Étude de codes LDPC pour applications spatiales optiques et conception des décodeurs associés

Vincent Pignoly

université de BORDEAUX

THÈSE PRÉSENTÉE
POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE N° 209 : SCIENCES PHYSIQUES ET DE L'INGÉNIEUR
SPÉCIALITÉ : ÉLECTRONIQUE

par **Vincent PIGNOLY**

Étude de codes LDPC pour applications spatiales optiques et conception des décodeurs associés

Directeur de thèse : Christophe JÉGO
Co-encadrants de thèse : Bertrand LE GAL
Benjamin GADAT

préparée au Laboratoire IMS

soutenue le 22 janvier 2021

Jury :

Jean-Pierre CANCES	- Professeur des Universités	- Université de Limoges	*Rapporteur*
Frédéric GUILLOUD	- Professeur IMT	- IMT Atlantique	*Rapporteur*
Charly POULLIAT	- Professeur des Universités	- Toulouse INP	*Examinateur*
Benjamin GADAT	- Ingénieur Docteur	- Airbus Defence & Space	*Co-encadrant*
Bertrand LE GAL	- Maître de Conférences	- Bordeaux INP	*Co-encadrant*
Christophe JÉGO	- Professeur des Universités	- Bordeaux INP	*Directeur*

Thèse réalisée au

Laboratoire de l'Intégration du Matériau au Système (IMS)
de Bordeaux, au sein de l'équipe CSN du groupe Conception.

Université de Bordeaux, Laboratoire IMS
UMR 5218 CNRS - Bordeaux INP
351 Cours de la Libération
Bâtiment A31
33405 Talence Cedex
FRANCE

Remerciements

Tout d'abord, je tiens à remercier les membres qui composent mon jury, le professeur Charly Poulliat, de l'ENSEEIHT - Toulouse INP, pour l'évaluation de mon travail et puis le professeur Jean-Pierre Cances, de l'Université de Limoges et le professeur Frédéric Guilloud, de l'IMT Atlantique, d'avoir accepté de rapporter ma thèse.

Merci à Christophe pour m'avoir donné la possibilité de me lancer dans cette expérience incroyable. Ton suivi sérieux et nos discussions m'ont beaucoup apporté et ce fut un réel plaisir que de travailler avec un tel directeur de thèse. Merci à Benjamin pour avoir encadré ma thèse depuis Airbus à Toulouse. Merci de m'avoir aussi bien reçu dans ton équipe. Nos échanges, toujours agréables et pertinents, m'ont beaucoup enrichi. Merci à Bertrand. Depuis que tu m'as accueilli au sein de l'équipe CSN au début de mon stage de fin d'études, tu n'as cessé de faire preuve de gentillesse, de patience, de pédagogie, d'humour et de soutien. Merci beaucoup !

Du côté de Toulouse, je remercie encore Benjamin, mais aussi Hugo, Jean-Adrien, Jean-Frédéric, Lyonel et Sylvain pour l'accueil dans votre équipe au cours des huit mois que j'ai eu la chance de passer chez Airbus. Du côté de l'IMS, je remercie Logan, mon binôme, mon collègue, mais surtout mon ami. Je ne compte plus les bons moments passés ensemble. Merci aux autres camarades de l'équipe CSN, la *dream team*, Adrien, Camille L., Camille M., Clémence G., Dominique, Guillaume, Hugues, Jérémie, Jonathan, Maël, Mathieu, Olivier, Rémi, Thibaud, Yann, Yassine et les autres. Vous avez rendu ces années dans l'*open space* super agréables. Les compagnons d'aventures vidéo-ludiques se reconnaîtront.

Je tiens aussi à remercier mes proches. Merci à mes parents, je ne pouvais pas souhaiter une meilleure famille. Merci aussi à Camille, ma petite sœur adorée, et à mamie Colette.

Enfin, merci à mon amour Clémence, qui a toujours été à l'écoute, m'a toujours soutenu, motivé et encouragé dans ce que j'ai pu entreprendre. Il s'est passé de belles choses pendant ces années avec toi. Merci d'être là, à mes côtés au quotidien.

Résumé

Les systèmes de communications numériques sont omniprésents dans notre quotidien. L'évolution des besoins implique la recherche et le développement de solutions innovantes pour les futurs systèmes de communications. Dans le cadre des communications numériques satellitaires, la plupart des satellites utilisent des liens par radiofréquences pour communiquer avec la Terre. Pour limiter l'utilisation de bande passante et augmenter les débits, les technologies de communications numériques via des liens optiques constituent une alternative intéressante. Ces technologies utilisent des lasers pour l'émission des données et des télescopes en réception. Cependant, l'énergie lumineuse est absorbée ou déviée par les particules présentes dans l'atmosphère terrestre. Ces perturbations sont à l'origine de nouvelles problématiques et de nouveaux schémas de codage doivent être mis au point pour y remédier.

Les codes LDPC sont une famille de codes correcteurs d'erreurs. Leurs performances proches de la limite de Shannon en font des solutions très attractives pour les systèmes de communications numériques. Ils ont notamment été sélectionnés dans le standard Wifi et pour la 5G, permettant d'atteindre de très hauts débits (plusieurs Gbit/s). Ils ont aussi été retenus par les standards CCSDS et DVB-S2 pour des applications spatiales.

Cette thèse porte sur l'étude et l'implantation matérielle de schémas de codage appliqués à des communications numériques satellitaires via des liens optiques. La première contribution est l'étude d'un schéma de codage pour un lien optique descendant avec un décodage canal à entrées souples au sol. Dans le cadre de cette étude, une architecture matérielle permettant d'implanter le processus de décodage sur FPGA et capable d'atteindre un débit attendu de 10 Gbit/s a été développée. Une deuxième contribution porte sur le lien optique montant impliquant un décodeur canal à entrées dures embarqué dans un satellite. Les contraintes qui en découlent ont amené à repenser l'algorithme Gallager B étendu. Cela a permis la conception d'une nouvelle architecture permettant d'effectuer efficacement un décodage à entrées dures tout en respectant les contraintes spatiales au niveau de la complexité matérielle, de la dissipation thermique et du débit (10 Gbit/s).

Mots clefs : Communications numériques spatiales optiques, Codes LDPC, Architecture matérielle, FPGA.

Abstract

Digital communication systems are everywhere in our daily life. The evolution of needs implies the research and development of innovative solutions for future communication systems. Considering space digital communications, most satellites use radio frequency links to communicate with Earth. To minimize bandwidth usage and increase the throughput, digital communication technologies based on optical links represent an interesting alternative. However, luminous energy is absorbed by particles that are present in the Earth's atmosphere. These perturbations imply new issues and new coding schemes must be developed to cope with them.

LDPC codes are an error correction code family. Their performance near Shannon's limit makes them an attractive solution for digital communication systems. They have been selected in Wifi and 5G standards to achieve very high throughputs (several Gbps). They were also adopted in CSSDS and DVB-S2 standards for space applications.

This thesis is about the study and the hardware implementation of coding schemes applied on optical links for space digital communication systems. The first contribution is the study of a coding scheme for an optical downlink with a soft input decoder on Earth. In this study, we developed a hardware architecture capable of implementing the decoding process on FPGA. The designed decoder reaches the expected throughput of 10 Gbps. A second contribution is about the optical uplink that implies hard input decoding in a satellite. Resulting constraints led us to rethink the extended Gallager B algorithm. It made possible the development of a new architecture that manages the hard input decoding process efficiently while being compliant with space constraints, such as hardware complexity, heat dissipation and throughput (10 Gbps).

Key words: Space optical digital communications, LDPC codes, Hardware architecture, FPGA

Table des matières

Liste des figures

Liste des tableaux

Liste des acronymes

ALU	Arithmetic Logic Unit
ASIC	Application Specific Integrated Circuit
ASIP	Application Specific Instruction set Processor
AWGN	Additive White Gaussian Noise
BCJR	Algorithme de décodage MAP nommé d'après ses inventeurs : Bahl, Cocke, Jelinek et Raviv
BER	Bit Error Rate
BI – AWGN	Binary Input - AWGN
BPSK	Binary Phase-Shift Keying
BRAM	Block Random Access Memory
CN	Check Node
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
DSP	Digital Signal Processor
ECC	Error Correction Codes
FB	Frozen Bit
FER	Frame Error Rate
FF	Flip-Flop
FIFO	First In, First Out
FPGA	Field-Programmable Gate Array
FSM	Finite State Machine
HDL	Hardware Description Language
IoT	Internet of Things
LDPC	Low Density Parity Check
LLR	Log Likelihood Ratio
LUT	Look-Up Table
MAP	Maximum A Posteriori
QC	Quasi-Cyclic
QC – LDPC	Quasi-Cyclic LDPC

RAM	Random Access Memory
SC	Successive Cancellation
SCL	Successive Cancellation List
SNR	Signal-to-Noise Ratio
SoC	System-on-Chip
SPA	Sum-Product Algorithm
VHDL	VHSIC Hardware Description Language
VN	Variable Node

Introduction

Motivations des travaux de thèse

Actuellement, il n'y a pas moins de 2600 satellites actifs en orbite autour de la Terre. C'est un chiffre en constante augmentation. Nous pouvons notamment penser au programme Starlink de SpaceX qui compte déjà plus de 360 satellites en orbite et prévoit à terme une constellation de 12 000 satellites en 2025, et même de 42 000 dans un futur plus lointain. Ce nombre élevé de satellites, associé à des besoins en débit toujours plus importants, pousse à la recherche de nouvelles alternatives aux systèmes radio-fréquences. Ces alternatives sont nécessaires, notamment en raison des problèmes d'interférences et d'occupation spectrale de plus en plus prégnant dans les bandes passantes naturelles et efficaces.

Dans ce contexte, l'utilisation de lasers pour transmettre de l'information semble pertinent. En effet, grâce à la finesse du rayon laser, des interférences négligeables sont engendrées et le signal n'occupe pas la bande hertzienne. Cette solution technologique présente d'autres avantages. Par exemple, l'interception de l'information transmise est plus difficile. Cela permet d'augmenter la sécurité du système de communications. Cela facilite également la réduction de la consommation énergétique des systèmes d'émission et de réception comme cela est détaillé dans [2].

Les systèmes de communications à base de laser intéressent l'ingénierie spatiale depuis la fin des années 70. La première communication via un lien optique a été réalisée en 1992 quand la sonde Galileo en route pour Jupiter a reçu de l'information envoyée depuis la Californie [3]. Trois ans plus tard, en 1995, le premier lien optique bidirectionnel a été mis en œuvre entre le satellite japonais ETS-VI et la Terre [4]. Puis en 2001, l'Europe a mis en place le premier lien optique direct entre deux satellites [5]. Depuis, d'autres démonstrations de déploiement de liens optiques ont été faites dans les régions les plus avancées technologiquement, c'est à dire aux États-Unis, en Europe, au Japon, en Russie et en Chine.

Les avancées technologiques ont rendu l'utilisation de liens optiques plus robuste et plus abordable pour aboutir à des démonstrateurs fonctionnels. Le Consultative Committee for Space Data Systems (CCSDS) a d'ailleurs débuté la définition d'un standard dédié aux communications optiques pour les applications spatiales. Son apparition dans les feuilles de route des acteurs industriels du domaine spatial est un autre facteur démontrant l'engouement grandissant pour cette technologie.

Dans le cadre de deux études R&T (Recherche & Technologie) financées par le CNES (Centre National d'Études Spatiales) sur les liens optiques, deux applications ont été envisagées. La première application consiste à transmettre des images acquises par un satellite en orbite basse vers une station de base terrestre. La seconde application est un *feeder* optique. Cette application est composée d'un lien optique entre la station de base et le satellite et de liens radiofréquences vers des utilisateurs terrestres.

