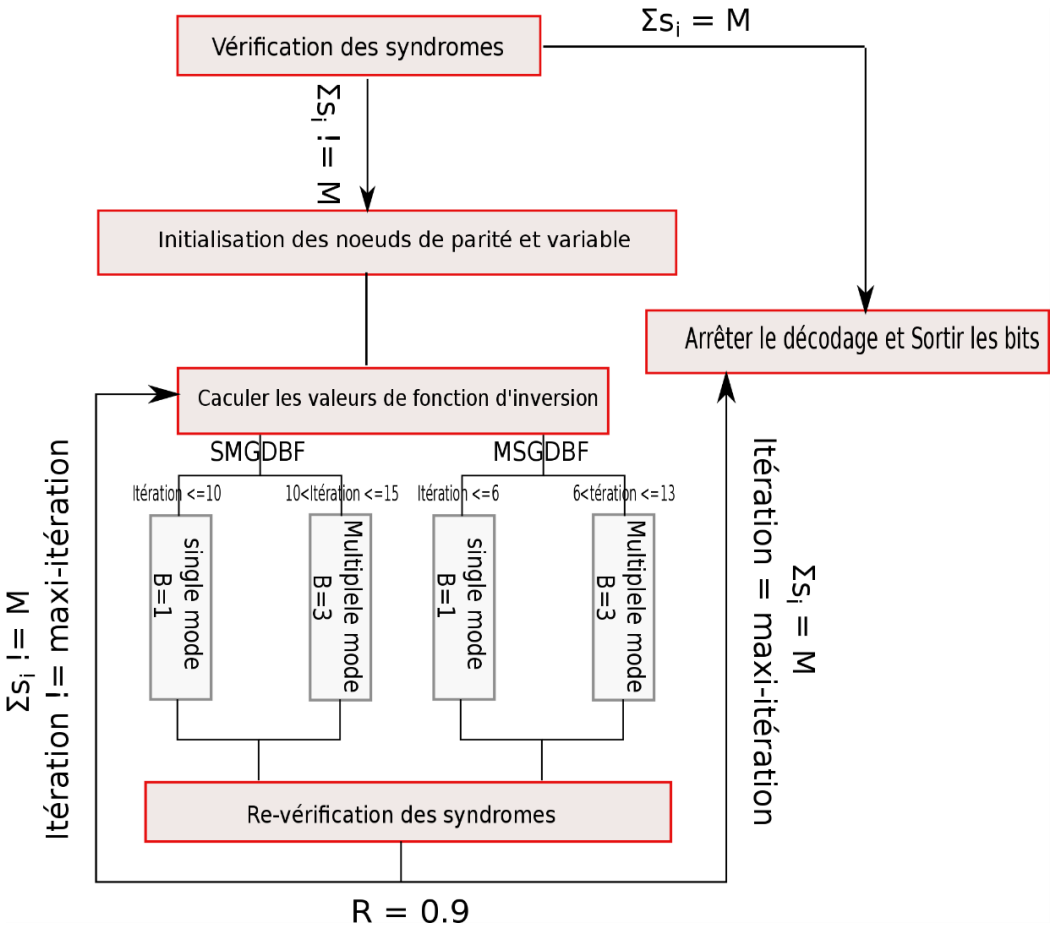


Figure IV-9 : Algorithmes SMGDBF et MSGDBF pour un rendement de code égal à 0.9

IV.3.1. Performance des algorithmes améliorés

Dans cette partie, on va montrer les performances en faisant varier le paramètre delta avec les méthodes SMGDBF et MSGDBF.

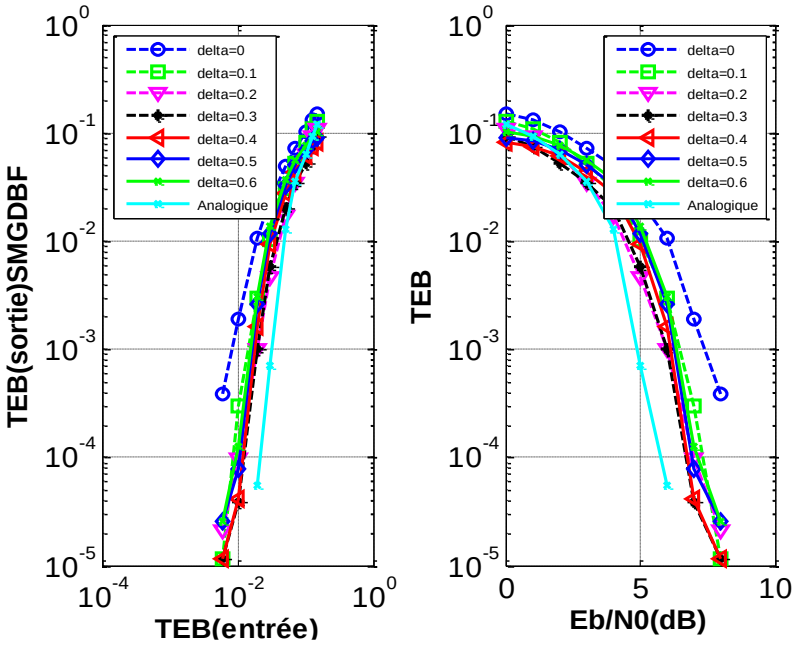


Figure IV-10 : Performances des différents deltas pour la méthode SMGDBF avec rendement du code 0.7

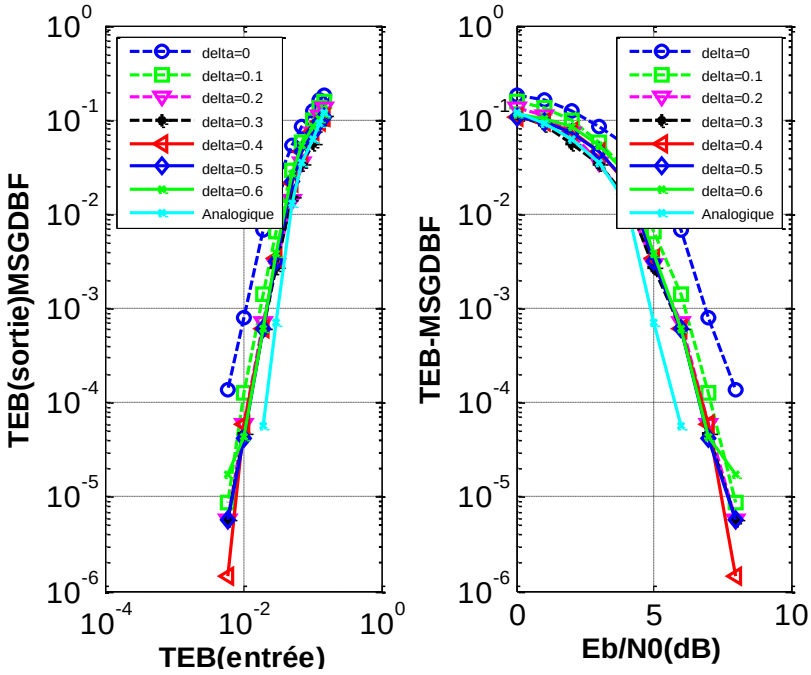


Figure IV-11 : Performances des différents deltas pour la méthode MSGDBF avec le rendement du code 0.7

Les Figure IV-10 et Figure IV-11 montrent les performances en variant la valeur de delta pour les deux différentes méthodes : SMGDBF et MSGDBF. On peut observer que pour ces deux méthodes, les courbes atteignent les meilleures performances quand delta vaut 0.4.

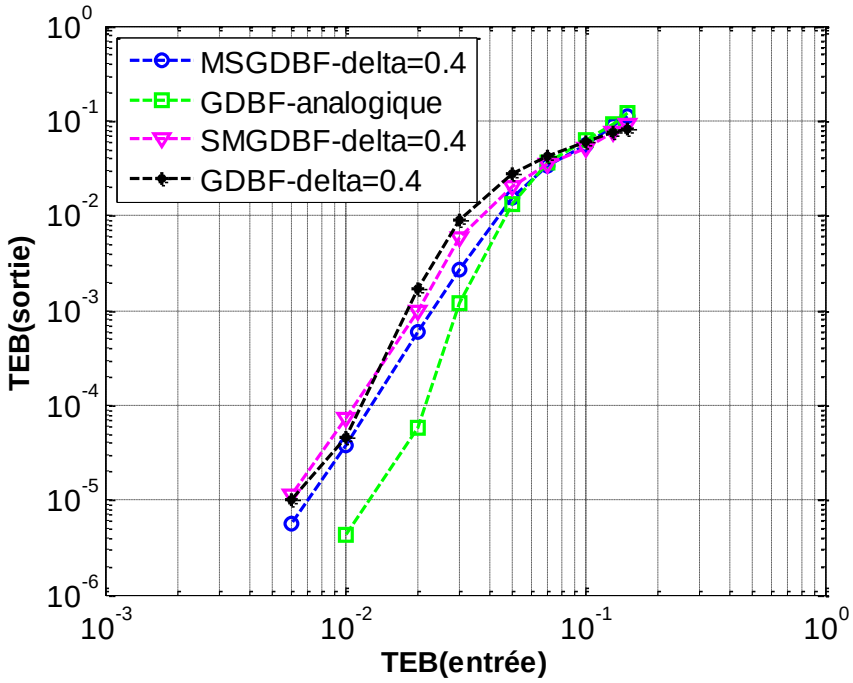


Figure IV-12 : Comparaison des performances entre les algorithmes MSGDBF, GDBF, SMGDBF avec CAN et l'algorithme GDBF sans CAN (analogique) pour rendement du code 0.7

On peut observer, en Figure IV-12, que les performances des algorithmes avec CAN comme MSGDBF, SMGDBF, GDBF se dégradent par rapport à la performance de l'algorithme GDBF analogique (sans CAN). Parmi ces méthodes avec CAN, c'est l'algorithme MSGDBF qui permet d'obtenir la meilleure performance.

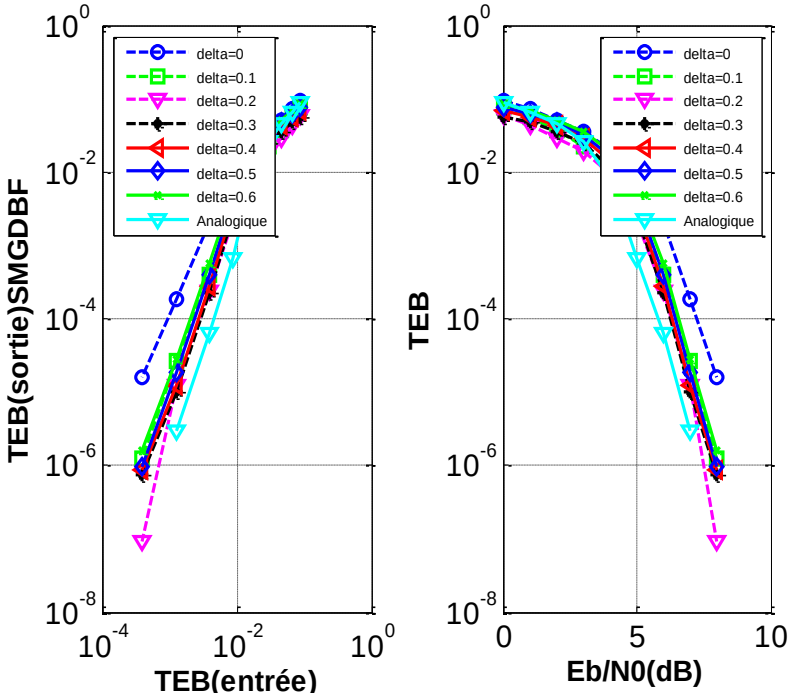


Figure IV-13 : Performances des différents δ pour la méthode SMGDBF avec le rendement du code 0.9

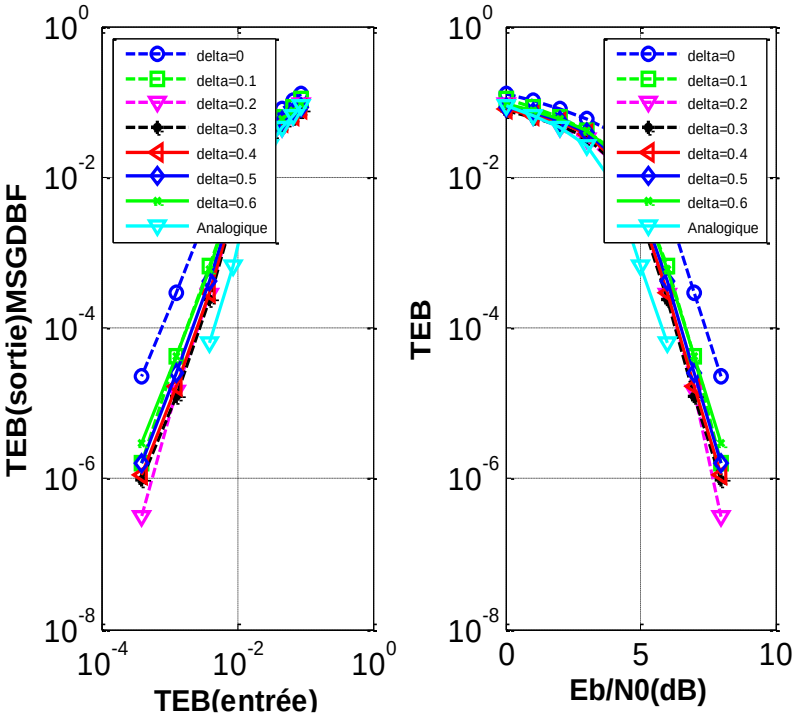


Figure IV-14 : Performances des différents δ pour la méthode MSGDBF avec le rendement du code 0.9

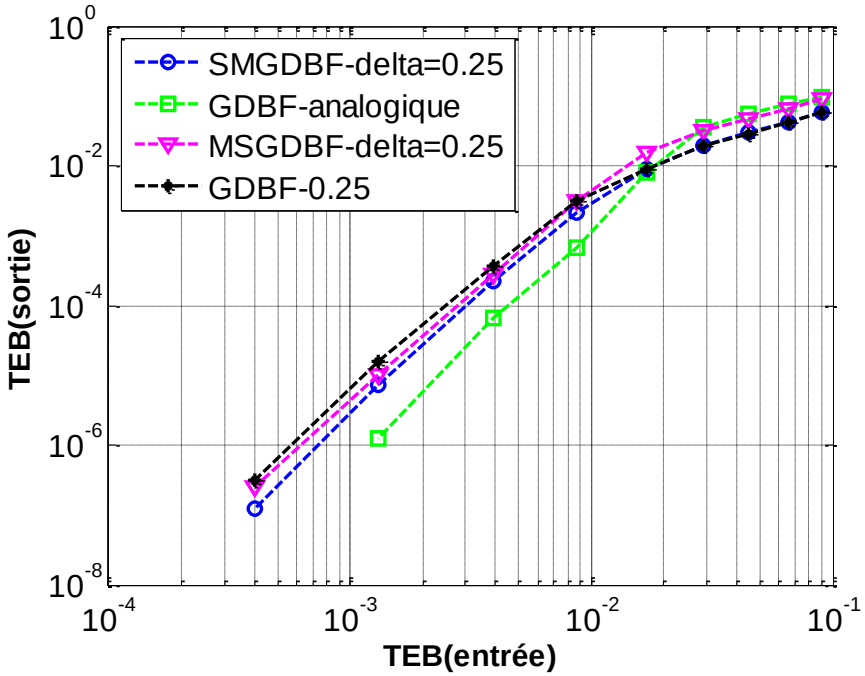


Figure IV-15 : Comparaison des performances entre les algorithmes MSGDBF, GDBF, SMGDBF avec CAN et l'algorithme GDBF sans CAN (analogique) pour le rendement du code 0.9

Comparé avec les performances pour le rendement du code 0.7, les algorithmes SMGDBF et MSGDBF atteignent la meilleure performance quand delta est entre 0.2 et 0.3 pour le

rendement du code 0.9. En vue d'une implémentation pratique on prendra delta égal à 0.25. Selon la Figure IV-13, Figure IV-14 et Figure IV-15, on peut observer que la méthode SMGDBF converge plus rapidement que la méthode MSGDBF dans la zone des SNR's faibles. En plus, les performances des MSGDBF, SMGDBF, GDBF et GDBF sans CAN sont très proches. Pour les méthodes avec CAN, la méthode SMGDBF est legèrement meilleure que les autres méthodes.

Dans les parties suivantes, on va surtout faire des études sur le rendement du code 0.9, donc on choisira la méthode SMGDBF avec une valeur de delta égale à 0.25.

IV.4. Optimisation

IV.4.1. Nombre des bits optimisés en mode multiple

Le but des décodages itératifs est de faire chercher parmi les mots de code ceux qui sont les plus proches des messages reçus comme il est montré en Figure IV-16.

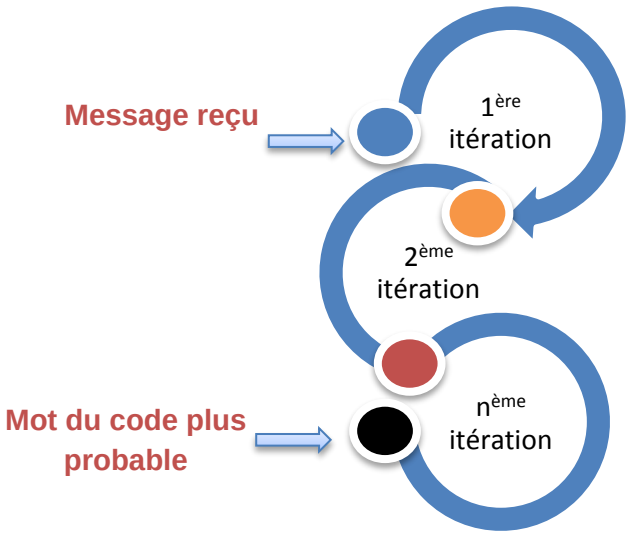


Figure IV-16 : Shéma du décodage

On a défini dans la dernière partie un nombre des bits égal à 3 en mode multiple afin d'éviter de créer le nouveau cycle. Si la matrice de parité est sans cycle (cycle free), on peut très facilement voir que s'il y a apparition d'une erreur la sommation des syndromes se réduit par un facteur qui est égal à la distance entre le syndrome correct et le syndrome erroné multiplié par D_s et le poid de la colonne P_c. Par exemple, on prend une matrice de parité $(3{,}30)$ donc avec un poids de colonne égal à 3 et un poids de ligne qui est égal à 30. On utilise une modulation BPSK, c'est à dire le syndrome correct vaut 1 et le syndrome erroné vaut -1. Par conséquent, la sommation des syndromes se reduit à: $6(3 * (1 - (-1)))$ quand il y a un bit erroné. Donc le nombre des bits N_e à inverser s'exprime par :

$$N_e = ceil(^{(M-S)}/_{D_s * P_c}) \qquad (IV\text{-}6)$$

Où $ceil(x)$ représente l'entier immédiatement supérieur à x, M est la valeur de la sommation des sydromes quand tous les bits sont corrects (il est toujours égal au nombre des noeuds de parité), S est la valeur de la sommation des syndromes calculée dans cette itération.

Cette équation n'est pas vraie quand il y a des cycles dans la matrice de parité. Dans la zone à faibles SNR's, il est beaucoup plus facile de se tromper. Comme montré en Figure IV-17, s'il y a 3 erreurs apparues dans le cycle 6 de la matrice de parité. Au lieu des 3 bits détectés comme les bits erronés, on détecte just un bit erroné $((6-0)/6)$.

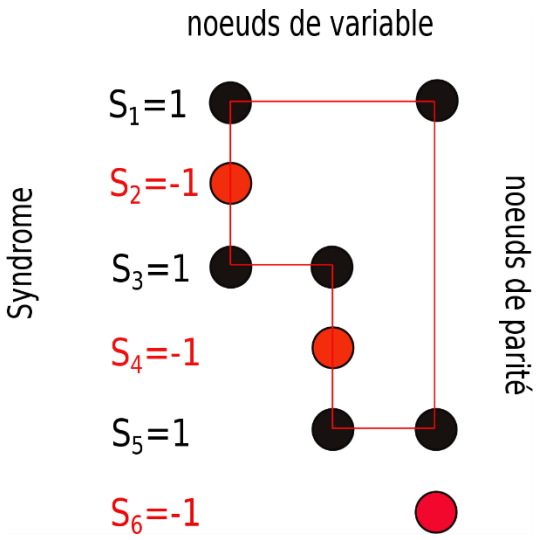


Figure IV-17 : Défaillance de la détection des bits

Les Figure IV-18, Figure IV-19 et Figure IV-20 montrent les taux de détection corrects des bits erronés avec différents TEB(entrée) par la formule (IV-6). On peut observer que à cause des bits nombreux erronés quand TEB(entrée) est très grand ($\frac{E_b}{N_0} < 4.5dB$ ou TEB(entrée) $\geq 5 * 10^{-2}$), la formule n'arrive pas à détecter des bits erronés à cause de la présence de trapping set et stopping set. En revanche, cette équation est quand même une façon d'estimer les nombres des bits à inverser quand TEB(entrée) est très petit ($\frac{E_b}{N_0} \geq 4.5dB$ ou TEB(entrée) $< 5 * 10^{-2}$).