Dans ces deux cadres applicatifs, le lien optique doit faire face à des perturbations lors de sa traversée de l'atmosphère. En effet, les particules présentes dans l'atmosphère absorbent et dévient le signal. Ainsi l'utilisation de codes correcteurs d'erreurs s'avère nécessaire afin d'augmenter la robustesse des liens de communications optiques.

Les codes LDPC sont une famille de codes correcteurs d'erreurs découverts par Robert G. Gallager [6] au début des années 60. Cependant, ils ne suscitèrent pas l'engouement au moment de leurs découvertes en raison de la complexité calculatoire des algorithmes de décodage qui était trop élevée par rapport aux capacités calculatoires de l'époque. En 1995, suite à la découverte des turbo-codes, David J. C. MacKay exhume les codes LDPC [7] inventés par Gallager. Depuis, de nombreux travaux se sont intéressés à cette famille de codes correcteurs d'erreurs capable d'atteindre des performances proches de la limite de Shannon.

Dans ce contexte, l'implantation matérielle de décodeurs LDPC pour les communications numériques spatiales optiques est un domaine émergent qui soulève de nombreux défis. Un premier objectif de cette thèse fût donc d'étudier différents schémas de codage pour les liens optiques montant et descendant entre la Terre et un satellite. Le second objectif fût d'aboutir à la conception de décodeurs LDPC matériels qui répondront aux différentes problématiques découlant du domaine applicatif.

Organisation du manuscrit

Le premier chapitre a pour thème les différentes notions associées aux codes correcteurs d'erreurs. Il introduit la composition d'une chaîne de communications numériques, la

notion de codage canal ainsi que la notion de codes en bloc. Dans un second temps, les codes convolutifs, les turbo-codes et les codes polaires sont successivement présentés. Les caractéristiques relatives aux codes LDPC et aux algorithmes de décodage associés sont alors détaillées. Ce premier chapitre se termine par une justification des besoins en codage canal dans le cadre des applications spatiales.

Une présentation des différentes solutions technologiques pour l'implantation matérielle temps réel de décodeurs LDPC est faite dans le second chapitre. Les algorithmes de décodage sous-optimaux permettant de réduire drastiquement la complexité calculatoire du processus de décodage y sont détaillés. De plus, les différentes familles d'architectures de décodeurs LDPC sont récapitulées en fonction des ordonnancements des calculs.

Dans le chapitre 3, une étude du lien optique descendant est présentée. Elle met en exergue, dans un premier temps, les contraintes applicatives. Ensuite, le choix de l'algorithme de décodage est argumenté à l'aide d'une étude comparative. Plusieurs familles de codes LDPC sont alors analysées afin d'obtenir des codes permettant d'atteindre des pouvoirs de correction satisfaisants. À partir des codes LDPC et de l'algorithme de décodage sélectionné, une architecture de décodeur LDPC est proposée et son implantation matérielle sur FPGA est mise en œuvre. Une étude de performance démontre que cette dernière satisfait les besoins applicatifs.

Le quatrième chapitre adresse la problématique du lien optique montant. Cette application induit de nouvelles contraintes liées au positionnement du décodeur dans un circuit FPGA embarqué dans le satellite. Ainsi le processus de décodage doit être réalisé à partir de bits (entrées dures). Pour répondre à cette problématique, une nouvelle formulation de l'algorithme Gallager B étendu (Gallager E) est proposée. Celle-ci est comparée aux algorithmes de décodage à entrées dures présents dans la littérature. Cette formulation permet la conception d'un décodeur Gallager E matériel qui, après évaluation, satisfait les contraintes de cette seconde application.

Contributions des travaux de thèse

Les différentes contributions de ces travaux de thèse se résument comme suit :

1. Une étude des différents schémas de codage dans le cadre d'un lien optique descendant est proposée. Cette étude a permis de trouver le schéma de codage présentant un compromis au niveau décodage entre complexité calculatoire et pouvoir de correction. Il est ainsi possible de corriger efficacement les erreurs de transmission spécifiques à cette application.

2. Une architecture flexible de décodeurs LDPC matériels a été conçue. Basée sur les algorithmes OMS et NMS avec un ordonnancement par couches horizontales, elle est adaptée au décodage efficace des codes QC-LDPC tout en atteignant de hauts débits (10 Gbit/s).

3. Une seconde étude des schémas de codage est proposée pour des transmissions sur un lien optique montant. Le processus décodage se fait, cette fois-ci, à partir d'entrées binaires dites entrées dures. Le décodage est basé sur une nouvelle formulation de l'algorithme Gallager E pour permettre un ordonnancement par couches horizontales. Une analyse des performances de cet algorithme de décodage est aussi présentée. Les résultats obtenus satisfont les besoins applicatifs.

4. Une seconde architecture de décodage à entrées dures, dérivée de l'algorithme préalablement validé, a été développée. Elle est basée sur le Gallager E et permet de décoder efficacement des codes QC-LDPC.

Ces différentes contributions ont été valorisées au travers de publications et présentations scientifiques au niveau national et au niveau international :

- Conférences internationales avec comité de lecture :
  - V. Pignoly, B. Le Gal, C. Jégo et B. Gadat, "High data rate and flexible hardware QC-LDPC decoder for satellite optical communications," in proceedings of the IEEE 10th International Symposium on Turbo Codes & Iterative Information Processing (ISTC), Hong Kong, Hong Kong, 2018.
  - V. Pignoly, B. Le Gal, L. Barthe, B. Gadat et C. Jégo, "High Speed LDPC Decoding for Optical Space Link," in proceedings of the 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Glasgow, Scotland, 2020.
  - V. Pignoly, B. Le Gal, L. Barthe, B. Gadat et C. Jégo, "Fair Comparison of Hardware and Software LDPC Decoder Implementations for SDR Space Links," in proceedings of the 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Glasgow, Scotland, 2020.
- Présentations nationales :
  - V. Pignoly, "Décodeurs LDPC à haut débit pour lien optique descendant" à la journée "Codes correcteurs d'erreurs pour les communications spatiales" du COMET (Comité d'experts du CNES), Toulouse, 2019.

    - V. Pignoly, "Implantation matérielle de décodeurs LDPC pour communications numériques optiques par satellite à haut débit" à la journée "Codage, modulation et traitement du signal pour les communications optiques" du GdR ISIS (Groupement de Recherche Information, Signal, Image et viSion), Paris, 2019.

- Revue internationale avec comité de lecture :

    - Adrien Cassagne, Olivier Hartmann, Mathieu Léonardon, Kun He, Camille Leroux, Romain Tajan, Olivier Aumage, Denis Barthou, Thibaud Tonnellier, Vincent Pignoly, Bertrand Le Gal et Christophe Jégo, "AFF3CT: A Fast Forward Error Correction Toolbox!", SoftwareX, Volume 10, 2019.

1 Introduction au codage canal

Ce premier chapitre a pour but de présenter les codes Low Density Parity Check (LDPC), une technique avancée de codage canal. Tout d'abord, le concept de code correcteur d'erreurs est introduit. Des techniques avancées de codage canal sont présentées dans la deuxième partie. La troisième partie introduit les notions relatives aux codes LDPC. Ces notions sont nécessaires à la compréhension du reste du manuscrit. Enfin, la dernière partie présente différents schémas de codage utilisés dans le domaine spatial.

1.1 Les codes correcteurs d'erreurs

1.1.1 Chaîne de communications numériques

Les systèmes de communications numériques transmettent de l'information numérique depuis une source vers un ou plusieurs destinataires. Ces systèmes de communications numériques peuvent se décomposer de manière simplifiée comme cela est illustré dans la figure 1.1.

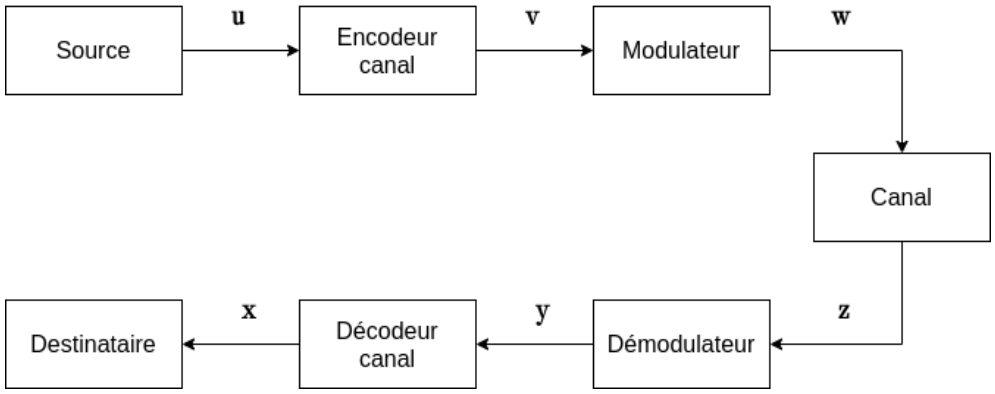

Figure 1.1 – Schéma simplifié d'une chaîne de communications numériques

Les principales étapes associées à une transmission de l'information numérique y sont représentées. Les données numériques en entrée de la chaîne **u** sont disponibles au format binaire. Elles sont émises par une source pour être transmises à un destinataire à travers un canal de transmission. Ce dernier est le support sur lequel transite l'information. Dans le cadre des communications optiques depuis/vers un satellite en orbite, le canal est composé d'un émetteur utilisant des lasers, d'un récepteur équipé d'un télescope et de l'atmosphère terrestre présente entre les deux. L'information à transmettre étant constituée de séquences de bits, le but du modulateur est de transformer le flux binaire en signaux continus pouvant être transmis via le canal. Au cours de la transmission à travers le canal, les signaux physiques subissent de nombreuses perturbations. Ces perturbations sont issues du bruit lié à la traversée de l'atmosphère et du bruit thermique des composants électroniques des équipements de transmission. Ces perturbations dégradent la qualité des signaux transmis. Le démodulateur a la fonction opposée à celle du modulateur : à partir des signaux continus reçus, il estime la valeur binaire des bits reçus. Ses estimations peuvent être erronées si les perturbations induites par le canal sont trop importantes. Cela pose problème car l'objectif est que les données binaires **x** reçues par le destinataire soient identiques aux données binaires **u** émises par la source. Le codage canal, basé sur l'utilisation de codes correcteurs d'erreurs, vise à faciliter la correction des erreurs de transmission. Le codeur et le décodeur canal ont donc pour objectif de fiabiliser la

transmission numérique.

1.1.2 Le codage canal

Pour fiabiliser la transmission de données numériques, le codage canal consiste à ajouter de la redondance à l'information binaire à émettre. À partir du mot d'information utile **u**, le codeur génère un mot de code **v** contenant cette redondance. La redondance et l'information utile sont liées par une loi donnée. Le décodeur canal se sert de la redondance et de la loi d'encodage à la réception pour détecter et corriger les erreurs de transmission.

L'ajout de redondance augmente le nombre de bits à transmettre via le canal et diminue ainsi l'efficacité du système. Cependant, cette diminution d'efficacité est contrebalancée par le gain important lié à la correction d'erreurs introduites lors de la transmission.

1.1.3 Caractéristiques des codes en blocs

Cette sous-partie présente les notions de base des codes en blocs. L'ensemble des solutions de codage canal étudiées dans ce manuscrit est réalisé à partir de codes en blocs. Le codage en blocs consiste à transformer un bloc d'information utile **u** composé de K bits en un bloc d'information codée **v** composé de N bits. Le codeur génère $M = N - K$ bits de redondance.

Le rendement du code noté R est le rapport entre le nombre de bits d'information utiles K entrant dans le codeur et la taille N du mot de code généré. Il est défini par :

$$R = \frac{K}{N} \tag{1.1}$$

Si la somme binaire de deux mots de codes est un mot de code, nous parlons alors de codes en blocs linéaires. Un code est dit systématique si les bits présents dans le vecteur **u** sont aussi présents dans le vecteur **v**.

Pour expliquer les notations suivantes, un code en bloc ayant pour matrice de parité **H** définie dans l'équation 1.2, et pour matrice génératrice **G** définie dans l'équation 1.3, est retenu.

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{1.2}$$

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \tag{1.3}$$

La matrice de parité $\mathbf{H}$ est de taille $N \times M$, par exemple 7×3. La matrice génératrice $\mathbf{G}$ possède quant à elle une taille $N \times K$, c'est à dire 7×4 dans notre exemple. Nous notons $\mathbf{H}^t$ la matrice transposée de $\mathbf{H}$ et $\mathbf{0}$ la matrice nulle.

La matrice génératrice permet de générer les mots de code à partir des bits d'informations de la source. Le codeur effectue le codage à l'aide d'une multiplication matricielle comme suit :

$$\mathbf{v} = \mathbf{u} \times \mathbf{G} \tag{1.4}$$

Les matrices $\mathbf{G}$ et $\mathbf{H}$ sont associées à partir de la formule suivante :

$$\mathbf{G} \times \mathbf{H}^t = \mathbf{0} \tag{1.5}$$

Nous pouvons déduire des formules 1.4 et 1.5 que :

$$\mathbf{u} \times \mathbf{G} \times \mathbf{H}^t = \mathbf{u} \times \mathbf{0} \tag{1.6}$$

et donc que :

$$\mathbf{v} \times \mathbf{H}^t = \mathbf{0} \tag{1.7}$$

soit :

$$\mathbf{H} \times \mathbf{v}^t = \mathbf{0} \tag{1.8}$$

Les mots de code sont donc l'ensemble des vecteurs qui vérifient l'équation 1.8. Si nous considérons un vecteur de bits $\mathbf{x}$, alors chaque ligne de l'équation $\mathbf{H} \times \mathbf{x}^t = \mathbf{0}$ est définie comme étant une équation de parité. Si toutes les équations de parité sont vérifiées, alors $\mathbf{x}$ est un mot de code.

Soit $\mathbf{u}$ un vecteur de bits d'information créé par la source tel que $\mathbf{u} = [0\ 1\ 0\ 1]$. Comme les calculs ont lieu dans le corps binaire $\mathbf{GF}(2)$, les additions binaires correspondent à des OU exclusifs (XOR). Le mot de code obtenu après multiplication par la matrice génératrice de l'équation 1.4 est $\mathbf{v} = [0\ 1\ 0\ 1\ 1\ 0\ 1]$. Ce mot de code est valide car il vérifie $\mathbf{H} \times \mathbf{v}^t = \mathbf{0}$. Maintenant, considérons le vecteur $\mathbf{y} = [0\ 1\ 0\ 1\ 1\ 0\ 0]$ qui correspond au vecteur $\mathbf{v}$ auquel la valeur du dernier bit est inversée. Cela correspond à une erreur introduite par le canal de transmission. Nous pouvons calculer le vecteur syndrome $\mathbf{s}$

défini de la manière suivante :

$$\mathbf{s}^t = \mathbf{H} \times \mathbf{y}^t \tag{1.9}$$

Le vecteur syndrome obtenu est égal à $\mathbf{s} = [0\ 0\ 1]$. Nous définissons le poids du syndrome comme le nombre de '1' dans le vecteur $\mathbf{s}$. Si le poids est nul, alors toutes les équations de parités sont vérifiées. Cela signifie que les bits testés correspondent à un mot de code. Ici, nous remarquons que le poids de $\mathbf{s}$ est non nul. Nous pouvons donc affirmer que $\mathbf{y}$ n'est pas un mot de code. Cela indique qu'il y a eu une erreur de transmission. Cependant, trouver un mot de code valide ne signifie pas que le message reçu est identique à celui qui a été généré lors du codage. En effet, plusieurs erreurs de transmission peuvent éventuellement faire aboutir le processus sur un autre mot de code. Il est donc important d'avoir des mots de code suffisamment éloignés les uns des autres pour éviter ce mauvais décodage.

Pour quantifier ces différences, nous parlons de distance de Hamming. Elle correspond au nombre de positions pour lesquelles deux vecteurs ont des valeurs binaires qui diffèrent. Nous notons d_{min} la distance minimale d'un code, c'est à dire la plus petite distance de Hamming entre deux mots de code. Si nous considérons un décodage par *maximum likelihood* ou maximum de vraisemblance en français, l'objectif est de trouver le mot de code ayant la distance de Hamming la plus faible par rapport au vecteur de bits estimés. Ainsi, plus d_{min} est grand, alors plus le décodage est précis. La distance minimale du code permet donc d'évaluer le pouvoir de correction pour ce type de décodage.

L'exemple précédent est uniquement basé sur la matrice $\mathbf{H}$ pour détecter une erreur. D'autres algorithmes permettant de détecter et de corriger les erreurs de transmission au cours d'un processus de décodage, seront mis en avant par la suite.

1.1.4 Évaluation du pouvoir de corrections d'un code

Le pouvoir de correction d'une technique de codage (encodage et décodage) peut être évalué à partir de courbes de performance obtenues lors de simulations de type Monte Carlo. Dans notre étude, les taux d'erreurs binaires BER (Bit Error Rate) et les taux d'erreurs trames FER (Frame Error Rate) servent de métriques afin d'évaluer les performances d'un schéma codage pour un niveau de bruit donné. Lors de simulations de Monte Carlo, différentes techniques permettent de modéliser les perturbations liées au canal. Pour un canal Additive White Gaussian Noise (AWGN), le rapport signal à bruit (SNR : Signal-Noise Ratio) permet de quantifier ce niveau de bruit. Le BER correspond au rapport entre le nombre de bits erronés et le nombre total de bits transmis et le FER correspond au rapport entre le nombre de trames erronées et le nombre de trames transmises. Dans

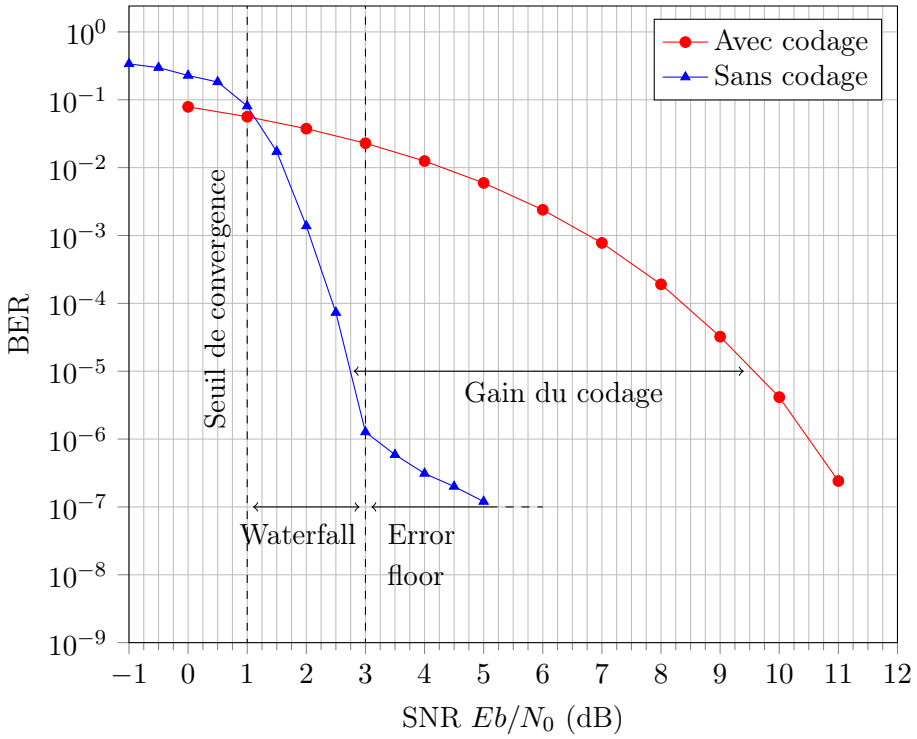

Figure 1.2 – Évaluation des performances

le cadre d'un passage par un canal AWGN, le SNR correspond au rapport entre l'énergie moyenne émise par bit d'information E_b et la densité spectrale de puissance du bruit N_0.

La figure 1.2 illustre une différence de performance de transmission entre des systèmes de communications numériques avec et sans codage canal. Le gain de codage pour un BER égal à 10^{-5} est dans cet exemple d'environ 6.5 dB.

1.2 Techniques de codage canal

Si nous considérons la courbe correspondant au système avec code correcteur d'erreurs, nous distinguons trois zones distinctes. La première zone se situe sur la courbe lorsque SNR est inférieur à 1 dB. Elle correspond au moment où le SNR est trop faible (trop de bruit) pour que le processus de décodage converge vers des mots de code ou vers le bon mot de code. Une fois que la valeur du SNR dépasse 1 dB (le seuil de convergence), le processus de décodage converge de plus en plus facilement vers les bons mots de code. Cette région est surnommée le *waterfall* en raison de sa forme proche de celle d'une chute d'eau. Enfin, après 3 dB, le BER diminue moins rapidement tandis que le SNR continue d'augmenter. C'est la zone du plancher d'erreurs, soit de l'*error floor* en anglais. Le changement de pente apparu ici lorsque le SNR est égal à 3 dB est lié aux propriétés du

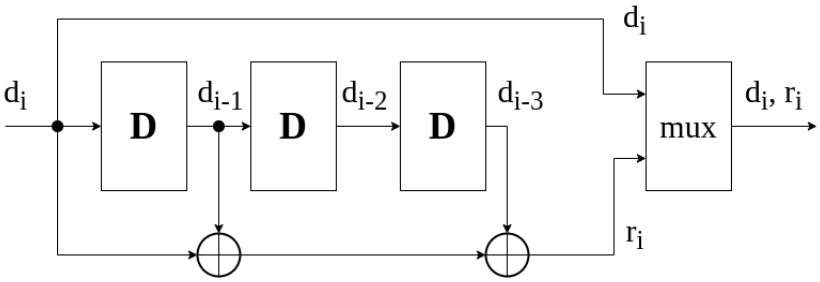

Figure 1.3 – Codeur convolutif de fonction de transfert [1, 1 + D + D^3]

code retenu.

Cette partie vise à introduire différentes familles de codes correcteurs d'erreurs usuelles. Actuellement, les turbocodes [8], les codes LDPC [6] et les codes polaires [9] sont les techniques avancées et populaires de codage canal.

Le but est de présenter une diversité de types de codes correcteurs d'erreurs mais surtout d'appréhender les notions de codage canal avant de se focaliser sur les codes LDPC, qui sont considérés dans le reste du manuscrit.

1.2.1 Introduction aux codes convolutifs

Les codes convolutifs ont été introduits par Peter Elias en 1955 [10]. Ils constituent une famille de codes qui a été adoptée par de nombreux systèmes de transmission. Pour un codeur convolutif, le symbole de sortie dépend non seulement du symbole d'entrée, mais aussi des symboles précédents. L'exemple de codeur introduit dans [10] est présenté dans la figure 1.3. Ce codeur convolutif, basé sur un registre à décalage, correspond à un code convolutif systématique de rendement 1/2. En effet, les données issues du vecteur d'entrée sont présentes dans le vecteur de sortie. Les codes convolutifs sont définis par un polynôme. Dans cet exemple, le bit de redondance r_i est obtenu à partir de d_i, d_{i-1} et d_{i-3}. Cela correspond au polynôme 1 + D + D^3. Nous appelons la fonction de transfert d'un code convolutif, l'ensemble des polynômes permettant de générer les sorties du codeur. Un multiplexeur est placé en sortie du codeur afin de sérialiser les bits d_i et r_i. Dans cet exemple, le débit de sortie du codeur est donc le double de celui d'entrée. Un codeur permet de générer des séquences continues de bits. Cependant les systèmes de communication opèrent sur des blocs de taille finie. Il faut donc mettre en place des techniques de fermeture de treillis. Une solution consiste à ajouter ν bits à la fin d'un bloc à transmettre tels que le codeur retrouve son état initial pour lequel les ν registres sont tous à zéro [11].