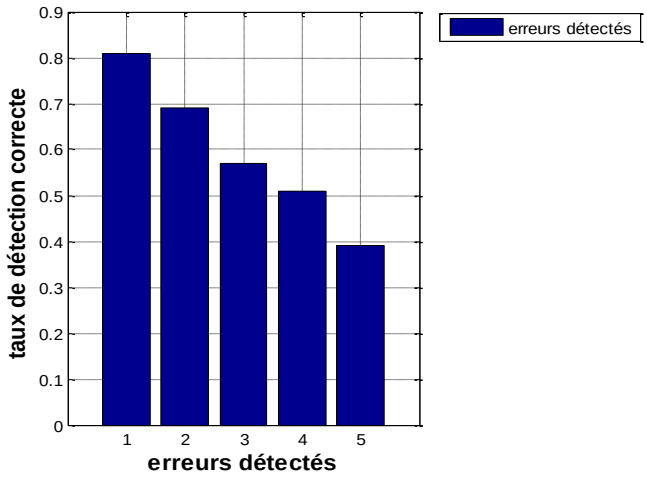


Figure IV-18 : Taux de détection des bits erronés avec TEB(entrée)=10^{-2} pour le rendement du code égal à 0.9

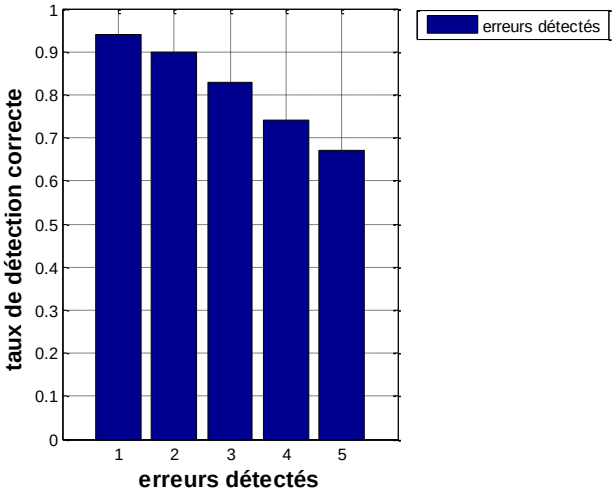


Figure IV-19 : Taux de détection des bits erronés avec TEB(entrée)=5.10^{-2} pour le rendement du code égal à 0.9

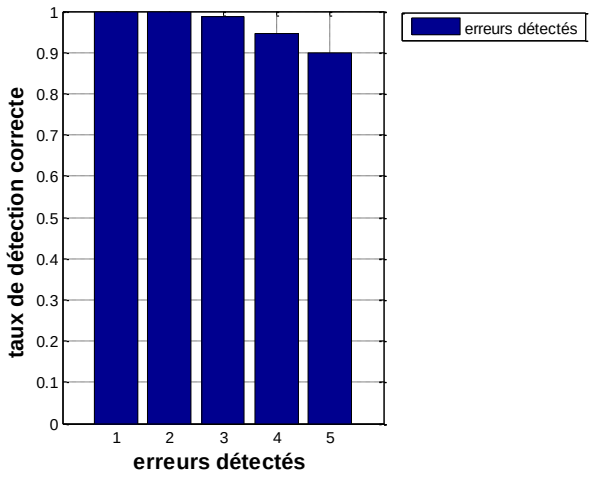


Figure IV-20 : Taux de détection des bits erronés avec TEB(entrée)=10^{-3} pour le rendement du code égal à 0.9

En conclusion, on peut finalement avoir deux modes en fonction des différents TEB(entrée) ou E_b/N_0 :

SMBF (single puis multiple) mode : $BER(entrée)$ supérieur à $5 * 10^{-2}$ou $Eb/N0 < 4.5dB$

Multiple mode : $BER(entrée)$ inférieur à $5 * 10^{-2}$ou $Eb/N0 \geq 4.5dB$

Pour le multiple-mode, on prend directement la formule (IV-6) afin de détecter les erreurs et reduire le nombre d'itérations utilisés dans le décodage.

IV.4.2. Algorithme proposé

Comme on a 2 bits pour le CAN, les symboles reçus dans le décodeur ont 4 valeurs $(-1, -\delta, \delta, 1)$, on n'a pas assez d'informations pour calculer les fiabilités des bits reçus $\emptyset(x_k, y_k)$ selon l'équation de fonction d'inversion $\Delta k^{général}$. Cependant, on peut essayer

d'évaluer la fiabilité des syndromes caculés en comptant le nombre des 1s dans les noeuds de parité (plus de 1s il apparaît dans un noeud de parité et plus fiable est ce noeud). Par rapport à la méthode GDBF, cette méthode crée des priorités entre les bits détectés.

On utilise alors deux méthodes WGDBF (Weighted GDBF) et BWGDBF (Balanced Weighted GDBF) qui sont basées sur ce principe et dont les équations s'écrivent :

$$\Delta k^{WGDBF}(x) \triangleq x_k y_k + \sum_{i\in M(k)} B_i \prod_{j\in N(i)} x_j$$
$$\Delta k^{BWGDBF}(x) \triangleq x_k y_k + \sum_{i\in M(k)} (\alpha + B_i) \prod_{j\in N(i)} x_j \qquad (IV\text{-}7)$$

Où α introduit une valeur expérimentale qui est autour de 1 pour ajuster l'importance entre les facteurs $x_k y_k$ et $\sum_{i\in M(k)} B_i \prod_{j\in N(i)} x_j$. B_i représente le rapport entre le nombre des $\pm$1s des noeuds de variable qui participent dans le même noeud de parité et le poids de la ligne de la matrice de parité.

IV.4.3. Performances

Les auteurs dans [156][157] ont proposé une méthode avec 2bit CAN pour l'algorithme MS. Les LLRs des noeuds de variable sont initialisés par $\{-W, -w, w, W\}$. En plus, la meilleure performance est atteinte quand $w = 1|W = 3$. En prenant le même principe, dans cette partie les LLRs pour l'algorithme MS sont intialisés par $4y_k|y_k \in \{-1, -\delta(-0.25), \delta(0.25), 1\}$. Les performances des algorithmes que l'on propose (BWGDBF, WGDBF) sont données page suivante.

D'abord les simulations sont lancées pour déduire les performances de BWGDBF en fonction de alpha sur la Figure IV-21, on peut obtenir que pour atteindre la meilleure performance de BWGDBF, alpha est pris à 1.

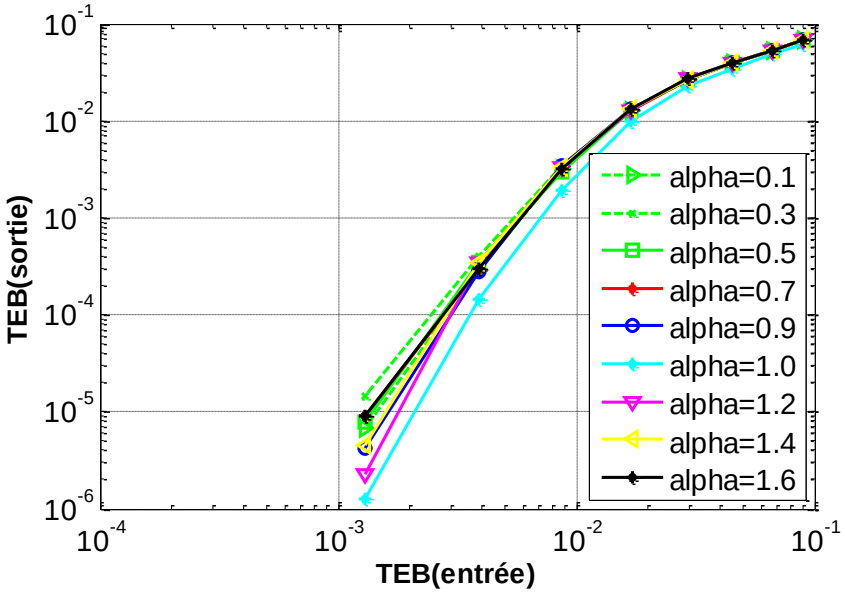


Figure IV-21 : Performances de BWGDBF en fonction de α

La Figure IV-22 a montré les comparaisons des performances entre différents algorithmes. Tous les algorithmes BF respectent deux modes (SMBF mode, multiple mode) que on a proposé dans la dernière partie. Selon cette figure, on peut observer que les algorithmes qu'on a proposés ici atteignent les meilleures performances parmi les différentes méthodes BF. De plus ils approchent de très près les performances de l'algorithme MS lorsque ce dernier a accompli à peu près dix itérations.

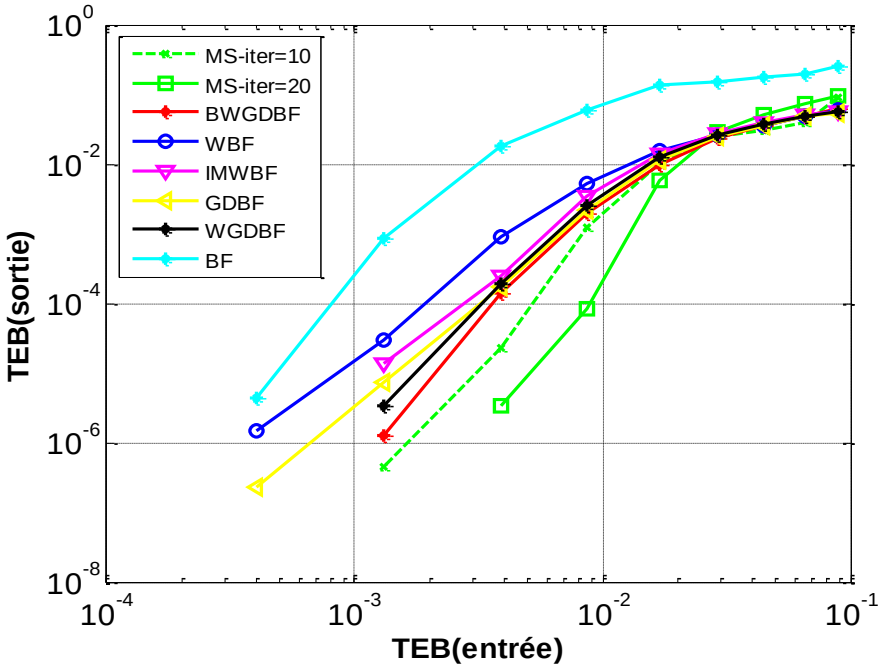


Figure IV-22 : Performances des différents algorithmes avec 20 itérations et 2bit CAN

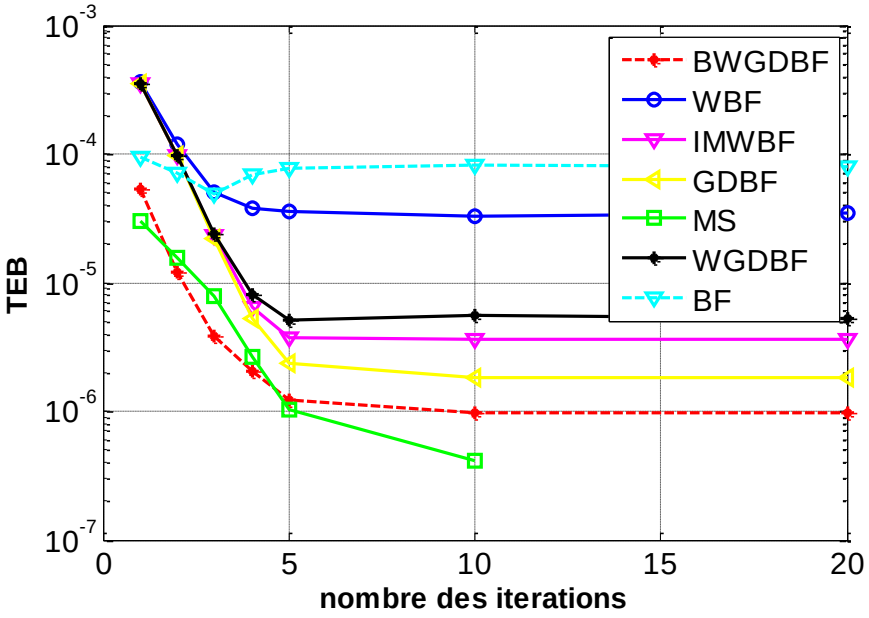


Figure IV-23 : Les performances des algortihmes en fonction du nombre d'itérations quand TEB(entrée) vaut 10^{-3}

La Figure IV-23 montre les performances des algorithmes en fonction du nombre d'itérations quand TEB(entrée) vaut 10^{-3}. On peut observer que l'algorithme BWGDBF converge plus rapidement que les autres algorithmes. A cause des trapping set et stopping set, il commence à saturer après 4 itérations de décodage. Comme montré en Figure III-9, s'il y a un trapping set qui apparaît, plus probablement il y a une boucle qui apparaît dans le décodage, donc on peut aussi faire la détection des boucles dans un certain nombre d'itérations afin d'améliorer la performance pour tous les algorithmes à décisions dures.

Le pouvoir de corrections en fonction du nombre d'itérations pour différents algorithmes est très intéressant à étudier et il est donné en Figure IV-24 et Figure IV-25. On peut obtenir que les méthodes BWGDBF et WGDBF arrivent à corriger les bits avec très peu d'itérations, mais WGDBF n'arrive pas à corriger plus de bits que la méthode BWGDBF, parce que sa fonction d'inversion ne peut pas bien détecter et justifier les bits erronés (c'est le même cas pour l'algorithme RRWGDBF).

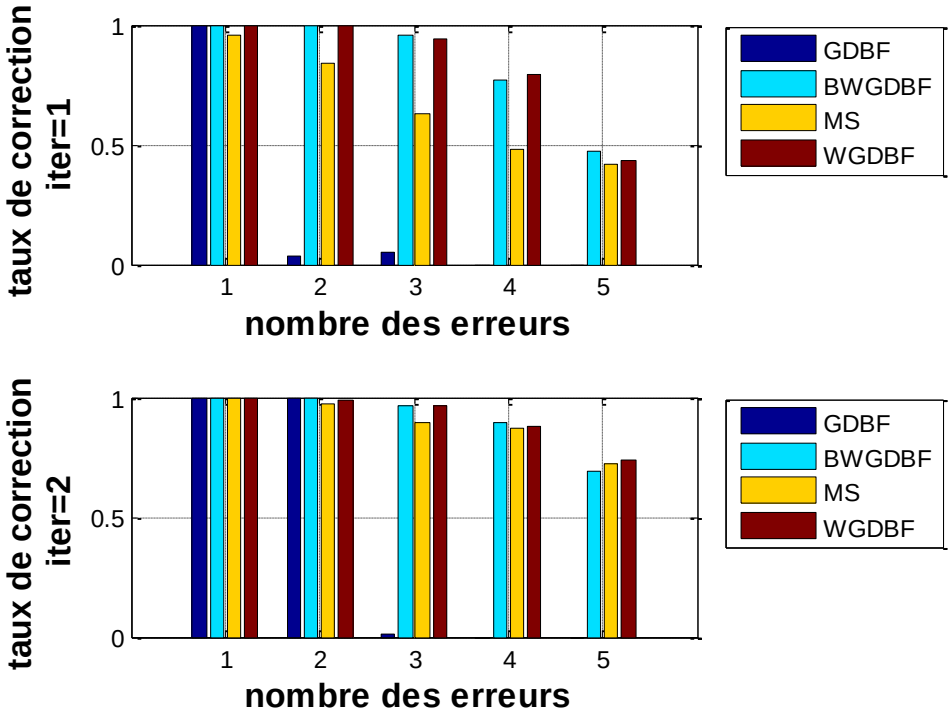


Figure IV-24 : Pouvoir de correction pour les différents algorithmes quand le nombre d'itérations vaut 1 et 2

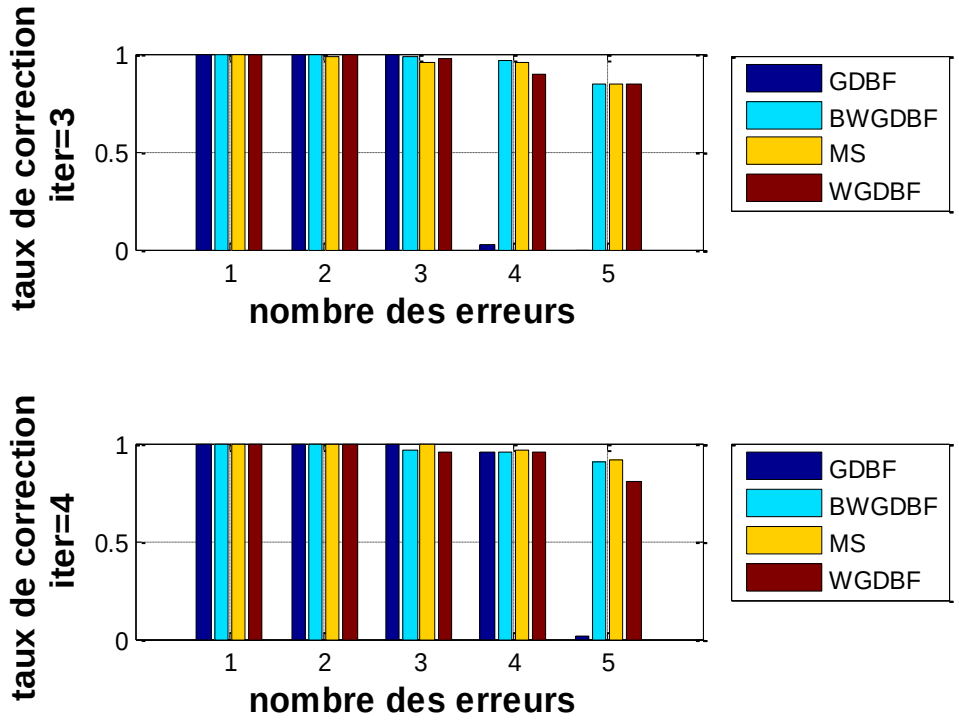


Figure IV-25 : Pouvoir de correction pour les différents algorithmes quand le nombre d'itérations vaut 3 et 4

Le TEB(entrée) expérimental dans le contexte C-RAN est donné entre 10^{-3}et 10^{-4}, donc on illustrer les performances dans Tableau IV-3. Comparé avec les Tableau III-5 et Figure III-19, on a observé un peu de dégradation avec 2 bits CAN par rapport aux simulations sans CAN. En revanche la méthode BWGDBF peut atteindre la même performance qui s'approche de la performance de l'algorithme GDBF sans CAN.

Tableau IV-3 : Comparaison des différents algorithmes quand TEB entrée vaut 10^{-3} avec 2 bit CAN

Algorithme(R=0.9) Avec 2 bit CAN	$TEB_S(TEB_e(10^{-3}))$
BF	$10^{-3} \rightarrow 10^{-4}$
WBF	$10^{-4} \rightarrow 10^{-5}$
IMWBF	$10^{-5} \rightarrow 10^{-6}$
GDBF	$10^{-5} \rightarrow 10^{-6}$
WGDBF	10^{-5}
BWGDBF	$10^{-6} \rightarrow 10^{-7}$
MS(10iter)	$10^{-7} \rightarrow 10^{-8}$

IV.5. Conclusion

Dans ce chapitre, un convertisseur analogique-numérique (CAN) est ajouté dans notre chaine de décodage dans le contexte C-RAN de la 5G. On a d'abord modélisé le canal AWGN par un canal discret, et puis on a fait des études du paramètre delta de ce canal discret en fonction des différents rendements du code pour les algorithmes GDBF. On a obtenu la meilleure valeur de delta pour différents rendements de code. D'ailleurs, deux modes (SMGDBF, MSGDBF) ont été proposés pour améliorer encore les performances de la méthode GDBF. Dans un canal fiable ($\frac{E_b}{N_0} \geq 4.5\ dB$ ou TEB(entrée)$\leq 10^{-2}$), on a aussi montré une formule pour détecter les bits errornés, ceci sert à réduire le nombre d'itérations afin de réduire la lantence du circuit. En plus, deux algorithmes (BWGDBF, WGDBF) sont proposés pour s'approcher de la performance de l'algorithme GDBF sans CAN en utilisant 4 itérations au lieu de 9 itérations quand TEB(entrée) vaut 10^{-3}. A la fin, l'algorithme BWGDBF est choisi pour l'implémentation qui sera discutée dans le prochain chapitre, et son nombre d'itérations est fixé à 4 pour obtenir un excellent compromis performance/complexité.

Chapitre V

Implémentation

Chapitre V. Implémentation

V.1. Introduction

De nos jours les codes QC-LDPC [158], dont nous avons parlé dans les chapitres précédents, sont largement appliqués dans de nombreuses normes de communication sans fil, comme IEEE 802.16e (WiMAX), IEEE 802.11n (Wi-Fi), IEEE 802.15.3c (WPAN), etc. Il est donc impotant de comparer les différerents paramètres d'implémentation pour différentes méthodes de décodage afin de réduire la complexité et la latence. Globalement, pour les codes LDPC, la complexité de décodage est proportionnelle à E/Rn [159], où E représente le nombre de liens entre les nœuds de parité et les nœuds de variable, R représente le rendement du code et n représente la taille du paquet.