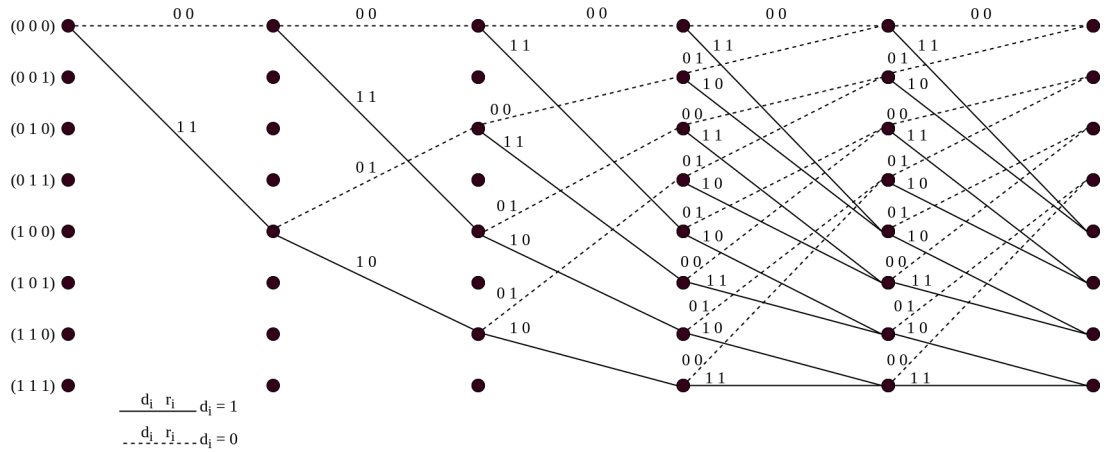

Figure 1.4 – Treillis du code convolutif de fonction de transfert [1, 1 + D + D^3]

Il existe des codes convolutifs récursifs [11] où la sortie de certains registres est réinjectée dans des registres précédents et des codes convolutifs non binaires [12] qui traitent plusieurs bits à la fois.

Les codes convolutifs peuvent être représentés sous la forme d'un diagramme en treillis. C'est la représentation la plus courante pour un code convolutif. Le code précédant est présenté sous cette forme dans la figure 1.4. Cela permet de visualiser les valeurs que peuvent prendre d et r à la réception d'un nouveau bit en fonction des bits précédemment reçus. La dernière partie du treillis (à droite) est une section complète. Elle est suffisante pour caractériser le code convolutif.

La représentation en treillis permet de définir le code convolutif mais est aussi utile pour le processus de décodage. En effet, l'algorithme de Viterbi [13] permet par exemple de trouver la séquence d'états la plus probable dans le treillis à partir des échantillons reçus. L'algorithme de Viterbi fournit des bits en sortie du processus de décodage. D'autre part, l'algorithme Maximum A Posteriori (MAP) permet de calculer les valeurs optimales des probabilités *a posteriori* des bits transmis à partir des valeurs souples reçues. Proposé par Bahl, Cocke, Jelinek and Raviv dans [14], il est aussi appelé BCJR en hommage à ses concepteurs. Les probabilités, fournies en sortie du décodeur et représentées par des valeurs souples, permettent de mettre en place des systèmes de décodage itératif, comme pour le décodage de turbocodes [8].

1.2.2 Turbocodes

Imaginés par Claude Berrou au début des années 90 [8], les turbocodes représentent une technique avancée de codage canal qui a permis d'accéder pour la première fois à des

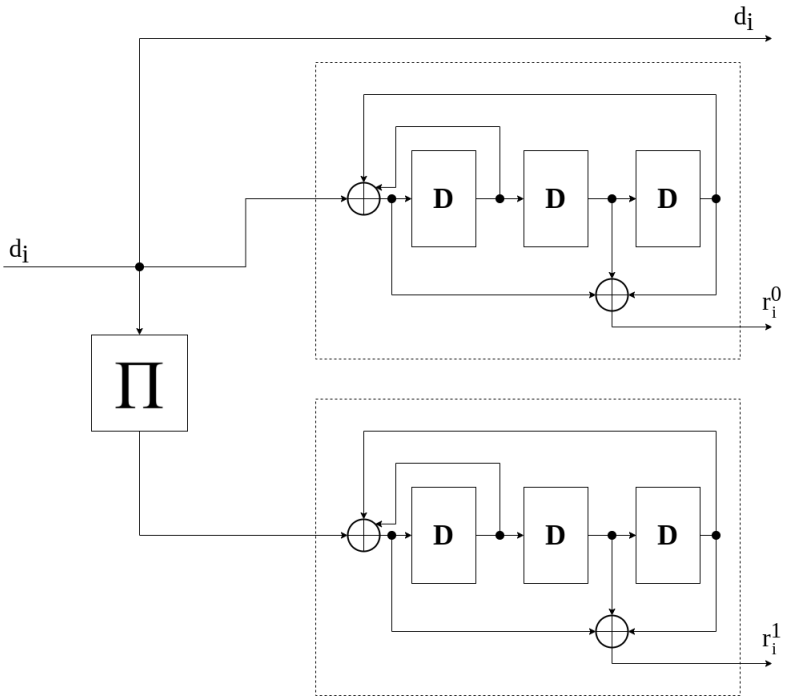

Figure 1.5 – Turbocodeur pour le polynôme $\frac{1+D^2+D^3}{1+D+D^3}$

pouvoirs de correction très proches de la limite théorique du canal défini par Shannon [15]. En effet, dans certaines conditions, les performances de décodage sont à moins de 1 dB de cette limite. Ce niveau de performances élevé représentait une réelle évolution lors de l'introduction des turbocodes.

Les codes convolutifs ont un pouvoir de correction qui augmente avec le nombre d'états. Cela est mis en évidence dans [11] qui soulève le fait qu'il faudrait des codes convolutifs avec plusieurs dizaines d'états pour pouvoir satisfaire les applications les plus courantes. Cependant, décoder de tels codes convolutifs avec de très grands nombres d'états est trop complexe pour être mis en œuvre. Toujours dans [11], un turbocode est présenté comme un stratagème permettant d'imiter un code convolutif avec un nombre d'états élevé. L'association de petits codes convolutifs avec des échanges judicieux d'informations lors du décodage permet d'obtenir un pouvoir de correction s'approchant de la limite théorique du canal.

Le codage des turbocodes implique la concaténation parallèle d'au moins deux codes convolutifs récursifs et systématiques (RCS pour Recursive Systematic Convolutional) liés par un entrelaceur. La figure 1.5 présente un turbocodeur basé sur deux codes RCS de polynôme $\frac{1+D^2+D^3}{1+D+D^3}$ pour la partie redondante. Le rendement du codeur est de 1/3 car pour 1 bit d'information utile, 2 bits de redondance sont générés. Pour augmenter le

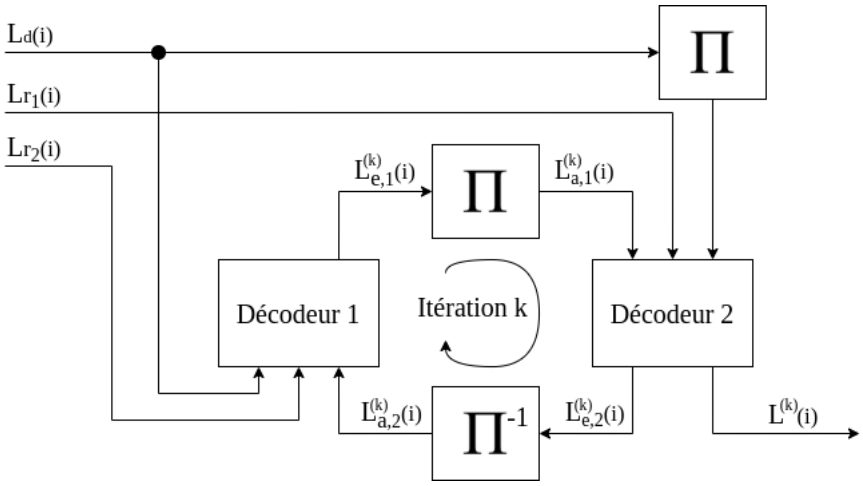

Figure 1.6 – Schéma de principe associé au processus de turbodécodage

rendement de ce type de codes, il est possible de poinçonner des bits qui, par définition, ne sont pas transmis. Souvent, les bits de redondance r_i^0 et r_i^1 sont poinçonnés [16].

L'entrelaceur entre les deux codeurs convolutifs permet d'augmenter la distance minimum de Hamming du code résultant en permutant l'ordre des données entrant dans le second codeur RCS.

Le décodage d'un turbocode ou turbodécodage s'appuie sur les traitements présentés dans la figure 1.6. Il comprend deux sous-décodeurs BCJR à sorties souples. Chacun est associé à un des deux codeurs convolutifs. Ici le décodeur 1 décode l'information codée par le codeur dans le domaine naturel et le décodeur 2, celle codé par le codeur dans le domaine entrelacé (figure 1.5). Les données d'entrées correspondent aux LLR (Log-Likelyhood Ratio) calculés à partir de l'information reçue par le récepteur. Le calcul des LLR *a priori* se fait à partir de la formule suivante :

$$LLR(y_i|v_i) = ln\left(\frac{p(y_i|v_i=0)}{p(y_i|v_i=1)}\right) \tag{1.10}$$

Les LLR mis à jour en sortie d'un décodeur élémentaire sont égaux à la somme du LLR initial, aussi appelé information intrinsèque, avec l'information extrinsèque. Les informations extrinsèques L_e sont échangées entre les décodeurs élémentaires. Elles transitent au travers d'un entrelaceur ou d'un desentrelaceur pour retrouver l'ordre qui correspond à chaque décodeur élémentaire. À chaque passage dans un décodeur, l'information extrinsèque gagne en fiabilité.

Les turbocodes souffrent en général de planchers d'erreurs assez prononcés en comparaison

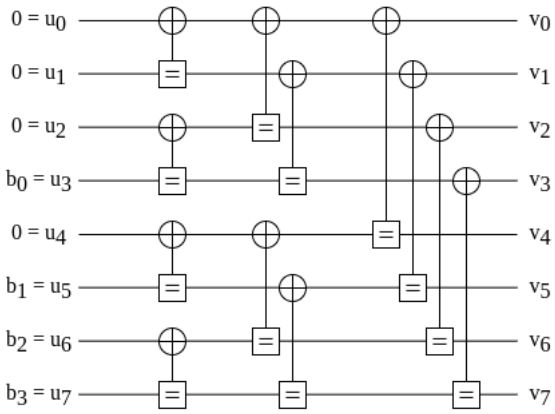

Figure 1.7 – Représentation du codage d'un code polaire de taille N = 8 sous la forme d'un graphe de factorisation

avec d'autres familles de codes correcteurs d'erreurs. Cependant, des travaux récents présentés dans [17] propose des turbocodes avec des planchers d'erreurs apparaissant à des BER inférieurs de deux décades par rapport aux turbocodes usuels.

La méthode intuitive qui a amené à la création des turbocodes est différente de l'approche mathématique qui a abouti à la création d'un autre schéma de codage en 2008 : les codes polaires.