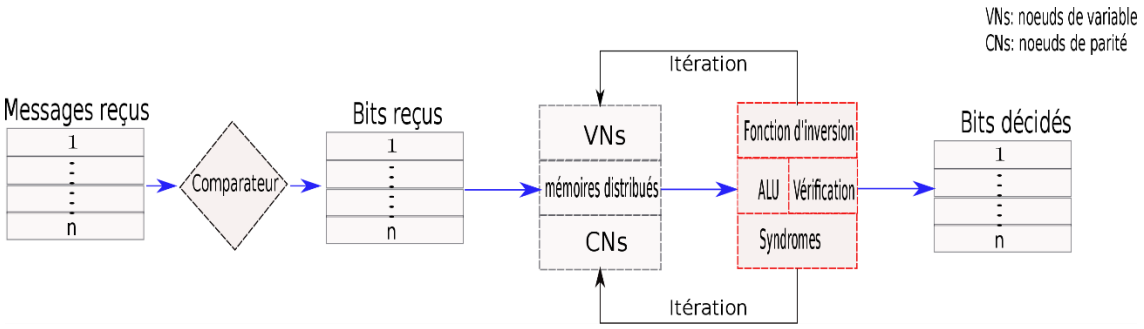


Figure V-1 : Chaine de décodage dur

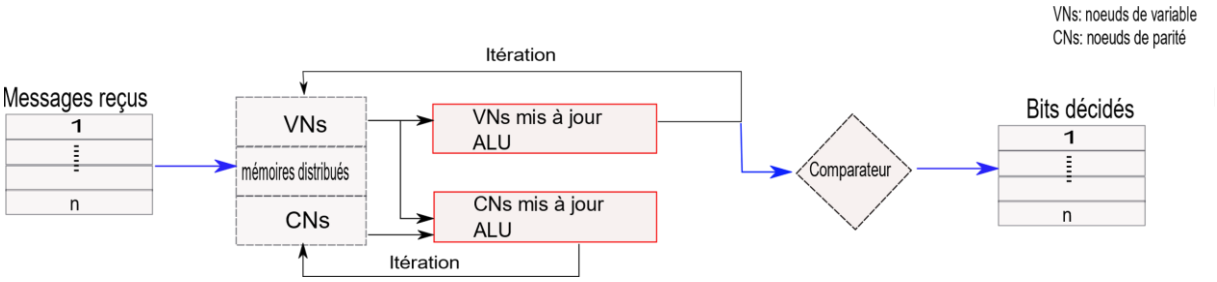


Figure V-2 : Chaine de décodage souple

Comme on a montré sur les Figure V-1 et Figure V-2, les résultats des calculs dans les nœuds de variable (VNs) et dans les nœuds de parité (CNs) sont enregistrés dans les mémoires distribués. Ce type de mémoire reste très limité dans un FPGA. Sachant que les données sont représentées par plus de bits dans les algorithmes souples que les algorithmes durs, les algorithmes souples utilisent largement plus de ressources que les algorithmes durs. Du point de vue de la latence, une architecture tout parallèle accélère le temps d'itérations, cependant la réalisation reste prohibitive en termes d'utilisation de ressources. Pour résoudre ce problème, des architectures partialement parallèles ont été proposées dans la littérature [160][161][162][163].

Selon le contexte des études (rendement 0.9 et la taille du code égale à 1000), la matrice a été optimisée pour maximiser le cycle dans la matrice H. Des simulations exhaustives montrent l'intérêt de l'algorithme BWGDBF par rapport à tous les autres algorithmes en termes de performance. C'est donc cette méthode-là qui sera implantée sur FPGA. Dans ce chapitre,

on va donc présenter les paramètres importants pour le décodeur, les différentes méthodes d'implémentation et nous allons démontrer l'intérêt de BWGDBF par rapport aux autres méthodes.

V.2. Conception de décodeur

Sachant que l'implantation de l'encodeur LDPC ne pose en général pas de problème particulier, on se concentre dans ce chapitre sur l'architecture du décodeur. Pour tester notre décodeur, on utilise matlab pour générer les messages de test (de manière off-line). Schlafer et al. dans [164] donnent un aperçu des méthodes de conception pour un décodeur LDPC à multi-giga bits par seconde. Comme montré en Figure V-3, le choix de décodeur dépend de plusieurs paramètres à prendre en compte comme les ressources utilisées, l'énergie consommée, la latence, le débit, et la performance.

Les sections suivantes présentent les interdépendences complexe qui doivent être prises en compte lors de la conception du décodeur. Les paramètres les plus importants à prendre en compte dans notre contexte sont la latence, le débit et les performances.

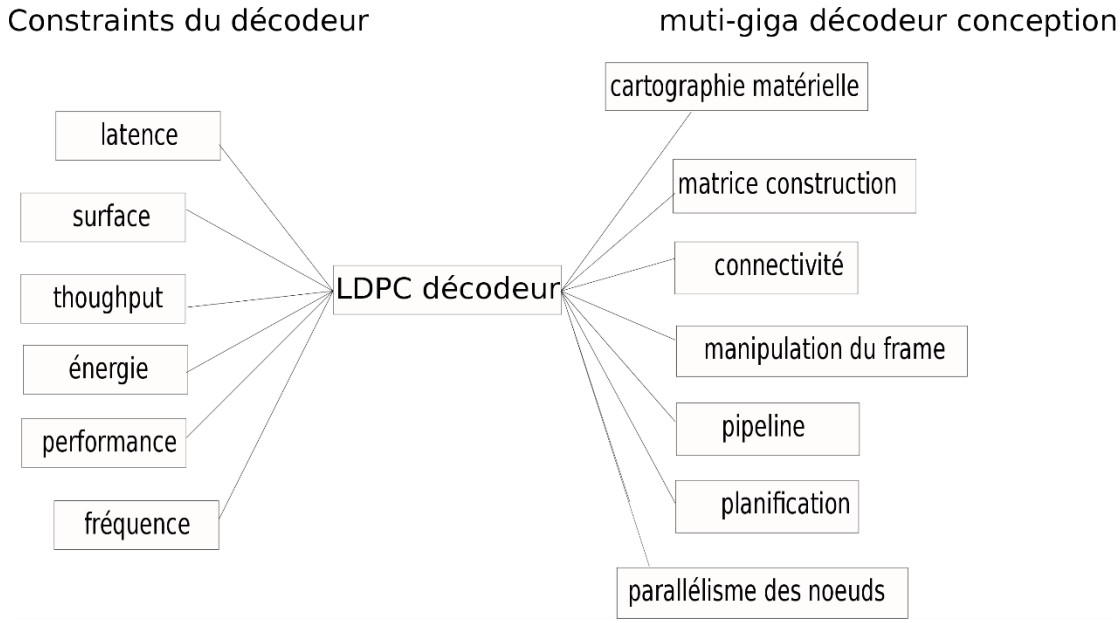


Figure V-3 : Conception et paramètres du décodeur

V.2.1. Planification (scheduling)

Il existe deux méthodes principales pour une planification partiellement parallèle. Soit la mise à jour des nœuds se fait en deux phases, soit on utilise la structure de planification par couche (layered scheduling). Pour la première méthode, tous les nœuds de parité traitent les informations dans un premier temps, et ensuite ils envoient les messages vers les nœuds de variable. Dans cette deuxième phase, les nœuds de variable produisent de nouveaux messages vers les nœuds de parité, comme montre la Figure V-4. En ce qui concerne la deuxième méthode (planification par couche), les nœuds de variable sont mis à jour de façon continue pendant que les informations sont traitées dans les nœuds de parité [165][166][167]. De plus, les CNs peuvent prendre en compte les résultats qui ont été générés dans la même itération. La méthode de planification par couche peut atteindre la même performance que la

méthode à deux phases tout en réduisant de 50% le nombre d'itérations nécessaires, comme le montre la Figure V-5. Ceci permet d'atteindre en théorie, pour la même fréquence d'horloge, un débit double par rapport à la méthode à deux phases. En revanche, la méthode planification par couche reste plus compliquée, et consomme beaucoup plus de ressource dans ce FPGA (elle a besoin de plus de variables intermédiaires pour gérer les accès à la mémoire). Un autre problème lié à l'implantation matérielle est qu'elle risque de générer des collisions dans la mémoire car selon la matrice de parité, on peut avoir besoin d'accéder à une même case mémoire plusieurs fois en même temps. Par conséquent, cette méthode ne sera pas utilisée dans nos travaux d'implantation.

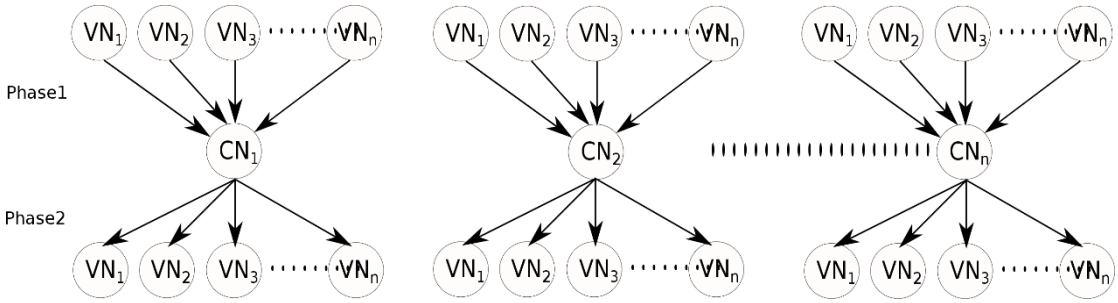


Figure V-4 : Planification à deux phases

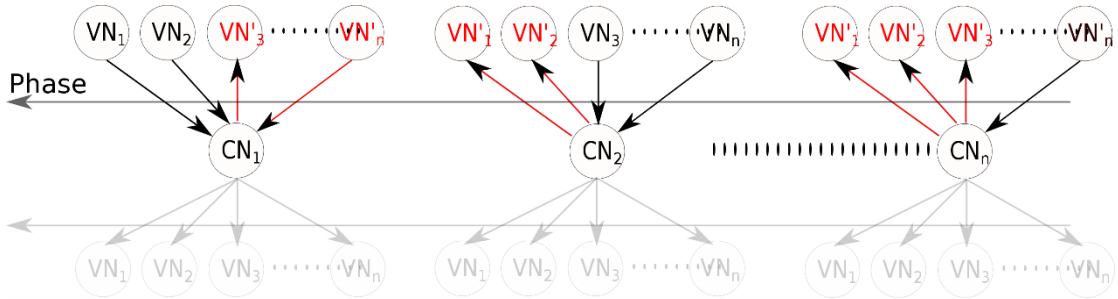


Figure V-5 : Planification par couche

V.2.2. Parallélisme des nœuds

Le degré d'un nœud est par définition le nombre de liens de ce nœud. En fait, il représente le nombre de lecture ou d'écriture que le nœud doit effectuer dans une itération. Cette opération ne peut pas être effectuée en un cycle d'horloge car le processeur ne peut pas accéder à toutes ces cases mémoires en même temps. Dans une implémentation en série le nœud de parité lit un message venant sur une flèche et produit une valeur par cycle. Cette architecture est flexible car chaque degré de nœud peut être facilement pris en charge. En revanche, dans une implémentation en série le délai de traitement des nœuds est important et il est difficile d'atteindre des débits multi-gigabit. Puisque notre contexte d'étude demande un débit élevé, le degré de parallélisme des nœuds doit être augmenté. Dans ce cas, chaque donnée produite par un nœud de variable est envoyée à tous les nœuds de parités auxquels elle est destinée. Comme montre la Figure V-6, le parallélisme des nœuds est donc nécessaire. Enfin, la valeur du degré des nœuds est également un facteur important qui décrit l'efficacité du décodeur. Plus le nombre de messages traités en parallèle est important plus complexe et plus efficace

est le décodeur. En conséquence, on doit faire un compromis entre la complexité et le parallélisme.

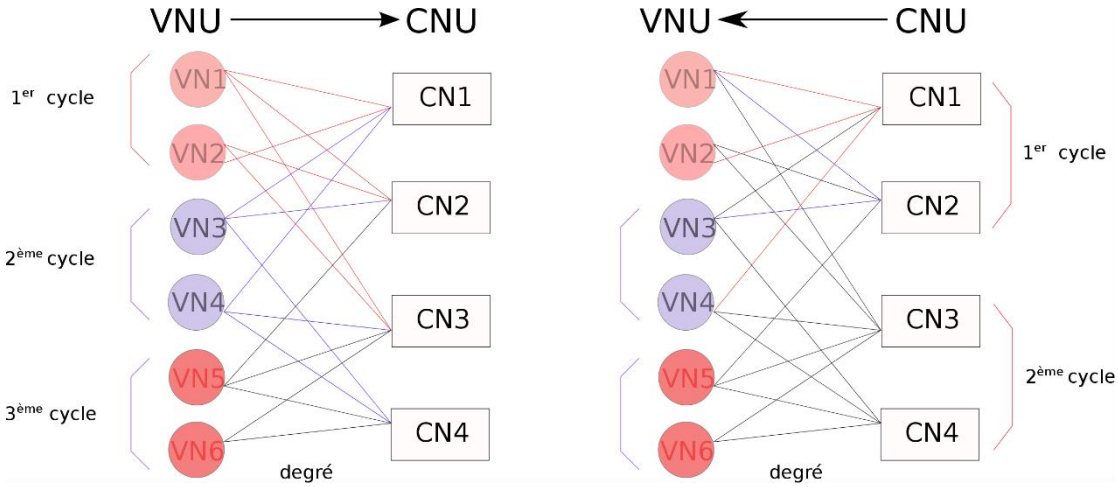


Figure V-6 : Parallélisme des nœuds

V.3. Cartographie matérielle (hardware Mapping)

Le mappage matériel est l'un des paramètres les plus importants pour la conception du décodeur. Globalement, il existe trois architectures pour réaliser le décodeur avec le graphe de Tanner : parallèle, partiellement parallèle et série.

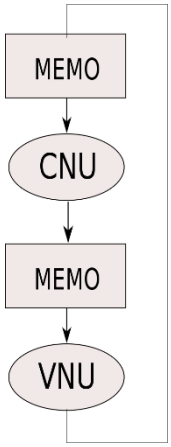


Figure V-7 : Décodeur série

Pour le décodeur série, comme présenté sur la Figure V-7, un seul VNU (module VN) et un seul CNU (module CN) sont implémentés. On obtient une grande flexibilité mais un très faible débit car les opérations sont itérées dans le temps. Par conséquent, nous ne nous sommes pas intéressés à cette méthode.

V.3.1. Décodeur parallèle (full parallel)

Comme la montre la Figure V-8, ce type du décodeur implémente tous les nœuds du graphe de Tanner en même temps, on obtient donc un débit élevé. Cependant, cette architecture matérielle manque de souplesse et se limite aux codes de petites tailles. L'utilisation prohibitive de ressources et la congestion de routage est un obstacle réel lors de l'implantation de cette

approche. Andrew J. Blanksby et al. [168] rapportent une implémentation parallèle sur un circuit de type ASIC, où une partie importante du circuit implémente le routage entre les nœuds. Ce circuit manque de souplesse et ne peut pas être adapté aux rendements différents proposés dans les standards actuels. Pour obtenir la flexibilité, on propose l'utilisation d'une méthode appelée « bit-série » (bit-serial) qui réduit les interconnexions entre les nœuds. Cette méthode peut aussi résourdre le problème de collison de mémoire. L'article [170] montre un décodeur parallèle très efficace pour l'algorithme min-sum. Cependant, comme l'autre, le type de décodeur n'est pas assez flexible pour être utilisé pour les normes les plus récentes. Par conséquent, dans la suite du mémoire, nous nous concentrerons uniquement sur les structures matérielles partiellement parallèles. Trois approches vont être présentées selon le rapport entre VNUs et CNUs : « légèrement parallèle », « basée sur les colonnes », et « basée sur les lignes » de la matrice H.

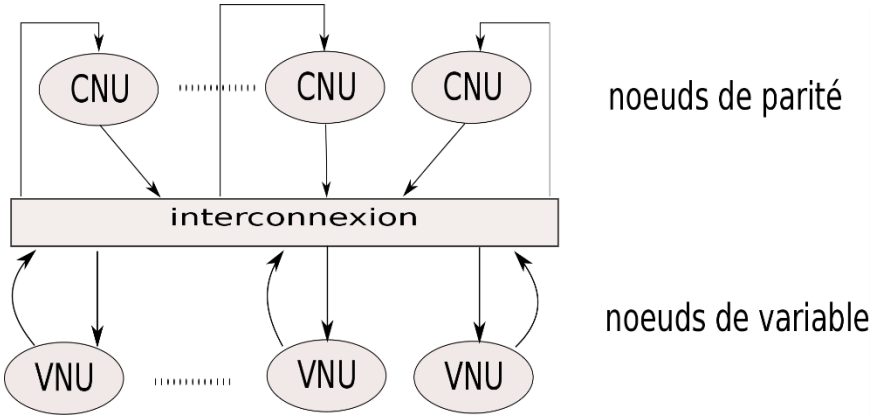


Figure V-8 : Décodeur full parallèle

V.3.2. Légèrement parallèle

Cette méthode est utile pour le débit allant jusqu'à 1 Gbit/s. Cela implique une multiplication du nombre des VNUs et CNUs, comme illustré sur la Figure V-9. Dans cet exemple, par rapport à la Figure V-7, nous avons réalisé 4 nœuds de variables qui travaillent en même temps. Ceci divise par quatre le nombre d'itérations et accélère donc le décodage. Il nécessite encore un grand nombre de cycles d'horloge par itération. On observe à partir de la Figure V-9 que dans le premier cycle, une partie de la première ligne de la matrice H est traitée, ensuite dans le deuxième cycle d'horloge c'est une partie de la quatrième ligne de la matrice qui est traitée. Globalement, seules quelques unités fonctionnelles doivent être instanciées. Une fois que tous les messages VN→CN sont transmis, les messages extrinsèques produits par CNs doivent être renvoyés aux VNs. Un des problèmes de cette méthode reste l'accès simultané à la mémoire contenants les valeurs des VNs. Deux méthodes sont envisageables pour résoudre ce problème. Soit on duplique la mémoire pour pouvoir accéder à des valeurs en même temps, soit on crée plusieurs mémoires chacune contenant une partie des VNs. On s'arrange pour n'adresser que les données dans les mémoires distinctes.

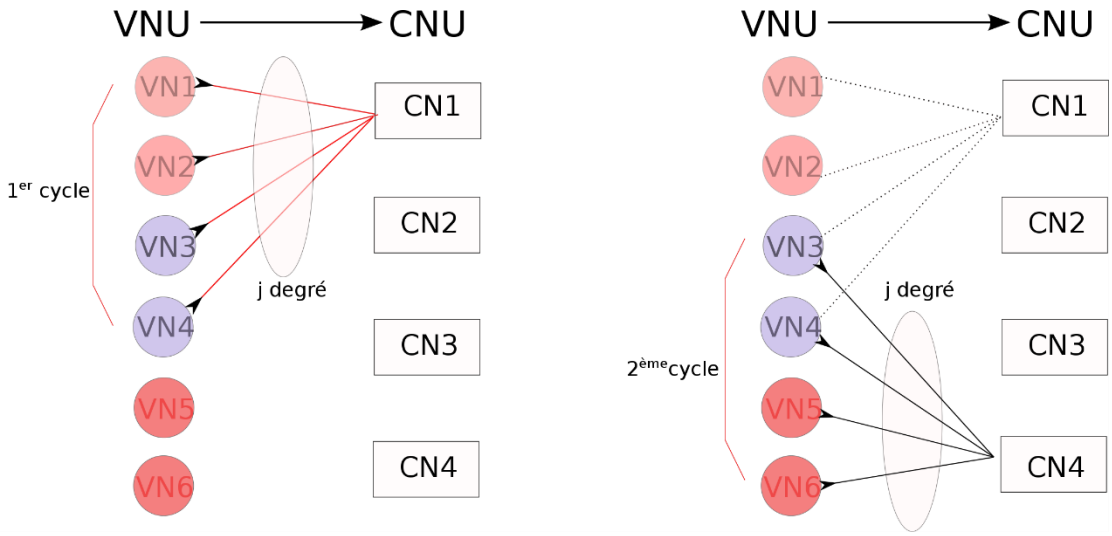


Figure V-9 : Décodeur low parallèle

V.3.3. Approche basée sur les lignes

Tous les N VNs sont instanciés comme VNU et P $(P < M)$ CN sont aussi instanciés comme CNUs. Tous les VNUs fonctionnent en parallèle, et P CNUs fonctionnent en parallèle qui traitent N degrés par cycle (c'est-à-dire N messages venant des VNUs).

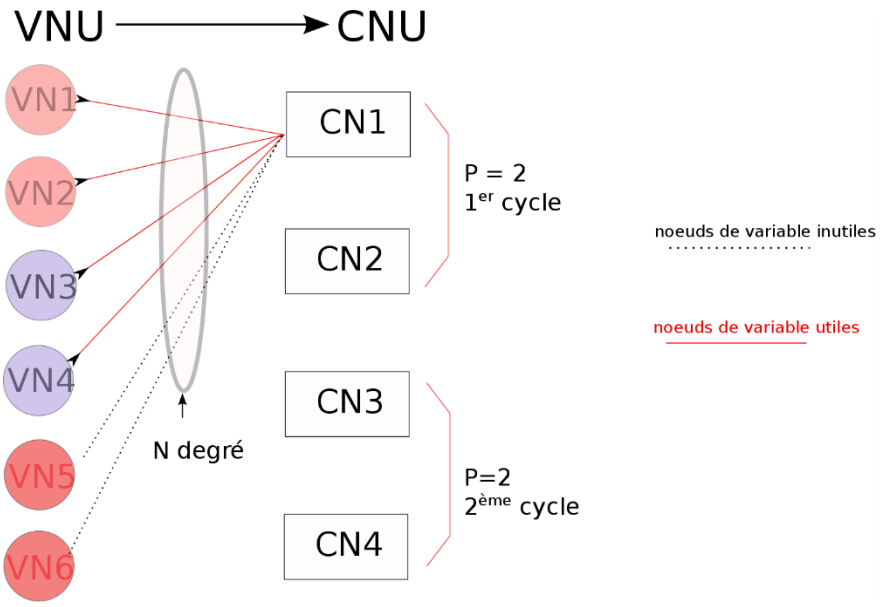


Figure V-10 : Décodeur basé sur les lignes

Comme montré en Figure V-10, $N = 6$ (nombre de VN), $P = 2$, $M = 4$ (nombre de CN), et chaque CN reçoit les messages envoyés par 4 différents VNs. Si on augmente la taille de la matrice, c'est la même signification que l'augmentation du nombre des VNs et des CNs, ce qui pourrait entraîner un problème de saturation des ressources dans le FPGA. Il est à noter que pour pouvoir accéder à tous les VNUs, ils doivent être implantés dans la mémoire distribuée du FPGA. Pour éviter ce problème, nous avons choisi la méthode basée sur les colonnes.