1.2.3 Codes polaires

Les codes polaires représentent une classe de codes correcteurs en blocs linéaires. Inventé par Erdal Arikan [9], ce schéma de codage récent est populaire et ses performances lui ont valu d'être adopté pour les canaux de contrôle de la cinquième génération du standard de communications mobiles 5G [18]. Dans ce standard, ils sont utilisés pour fiabiliser les canaux de contrôle.

La matrice génératrice d'un code polaire est le produit de deux matrices : $\mathbf{G} = \mathbf{E}\mathbf{F}^{\otimes n}$ avec $\mathbf{F}^{\otimes n}$ est la n-ième puissance de Kronecker de $\mathbf{F}$. $\mathbf{F}^{\otimes n}$ est définie récursivement par $\mathbf{F}^{\otimes n+1} = \begin{bmatrix} \mathbf{F}^{\otimes n} & \mathbf{0} \\ \mathbf{F}^{\otimes n} & \mathbf{F}^{\otimes n} \end{bmatrix}$ où $\mathbf{0}$ est la matrice nulle et $\mathbf{F} = \mathbf{F}^{\otimes 1} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ le noyau de Kronecker.

La matrice $\mathbf{E}$ permet d'introduire des bits dit "gelés" (*frozen bits* en anglais), support de la redondance. Le codage peut être représenté sous la forme d'un graphe de factorisation qui permet de visualiser simplement les procédés de codage et de décodage. La figure 1.7 montre le graphe de factorisation pour une taille N = 8.

Les codes polaires se basent sur le phénomène de polarisation. Si nous considérons que chacun des N bits est transmis sur un canal différent, nous pouvons diviser ces canaux en un groupe de canaux fiables et un groupe de canaux à fortes probabilités d'erreurs. Nous parlons de polarisation. Il est possible de démontrer que lorsque N tend vers l'infini, la capacité des canaux fiables tend vers la capacité limite du canal. Le codage polaire consiste donc à associer des bits gelés aux canaux peu fiables. Le décodage profite du fait que les valeurs des bits gelés sont connues (égales à zéro).

Les codes polaires ont initialement été développés pour être décodés par l'algorithme Successive Cancellation (SC) [9]. Les performances de cet algorithme de décodage sont limitées pour des codes de petites tailles. En conséquence, d'autres algorithmes comme le SC List (SCL) [19], le SC Flip (SCF) [20] ou le SC Stack (SCS) [21] ont été proposés afin de palier à ce problème de performance de correction en échange d'une complexité calculatoire accrue.

Malgré leur découverte récente, les bonnes performances des codes polaires font qu'ils ont été sélectionnés pour protéger les canaux de contrôle de la 5G. Cependant, ils ne sont pas encore très utilisés pour des applications à très haut débit avec de grandes tailles de trames (plusieurs Gbit/s). Il existe tout de même des implantations matérielles de l'algorithme SCL avec des débits estimés à 1 Gbit/s sur technologie ASIC [22].

1.3 Codes Low Density Parity Check

Inventés par Gallager en 1962 [6], les codes Low Density Parity Check (LDPC) sont une famille populaire de codes correcteurs en blocs linéaires. Cependant les capacités d'intégration de l'époque ne permettait pas d'effectuer le processus de décodage de ces codes du fait de sa grande complexité calculatoire. Il faut attendre 1995 pour que MacKay les réintroduise sur le devant de la scène [7] en démontrant que le décodage des codes LDPC peut se faire par propagation de croyances. Les codes LDPC ont depuis été adoptés dans plusieurs standards de communications numériques. Nous les retrouvons notamment au sein du standard DVB-S2 [23] pour la diffusion vidéo par satellite, mais aussi dans les standards IEEE 802.16e Wimax [24] ou IEEE 802.11n Wifi [25]. Plus récemment, ils ont été sélectionnés par l'organisme 3GPP (3rd Generation Partnership Project) pour la 5G [18]. Un codage LDPC est mis en œuvre pour les canaux de données de la partie *enhanced Mobile BroadBand* (eMBB) de la 5G, ce qui correspond aux scenarii où il y a de fortes contraintes en termes de débit. Les codes LDPC sont donc très présents dans l'industrie. Par ailleurs, les nouvelles problématiques liées aux applications modernes font qu'ils sont aussi très étudiés par la communauté scientifique.

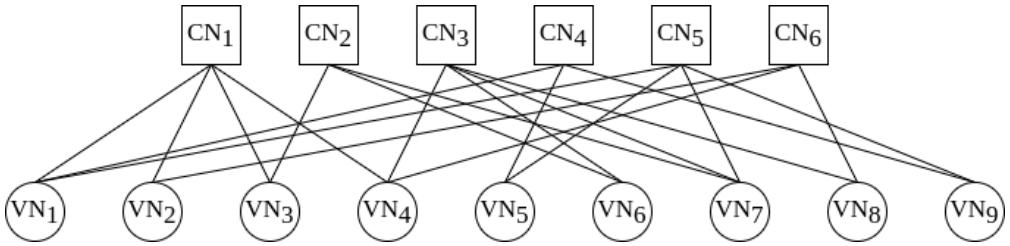

Figure 1.8 – Graphe de Tanner associé à un code LDPC de taille 9 × 6

1.3.1 Définition et notations

Un code LDPC est défini par une matrice de parité **H** de taille N × M. Cette matrice binaire possède une faible densité. En effet, comme son nom l'indique, le nombre de '1' dans la matrice **H** est faible par rapport au nombre de '0'. Cette faible proportion de '1' dans la matrice de parité est essentielle car la complexité du processus de décodage augmente linéairement avec le nombre de '1'. Un code LDPC peut être représenté sous la forme d'un graphe bipartite, nommé graphe de Tanner [26]. Il permet de visualiser les connexions entre les N nœuds de variable ou Variable Nodes (VN) et les M nœuds de parité ou Check Nodes (CN). La figure 1.8 représente le graphe de Tanner d'un code LDPC de taille 9 × 6.

Dans un graphe de Tanner, chaque branche du graphe liant un élément VN et un élément CN correspond à un '1' dans la matrice **H**. S'il y a un '1' en position (i, j) dans **H**, alors le VN_i est connecté au CN_j dans le graphe de Tanner. Nous parlons d'un cycle dans un graphe de Tanner quand il existe un chemin permettant de partir d'un nœud VN_i et de revenir à ce même nœud VN_i sans repasser par les mêmes branches. Nous notons M(i) l'ensemble des indices des CN connectés au nœud de variable VN_i et N(j) l'ensemble des indices des VN connectés au nœud de parité CN_j. Nous appelons le degré d'un nœud le nombre de branches connectées à ce nœud. Nous notons dv_i le degré du nœud de variable VN_i et dc_j le degré du nœud de parité CN_j. Nous définissons un code LDPC comme régulier si le degré de nœuds de variable et le degré des nœuds de parité sont constants :

$$\exists c_1 \in \mathbb{N}^*, \forall i \in [\![0, N-1]\!], dv_i = c_1 \tag{1.11}$$

$$\exists c_2 \in \mathbb{N}^*, \forall j \in [\![0, M-1]\!], dc_j = c_2 \tag{1.12}$$

Dans la suite du document, les techniques de codage canal retenues seront basées sur des codes LDPC pour deux raisons principales.

1. La première est qu'ils permettent d'atteindre des performances en termes de cor-

rection d'erreurs proches de la capacité limite du canal [27, 28]. En effet, il a été montré dans [29] qu'il est possible d'obtenir des codes LDPC dont le gain de codage est à seulement 0.0045 dB de la capacité théorique du canal.

2. La seconde raison est qu'il existe déjà beaucoup de travaux antérieurs. Nous avons donc un certain recul sur l'utilisation et l'implantation matérielle de codeurs et de décodeurs LDPC. Les innovations proposées dans ce manuscrit portent sur la définition de schémas de codage efficaces pour les systèmes de communications optiques par satellites. Ce type de communications constitue un nouveau domaine applicatif pour les codes correcteurs d'erreurs.

1.3.2 Codes LDPC quasi-cycliques

L'implantation matérielle en temps réel et efficace de décodeurs LDPC est une tâche ardue. La complexité d'implantation est principalement liée à la structure de la matrice $\mathbf{H}$. Afin de permettre le développement de systèmes de communications numériques basés sur les codes LDPC, des structures de codes avec des propriétés facilitant l'implantation de décodeurs matériels ont été proposées. Les codes LDPC quasi-cycliques (QC-LDPC), par exemple, sont définis par des matrices $\mathbf{H}$ structurées. Cette dernière est composée de sous-matrices de taille $Z \times Z$. Dans le cadre des matrices quasi-cycliques, Z est appelé le facteur d'expansion. Les sous-matrices de taille $Z \times Z$ appelées circulantes sont soit obtenues par décalage de la matrice identité de taille $Z \times Z$, soit égales des matrices nulles.

La figure 1.9 montre un exemple de code QC-LDPC de taille 448×160 et de facteur d'expansion $Z = 32$. Dans cette description compressée de la matrice $\mathbf{H}$, les sous-matrices sont représentées par des nombres compris entre -1 et 31 qui ont la signification suivante :

- -1 : la sous-matrice est nulle
- 0 : la sous-matrice est une matrice identité
- $\xi = 1$ à 31 : la sous-matrice est une matrice identité décalée ξ fois vers la droite.

Les codes QC-LDPC présentent un pouvoir de correction proche des codes LDPC non structurés [30]. Ils sont plébiscités car leur structure facilite l'implantation de décodeurs matériels. En effet, ils possèdent un degré de parallélisme constant et autorisent l'utilisation d'entrelaceurs moins complexes [31, 32, 33, 34]. C'est pour ces raisons que les études présentées dans les chapitres suivants se basent sur des codes QC-LDPC.

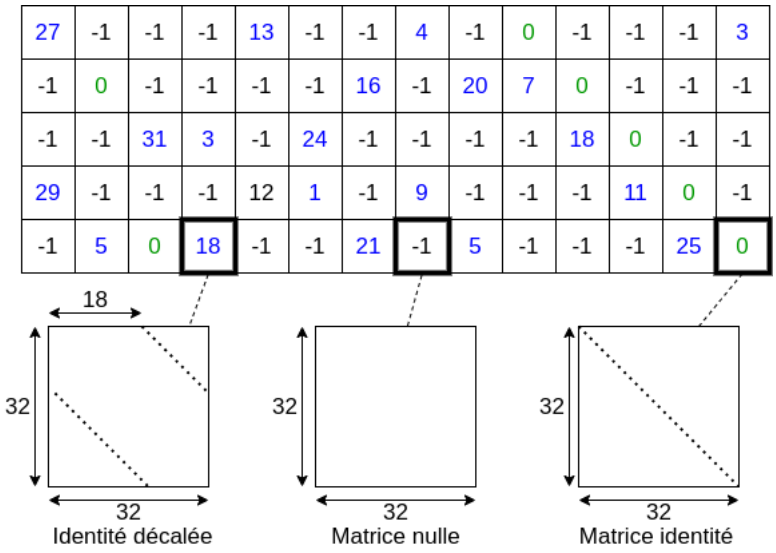

Figure 1.9 – Représentation d'un code QC-LDPC de taille 448 × 160 pour un facteur d'expansion $Z = 32$

Il existe plusieurs méthodes pour construire des codes QC-LDPC. Certaines d'entre elles se basent sur l'utilisation de protographes. Un protographe est un graphe bipartite comme présenté sur la figure 1.10.1. La construction d'un code LDPC à partir d'un protographe se fait en deux étapes. Le protographe est répété autant de fois que nécessaire pour obtenir la taille de code souhaitée comme présenté sur la figure 1.10.2. Dans un second temps, les branches sont permutées en suivant des règles précises comme dans la figure 1.10.3. Pour construire un code QC-LDPC à base de protographes, les permutations sont définies par des matrices circulantes. En effet, chaque nouvelle branche est associée à une matrice circulante et les valeurs de décalage peuvent être modifiées afin d'obtenir un code QC-LDPC avec les propriétés souhaitées.