V.3.4. Approche basée sur les colonnes

Par rapport à la méthode précédente, les CNUs doivent être implémentées en parallèle. On parcourt les nœuds VNU un par un. Le message correspondant à chaque VNU est envoyé à tous les CNUs connectés à ce VNU, comme montre la Figure V-11. Nous avons le même problème d'accès mémoire que la méthode précédente mais le nombre de CNUs est beaucoup moins important que le nombre de VNUs (vu le rendement proche de 1 du code). Une duplication de mémoire coutera donc moins que précédemment. Ainsi, il est possible de réaliser les valeurs stockées des CNUs dans la mémoire distribuée. Cette méthode (tout comme la méthode précédente) donne la possibilité de planifier en deux phase car les CNUs fonctionnent tous en parallèle (notez que la mise à jour intermédiaires des VNUs n'est pas possible et il faudra attendre l'accomplissement des calculs des CNU).

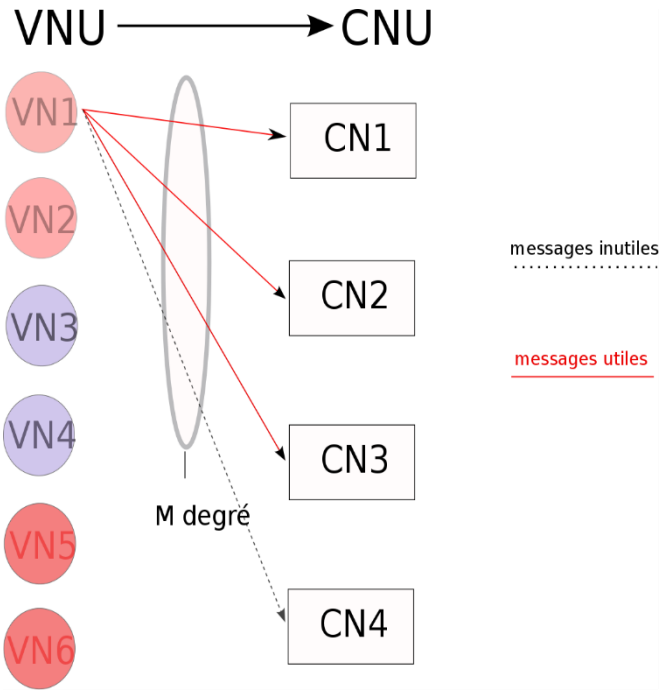


Figure V-11 : Décodeur basé sur les lignes

V.4. Décodeur à décisions dures

Comme on a introduit dans les dernières parties, la méthode basée sur les colonnes avec une planification en deux phases a été finalement choisie pour être implantée dans le décodeur LDPC. Dans cette partie, l'architecture du décodeur à décisions dures et le résultat obtenu en simulation vont être présentés. Les simulations ont été effectuées sur l'outil Xilinx ISE, qui permettra au développeur de synthétiser le circuit, d'effectuer une analyse temporelle, et d'examiner les diagrammes de temps, etc.

Dans ces études, on a réalisé les décodeurs sur le FPGA spartan6 (xc6slx16) disponible au laboratoire. Il est à noter que cette carte n'est pas exactement la carte cible du projet LAMPION. Notre contribution a consisté à démontrer la faisabilité d'implémentation du décodeur avec l'algorithme ciblé (BWGDBF) puis à transférer tous les programmes au partenaire Ekinops qui effectuera les optimisations nécessaires à l'implémentation sur la

plateforme développée dans le cadre du projet LAMPION. En outre ils vont mettre en oeuvre une parallélisation des cibles pour pouvoir atteindre le débit de 2,5 Gbit/s et une latence maximale de 5 µs.

Comme le montre la Figure V-12, on peut observer l'architecture globale qu'on va construire dans cette partie. Les valeurs de x_k sont d'abord utilsées pour calculer les syndromes. En combinant les sommations des syndromes avec les valeurs de x_k et y_k on détermine une table de vérité pour justifier les bits à inverser afin de reconstituer les valeurs de y_k.

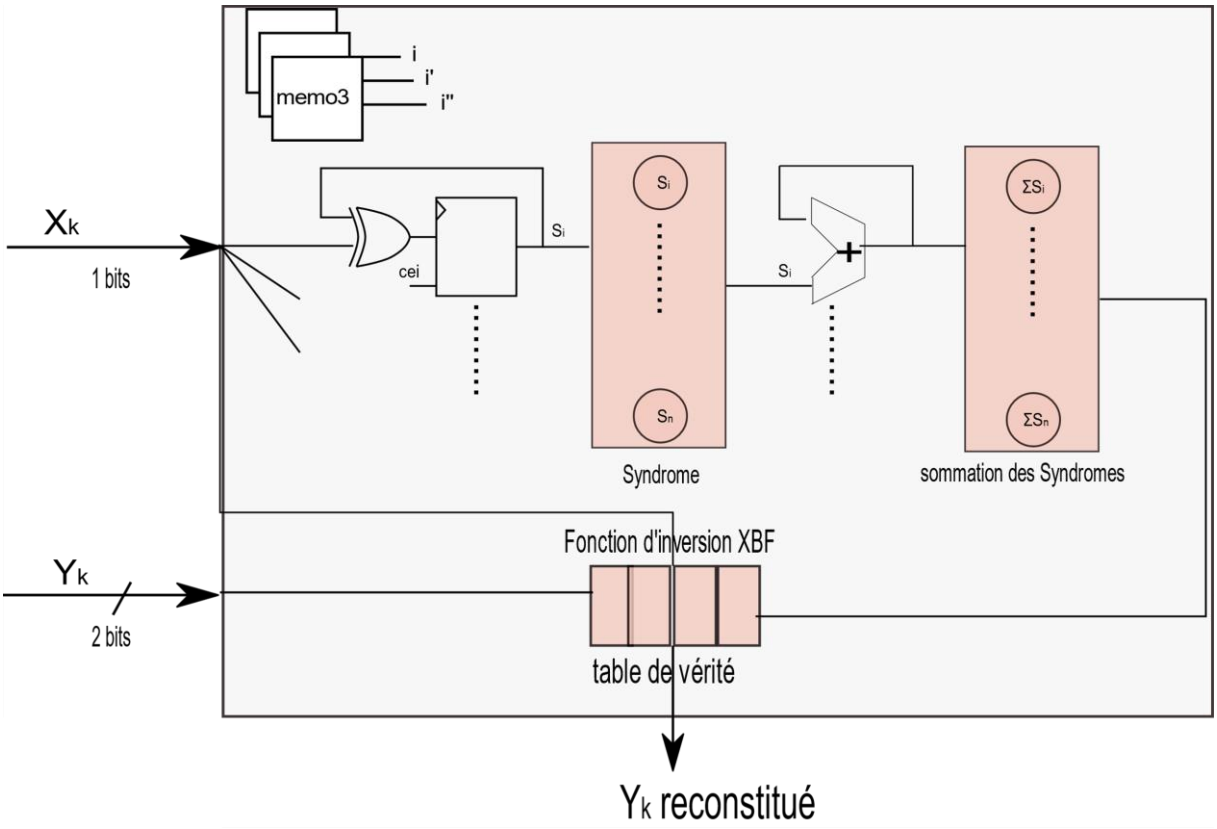


Figure V-12 : Architecture générale

Décodeur BF (Bit Flipping)

On commence d'abord par le décodeur BF car c'est la base de toutes les méthodes présentées dans ce mémoire. Donc, si on implante la fonction d'inversion de cet algorithme, tous les autres algorithmes pourront l'utiliser. On note ici que les zéros logiques sont présentés par un niveau 0 volt et de la même façon les uns logiques par le niveau 3,3 volts. Pour être plus explicite comme le montre la Figure V-13, on utilisera tout au long de ce mémoire un code LDPC régulier avec une matrice de parité de taille 100x1000, un poids de colonne égal à 3 et un poids de ligne égal à 30.

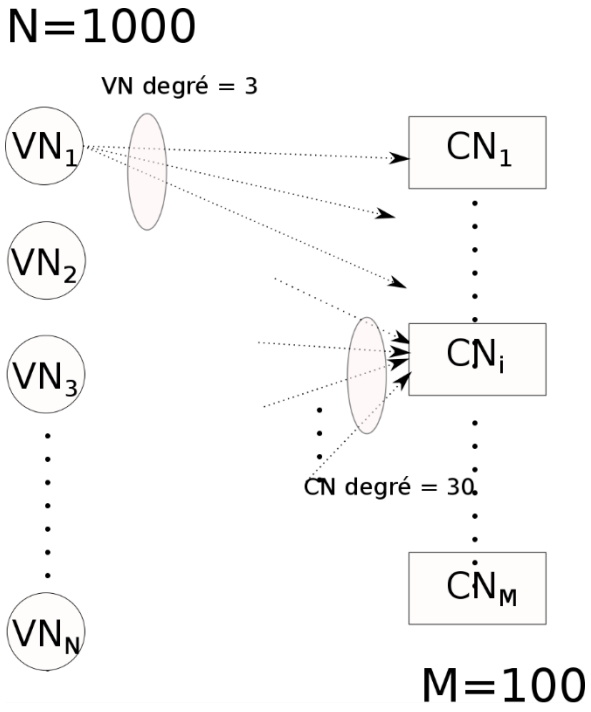


Figure V-13 : (100 ,1000) Matrice utilisée

Comme le montre la Figure V-14, le buffer d'entrée (appelé X) représente les valeurs récupérées du canal (valeurs décidées donc 0 ou 1), il contient donc 1000 bits. Les syndromes sont représentés par un autre buffer avec une taille égale à 100 bits, appelé S. Le syndrome $S(i)$ est égal à 1 quand la $i^{ème}$ équation de parité n'est pas satisfaite.

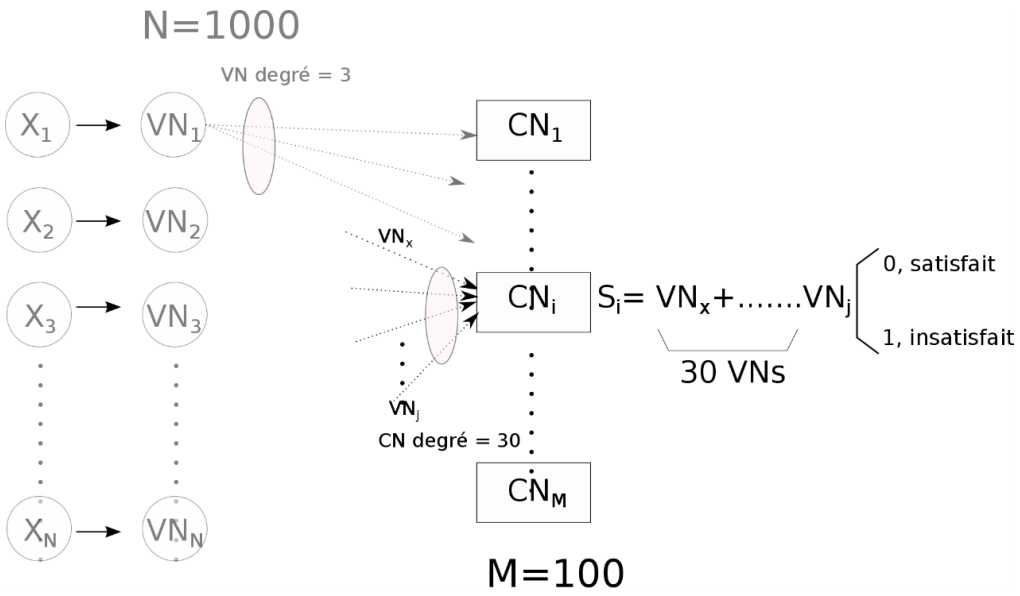


Figure V-14 : Calcul des syndromes

En choisissant les VNs dans l'ordre, à chaque fois 3 CNs sont sélectionnés. Les indices de ces CNs correspondent aux emplacements des 1s dans la colonne i de la matrice H. Ce qui signifie également que la valeur de ce nœud de variable est à envoyer aux 3 CNUs pour calculer les syndromes. Le parcours des éléments du buffer X est géré par un contrôleur d'adresse. L'opération itérative effectuée dans chaque CN pour calculer le syndrome est la suivante :

$$S_i = S_i + VN_{j \to i} \; avec j \in \{j \,|h_{ij} = 1\} \qquad (V\text{-}1)$$

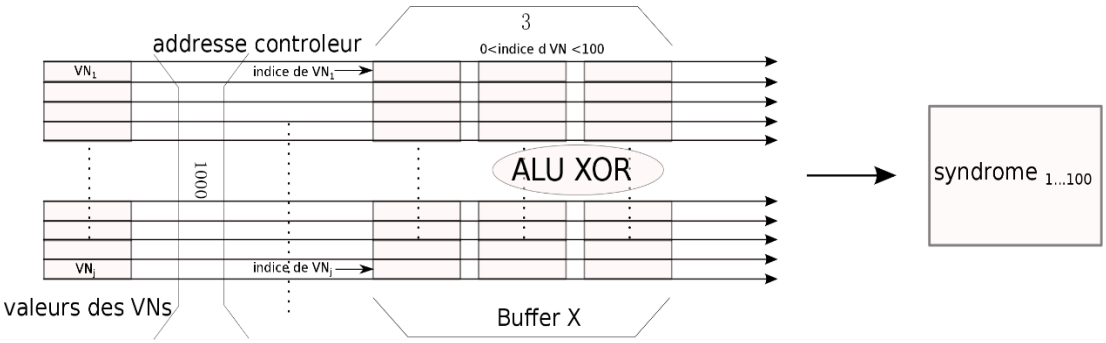


Figure V-15 : calcul du syndrome avec buffer

Pour pouvoir avoir accès aux trois CN, on propose de mettre dans trois mémoires, les trois indices des CN reliés au VN_j. Pour la première itération, ($j = 1$), on active les trois « ce » (Chip Enable) de la Figure V-16. Le générateur d'adresse fera varier le j jusqu'à la fin (ici dans cet exemple il va jusqu'à 1000) et une fois terminé, tous les syndromes sont calculés et se trouvent dans les bascules (Figure V-16). Cette opération durera pendant 1000 cycles d'horloge.

Il est à noter que toutes ces mémoires sont contrôlées par le même contrôleur d'adresse.

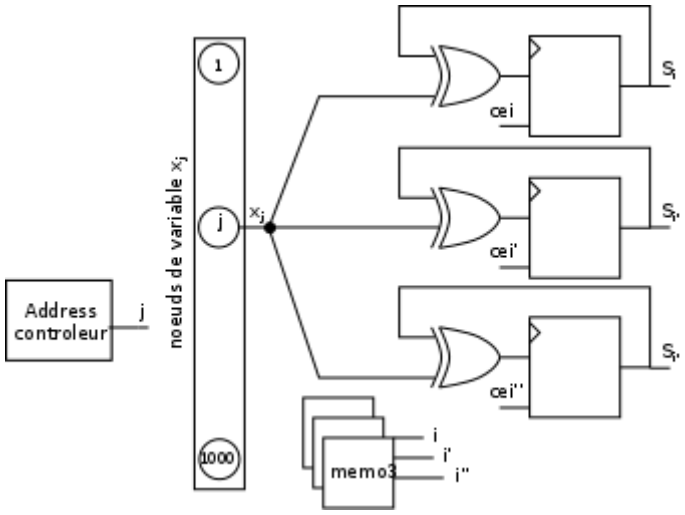


Figure V-16 : Circuit pour syndromes

Les procédures pour obtenir la valeur de la fonction d'inversion pour chaque bit est très similaire aux calculs du syndrome. La fonction d'inversion pour chaque bit représente la somme des syndromes qui sont liés au bit en question (Figure V-17). Ceci est calculée, pour le $k^{\text{ième}}$ bit, par :

$$\Delta k^{BF} = \sum_{i \in M(k)} \prod_{j \in N(i)} x_j \qquad (V\text{-}2)$$

Pour trouver les bits les plus susceptibles d'être erronés, il suffit de sauvegarder les valeurs du contrôleur d'adresse à chaque fois que la valeur de la fonction d'inversion est importante (2 ou 3). La politique d'inversion de bits douteux (multiple bit-flipping ou single bit-flipping) peut influencer la performance et la convergence de l'algorithme, comme on a vu dans les chapitres précédents. On suivra la méthode multiple bit-flipping par la suite vu sa meilleure performance.

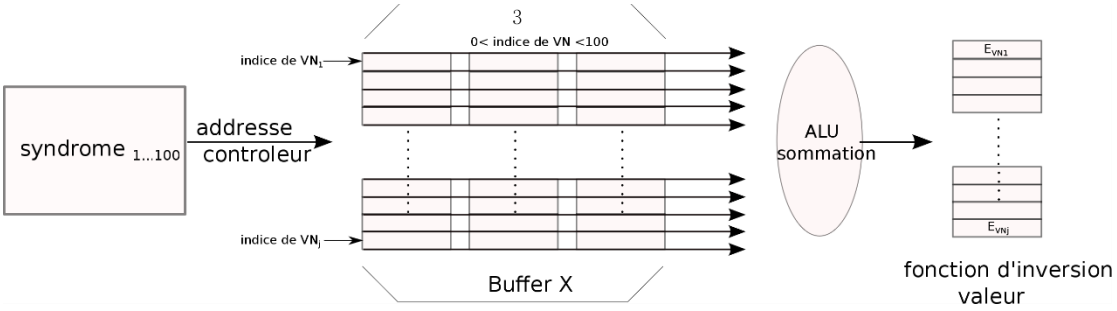


Figure V-17 : Fonction d'inversion de BF

En comparaison avec le circuit qui calcule les syndromes, pour la fonction d'inversion, les portes XOR sont remplacées par des additionneurs à deux bits. Après 100 cycles d'horloge, les registres contiendront les Δk^{BF} pour tout k.

V.4.1.1. Résultats des simulations

Nous utilisons l'outil de développement Xilinx ISE à la fois pour la synthèse et pour la simulation. Le FPGA cible est le spartan6 (xc6slx16) qui se trouve sur les cartes disponibles au laboratoire. Pour tester le décodeur, on utilise Matlab pour générer une séquence d'informations, la coder en LDPC avec la matrice décrite dans les chapitres précédents, y ajouter des erreurs simulant les erreurs de transmission avec des TEB fixés (TEB est egale à 10^{-3} ce qui a été fixé par Orange vu les contraintes pratiques dans le contexte C-RAN), et finalement générer les deux séquences x et y. La séquence y est quantifiée sur deux bits conformément aux études mentionnées aux chapitres précédents. Ces séquences sont enregistrées dans plusieurs blocs mémoires afin d'être manipulées par le circuit de décodeur comme montre la Figure V-18.

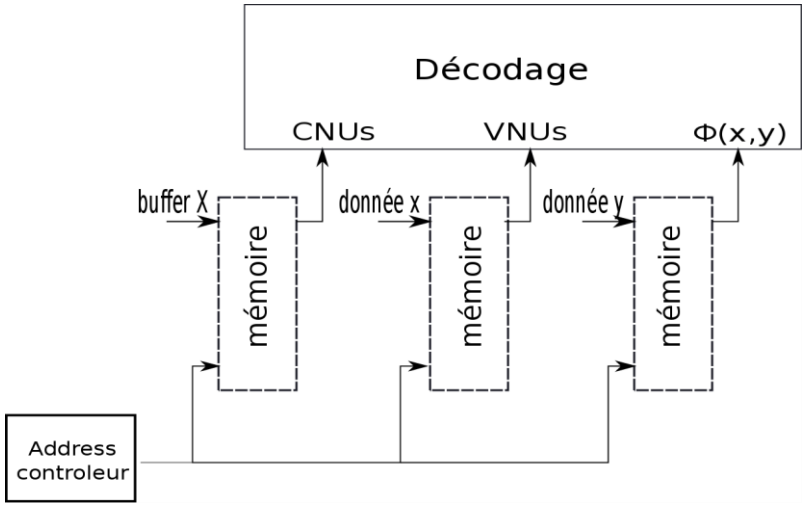


Figure V-18 : Mémoire prédéfinie avant décodage

La Figure V-19 montre quelques chronogrammes obtenus en simulation circuit. La machine d'états conçue contient 9 états, listés ci-dessous :

1. Ide : état initial
2. Compute : calcul des valeurs des syndromes
3. Tempo : initialisation du contrôleur d'adresse et des autres variables qui sont utilisées dans les états suivants
4. Sum_syndrom : calcul des valeurs de fonction d'inversion pour chaque bit, et en même temps stockage des bits les plus douteux dans la mémoire
5. Correction : vérification des équations de parité, si le mot reçu est un mot valide du code on arrête le décodage, sinon on passe à l'inversion des bits
6. Attent : synchronisation des signaux
7. Ecriture : inverser les bits choisis pour remplacer les valeurs d'origine
8. Nouvelle_iteration : initialisation des signaux et commencer une nouvelle itération
9. Fin : fin du décodage

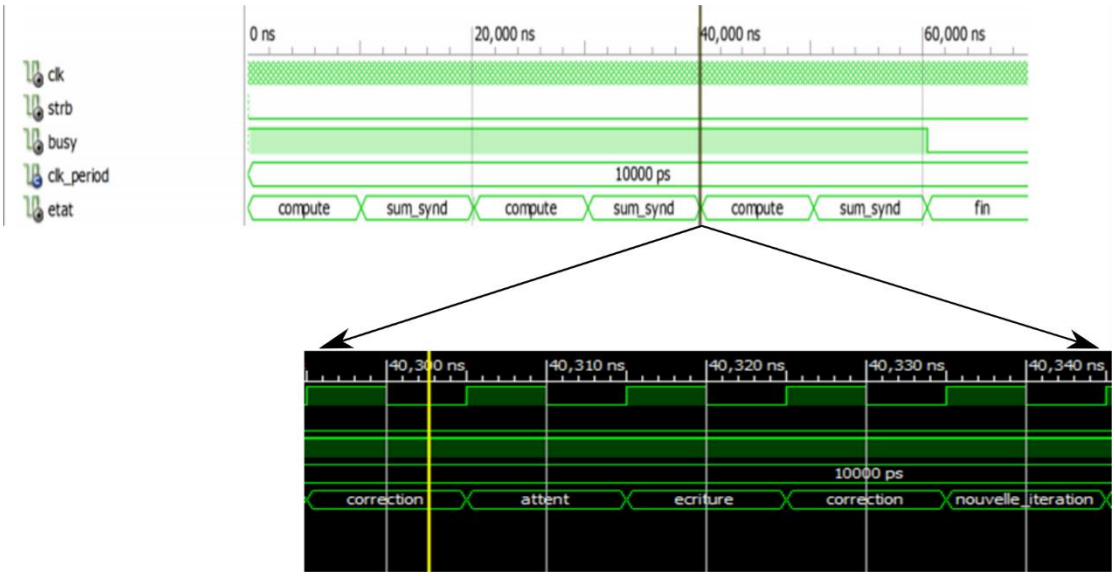


Figure V-19 : Machine à état

La Figure V-20 présente le résultat de simulation. On sauvegarde les valeurs dans les cases mémoires nommées : error_indice, error_indice_2, error_indice_3, lorsque les valeurs de Δk^{BF} valent 1, 2 et 3. Dans cet exemple, le programme Matlab avait généré 3 erreurs en positions 0, 2 et 883 (le résultat pour toute autre position reste identique). Comme le montre la figure, les positions de ces erreurs ont été identifiées et ces erreurs ont été corrigées au bout de la troisième itération. Le nombre maximum d'itérations a été fixé à 4. Sachant que l'horloge de la carte est de 100 MHz, le temps nécessaire pour effectuer ces quatre itérations est égal à 60 µs.

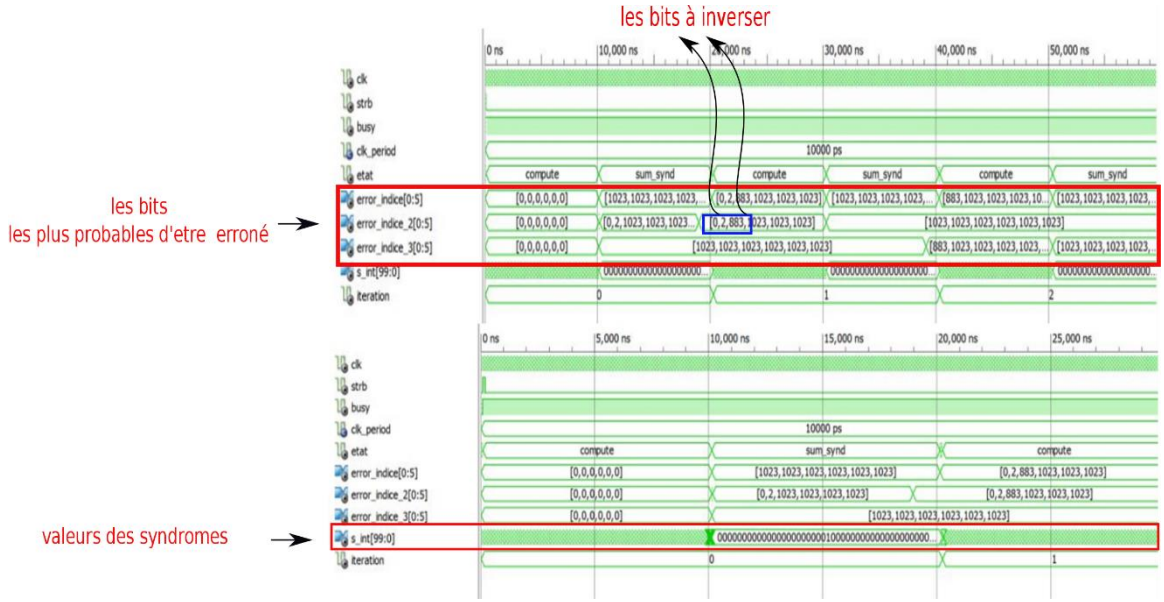


Figure V-20 : Simulation de l'algorithme BF dans ISE

V.4.1.2. Utilisation de ressource dans le FPGA

En faisant une synthèse à l'aide de l'outil ISE, nous avons estimé le besoin de ressources dans un FPGA de type Spartan 6.

Tableau V-1 Consommation de ressource pour le décodeur BF

Logic utilisation	Used	Available	Utilization
Number of slice register	162	18224	1%
Number of slice LUTS	626	9112	6%
Number of occupied slices	211	2278	9%

Le Tableau V-1 est basé sur le FPGA Spartan-6 où chaque slice contient 4 LUTs (look up table) et 8 bascules (flip-flops). On peut conclure que le décodeur BF consomme très peu de ressources d'un FPGA de type spartan 6.

V.4.2. Décodeur GDBF (Gradient Descent BF)

L'implémentation de l'algorithme GDBF est un peu plus complexe que celle de l'algorithme de BF. Ici, on ajoute une nouvelle mémoire afin de mémoriser sur 2 bits, le signal reçu y. Conformément à l'équation d'inversion, on réalise la multiplication des buffers x et y. Cette multiplication fait la corrélation entre le signal reçu et le bit décodé et fournit donc une fiabilité sur le message reçu.

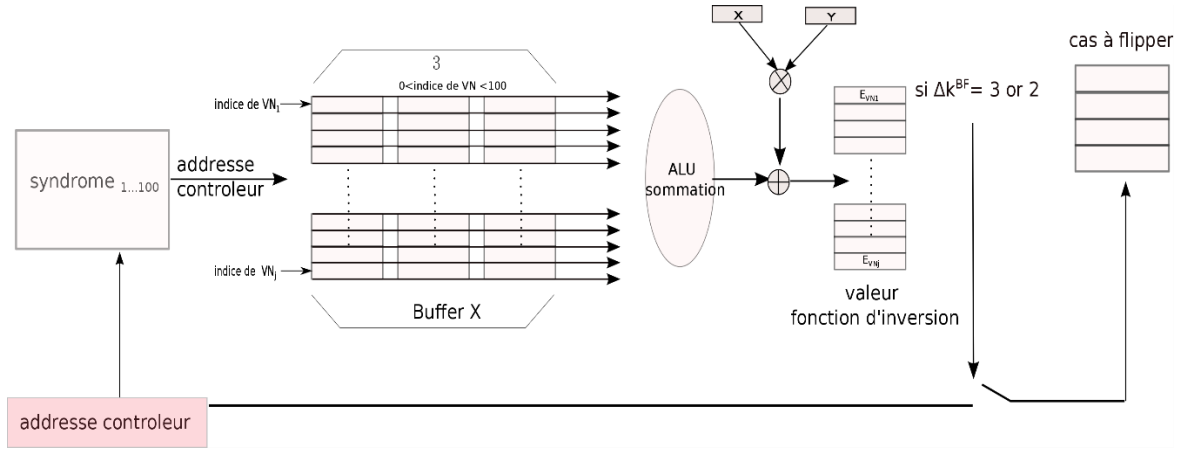


Figure V-21 : Fonction d'inversion de l'algorithme GDBF

Basé sur l'algorithme BF, la formule Δk^{GDBF} peut s'exprimer par $x.y + \Delta k^{BF}$. Afin de réduire la complexité du circuit, on effectue l'évaluation de cette métrique en deux phases. D'abord on applique la méthode BF pour identifier les bits les plus douteux suivant cette métrique (où la valeur de Δk^{BF} vaut 2 ou 3). Dans certaines situations nous trouverons plusieurs bits qui donnent les mêmes métriques BF. C'est uniquement pour ces cas que nous calculons la corrélation pour les différencier. Sachant que le nombre de bits à inverser a déjà été calculé,

on commence par les métriques égales à 3, et ensuite, on élargit vers les métriques égales à 2.

Pour calculer le nombre de bits pour présenter le résultat de multiplication, on remarque que 3 bits sont suffisants, comme le montre la Figure V-22.

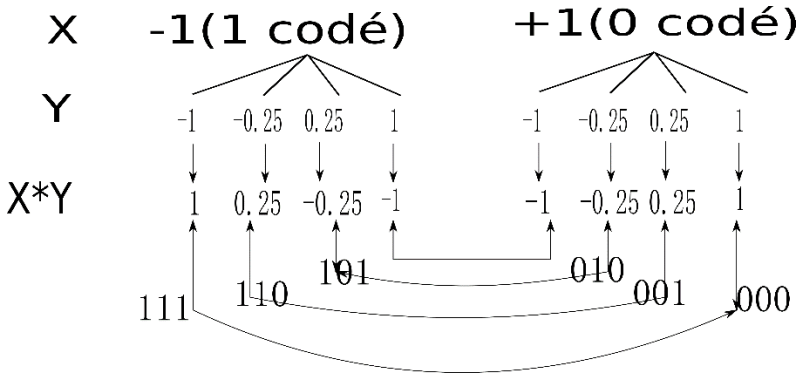


Figure V-22 : Multiplication de x, y en 3bits

Pour tenir compte des deux cas de métrique (Δk^{BF} vaut 2 ou 3), il suffit juste d'ajouter un seul bit, comme illustré par le Tableau V-2 pour représenter les valeurs de Δk^{GDBF}.

Tableau V-2 : Table vérité de Δk^{GDBF}

X	Y	$X * Y$	$\lvert X * Y + \Delta k^{BF} \rvert$ $(\Delta k^{BF} = -3)$	$bin\lvert X * Y + \Delta k^{BF} \rvert$ $(\Delta k^{BF} = -3)$	$\lvert X * Y + \Delta k^{BF} \rvert$ $(\Delta k^{BF} = -2)$	$bin\lvert X * Y + \Delta k^{BF} \rvert$ $(\Delta k^{BF} = -2)$
1	1	000	2	1000	1	0100
	0.25	001	2.75	1011	1.75	0111
	-0.25	010	3.25	1101	2.25	1001
	-1	011	4	1111	3	1100
-1	1	100	4	1111	3	1100
	0.25	110	3.25	1101	2.25	1001
	-0.25	101	2.75	1011	1.75	0111
	-1	111	2	1000	1	0100

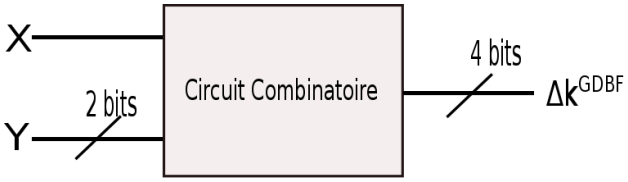


Figure V-23 : Circuit combinatoire

Selon leTableau V-2, les bits qui ont les valeurs maximales doivent être inversés. Il faut donc comparer ces valeurs afin d'obtenir la valeur maximale de la fonction d'inversion de ces bits. Pour réduire la latence du circuit, on propose de réserver plusieurs bascules (mémoire distribuée) pour enregistrer les indices des bits à inverser. Pour cela il faudra créer une boucle afin de comparer les valeurs en parallèle sous forme arborescente comme le montre la Figure V-24. Nous avons ajouté dans la machine d'état, un nouvel état « calcul_max » pour trier ces métriques.

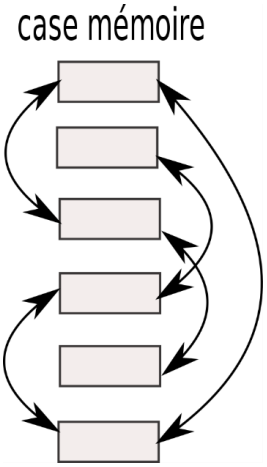


Figure V-24 : Comparaison des cas

Idéalement, il faut réserver 1000 cases mémoires pour trier toutes les métriques. Pour réduire l'utilisation des ressources FPGA, on étudiera l'impact de la réduction de taille de mémoire sur la probabilité d'erreur en sortie du décodeur. Nous avons effectué une simulation système utilisant Matlab pour optimiser le nombre de cases mémoires qu'il faut réserver.

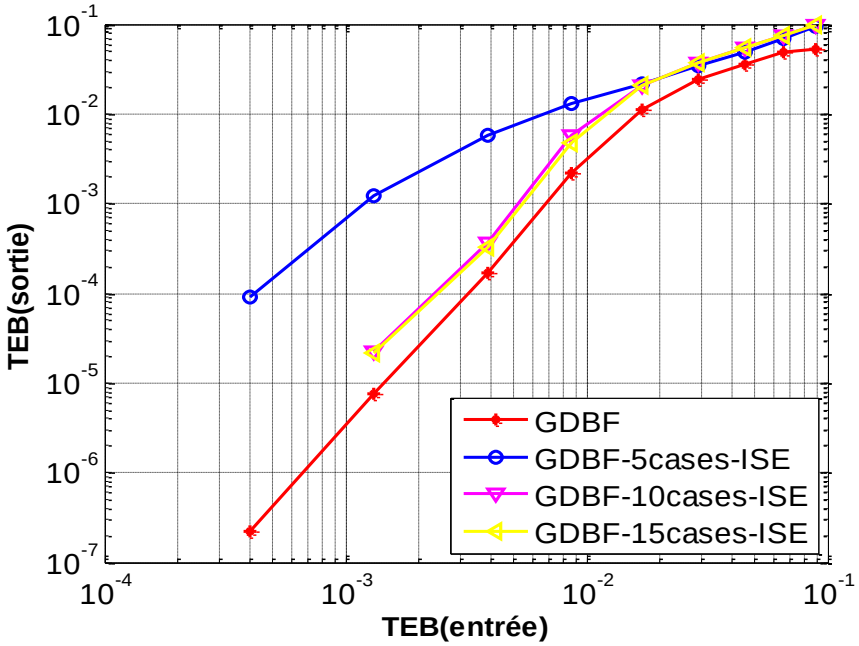


Figure V-25 : Performance de GDBF avec différents nombres de cases mémoires

La Figure V-25 montre le résultat de simulation et présente les performances de l'algorithme GDBF avec différents nombres de cases mémoires. La dégradation par rapport à la courbe idéale se justifie par :

1. Comparaisons d'un nombre insuffisant de cases. Idéalement, toutes les valeurs de Δk^{GDBF}, représentant tous les bits, doivent être comparées, alors ici on ne compare qu'un certain nombre de bits qui ont donné les valeurs de Δk^{BF} prédéfinies (2 ou 3).
2. Dans certains cas particuliers, nous avons constaté que la corrélation pouvait augmenter ou diminuer la valeur de la métrique de manière importante. C'est-à-dire qu'une présélection basée uniquement sur Δk^{BF} n'est toujours pas justifiée.

Nous avons trouvé de manière empirique qu'un nombre de comparaison égal à 10 donnait un résultat presque idéal sans trop dégrader la performance en termes d'erreur binaire.

V.4.2.1. Résultats des simulations

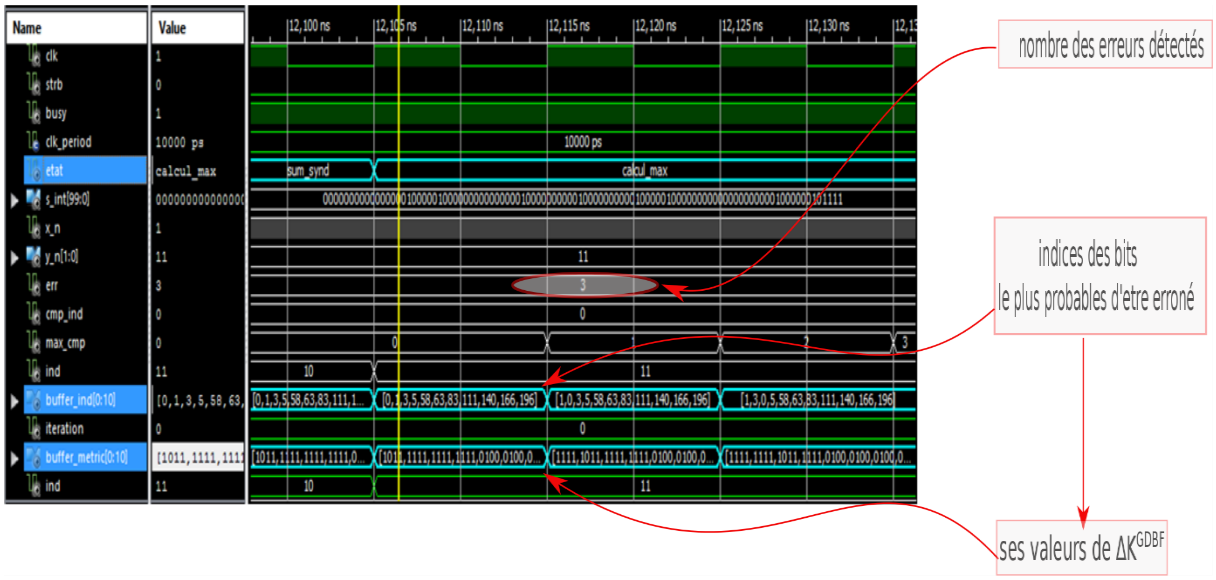


Figure V-26 : Etat du calcul_max

Comme le montre la Figure V-26, le nombre des bits incorrects détectés est de 3. Dans l'état de calcul_max, les bits les plus douteux sont mis en ordre descendant selon les valeurs de Δk^{GDBF}. En Figure V-27, on peut observer comment les bits sont inversés.

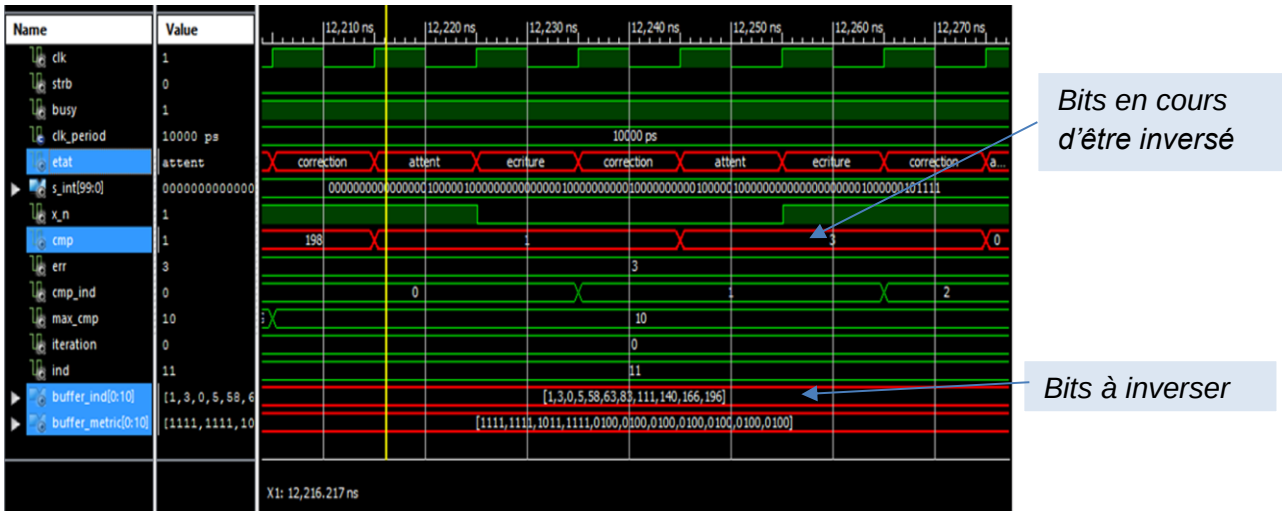


Figure V-27 : Etat d'inversion des bits

L'occupation de la ressource pour cette méthode est montrée en Tableau V-3. On peut observer que par rapport à la consommation de la méthode BF, le nombre de slice est consommé est augmenté de 33%.

Tableau V-3 : Consommation de ressource pour le décodeur GDBF

Logic utilisation	Used	Available	Utilization
Number of slice register	1309	18224	7%
Nulber of slice LUTS	2815	9112	30%
Number of occupied slice	959	2278	42%

V.4.3. BWGDBF (balanced weighted GDBF)

La différence entre BWGDBF et GDBF est qu'on ajoute la fiabilité du syndrome, donc il peut être écrit comme :

$$\Delta k^{BWGDBF} = \Delta k^{GDBF} + \sum_{i}^{M} B_i * S_i \quad \text{(V-3)}$$

La fonction B_i est définie par :

$$B_i = \frac{sum(H(i,:) = 1 | x = \pm 1)}{poid\ du\ colonne} \quad \text{(V-4)}$$

D'où B_i peut être calculé en même temps que les syndromes comme montré en Figure V-28.