1.3.3 Algorithmes de décodage de codes LDPC

Il existe plusieurs algorithmes permettant de décoder les codes LDPC. Ces algorithmes présentent une grande diversité au niveau de la complexité calculatoire et des performances en termes de pouvoir de correction d'erreurs. Dans le cadre du décodage canal, il est déjà possible de différencier deux principaux types de décodage :

- Le décodage à **entrées dures** : le processus de décodage reçoit de l'information dure en entrée. Cela signifie qu'il doit se baser sur les bits estimés par le démodulateur pour corriger les éventuelles erreurs de transmission. Gallager propose dans [6] trois algorithmes pour le décodage de codes LDPC à partir d'entrées dures pour un canal

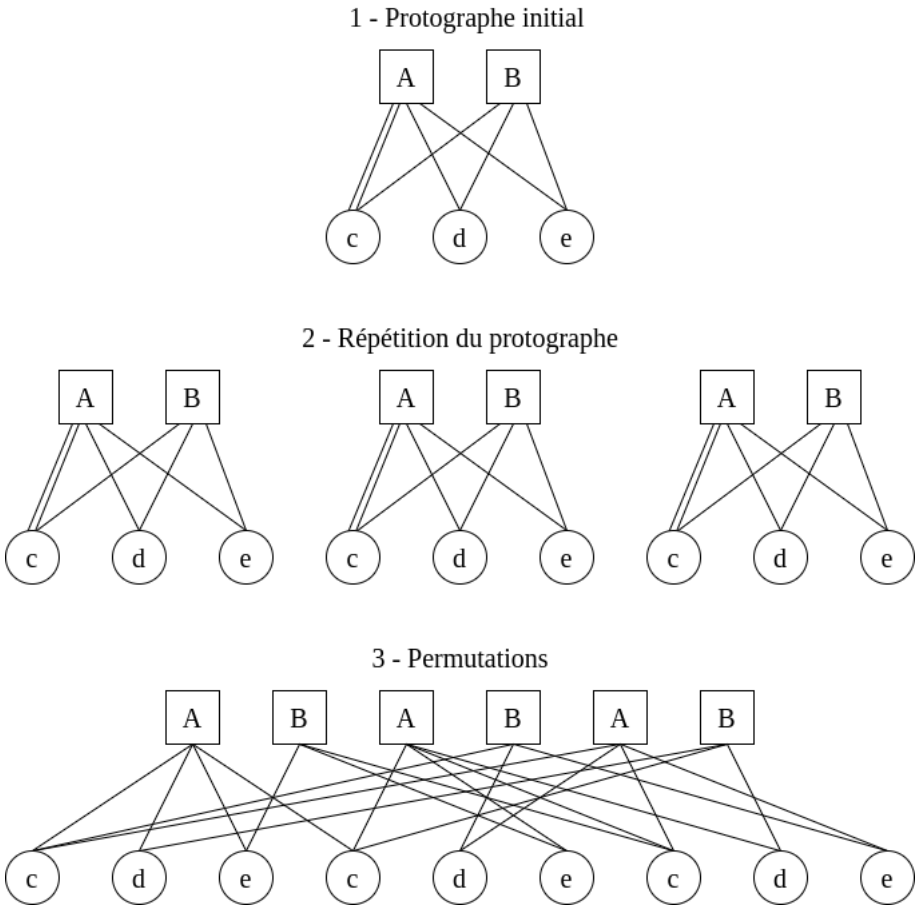

Figure 1.10 – Exemple de construction de code LDPC à partir d'un protographe

BSC (Binary Symmetric Channel) : Gallager Algorithm A, Gallager Algorithm B, et Bit-Flipping Algorithm. Ce type d'algorithmes de décodage a été utilisé par exemple dans le cadre de communications numériques par fibre optique [35]. Le récepteur ne permet d'évaluer l'information du canal que sur un bit pour des raisons technologiques. C'est le cas pour le lien optique montant retenu comme cadre applicatif dans le chapitre 4.

- Le décodage à **entrées souples** : le processus de décodage se fait avec des données d'entrée quantifiées sur plusieurs bits. En plus de l'information indiquant le bit décidé après démodulation, une information de fiabilité est fournie. Cela permet d'améliorer le pouvoir de correction en contrepartie d'une complexité calculatoire supérieure. De nombreux algorithmes basés sur la propagation de croyances ont été mis au point et leur diversité permet de subvenir aux besoins d'un grand nombre d'applications comme c'est le cas dans l'application de lien descendant optique retenu comme cadre applicatif dans le chapitre 3.

1.3.3.1 Algorithme de décodage par propagation de croyances

L'algorithme Belief Propagation (BP), aussi appelé Sum-Product Algorithm (SPA), ou algorithme par propagation de croyances permet d'effectuer le décodage de codes LDPC. L'objectif de cet algorithme est de propager la confiance dans les bits reçus dans le graphe en échangeant des messages entre les VN et CN tout en appliquant les règles des équations de parité. Nous parlons alors d'algorithme de *message passing*. Dans le cadre d'un décodage à entrées souples, c'est la probabilité *a priori* des bits reçus qui est envoyée au décodeur. La formule permettant de calculer les logarithmes du rapport de vraisemblance (LLR pour Log-Likelihood Ratio) *a priori* est la suivante :

$$LLR(y_i|v_i) = ln\left(\frac{p(y_i|v_i=0)}{p(y_i|v_i=1)}\right) \tag{1.13}$$

avec y_i le i-ème échantillon reçu et v_i le i-ème bit du mot de code envoyé. Dans le cadre d'un canal Binary Input AWGN (BI-AWGN) de variance σ^2, les probabilités sont calculées de la manière suivante :

$$p(y_i|v_i) = \frac{1}{\sqrt{2\pi\sigma^2}}\left[\frac{(y_i-(2v_i-1))^2}{2\sigma^2}\right] \tag{1.14}$$

A partir des équations 1.13 et 1.14, nous pouvons déduire que :

$$LLR(y_i|v_i) = \frac{-2y_i}{\sigma^2} \tag{1.15}$$

L'algorithme de décodage BP-SPA est détaillé dans l'algorithme 1. Cet algorithme tient son nom du calcul des messages sortant des CN dans le domaine des probabilités. Dans un premier temps, les informations intrinsèques Y_i sont initialisées avec les valeurs des LLR tout comme les messages $v2c_{ij}$ à Y_i. Ensuite le processus de décodage itératif est appliqué. Si un ordonnancement par inondation [36] des calculs est exécuté comme dans l'algorithme 1, alors l'ensemble des messages des CN vers les VN (c2v) est calculé dans un premier temps. Puis, dans un second temps, les messages des VN vers les CN (v2c) sont évalués. Un critère d'arrêt peut être mis en place pour stopper le processus de décodage si un mot de code est trouvé. Le calcul des bits X_i correspond au signe de la somme de l'information intrinsèque Y_i avec les informations extrinsèques $c2v_{ji}$. Enfin, si le nombre maximum d'itérations est atteint ou si un mot de code a été trouvé, alors le processus de décodage s'arrête et les bits de sortie correspondant aux X_i sont décidés.

Algorithm 1: Sum-Product Algorithm

▷ Input : $\boldsymbol{y} = (y_1, y_1, ..., y_N)$
init
$t = 0$, $Y_i = -\frac{2y_i}{\sigma^2}$ for $i \in [1, .., N]$, $v2c_{ij} = Y_i$ for $i \in [1, .., N]$ and $j \in [1, .., M]$
repeat
for all j $\in [1, .., M]$ **do**
for all i $\in$ N(j) **do**
▷ Calcul des messages $c2v_{ji}$

$$c2v_{ji} = 2tanh^{-1}\left(\prod_{k \in N(j)-\{i\}} tanh\left(\tfrac{1}{2} v2c_{kj}\right)\right)$$

end for
end for
for all i $\in [1, .., N]$ **do**
for all j $\in$ M(i) **do**
▷ Calcul des messages $v2c_{ij}$

$$v2c_{ij} = Y_i + \sum_{k \in M(i)-\{j\}} c2v_{ik}$$

end for
end for
for all i $\in [1, .., N]$ **do**
▷ Somme des informations intrinsèques et extrinsèques pour le critère d'arrêt

$$X_i = \begin{cases} 1, & \left(Y_i + \sum_{k \in M(i)} c2v_{ik}\right) < 0 \\ 0, & \text{otherwise} \end{cases}$$

end for
$t = t + 1$
until $t \leq t_{max}$ or $X\mathbf{H}^t = \mathbf{0}$
▷ Output : $\boldsymbol{x} = (x_1, x_2, ..., x_N) \in \{0, 1\}^N$ with $x_i = X_i$

Le calcul des messages $c2v$ peut se faire en représentation signe/amplitude avec la formule suivante :

$$c2v_{ji} = \left(\prod_{k \in N(j)-\{i\}} sign\left(v2c_{kj}\right)\right) \times f\left(\sum_{k \in N(j)-\{i\}} f\left(|v2c_{kj}|\right)\right) \tag{1.16}$$

La fonction f est définie par :

$$f(x) = ln\frac{e^x + 1}{e^x - 1} \tag{1.17}$$

Le pouvoir de correction de l'algorithme BP-SPA est illustré sur la figure 1.11. La courbe

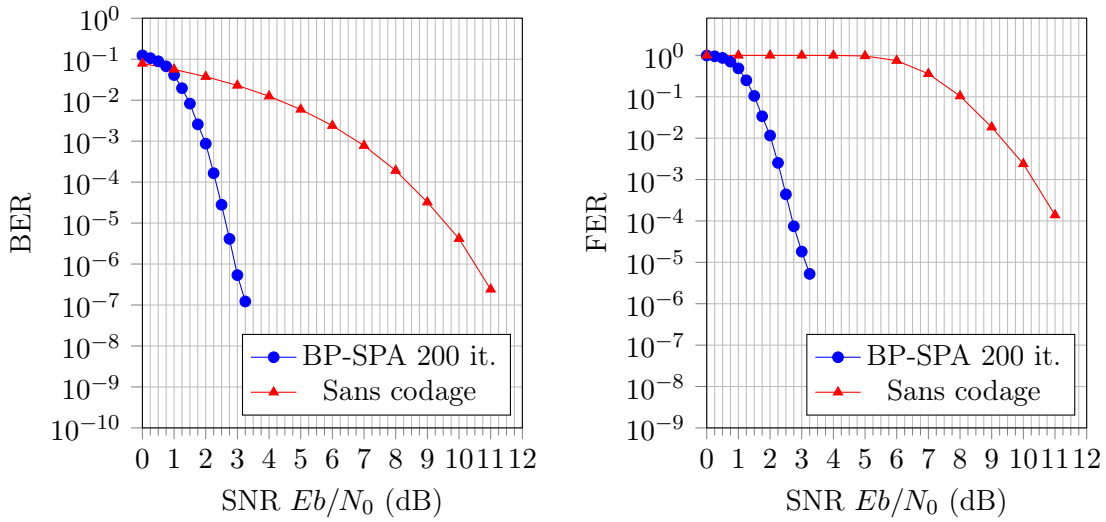

Figure 1.11 – Performances en correction d'erreurs de l'algorithme SPA avec un code LDPC (576, 288) du standard Wimax

correspond aux performances de décodage obtenues lorsque 200 itérations sont exécutées sur un canal AWGN avec une modulation BPSK. Le code LDPC utilisé est de taille (576, 288). Il est issu du standard IEEE 802.16e Wimax [24]. La simulation a été effectuée à l'aide du logiciel open-source AFF3CT, A Fast Forward Error Correction Toolbox [37], proposant un environnement de simulation de codes correcteurs d'erreurs.