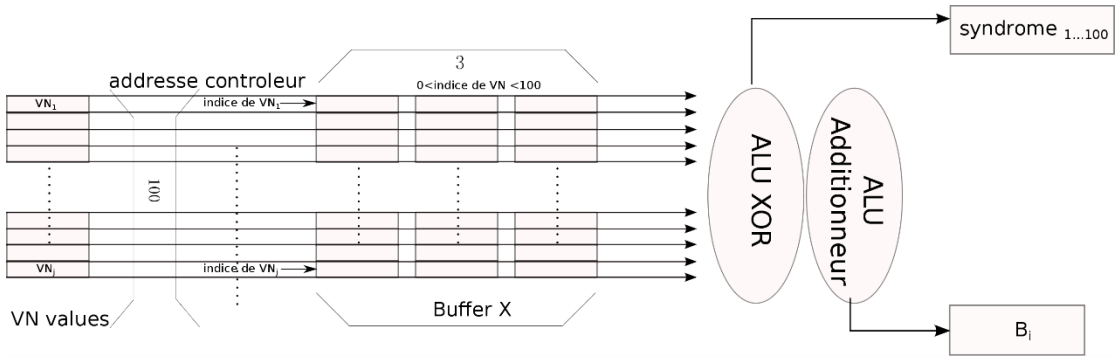


Figure V-28 : Calcul de la fonction B_i

Sachant que B_i n'est pas une valeur entière ($0 \leq B_i \leq 1$), pour garder une précision suffisante, on multiplie les valeurs par une constante (32) et on ne prend que la partie entière[1]. Ceci revient à placer une virgule virtuelle à partir du cinquième bit :

$$32 * \Delta k^{BWGDBF}(x) \triangleq (x_k y_k + \sum_{i \in M(k)} (1 + B_i) \prod_{j \in N(i)} x_j) * 32 \quad \text{(V-5)}$$

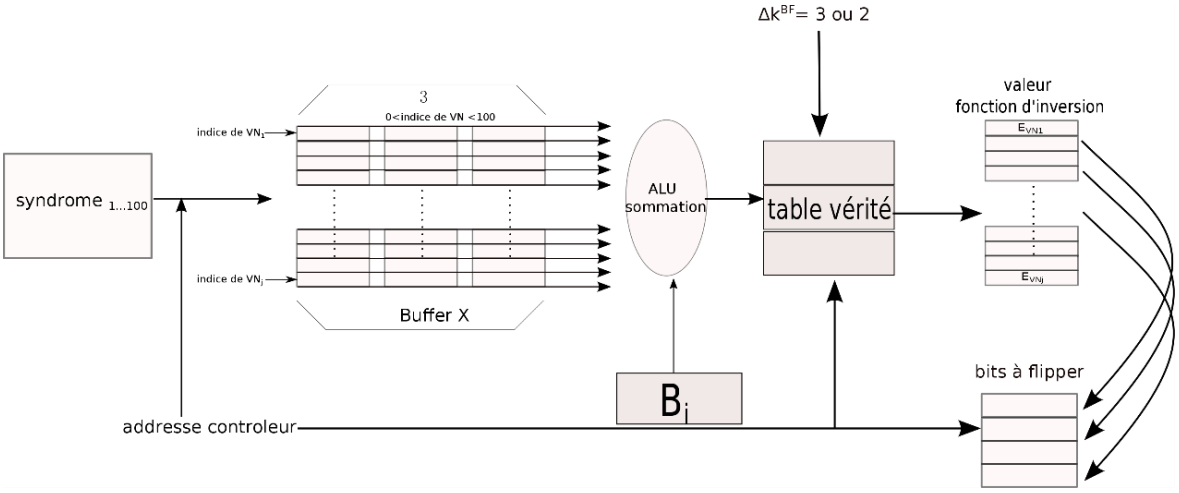


Figure V-29 : Fonction d'inversion de BWGDBF

La Figure V-29 montre la fonction d'inversion de BWGDBF. On peut aussi établir la table de vérité en Tableau V-4 pour cette fonction d'inversion afin d'identifier les bits à inverser :

Tableau V-4 : Table de vérité pour générer la fonction d'inversion BWGDB

$X * Y$	$X * Y + \Delta k^{BF} + B_i$ $(\Delta k^{BF} = 3)$	$X * Y + \Delta k^{BF} + B_i$ $(\Delta k^{BF} = 2)$
000|111	01000000 + 000&B_i	00100000 + 000&B_i
001|110	01011000 + 000&B_i	00111000+ 000&B_i
010|101	01101000 + 000&B_i	01001000+ 000&B_i
Others	01111000+ 000&B_i	01100000+ 000&B_i

[1] En fait, la valeur de B_i pour ce code se trouve dans l'ensemble $\left\{\frac{1}{32}, \frac{2}{32}, \ldots, \frac{30}{32}\right\}$

Comme on l'a fait remarquer auparavant, le nombre maximal de bits à inverser est égal à 10. La Figure V-30 montre le résultat d'une simulation matlab avec exactement les mêmes contraintes que le circuit réel avec des nombres limités de bits, et une taille de buffer identique à celle du circuit. On peut y observer que la performance de cet algorithme dégradée par rapport à celle de l'algorithme idéal (taille du buffer non limitée). La méthode BWGDBF donne un meilleur résultat que la méthode GDBF.

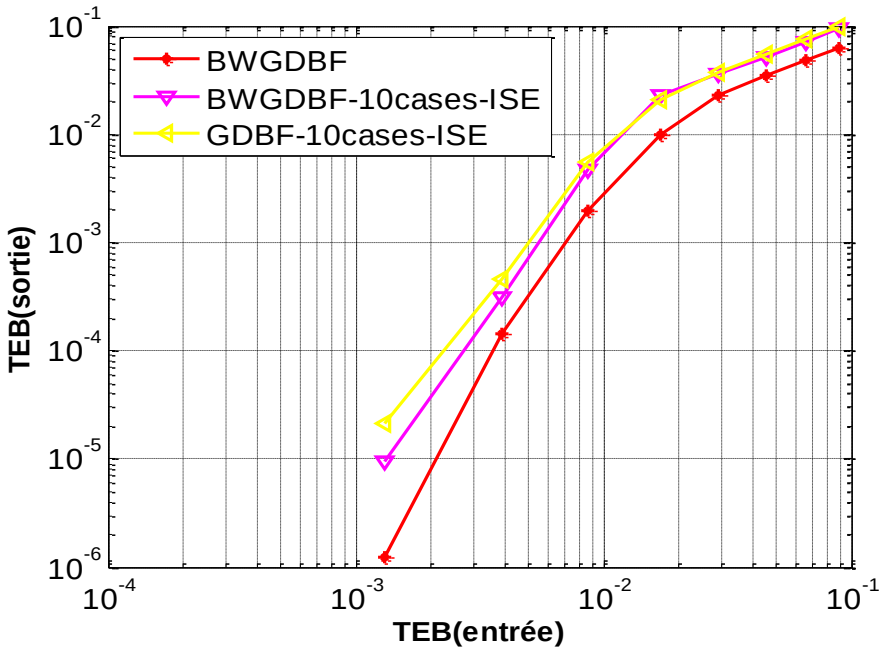


Figure V-30 : Performance des algorithmes avec 10 cases mémoires

V.4.3.1. Résultats de simulations

Figure V-31 : Détail de l'état « calcul_max » pour la méthode BWGDBF

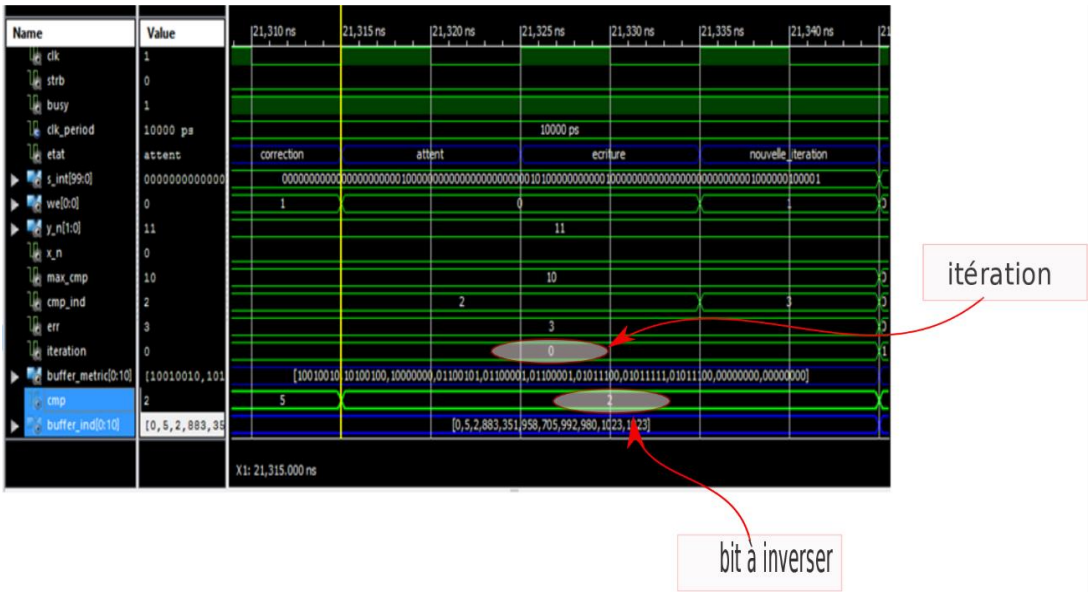


Figure V-32 : Détail de l'état « correction » du bit

Comme le montre les Figure V-31 et Figure V-32, on a réussi à corriger les bits erronés dans la séquence. Un tableau de l'utilisation de ressources de FPGA est fourni en Tableau V-5 On peut observer que l'utilisation de ressources a été augmentée par rapport à la méthode GDBF (plus de 23%).

Tableau V-5 : Utilisation de la ressource pour la méthode BWGDBF

Logic utilisation	Used	Available	Utilization
Number of slice register	1743	18224	9%
Number of slice LUTS	3350	9112	36%
Number of occupied slice	1182	2278	57%

V.4.3.2. Réduction de la latence

Selon la méthode précédente, la latence est estimée aux alentours de $60\ \mu s$ (avec une horloge de 100 MHz)[2]. Pour réduire encore la latence, on propose de dédoubler des registres pour réduire la latence à $30\ \mu s$. Pour ce faire, on casse les buffers x et y chacun en deux buffers de taille moitiée. On parcourt parallèlement ces deux buffers. Pour éviter le conflit d'accès au niveau des nœuds de parité, on utilisera deux jeux de buffer dont les sorties seront combinées en un seul cycle d'horloge. En revanche, cette méthode utilisera plus de ressources, comme le montre le Tableau V-6.

[2] La fréquence d'horloge peut être multipliée par quatre sans changer la carte de FPGA. Dans ce cas la latence est divisée par 4 pour arriver à 15 µs.

Tableau V-6 : Utilisation de ressources pour la méthode BWGDBF en doublant des registres

Logic utilisation	Used	Available	Utilization
Number of slice register	3809	18224	20%
Nulber of slice LUTS	7268	9112	79%
Number of occupied slice	2249	2278	98%

V.5. Conclusion

Dans ce chapitre, nous avons expliqué les principes de nos différents décodeurs durs. Ces décodeurs ont été réalisés en langage VHDL. Pour cela, une machine à 9 états a été programmée. Pour simplifier et accélérer le calcul des fonctions d'inversion (pour BWGDBF et GDBF), deux tables de vérité ont été proposées afin de développer des fonctions logiques simples réalisables par des LUT disponibles dans des slices des FPGA. Pour réduire l'utilisation de ressources FGPA, nous n'avons pas chercher les fonctions min-max globales, mais nous avons proposé donc de fixer un nombre limité de cases mémoires pour mémoriser les erreurs qui sont les plus susceptibles d'être inversés en une itération. Les performances ont été évaluées en utilisant Matlab, en tenant compte des contraintes circuit. Enfin, les circuits ont été synthétisés dans l'environnement Xilinx/ISE, pour la cible Spartan 6 (xc6slx16). A l'aide de ce travail de synthèse logique, nous avons quantifié précisément l'utilisation de ressources dans ce FPGA pour chaque méthode. On peut observer que l'algorithme BWGDBF surpasse d'autres méthodes en termes de TEB, mais consomme bien évidemment plus de ressources. La latence obtenue en circuit pour implanter cet algorithme a été évoluée aux alentours de $60\ \mu s$. Une stratégie de parallélisme a été proposée, qui consistait à dupliquer des registres, pour diviser par deux la latence.

Conclusion Générale et Perspectives

Ce mémoire a présenté l'étude du codage de canal pour les « fronthaul » sur fibre optique dans le contexte du C-RAN ; l'impact du codage et décodage ont été considérés au niveau de la couche physique. Les innovations et optimisations ont été réalisées dans le but d'augmenter les performances, faciliter l'implémentation et réduire la latence.

Le premier chapitre a introduit les évolutions de différentes technologies de RAN. Une solution pour les réseaux modernes C-RAN est proposée qui satisfait à différentes demandes des utilisateurs. Les technologies RSOA en configuration self-seeded avec les émetteurs sont aussi passées en revue pour les réseaux DWDM-PON (Passive Optical Network) afin d'atteindre 2,5Gbit/s. Etant donné que les pertes sont importantes en fonction de la longueur de la cavité et à cause du désaccord entre MUX et DeMUX, l'ajout d'un codage FEC est considéré comme nécessaire pour assurer la performance demandée.

La capacité du canal fibré et différents types de codage FEC ont été étudiés dans le contexte C-RAN dans le second chapitre. Plusieurs codes récents et utilisés dans les standards actuels sont comparés tels que les codes polaires, les codes LDPC et les codes de Reed-Solomon. La taille des séquences considérées a été fixée à 1000 bits et le rendement du code figé à 0,9. Les codes LDPC ont finalement été choisis pour être étudiés dans le contexte C-RAN. Nous avons décidé de nous intéresser au contexte des codes LDPC à décisions dures afin de préserver une complexité raisonnable.

Le troisième chapitre a présenté le principe des algorithmes à décisions dures pour les codes LDPC. Trois méthodes ont été principalement étudiées : BF, WBF et GDBF. Ces méthodes et leurs variantes ainsi que les performances sont intégralement listées dans ce chapitre, Finalement la méthode GDBF est choisie comme représentant le meilleur compromis complexité/performances.

Dans le quatrième chapitre nous avons considéré l'ajout d'un convertisseur CAN 2 bits. Nous avons montré comment optimiser le choix du seuil de décision. En outre, la méthode BWGDBF a été proposée pour atteindre la meilleure performance parmi les codes hard-décision LDPC, il est aussi choisi pour la partie implémentation dans le chapitre cinq.

Le dernier chapitre a présenté différentes méthodes d'implémentation pour les codes LDPC, la méthode partial-parallèle a été retenue pour son compromis complexité/ressource occupation. La méthode BWGDBF a été implémentée sur la carte spartan6 xc6slx16 en utilisant Xilinx ISE. Le problème qui s'est posé a été de trouver une solution pour réduire la latence. Deux méthodes ont été proposées par rapport aux mémoires distribuées. Pour finir, différentes pistes d'amélioration sont proposées en perspective.

Les différentes contributions de ce mémoire de thèse sont les suivantes :

- Une comparaison précise et quasi-exhaustive des différents algorithmes de décodage à décisions dures pour les codes LDPC
- La prise en compte d'un convertisseur CAN dans les performances.
- La définition d'un nouvel algorithme de décodage BWGDBF dans le contexte C-RAN
- L'implémentation de cet algorithme sur plateforme FPGA Xilinx.

Le problème principal qui empêcherait l'implantation directe de notre circuit dans un système C-RAN est la latence trop importante obtenue. En fait il faudra une latence de l'ordre de 5 µs. Alors que nous obtenons une latence d'environ 60 µs. En fait, la fréquence de la carte disponible au laboratoire était de 100 MHz. Dans la plupart des FPGA d'aujourd'hui, la fréquence d'horloge s'élève à 400 MHz, ce qui permettrait de réduire la latence d'un facteur 4 soit la ramener à 15 µs.

Le fait de dédoubler des registres pour paralléliser les opérations pourra faire gagner un facteur 2 supplémentaire pour atteindre 7,5 µs de latence.

En ce qui concerne Rapidité du décodage (RD), nous avons actuellement :

$$RD = \frac{1}{temps\ de\ traitment} * taille\ du\ frame$$

$$temps\ de\ traitment\ = n_c * n_i / f_p$$

Où n_c représente nombre de cycles d'horloge utilisés dans une itération, n_i représente le nombre d'horloges nécessaires dans chaque itération et f_p représente la fréquence de l'horloge.

La taille de « la frame » étant fixée à 1000 bits, avec une horloge de 100 MHz et donc une latence de 60 µs, on arrive à un RD de 16,6 Mbits/s. Les perspectives de ce travail seraient d'utiliser les techniques citées ci-dessus, rapidement exploitables. On pourrait ainsi obtenir un débit être multiplié par 8 ou plus, pour attendre un RD supérieur à 133,3 Mbits/s (inférieur à 5 μs de latence). Pour ce faire il faudrait augmenter la fréquence d'horloge et diviser le buffer X en deux ou plus pour paralléliser et parcourir plus rapidement les mémoires en utilisant une duplication du port XOR.

Les perspectives qui pourraient être envisagées pour ce travail sont :

- Implémentation sur un FPGA à forte capacité pour augmenter le parallélisme
- Augmentation de l'efficacité spectrale avec l'utilisation des codes LDPC q-aire.
- Développement de la formule arithmétrique de delta en fonction du nombre de bits de quantification dans la matrice H afin d'atteindre la meilleure performance.
- Examen des méthodes de rajout de bruit pour améliorer la convergence des algorithmes de décodage : NBWGDBF(Noisy).

Références bibliographiques

[1] https://www.bearingpoint.com/fr-fr/qui-sommes-nous/actualites/presse/communiques-de-presse/en-2020-4-voitures-sur-5-seront-connectees-selon-bearingpoint/

[2] https://fr.idate.org/news-esport/

[3] http://www.gfk.com/fr/

[4] CPRI Specification V6.0 ; August 30, 2013; http://www.cpri.info

[5] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2016–2021 White Paper, https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.pdf

[6] China Mobile Research Institute. C-RAN The Road Towards Green RAN. White Paper, Version 2.5, October 2011 (cit. on pp. 7, 8, 14–17, 20, 21, 24, 25, 28–30, 33, 34, 41, 42, 49, 56, 62, 64, 68, 84, 85).

[7] P. Chanclou et al.," Optical fiber solution for mobile fronthaul to achieve cloud radio access network" Future Network & Mobile Summit Conference, 2013.

[8] F.J. Effenberger, "Mobile backhaul and fronthaul systems". OptoElectronics and Communications Conference (OECC) held jointly with 2016 International Conference on Photonics in Switching (PS), 2016 21st. IEEE, Japan, 2016.

[9] Xavier Lagrange, Philippe Godlewski, "Réseaux GSM, des principes à la norme", Hermes Science Publications, sept 2000

[10] X. L. Philippe Godlewski, Sami Tabbane, Réseaux GSM, H. S. Publication, Ed.,1999.

[11] Zièd Choukair et Sami Tabbane, « Ingénierie des services télécoms : UMTS et Wi-Fi ». Hermès Science, 2005.

[12] Pierre Lescuyer, UMTS : Les origines, l'architecture, la norme", Editions Dunod, 2001.

[13] Heikki Kaaranen et al. "UMTS Networks : Architecture, Mobility and Services", Editions Wiley, 2001.

[14] Harri Holma, Antti Toskala, "UMTS : Les réseaux mobiles de 3ème génération", Editions Osman Eyrolles Multimedia, 2000.

[15] Harri Holma et Antti Toskala, "WCDMA for umts : hspa evolution and lte", john Wiley & sons ed, 2007.

[16] Harri Holma, Antti Toskala, Karri Ranta-Aho, *et al.*, "High-speed packet access evolution" in 3gp release 7 [topics in radio communications]. IEEE Communications Magazine, 2007, vol. 45, no 12, p. 29-35.

[17] Y. Bourguen *et.al.* "LTE et les réseaux 4G" EYROLLES, 2015

[18] Moray Rumney, *et al.,* "LTE and the evolution to 4G wireless : Design and measurement challenges", John Wiley & Sons (ed.), 2013.

[19] A. Ghosh, R. Ratasuk, B. Mondal, N. Mangalvedhe, and T. Thomas, "Lte-advanced : next-generation wireless broadband technology [invited paper]," IEEE Wireless Communications, vol. 17, no. 3, pp. 10-22, June 2010.

[20] E. Dahlman, S. Parkvall, and J. Skold, 4G: LTE/LTE-advanced for mobile broadband. Academic press, 2013.

[21] Christopher Cox, “An introduction to LTE” WILEY, 2014

[22] Visual Networking Index : Global Mobile Data Traffic Forecast Update, 2012-2017.Tech. rep. Cisco, February 2013 (cit. on pp. 2, 64)

[23] Mobility Report. Tech. rep. Ericsson, November 2015 (cit. on pp. 2, 73)

[24] J. Gozalvez. “Tentative 3GPP Timeline for 5G [Mobile Radio]”. In: Vehicular Vechnology Magazine, IEEE 10.3 (Sept. 2015), pp. 12–18. ISSN : 1556-6072. DOI : 10.1109/MVT.2015.2453573.

[25] 3GPP TS 36.300 version 10.9.0 Release 10, " LTE; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2” ETSI TS 136 300 V10.9.0 (2013-02) technical specification, http://www.etsi.org/deliver/etsi_ts/136300_136399/136300/10.09.00_60/ts_136300v100900p.pdf

[26] Rachid El Hattachi, Javan Erfanian, “NGMN 5G White Paper”, Feb 2015 https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0.pdf

[27] Aleksandra Checko, “Cloud Radio Access Network architecture. Towards 5G mobile networks”, PhD thesis, Technical University of Denmark, Kgs. Lyngby, Denmark 2016

[28] Ekram Hossain et Monowar Hasan, “5G cellular : key enabling technologies and research challenges”, IEEE Instrumentation & Measurement Magazine, 2015, vol. 18, no 3, p. 11-21.

[29] Nicola Carapellese, « BaseBand Unit Hotelling Architectures for Fixed-Mobile Converged Next-Generation Access and Aggregation Networks” Phd of Politecnico di Milano, Dipartimento di Elettronica, Informazione e Bioingegneria, 2015.

[30] A. Pizzinat, P. Chanclou, F. Saliou, *et al.*, "Things you should know about fronthaul”, Journal of Lightwave Technology, 2015, vol. 33, no 5, p. 1077-1083.

[31] DIALLO, Thierno. La fibre en support du Mobile Cloud. 2016. Thèse de doctorat. Limoges.

[32] SPECIFICATION, C. P. R. I. V3. 0 common public radio interface (cpri); interface specification, oct. 20, 2006, 89 pages, ericsson ab, huawei technologies col ltd, nec corporation, nortel networks sa and siemens networks gmbh & co. Ltd, NEC Corporation, Nortel Networks SA and Siemens Networks GmbH & Co. KG, 2006

[33] OBSAI ; http://www.obsai.org

[34] OPEN BASE STATION ARCHITECTURE INITIATIVE, et al. The Development and Benefits of an Open Base Station Architecture. White Paper MWG-040318-006-01. Available from http://www. obsai. org/latests/publicdocuments. Htm

[35] ETSI, Open Radio equipment Interface (ORI) ; ORI interface Specification ; Part 1: Low Layers (Release 4), ETSI Std., Oct. 2014. [Online]. Available : http://www.etsi.org/deliver/etsi_gs/ORI/001_099/00201/04.01.01_60/gs_ori00201v040101p.pdf

[36] Zakaria Tayq, “Intégration et supervision des liens Fronthaul dans les réseaux 5G”, these de doctorat de l’Université de Limoges, décembre 2017.

[37] N. Carapellese ARAPELLESE, N., PIZZINAT, A., TORNATORE, M., *et al.* « An energy consumption comparison of different mobile backhaul and fronthaul optical access architectures. In : Optical Communication (ECOC), 2014 European Conference on. IEEE, 2014. p. 1-3.

[38] F Saliou *et al.,* “ Spécifications Système”, Livrable D4.1 du projet ANR LAMPION, (confidentiel), avril 2014

[39] Z. Tayq, P. Chanclou, T. Diallo, K. Grzybowski, F. Saliou, S. Gosselin, O. Foucault, C. Aupetit-Berthelemot, L. Bellot, T. Boukour, J. C. Plumecoq, and J. Sayed, “Performance demonstration of ber and wireless fronthaul combination with remote powering”, Optical Fiber Communications Conference and Exhibition (OFC), March 2016.

[40] ITU-T, Recommendation ITU-T G.709/Y.1331 : Interfaces for the optical transport network, ITU-T Std., 2016. [Online]. Available : https://www.itu.int/rec/T-REC-G. 709-201606-I/fr

[41] Recommendation ITU-T G798 Characteristics of optical transport network hierarchy equipment functional blocks, ITU-T Std., 2012. [Online]. Available : https://www.itu.int/rec/T-REC-G.798-201212-I/fr

[42] D. Temple and L. Pedersen, Superior cpri over otn front-haul solutions, ALTERA, White Paper, 2016, wP-01263-1.0. [Online]. Available : https://www.altera.com/content/dam/altera-www/global/en_US/ pdfs/literature/wp/wp-01263-superior-cpri-over-otn-front-haul-solutions.pdf

[43] ITU-T, Recommendation ITU-T G Supplement 56 OTN transport of CPRI signals, ITU-T Std., 2016. [Online]. Available : https://www.itu.int/rec/T-REC-G. Sup56-201602-I/en

[44] M. Jamgochia and S. Trowbridge, “Understanding otn, optical transport network” (g.709), Alcatel Lucent, Tech. Rep., 2010. [Online]. Available : http://www.cvt-dallas.org/March2010.pdf

[45] T. P. Walker, Optical transport network (otn) tutorial, AMCC, techreport, 2005. [Online]. Available : https://www.itu.int/ITU-T/studygroups/com15/otn/OTNtutorial.pdf

[46] F. Labs, “Dense wavelength division multiplexing (dwdm) iut grid : C-band 100 ghz spacing”, 2014. [Online]. Available : http://www.berdyne.com/products/itu-grid.html

[47] ITU-T, ITU-T G.694.1 Spectral grils for WDM applications : DWDM frequency grid, ITU-T Std., 2012, geneva. [Online]. Available : http://www.itu.int/rec/T-REC-G.694.1-200206-S/en

[48] WONG, Elaine, LEE, Ka Lun, et ANDERSON, Trevor B, “Directly modulated self-seeding reflective semiconductor optical amplifiers as colorless transmitters in wavelength division multiplexed passive optical networks”, Journal of Lightwave Technology, 2007, vol. 25, no 1, p. 67-74.

[49] Fabienne Saliou et al., Projet ANR Lampion, document scientifique de l’appel à projet ANR INFRA 2013 (confidentiel).

[50] G. Simon, F. Saliou, P. Chanclou, Q. Deniel, D. Erasme, R. Brenot, “70 km external cavity DWDM sources based on O-band Self Seeded RSOAs for transmissions at 2.5 Gb/s”, Optical Fiber Communication Conference, 2014, W3G.5.

[51] HAMMING, Richard W. Error detecting and error correcting codes. Bell System technical journal, 1950, vol. 29, no 2, p. 147-160.

[52] VUCETIC, Branka et YUAN, Jinhong. Turbo codes : principles and applications. Springer Science & Business Media, 2012.

[53] SHANNON, Claude Elwood. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 2001, vol. 5, no 1, p. 3-55.

[54] COVER, Thomas M. et THOMAS, Joy A. *Elements of information theory*. John Wiley & Sons, 2012.

[55] MACKAY, David JC. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[56] NAKAMORI, Masami. *Fiber optics communication system*. U.S. Patent No 4,249,266, 3 févr. 1981.

[57] COVER, Thomas M. et THOMAS, Joy A. Elements of information theory. John Wiley & Sons, 2012.

[58] WANG, Yuanye et PEDERSEN, Klaus I. Performance analysis of enhanced inter-cell interference coordination in LTE-Advanced heterogeneous networks. In : Vehicular Technology Conference (VTC Spring), 2012 IEEE 75th. IEEE, 2012. p. 1-5.

[59] HOSSAIN, Ekram et HASAN, Monowar. 5G cellular : key enabling technologies and research challenges. IEEE Instrumentation & Measurement Magazine, 2015, vol. 18, no 3, p. 11-21.

[60] BERROU, Claude, GLAVIEUX, Alain, et THITIMAJSHIMA, Punya. Near Shannon limit error-correcting coding and decoding : Turbo-codes. 1. In : Communications, 1993. ICC'93 Geneva. Technical Program, Conference Record, IEEE International Conference on. IEEE, 1993. p. 1064-1070.

[61] FRANZ, Volker et ANDERSON, John B... Concatenated decoding with a reduced-search BCJR algorithm. IEEE Journal on selected Areas in Communications, 1998, vol. 16, no 2, p. 186-195.

[62] HAGENAUER, Joachim et HOEHER, Peter. A Viterbi algorithm with soft-decision outputs and its applications. In : Global Telecommunications Conference and Exhibition'Communications Technology for the 1990s and Beyond'(GLOBECOM), 1989. IEEE. IEEE, 1989. p. 1680-1686.

[63] REED, Irving S. et SOLOMON, Gustave. Polynomial codes over certain finite fields. Journal of the society for industrial and applied mathematics, 1960, vol. 8, no 2, p. 300-304.

[64] BOSE, Raj Chandra et RAY-CHAUDHURI, Dwijendra K. On a class of error correcting binary group codes. Information and control, 1960, vol. 3, no 1, p. 68-79.

[65] HAGENAUER, Joachim, OFFER, Elke, et PAPKE, Lutz. Iterative decoding of binary block and convolutional codes. IEEE Transactions on information theory, 1996, vol. 42, no 2, p. 429-445.

[66] GALLAGER, Robert. Low-density parity-check codes. IRE Transactions on information theory, 1962, vol. 8, no 1, p. 21-28.

[67] ARIKAN, Erdal. Channel polarization : A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. IEEE Transactions on Information Theory, 2009, vol. 55, no 7, p. 3051-3073.

[68] CALDERBANK, A. Robert. The art of signaling : Fifty years of coding theory.IEEE Transactions on Information Theory, 1998, vol. 44, no 6, p. 2561-2595.

[69] HAYKIN, Simon. Communication systems. John Wiley & Sons, 2008.

[70] BERLEKAMP, Elwyn R., MCELIECE, Robert J., et VAN TILBORG, Henk CA. On the inherent intractability of certain coding problems. IEEE Transactions on Information Theory, 1978, vol. 24, no 3, p. 384-386.

[71] JOHNSON, Sarah J. Iterative error correction : turbo, low-density parity-check and repeat-accumulate codes. Cambridge University Press, 2009.

[72] REED, Irving S. et SOLOMON, Gustave. Polynomial codes over certain finite fields. Journal of the society for industrial and applied mathematics, 1960, vol. 8, no 2, p. 300-304.

[73] BERROU, Claude et GLAVIEUX, Alain. Turbo codes. Encyclopedia of Telecommunications, 2003.

[74] GLAVIEUX, Alain, ADDE, Patrick, BATTAIL, Gérard, et al. Codage de canal-des bases théoriques aux turbocodes (sous la direction de Alain GLAVIEUX). 2005.

[75] PETERSON, William Wesley et BROWN, Daniel T. Cyclic codes for error detection. Proceedings of the IRE, 1961, vol. 49, no 1, p. 228-235.

[76] BOSE, Raj Chandra et RAY-CHAUDHURI, Dwijendra K. On a class of error correcting binary group codes. Information and control, 1960, vol. 3, no 1, p. 68-79.

[77] HOCQUENGHEM, Alexis. Codes correcteurs d'erreurs. Chiffres, 1959, vol. 2, no 147-156, p. 8.5.

[78] TANNER, R. A recursive approach to low complexity codes. IEEE Transactions on information theory, 1981, vol. 27, no 5, p. 533-547.

[79] SUGIYAMA, Yasuo, KASAHARA, Masao, HIRASAWA, Shigeichi, et al. A method for solving key equation for decoding Goppa codes. Information and Control, 1975, vol. 27, no 1, p. 87-99.

[80] MASSEY, James. Shift-register synthesis and BCH decoding. IEEE transactions on Information Theory, 1969, vol. 15, no 1, p. 122-127.

[81] BERLEKAMP, Elwyn R. Coding theory and the Mathieu groups. Information and Control, 1971, vol. 18, no 1, p. 40-64.

[82] MACKAY, David JC. Information theory, inference and learning algorithms. Cambridge university press, 2003.

[83] SONG, Leilei, YU, Meng-Lin, et SHAFFER, Michael S. 10-and 40-Gb/s forward error correction devices for optical communications. IEEE journal of solid-state circuits, 2002, vol. 37, no 11, p. 1565-1573.

[84] CHANG, Frank, ONOHARA, Kiyoshi, et MIZUOCHI, Takashi. Forward error correction for 100 G transport networks. IEEE Communications Magazine, 2010, vol. 48, no 3, p. S48-S55.

[85] FORNEY, G. David et FORNEY, G. David. Concatenated codes. Cambridge : MIT press, 1966.

[86] GLAVIEUX, Alain, ADDE, Patrick, BATTAIL, Gérard, et al. Codage de canal-des bases théoriques aux turbocodes (sous la direction de Alain GLAVIEUX). 2005.

[87] KORADA, Satish Babu. Polar codes for channel and source coding. 2009. Thèse de doctorat. Ecole Polytechnique Fédèrale de Lausanne.

[88] BERHAULT, Guillaume. Exploration architecturale pour le décodage de codes polaires. 2015. Thèse de doctorat. Université de Bordeaux.

[89] KABORE, Abraham Wendyida, MEGHDADI, Vahid, CANCES, Jean-Pierre, et al. Performance of Gabidulin codes for narrowband PLC smart grid networks. In : Power Line Communications and its Applications (ISPLC), 2015 International Symposium on. IEEE, 2015. p. 262-267.

[90] GARRAMMONE, Giuliano. On decoding complexity of reed-solomon codes on the packet erasure channel. IEEE Communications Letters, 2013, vol. 17, no 4, p. 773-776.

[91] 3GPP (R1-1701552), *Final Report of 3GPP TSG RAN WG1 #87 Final report, August, 2017*

[92] PAMUK, Alptekin. An FPGA implementation architecture for decoding of polar codes. In : Wireless Communication Systems (ISWCS), 2011 8th International Symposium on. IEEE, 2011. p. 437-441.

[93] PAMUK, Alptekin et ARIKAN, Erdal. A two phase successive cancellation decoder architecture for polar codes. In : Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on. IEEE, 2013. p. 957-961.

[94] BERHAULT, Guillaume. Exploration architecturale pour le décodage de codes polaires. 2015. Thèse de doctorat. Université de Bordeaux.

[95] LI, Bin, SHEN, Hui, et TSE, David. An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check. IEEE Communications Letters, 2012, vol. 16, no 12, p. 2044-2047.

[96] GIARD, Pascal, SARKIS, Gabi, THIBEAULT, Claude, et al. Multi-mode unrolled architectures for polar decoders. IEEE Transactions on Circuits and Systems I: Regular Papers, 2016, vol. 63, no 9, p. 1443-1453.

[97] LEROUX, Camille, RAYMOND, Alexandre J., SARKIS, Gabi, et al. A semi-parallel successive-cancellation decoder for polar codes. IEEE Transactions on Signal Processing, 2013, vol. 61, no 2, p. 289-299.

[98] HUANG, Z. L., DIAO, C. J., et CHEN, M. Latency reduced method for modified successive cancellation decoding of polar codes. Electronics letters, 2012, vol. 48, no 23, p. 1505-1506.

[99] ARIKAN, Erdal. Systematic polar coding. IEEE Commun. Lett, 2011, vol. 15, no 8, p. 860-862.

[100] YUAN, Bo et PARHI, Keshab K. Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders. IEEE Transactions on Signal Processing, 2014, vol. 62, no 24, p. 6496-6506.

[101] YUAN, Bo et PARHI, Keshab K. Low-latency successive-cancellation polar decoder architectures using 2-bit decoding. IEEE Transactions on Circuits and Systems I : Regular Papers, 2014, vol. 61, no 4, p. 1241-1254.

[102] SARKIS, Gabi, GIARD, Pascal, VARDY, Alexander, et al. Fast polar decoders : Algorithm and implementation. IEEE Journal on Selected Areas in Communications, 2014, vol. 32, no 5, p. 946-957.

[103] JOHNSON, Sarah J. Iterative error correction : turbo, low-density parity-check and repeat-accumulate codes. Cambridge University Press, 2009.

[104] CHUNG, Sae-Young, FORNEY, G. David, RICHARDSON, Thomas J., et al. On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit. IEEE Communications letters, 2001, vol. 5, no 2, p. 58-60.

[105] FU, Minyue. On Gaussian approximation for density evolution of low-density parity-check codes. In : 2006 IEEE International Conference on Communications. IEEE, 2006. p. 1107-1112.

[106] CHUNG, Sae-Young, RICHARDSON, Thomas J., et URBANKE, Rüdiger L. Analysis of sum-product decoding of low-density parity-check codes using a Gaussian approximation. IEEE Transactions on Information Theory, 2001, vol. 47, no 2, p. 657-670.

[107] RICHARDSON, Thomas J., SHOKROLLAHI, Mohammad Amin, et URBANKE, Rüdiger L. Design of capacity-approaching irregular low-density parity-check codes. IEEE transactions on information theory, 2001, vol. 47, no 2, p. 619-637.

[108] MACKAY, David JC et NEAL, Radford M. Near Shannon limit performance of low density parity check codes. Electronics letters, 1996, vol. 32, no 18, p. 1645-1646.

[109] RICHARDSON, Tom. Error floors of LDPC codes. In : Proceedings of the annual Allerton conference on communication control and computing. The University ; 1998, 2003. p. 1426-1435.

[110] LANDNER, Stefan et MILENKOVIC, Olgica. Algorithmic and combinatorial analysis of trapping sets in structured LDPC codes. In : 2005 International Conference on Wireless Networks, Communications and Mobile Computing. IEEE, 2005. p. 630-635.

[111] CHILAPPAGARI, Shashi Kiran, SANKARANARAYANAN, Sundararajan, et VASIC, Bane. Error floors of LDPC codes on the binary symmetric channel. In : 2006 IEEE International Conference on Communications. IEEE, 2006. p. 1089-1094.

[112]KARIMI, Mehdi et BANIHASHEMI, Amir H. Efficient algorithm for finding dominant trapping sets of LDPC codes. IEEE Transactions on Information Theory, 2012, vol. 58, no 11, p. 6942-6958.

[113]HUANG, Qin, DIAO, Qiuju, LIN, Shu, et al. Cyclic and quasi-cyclic LDPC codes on constrained parity-check matrices and their trapping sets. IEEE Transactions on Information Theory, 2012, vol. 58, no 5, p. 2648-2671.

[114]VASIC, Milos Ivkovic Shashi Kiran Chilappagari, et al. Eliminating trapping sets in low-density parity check codes by using Tanner graph covers. arXiv preprint arXiv:0805.1662, 2008.

[115]MCGREGOR, Andrew et MILENKOVIC, Olgica. On the hardness of approximating stopping and trapping sets. IEEE Transactions on Information Theory, 2010, vol. 56, no 4, p. 1640-1650.

[116]DI, Changyan, PROIETTI, David, TELATAR, I. Emre, et al. Finite-length analysis of low-density parity-check codes on the binary erasure channel. IEEE Transactions on Information theory, 2002, vol. 48, no 6, p. 1570-1579.

[117]HU, Xiao-Yu, ELEFTHERIOU, Evangelos, et ARNOLD, D.-M. Progressive edge-growth Tanner graphs. In : Global Telecommunications Conference, 2001. GLOBECOM'01. IEEE. IEEE, 2001. p. 995-1001.

[118]HU, Xiao-Yu, ELEFTHERIOU, Evangelos, et ARNOLD, Dieter-Michael. Regular and irregular progressive edge-growth tanner graphs. IEEE Transactions on Information Theory, 2005, vol. 51, no 1, p. 386-398.

[119]HU, Xiao-Yu, FOSSORIER, Marc PC, et ELEFTHERIOU, Evangelos. On the computation of the minimum distance of low-density parity-check codes. In : Communications, 2004 IEEE International Conference on. IEEE, 2004. p. 767-771.

[120]BANGERTER, Boyd, JACOBSEN, Eric, HO, Minnie, et al. High-Throughput Wireless LAN Air Interface. Intel Technology Journal, 2003, vol. 7, no 3.

[121]URARD, P., YEO, E., PAUMIER, L., et al. A 135Mb/s DVB-S2 compliant codec based on 64800b LDPC and BCH codes. In : ISSCC. 2005 IEEE International Digest of Technical Papers. Solid-State Circuits Conference, 2005. IEEE, 2005. p. 446-609.

[122]BRACK, Torben, ALLES, Matthias, KIENLE, Frank, et al. A synthesizable IP core for WIMAX 802.16 e LDPC code decoding. In : 2006 IEEE 17th international symposium on personal, indoor and mobile radio communications. IEEE, 2006. p. 1-5.

[123]WANG, Yige, YEDIDIA, Jonathan S., et DRAPER, Stark C. Construction of high-girth QC-LDPC codes. In : 5th International Symposium on Turbo Codes and Related Topics. Lausanne, Switzerland, 2008.

[124]FOSSORIER, Marc PC. Quasicyclic low-density parity-check codes from circulant permutation matrices. IEEE Transactions on Information Theory, 2004, vol. 50, no 8, p. 1788-1793.

[125]YANG, Lei, LIU, Hui, et SHI, C.-JR. VLSI implementation of a low-error-floor and capacity-approaching low-density parity-check code decoder with multi-rate capacity. In : GLOBECOM'05. IEEE Global Telecommunications Conference, 2005. IEEE, 2005. p. 6 pp.

[126]WYMEERSCH, Henk, STEENDAM, Heidi, et MOENECLAEY, Marc. Log-domain decoding of LDPC codes over GF (q). In : Communications, 2004 IEEE International Conference on. IEEE, 2004. p. 772-776.

[127]FOSSORIER, Marc PC, MIHALJEVIC, Miodrag, et IMAI, Hideki. Reduced complexity iterative decoding of low-density parity check codes based on belief propagation. IEEE Transactions on communications, 1999, vol. 47, no 5, p. 673-680.

[128]YAZDANI, Mohammad R., HEMATI, Saied, et BANIHASHEMI, Amir H. Improving belief propagation on graphs with cycles. IEEE Communications Letters, 2004, vol. 8, no 1, p. 57-59.

[129]CHEN, Jinghu et FOSSORIER, Marc PC. Density evolution for two improved BP-based decoding algorithms of LDPC codes. IEEE Communications Letters, 2002, vol. 6, no 5, p. 208-210.

[130]JIANG, Ming, ZHAO, Chunming, ZHANG, Li, et al. Adaptive offset min-sum algorithm for low-density parity check codes. IEEE communications letters, 2006, vol. 10, no 6, p. 483-485.