Nous observons un gain de codage d'environ 7.5 dB pour un BER de 10^{-6}. L'intérêt du codage est plus intuitif en se plaçant à SNR fixe. Par exemple, pour un SNR de 3 dB, environ 1 bit reçu sur 10 est erroné sans codage canal alors qu'avec l'utilisation du code LDPC, seulement 1 bit sur plus d'un million est erroné.

L'algorithme BP-SPA est optimal quand le graphe du code en blocs ne contient pas de cycles. Cependant, les codes LDPC possèdent, par construction, des cycles. Leur décodage par les algorithmes de propagation de croyance peut présenter des cas où la valeur d'un message finit par s'autoconfirmer en raison de ces cycles. Les cycles diminuent le pouvoir de correction. Cela provoque l'apparition d'*error floor* même pour des codes ayant une grande distance minimum de Hamming. Dans ce cas, nous parlons de *trapping sets*.

Un *trapping set* [38] noté T(ω, ν) est l'ensemble des ω VN connectés qui forment un sous-graphe avec ν CN de degré impair tel que, si tous les bits associés à ces VN sont erronés, alors il y a ν équations de parité non vérifiées. Quand les valeurs de ω et ν sont faibles ($\omega < \sqrt{N}$ et $\nu < 4\omega$), les *trapping sets* génèrent des configurations pour lesquelles les algorithmes basés sur la propagation de croyance sont inefficaces.

La présence de *stopping sets* [39] est un autre critère qui influe les performances d'un code LDPC. Un *stopping set* est un groupe de VN tel que si un CN est connecté à un VN de ce groupe, alors il est connecté à au moins un autre VN du groupe. Ces configurations sont particulièrement intéressantes pour des canaux à effacement comme le BEC (Binary Erasure Channel). En effet, il est possible d'estimer théoriquement les courbes de BER et FER sur ce canal uniquement à partir des différents *stopping sets* présents dans la structure du code LDPC [40].

Les caractéristiques des codes LDPC font qu'ils représentent une classe variée de codes correcteurs d'erreurs permettant de satisfaire les besoins de différentes applications. D'autres algorithmes de décodage basés sur l'algorithme BP-SPA permettent d'atteindre des performances proches tout en diminuant drastiquement la complexité calculatoire lors du décodage. Ils facilitent l'implantation matérielle des décodeurs. Ils sont détaillés dans le chapitre 2.

1.3.4 Choix du schéma de codage

Plusieurs familles de codes correcteurs d'erreurs ont été présentées dans ce chapitre. Chacune a ses spécificités et il est impossible de généraliser et de dire que telle famille de code est toujours meilleure qu'une autre. C'est en fonction de l'application qu'il faut sélectionner un ou plusieurs schémas de codage en fonction des différentes contraintes. En effet, selon les besoins en termes de débits, de complexités matérielles ou de pouvoir de correction, différents schémas de codage présentant des compromis satisfaisant ces besoins peuvent être retenus.

Les codes LDPC ont su tout de même se démarquer des turbocodes et des codes polaires pour des applications nécessitant à la fois de très hauts débits (10 Gbit/s jusqu'à plusieurs Tbit/s), une faible complexité de décodage et peu d'erreurs (disques SSD et transmission par fibre optique [41, 42]).

1.4 Le codage canal pour les transmissions spatiales

1.4.1 Bref historique

Les codes correcteurs d'erreurs ont toujours eu une place prépondérante dans les communications spatiales. À partir 1958, le projet Pioneer d'exploration spatiale de la NASA permet l'envoi de sondes vers la lune avec des transmissions de données fiabilisées par un code convolutif.

Plus tard, entre 1969 et 1973, le programme Mariner de la NASA est mis en œuvre afin d'explorer les planètes du système solaire intérieures à la ceinture d'astéroïdes. Pour l'occasion, un code linéaire de type Reed-Salomon (RS) a permis d'assurer l'envoi de photos en provenance de Venus, Mars et enfin Mercure vers la Terre.

Ensuite, à partir de 1977, les sondes Voyager 1 et 2 de la NASA partent explorer les planètes extérieures à la ceinture d'astéroïdes. La concaténation d'un code convolutif avec un code RS est un élément qui a permis le succès de cette mission.

En France, le CNES anime différents projets comme SPOT comprenant des satellites d'observation de la Terre lancés depuis 1986. Certains de ces satellites fiabilisent le lien descendant par une concaténation d'un code convolutif et d'un code RS.

Depuis le début des années 90 avec l'apparition des turbocodes et la redécouverte des codes LDPC, des standards ont été proposés pour fiabiliser les communications spatiales. Nous présentons à présent ces standards.

1.4.2 Les codes correcteurs d'erreurs dans les standards de communications spatiales

Plusieurs organismes dans le monde sont à l'origine de normes ciblant spécifiquement le domaine des communications spatiales. En Europe, les télécommunications sont régies par les normes produites par l'ETSI (European Telecommunications Standards Institute) créé en 1988. C'est l'organisme à l'origine des standards DVB (Digital Video Broadcasting) dont plusieurs parties se concentrent sur les communications par satellites. Spécialisé dans les équipements spatiaux, le CCSDS (Consultative Committee for Space Data Systems) a été créé à l'échelle mondiale par les principales agences spatiales en 1982. Il est à l'origine de normes mises en place pour mener à bien le développement et la maintenance des systèmes de données dans l'espace.

Dans un premier temps, les turbocodes ont été sélectionnés pour le standard DVB-RCS (Return Channel to Satellite) [43] dès 1999. C'est une norme mise en place en Europe pour l'accès Internet par satellite. En outre, les turbocodes ont aussi été retenus par le CCSDS dans ses recommandations pour les communications en espace lointain [44].

Les codes LDPC ont été sélectionnés plus tardivement pour le standard DVB-S2 [23] ratifié en 2005 puis dans son extension DVB-S2X [45] en 2014. Ces standards portent sur la diffusion de vidéos en haute définition par satellite.

Les codes LDPC ont aussi été retenus par le CCSDS dans ses recommandations pour les

communications spatiales proches de la Terre [46].

1.4.3 Les contraintes des communications spatiales sur le codage canal

Les communications numériques spatiales imposent de fortes contraintes sur le codage canal. Tout d'abord, les besoins au niveau du pouvoir de correction sont importants en raisons des nombreuses sources de bruit qui dégradent la qualité des signaux transmis et du coût élevé d'une éventuelle retransmission.

Les codes LDPC de grandes tailles présentent des caractéristiques qui les rendent attractifs pour ces applications. En effet, le pouvoir de correction augmente avec la taille du code. De plus, plus ces codes sont grands, plus la taille des trames est grande. Cela signifie que l'utilisation des *headers* a moins d'impact sur le débit utile. Par exemple, dans le cadre du DVB-S2 et du DVB-S2X, il existe trois valeurs possibles : N = {16200, 32400, 64800}.

Ensuite, les débits des communications numériques spatiales sont de plus en plus élevés. C'est le cas dans les applications d'observation de la Terre où les images prises et transmises depuis l'espace sont de plus en plus précises. Cela signifie que le débit nécessaire pour les redescendre sur Terre augmente. C'est aussi le cas de la connexion à Internet par satellites où le nombre d'utilisateurs augmente avec des besoins en débit toujours plus élevés. Dans le cadre de nos travaux, les débits visés par les applications présentées sont de 10 Gbit/s.

Dans le cadre des liens optiques, il est possible de perdre un paquet de plusieurs trames successives lors de la traversée de l'atmosphère. Ce phénomène de *fading* peut durer plusieurs millisecondes. Il est donc nécessaire d'entrelacer des trames pour répartir ces erreurs successives dans un nombre supérieur de trames. Ces perturbations sont alors dispersées dans différentes trames qui peuvent être plus efficacement corrigées.

Enfin, les communications numériques spatiales sont à l'origine de contraintes fortes sur le matériel embarqué dans l'espace. Les composants électroniques embarqués dans les satellites sont exposés à des rayonnements importants. Ils sont durcis pour résister à ces radiations. Leur robustesse est soumise à des certifications. De plus, la dissipation thermique est une problématique importante pour ces composants embarqués. Leur consommation énergétique est donc souvent limitée. Ces certifications et les limites de consommation énergétique engendrent le fait que les FPGA les plus récents et performants ne peuvent pas être retenus pour les applications spatiales.

Toutes ces contraintes font qu'il est nécessaire dans un premier temps, de concevoir des

codes et des algorithmes de décodage fournissant un pouvoir de correction satisfaisant les besoins applicatifs. Dans un second temps, les algorithmes de décodage doivent présenter des complexités calculatoires permettant l'implantation matérielle d'architectures temps réel de décodeurs compatibles avec les contraintes du spatial. Le chapitre suivant a pour but de présenter des algorithmes de décodage dont le pouvoir de correction se rapproche de l'algorithme BP-SPA tout en possédant des complexités calculatoires plus faibles. Les différents types d'architectures de décodeurs existant pour implanter ces algorithmes de décodage y sont aussi détaillés.

2 Implémentation de décodeurs LDPC

Ce deuxième chapitre a pour but principal de présenter les différents types d'architectures matérielles de décodeurs LDPC. Un second objectif est de détailler les différents éléments à prendre en compte dans la conception d'un décodeur LDPC sur FPGA. Dans un premier temps, les différents ordonnancements possibles des calculs sont détaillés afin de montrer leurs influences sur les architectures de décodeurs. Ensuite, des algorithmes de décodage sous-optimaux, employés usuellement pour la conception de systèmes de communications numériques, sont présentés. Certains aspects à considérer pour l'implémentation de décodeurs LDPC sont alors abordés. Enfin, un flot de conception de décodeurs LDPC sur FPGA est proposé à partir des éléments développés dans ce chapitre.

2.1 Ordonnancements et architectures

Les algorithmes de *message passing* utilisés pour le décodage de codes LDPC tel que le BP-SPA ont été proposés initialement avec une organisation des calculs basée sur un ordonnancement par inondation [36]. Il existe d'autres méthodes d'ordonnancement avec chacune un parallélisme particulier. Elles peuvent avoir un impact sur le pouvoir de correction du processus de décodage. Il est primordial de noter que le degré de parallélisation des architectures matérielles des décodeurs LDPC dépend de l'ordonnancement choisi.

2.1.1 Ordonnancement par inondation

Historiquement utilisé la première fois dans la formulation de l'algorithme BP-SPA, l'ordonnancement par inondation (*flooding* en anglais) consiste à décomposer l'exécution d'une itération de décodage en 2 étapes disjointes. La première étape évalue les messages $c2v$ provenant des CN (Figure 2.1 (a)). Au cours de la seconde étape, les messages $v2c$ sortant des VN sont calculés (Figure 2.1 (b)).

À partir de cet ordonnancement, une représentation d'une architecture générique est illustrée sur la figure 2.2 avec S VNP (processeur de nœuds de variable) et T CNP (processeur de nœuds de parité). Différents processeurs effectuent les calculs correspondant aux différents nœuds.

Cet ordonnancement des calculs permet d'utiliser des architectures massivement parallèles. En effet, ce type d'architecture alloue N VNP (processeur de nœuds de variable) et M CNP (processeur de nœuds de parité). Cela permet d'effectuer une itération en deux macro-cycles et ainsi d'atteindre des débits très élevés. Cependant, elle nécessite une empreinte mémoire élevée car il faut mémoriser $N + 2Md_c$ données pour une itération. En effet, un '1' dans la matrice $\mathbf{H}$ correspond à un message $c2v$ et un message $v2c$ au cours d'une itération. Or, la matrice $\mathbf{H}$ contient $2Md_c$ '1'. Il faut aussi conserver les N LLR a priori entrant dans le décodeur.