[131]XU, Meng, WU, Jianhui, et ZHANG, Meng. A modified offset min-sum decoding algorithm for LDPC codes. In : Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on. IEEE, 2010. p. 19-22.

[132]SAVIN, Valentin. Self-corrected Min-Sum decoding of LDPC codes. In : 2008 IEEE International Symposium on Information Theory. IEEE, 2008. p. 146-150.

[133]BALATSOUKAS-STIMMING, Alexios et DOLLAS, Apostolos. FPGA-based design and implementation of a multi-GBPS LDPC decoder. In : Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on. IEEE, 2012. p. 262-269.

[134]LI, Ao, MEGHDADI, Vahid, CANCES, Jean-Pierre, Christelle Aupetit-Berthelemot. High Rate LDPC Based Decoder Architectures with High Speed ADC for C-RAN Optical Fronthaul. In : Computer and Communication Engineering (ICCCE), 2016 International Conference on. IEEE, 2016. p. 386-391.

[135]LI, Ao, MEGHDADI, Vahid, CANCES, Jean-Pierre, Christelle Aupetit-Berthelemot High throughput LDPC decoder for C-RAN optical fronthaul based on improved bit-flipping algorithm. In : Communication Systems, Networks and Digital Signal Processing (CSNDSP), 2016 10th International Symposium on. IEEE, 2016. p. 1-5.

[136]MILADINOVIC, Nenad et FOSSORIER, Marc PC. Improved bit-flipping decoding of low-density parity-check codes. IEEE Transactions on Information Theory, 2005, vol. 51, no 4, p. 1594-1606.

[137]CHANG, Tofar C.-Y. et SU, Yu T. Dynamic Weighted Bit-Flipping Decoding Algorithms for LDPC Codes. IEEE Transactions on Communications, 2015, vol. 63, no 11, p. 3950-3963.

[138]NGUYEN, Dung Viet et VASIC, Bane. Two-bit bit flipping algorithms for LDPC codes and collective error correction. IEEE Transactions on Communications, 2014, vol. 62, no 4, p. 1153-1163.

[139]CHO, Junho et SUNG, Wonyong. Adaptive threshold technique for bit-flipping decoding of low-density parity-check codes. IEEE Communications Letters, 2010, vol. 14, no 9, p. 857-859.

[140]KOU, Yu, LIN, Shu, et FOSSORIER, Marc PC. Low-density parity-check codes based on finite geometries : a rediscovery and new results. IEEE Transactions on Information Theory, 2001, vol. 47, no 7, p. 2711-2736.

[141]ZHANG, Juntan et FOSSORIER, Marc PC. A modified weighted bit-flipping decoding of low-density parity-check codes. IEEE Communications Letters, 2004, vol. 8, no 3, p. 165-167.

[142]JIANG, Ming, ZHAO, Chunming, SHI, Zhihua, et al. An improvement on the modified weighted bit flipping decoding algorithm for LDPC codes. IEEE Communications Letters, 2005, vol. 9, no 9, p. 814-816.

[143]GUO, Feng et HANZO, Lajos. Reliability ratio based weighted bit-flipping decoding for low-density parity-check codes. Electronics Letters, 2004, vol. 40, no 21, p. 1356-1358.

[144]WU, Xiaofu, ZHAO, Chunming, et YOU, Xiaohu. Parallel weighted bit-flipping decoding. IEEE Communications letters, 2007, vol. 11, no 8.

[145]CHEN, Tso-Cho. Adaptive-weighted multibit-flipping decoding of lowdensity parity-check codes based on ordered statistics. IET communications, 2013, vol. 7, no 14, p. 1517-1521.

[146]ZHANG, Lina, YE, Zhihui, et FENG, Qi. An Improved Multi-Bit Threshold Flipping LDPC Decoding Algorithm. International Journal of Computer Theory and Engineering, 2014, vol. 6, no 6, p. 510.

[147]LI, Guangwen, LI, Dashe, WANG, Yuling, et al. Improved parallel weighted bit flipping decoding of finite geometry LDPC codes. In : Communications and Networking in China, 2009. ChinaCOM 2009. Fourth International Conference on. IEEE, 2009. p. 1-5.

[148]CHANG, Tofar C.-Y. et SU, Yu T. Dynamic Weighted Bit-Flipping Decoding Algorithms for LDPC Codes. IEEE Transactions on Communications, 2015, vol. 63, no 11, p. 3950-3963.

[149]WADAYAMA, Tadashi, NAKAMURA, Keisuke, YAGITA, Masayuki, et al. Gradient descent bit flipping algorithms for decoding LDPC codes. In : Information Theory and Its Applications, 2008. ISITA 2008. International Symposium on. IEEE, 2008. p. 1-6.

[150]BOYD, Stephen et VANDENBERGHE, Lieven. Convex optimization. Cambridge university press, 2004.

[151]SUNDARARAJAN, Gopalakrishnan, WINSTEAD, Chris, et BOUTILLON, Emmanuel. Noisy gradient descent bit-flip decoding for LDPC codes. IEEE Transactions on Communications, 2014, vol. 62, no 10, p. 3385-3400.

[152]PHROMSA-ARD, Tharathorn, ARPORNSIRIPAT, Jiratchaporn, WETCHARUNGSRI, Jutaphet, et al. Improved gradient descent bit flipping algorithms for LDPC decoding. In : Digital Information and Communication Technology and it's Applications (DICTAP), 2012 Second International Conference on. IEEE, 2012. p. 324-328.

[153]HAGA, Ryoji et USAMI, Shogo. Multi-bit flip type gradient descent bit flipping decoding using no thresholds. In : Information Theory and its Applications (ISITA), 2012 International Symposium on. IEEE, 2012. p. 6-10.

[154]TITHI, Tasnuva, WINSTEAD, Chris, et SUNDARARAJAN, Gopalakrishnan. Decoding LDPC codes via noisy gradient descent bit-flipping with re-decoding. ArXiv preprint arXiv:1503.08913, 2015.

[155]ISMAIL, Mohamed, AHMED, Imran, et COON, Justin. Low power decoding of LDPC codes. ISRN Sensor Networks, 2013, vol. 2013.

[156]CHANDRASETTY, Vikram Arkalgud et AZIZ, Syed Mahfuzul. FPGA implementation of high performance LDPC decoder using modified 2-bit min-sum algorithm. In : Computer Research and Development, 2010 Second International Conference on. IEEE, 2010. p. 881-885.

[157]KAVITHA, P. Modified min sum algorithm for low density parity check decoder for FPGA implementation. Matrix, 2015, vol. 1, no 8.

[158]FOSSORIER, Marc PC. Quasicyclic low-density parity-check codes from circulant permutation matrices. IEEE Transactions on Information Theory, 2004, vol. 50, no 8, p. 1788-1793.

[159]ARDAKANI, Masoud, SMITH, Benjamin, YU, Wei, et al. Complexity-optimized low-density parity-check codes. In : Proceedings of the Forty-Third Annual Allerton Conference on Communication, Control and Computing. 2005. p. 45-54.

[160]AL HARIRI, Alaa Aldin, MONTEIRO, Fabrice, SIÉLER, Loïc, et al. A high throughput configurable partially-parallel decoder architecture for Quasi-Cyclic Low-Density Parity-Check Codes. In : Design of Circuits and Integrated Circuits (DCIS), 2014 Conference on. IEEE, 2014. p. 1-6.

[161]DAI, Yongmei, CHEN, Ning, et YAN, Zhiyuan. Memory efficient decoder architectures for quasi-cyclic LDPC codes. IEEE Transactions on Circuits and Systems I: Regular Papers, 2008, vol. 55, no 9, p. 2898-2911.

[162]WANG, Zhongfeng et CUI, Zhiqiang. Low-complexity high-speed decoder design for quasi-cyclic LDPC codes. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2007, vol. 15, no 1, p. 104-114.

[163]KHAN, Zahid, ARSLAN, Tughrul, et MACDOUGALL, Scott. A real time programmable encoder for low density parity check code as specified in the IEEE P802. 16E/D7 standard and its efficient implementation on a DSP processor. In : SoC Conference, 2006 IEEE International. IEEE, 2006. p. 17-20.

[164]SCHLÄFER, Philipp, WEIS, Christian, WEHN, Norbert, et al. Design space of flexible multigigabit LDPC decoders. VLSI Design, 2012, vol. 2012, p. 4.

[165]ZHANG, Juntan et FOSSORIER, Marc. Shuffled belief propagation decoding. In : Signals, Systems and Computers, 2002. Conference Record of the Thirty-Sixth Asilomar Conference on. IEEE, 2002. p. 8-15.

[166]HOCEVAR, Dale E. A reduced complexity decoder architecture via layered decoding of LDPC codes. In : Signal Processing Systems, 2004. SIPS 2004. IEEE Workshop on. IEEE, 2004. p. 107-112.

[167]DIELISSEN, John, HEKSTRA, Andries, et BERG, Vincent. Low cost LDPC decoder for DVB-S2. In : Proceedings of the conference on Design, automation and test in Europe : Designers' forum. European Design and Automation Association, 2006. p. 130-135.

[168]BLANKSBY, Andrew J. et HOWLAND, Chris J. A 690-mW 1-Gb/s 1024-b, rate-1/2 low-density parity-check code decoder. IEEE Journal of solid-state circuits, 2002, vol. 37, no 3, p. 404-412.

[169]DARABIHA, Ahmad, CARUSONE, Anthony Chan, et KSCHISCHANG, Frank R. A bit-serial approximate min-sum LDPC decoder and FPGA implementation. In : Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on. IEEE, 2006. p. 4 pp.

[170]KORB, Matthias et NOLL, Tobias G. Area and latency optimized high-throughput Min-Sum based LDPC decoder architectures. In : ESSCIRC, 2009. ESSCIRC'09. Proceedings of. IEEE, 2009. p. 408-411.

Index

3GPP : Third Generation Partnership Project

A-BS : All-in-one BS

ADC : analog to digital converter

AMWBF : Adaptive Modified Weighted Bit-Flipping Algorithm

ARPU : Average Revenue Per User

AT-GDBF : Adaptive Threshold Gradient Descent Bit-Flipping Algorithm

AWGN : Additive White Gaussian Noise

BBU : Base Band Unit

BCH : Bose–Chaudhuri–Hocquenghem

BCJR : Bahl, Cocke, Jelinek and Raviv

BER : Bit Error Rate

BF : Bit-Flipping Algorithm

BP : Belief Propagation

BPSK : binary phase-shift keying

BS : Base Station ou Station de Base

BSC : Binary Symmetric Channel

BSC : contrôleur de stations de Base

BWGDBF : Balanced Weighted Gradient Descent Bit-Flipping Algorithm

CA : Carrier Aggregation

CAGR : Compound Annual Growth Rate

CAN : (convertisseur analogique numérique)

CAPEX : CAPital Expenditure

CCO : core central office

CE : Cycle Elimination

CN : Core Network ou Réseau Cœur

CNU : Check node Unit

CoMP : Coordinated Multipoint

CP : Polar Code

CPRI : Common Public Radio Interface

C-RAN : Cloud Radio Access Network

CRC : Cyclic Redundancy Check

CSG : Cell Site Gateway

D-BS (distributed BS

DE : Density Evolution

D-RoF : Digital Radio over Fiber

DU : Digital Unit

DWBF : Dynamic Weighted Bit-Flipping Algorithm

DWDM : Dense Wavelength Division Multiplexing

eNodeB : evolved-NodeB

EPC : Evolved Packet Core

FEC : Forward error correction

FH-C (FH to Core)

FH-I : (FrontHaul to intermediate

FPGA : Field-Programmable Gate Array

GDBF : Gradient Descent Bit-Flipping Algorithm

GPRS : General Packet Radio Service

GSM : Global System for Mobile communications ou système mondial de communications mobiles

HGDBF : Hybrid Gradient Descent Bit-Flipping Algorithm

ICO : intermediate Central Office

IHGDBF : Improve Hybrid Gradient Descent Bit-Flipping Algorithm

IMWBF : Improve Modified Weighted Bit-Flipping Algorithm

IOT : Internet of Things

IPWBF : Improved Parallel Weighted Bit-Flipping Algorithm

LDPC : Low Density Parity Check Code

LLR : Log-Likehood Ratios

LPWBF : Liu-pados Weighted Bit-Flipping Algorithm

LTE : Long Term Evolution

LTE-A : Long Term Evolution Avanced

M2M : Machine to Machine

MGDBF : Multple Flipping Gradient Descent Bit-Flipping Algorithm

MIMO : multi-input multiple output

MIWBF : Multiple improved Weighted Bit-Flipping Algorithm

ML : Maximum Likehood

M-NGDBF : Multiple-NGDBF

MOMS : Modified Offset Min-sum Algorithm

MPA : Message Passig Algorithm

MS : Min-Sum Algorithm

MSGDBF : Multiple then Singled Flipping Gradient Descent Bit-Flipping Algorithm

MWBF : Modified Weighted Bit-Flipping Algorithm

NCG : Net Coding Gain

NFV : Network Function Virtualisation

NGDBF : Noisy Gradient Descent Bit-Flipping Algorithm

NMS : Normalized Min-sum Algorithm

NSC : Non Systematic Convolution

OBSAI : Open Base Station Aechitecture Initiative

OFDM : Orthogonal Frequency Division Multiplexing

OMS : Offset Min--sum Algorithm

OPEX : OPerating EXpenditure

PEG : Progressive Edge Growth

PWBF : Parallel Weighted Bit-Flipping Algorithm

QC : Quasi Cyclic

RAN : Radio Access Network

RNC : Radio Network controller

RRH : Remote Radio Head

RRWBF : Reliability Based Ratio Weighted Bit-Flipping Algorithm

RRWGDBF : Reliability Ratio Weighted Gradient Descent Bit-Flipping Algorithm

RS : Reed-solomon

RSC : Recursive Systematic Convolution

RSOAs : Reflective Semi-conductor Optical Amplifiers

RU : Radio Unit

SC : Successive Cancelation

SCMS : Self corrected Min-sum Algorithm

SDN : Software Defined Networking

SFP : Small Form Factor

SGDBF : Singed Flipping Gradient Descent Bit-Flipping Algorithm

SLBWBF : Simplified Loop Break Weighted Bit-Flipping Algorithm

SMGDBF : singled then Multiple Gradient Descent Bit-Flipping Algorithm

SMS : Short Message Service

S-NGDBF : Singled -NGDBF

SNR : Signal to Noise Ratio

SOVA : Soft Output Viterbi Algorithm

SPA : Sum-product Algorithm

TCAC : Taux de Croissance Annuel Composé

TDMA : Time Division Multiple Access ou accès multiple par répartition dans le temps

TEB : Taux d'erreur

UMTS : Universal Mobile Telecommunications System

UTRAN : UMTS Terrestrial Radio Access Network

VNU : Variable node Unit

WBF : Weighted Bit-Flipping Algorithm

WGDBF : Weighted Gradient Descent Bit-Flipping Algorithm

WSWBF : Weighted-sum Weighted Bit-Flipping Algorithm

Table des illustrations

Table des tableaux

Publications de l'auteur

Soumission dans un journal :

Ao Li, Vahid Meghdadi, Jean-Pierre Cances and Christelle-Berthelemot, "High throughput 2-bit LDPC FEC for C-RAN optical front-haul based on hard decision algorithm", soumis à la revue IET Research Journal Papers, Circuits, Devices & Systems, en revision.

Conférences Internationales

Ao Li, Vahid Meghdadi, Jean-Pierre Cances and Christelle-Berthelemot, "High Throughput LDPC Decoder for C-RAN optical front-haul based on improved bit-flipping algorithm", 10th IEEE/IET International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), 20- 22 July 2016, Prague, République tchèque.

Ao Li, Vahid Meghdadi, Jean-Pierre Cances and Christelle-Berthelemot, "High Rate LDPC Based Decoder Architectures with High Speed ADC for C-RAN Optical Fronthaul", 6th International Conference on Computer and Communication Engineering 2016 (ICCCE 2016), 25-27 July 2016, Kuala Lumpur, Malaysia.

Mathilde Gay, Kamal Hussain, Claude Le Bouëtté, Jean-Luc Pamart, Laurent Schoc, Christelle Aupetit-Berthelemot, Li Ao, Vahid Meghdadi, Romain Brenot, Anaëlle Maho, Philippe Chanclou, Sylvain Barthomeuf, Fabienne Saliou, "Colourless Self-Seeded Source for CPRI3 Mobile Fronthaul over 70 km Reach", 43rd European Conference on Optical Communication (ECOC 2017), Sep 2017, Gothenburg, Sweden

Workshop

Ao Li, Vahid Meghdadi, Jean-Pierre Cances and Christelle-Berthelemot, "Modeling and simulation of LDPC-FEC in high speed optical transmission and implementation using FPGA", Worshop Interne Xlim, Limoges France, 2015, Malaysia.

Table des matières

Performances des codes correcteurs d'erreur LDPC appliqués au lien Fronthaul optique haut-débit pour l'architecture C-RAN du réseau 5G : conception et implantation sur FPGA

De nos jours, l'architecture du réseau mobile est en pleine évolution pour assurer la montée en débit entre les Centraux (CO) (réseaux cœurs) et différents terminaux comme les mobiles, ordinateurs, tablettes afin de satisfaire les utilisateurs. Pour faire face à ces défis du futur, le réseau C-RAN (Cloud ou Centralized-RAN) est connu comme une solution de la 5G. Dans le contexte C-RAN, toutes les BBUs (Base Band Units) sont centralisées dans le CO, seules les RRH (Remote Radio Head) restent situées à la tête de la station de base (BS). Un nouveau segment entre les BBUs et RRHs apparait nommé « fronthaul ». Il est basé sur des transmissions D-ROF (digital radio-overfiber) et transporte le signal radio numérique à un débit binaire élevé en utilisant le protocole CPRI (Common Public Radio Interface). En prenant en compte le CAPEX et l'OPEX, le projet ANR LAMPION a proposé la technologie RSOA (Reflective Semiconductor Optical Amplifier) auto alimentée afin de rendre la solution plus flexible et s'affranchir d'émetteurs/récepteurs colorés dans le cadre de transmission WDM-PON (Wavelength Division Multiplexing Passive Optical Network). Néanmoins, il est nécessaire d'ajouter un FEC (forward error corrector) dans la transmission pour assurer la qualité de service. Donc l'objectif de cette thèse est de trouver le FEC le plus adéquat à appliquer dans le contexte C-RAN. Nos travaux se sont focalisés sur l'utilisation de codes LDPC, choisis après comparaisons des performances avec les autres types de codes. Nous avons précisé les paramètres (rendement du code, taille de la matrice, cycle, etc.) nécessaires pour les codes LDPC afin d'obtenir les meilleures performances. Les algorithmes LDPC à décisions dures ont été choisis après considération du compromis entre complexités de circuit et performance. Parmi ces algorithmes à décisions duress, le GDBF (gradient descent bit-flipping) était la meilleure solution. La prise en compte d'un CAN 2-Bit dans le canal nous a amené à proposer une variante : le BWGDBF (Balanced weighted GDBF). Des optimisations ont également été faites en regard de la convergence de l'algorithme et de la latence. Enfin, nous avons réussi à implémenter notre propre algorithme sur le FPGA Spartan 6 xc6slx16. Plusieurs méthodes ont été proposées pour atteindre une latence de 5 µs souhaitée dans le contexte C-RAN. Cette thèse a été soutenue par le projet ANR LAMPION (Lambada-based Access and Metropolitan Passive Optical networks).

Mots-clés : Codes Correcteurs d'erreur, C-RAN, LDPC, Fronthaul optique

Modeling and simulation of high speed optical transmission and forward error correction design and implementation using FPGA

Nowadays, the architecture of the mobile network is in full evolution to ensure the increase in terms of bit rate between the Central (CO) (core networks) and various terminals such as mobiles, computers, tablets in order to satisfy the users. To address these challenges of the future, the C-RAN (Cloud or Centralized-RAN) network is known as a 5G solution. In the C-RAN context, all BBUs (Base Band Units) are centralized in the CO, only the RRH (Remote Radio Head) remain at the head of the base station (BS). A new segment between BBUs and RRHs appears called "fronthaul". It is based on D-ROF (digital radio-overfiber) transmissions and carries the digital radio signal at a high bit rate using the Common Public Radio Interface (CPRI) protocol. Taking into account CAPEX and OPEX, the ANR LAMPION project has

proposed the Self-seeded Reflective Semiconductor Optical Amplifier (RSOA) technology in order to make the solution more flexible and overcome the need for colored transmitters / receivers in the context of PON-WDM (Wavelength Division Multiplexing Passive Optical Network). Nevertheless, it is necessary to add a FEC (forward error corrector) in the transmission to ensure the quality of service. So the objective of this thesis is to find the most suitable FEC to apply in the C-RAN context. Our work has focused on the use of LDPC codes, chosen after performance comparisons with other types of codes. We have specified the parameters (code performance, matrix size, cycle, etc.) required for LDPC codes to obtain the best performance. Hard-decision LDPC algorithms were chosen after considering the tradeoff between circuit complexities and performance. Among these hard-decision algorithms, the GDBF (gradient descent bit-flipping) was the best solution. Taking into account a CAN 2-Bit in the channel led us to propose a variant: the BWGDBF (Balanced weighted GDBF). Optimizations have also been made with respect to the convergence of the algorithm and latency. Finally, we managed to implement our own algorithm on the Spartan FPGA 6 xc6slx16. Several methods have been proposed to achieve a latency of 5 µs desired in the C-RAN context. This thesis was supported by the project ANR LAMPION (Lambada-based Access and Metropolitan Passive Optical Networks).

Keywords : Forward Error Code, C-RAN, LDPC, Optical Fronthaul