Pour augmenter le débit, il est possible de dérouler les itérations de décodage en dupliquant

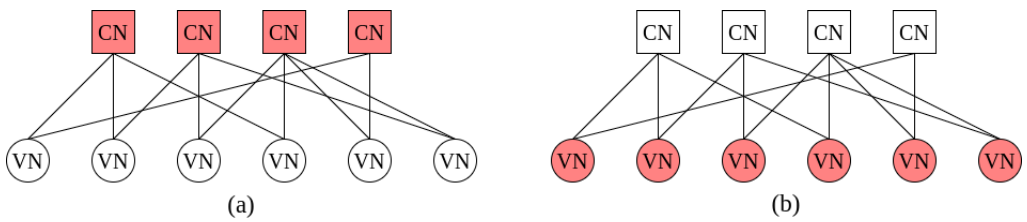

Figure 2.1 – Ordonnancement par inondation

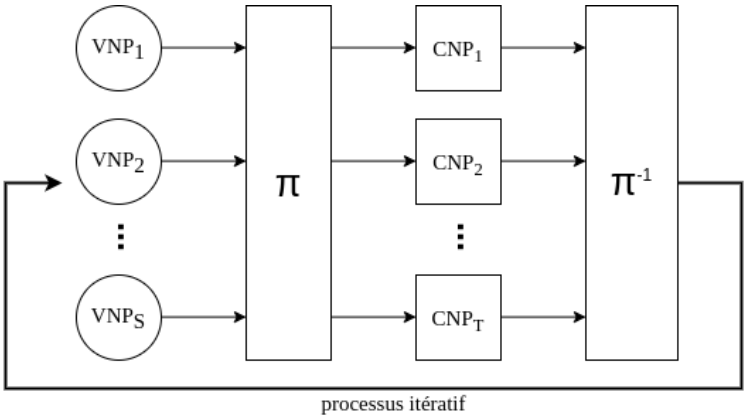

Figure 2.2 – Architecture générique pour les algorithmes de décodage suivant un ordonnancement par inondation

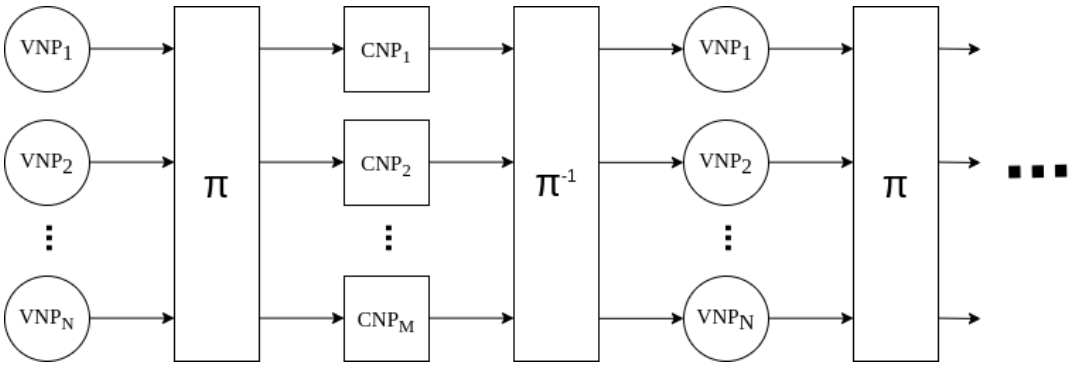

Figure 2.3 – Architecture parallèle et déroulée pour les algorithmes de décodage suivant un ordonnancement par inondation

i fois l'architecture de la figure 2.2 avec i le nombre d'itérations souhaitées. L'architecture qui en résulte est schématisée dans la figure 2.3. Ce type d'architectures entièrement parallèles et déroulées pose problème pour des codes LDPC de grandes tailles lors d'une implantation matérielle sur circuit FPGA. En effet, les outils de synthèse, de placement et de routage ne réussissent pas à générer le circuit logique correspondant au décodeur. Le logiciel Vivado 2018.2 de Xilinx n'arrive par exemple pas à synthétiser le circuit logique correspondant à un décodeur parallèle et déroulé selon l'algorithme Gallager B pour un code LDPC de taille $N = 20480$.

À l'opposé des architectures parallèles, les architectures séquentielles effectuent le processus de décodage à l'aide d'un unique ensemble de deux processeurs (un VNP et un CNP). Une représentation de ces architectures est proposée sur la figure 2.4. Cette famille d'architectures atteint des débits faibles mais présente une faible complexité matérielle.

Enfin, les architectures semi-parallèles se situent entres les architectures parallèles et

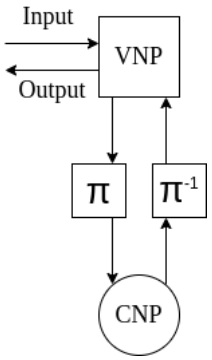

Figure 2.4 – Architecture entièrement séquentielle pour les algorithmes de décodage suivant un ordonnancement par inondation

séquentielles. Le choix du nombre S de processeurs VNP et du nombre T de processeurs CNP permet d'obtenir un compromis entre débit et complexité matérielle. Cette famille d'architecture est particulièrement intéressante lorsque des codes QC-LDPC sont considérés. Dans ce cas, fixer les valeurs de S et T à Z permet de simplifier d'une part les entrelaceurs mais également les accès aux mémoires.

Afin de diminuer l'empreinte mémoire et d'augmenter la vitesse de convergence, les ordonnancements par couches ont été proposés.

2.1.2 Ordonnancement par couches

L'ordonnancement par inondation alterne de manière séquentielle le calcul de tous les CN et de tous les VN. Les ordonnancements par couches visent à organiser les calculs autour des différents nœuds du graphe de Tanner. Dans le cadre d'un ordonnancement par couches horizontales, les calculs sont centrés sur les CN. Pour un ordonnancement par couches verticales, les calculs sont centrés sur les VN. L'utilisation de ces ordonnancements permet d'accélérer la propagation de croyances associée.

2.1.2.1 Ordonnancement par couches horizontales

Appelé *horizontal layered scheduling* en anglais, l'ordonnancement par couches horizontales concentre les calculs autour des nœuds de parité [47]. La figure 2.5 illustre graphiquement ce processus. Dans un premier temps, comme le montre la figure 2.5 (a), un CN est sélectionné. Le calcul de son état interne est réalisé permettant ainsi d'évaluer les valeurs des messages sortant de ce nœud. Puis, comme cela est illustré dans la figure 2.5 (b), les valeurs des VN auxquels il est connecté sont mises à jour. Ensuite, c'est au tour d'un autre CN d'être évalué (figure 2.5 (c)). Ce traitement est appliqué successivement à tous

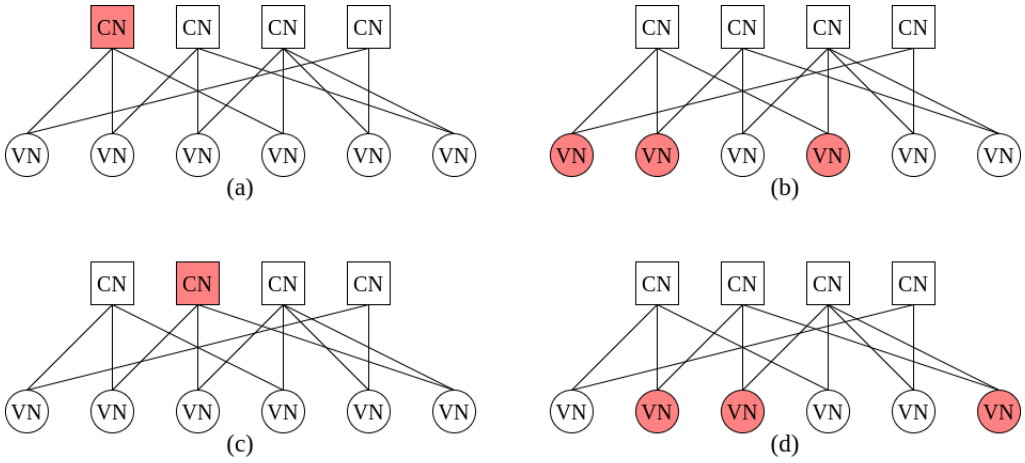

Figure 2.5 – Ordonnancement par couches horizontales

les nœuds de type CN du graphe. Les valeurs des VN sont mises à jour plusieurs fois au cours d'une même itération. Cette approche permet approximativement de doubler la vitesse de convergence du processus de décodage. En effet, dans [48], il est démontré qu'il faut deux fois moins d'itérations avec un ordonnancement par couches horizontales pour atteindre des performances équivalentes à celles obtenues avec un ordonnancement par inondation. De plus, l'empreinte mémoire des décodeurs associés est plus faible d'environ 30 %. En effet, il n'est plus nécessaire de stocker les messages $v2c$. Cependant, ces avantages ont un coût. Cet ordonnancement possède un degré de parallélisme plus faible. De plus, il ne permet pas l'utilisation d'architectures entièrement parallèles. Ce constat est lié au fait qu'il n'est possible d'exécuter en parallèle que des CN indépendants. Deux CN sont indépendants lorsqu'ils sont connectés à des groupes de VN distincts. Un taux de parallélisme élevé peut cependant être obtenu lorsque la structure du code LDPC est appropriée. Pour toutes ces raisons, l'ordonnancement par couches horizontales est retenu dans la plupart des implémentations de décodeurs LDPC qu'ils soient logiciels ou matériels [49, 50, 51, 52, 53, 54].

L'ordonnancement par couches horizontales permet d'utiliser des architectures qui concentrent l'ensemble des calculs autour d'unités de traitement centrées sur les CN. Une représentation générique de ces architectures est présentée sur la figure 2.6. Sur cette figure, les VNP et CNP sont détaillés, ce qui fait apparaître les différentes mémoires. En effet, un VNP correspond à une unité de stockage et un CNP correspond à une unité de traitement et sa mémoire associée pour les messages $c2v$. Les LLR sont initialement mémorisés dans S unités de stockage où seront accumulés les valeurs des messages c2v. Nous parlons d'accumulateurs VN. Ensuite, les calculs se font dans les unités de traitement dont le fonctionnement est schématisé sur la figure 2.7. Les messages $v2c$ sont calculés à

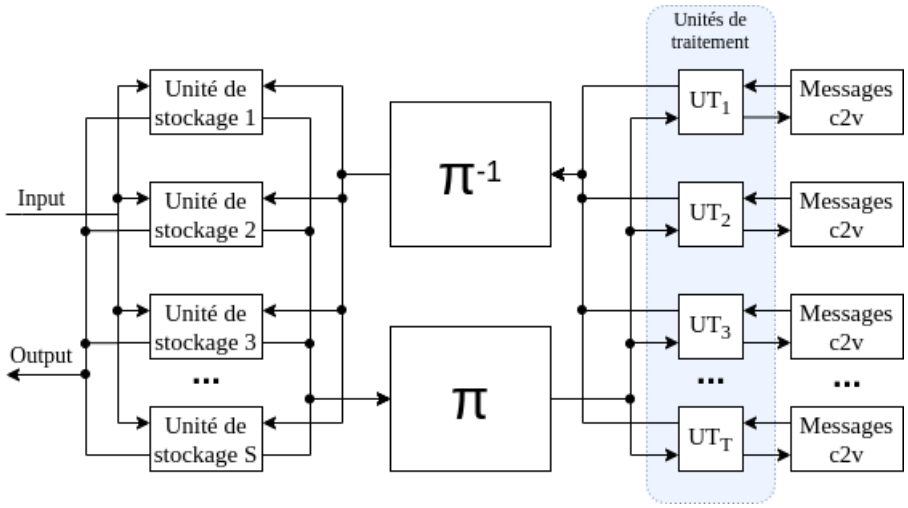

Figure 2.6 – Architecture générique de décodeur LDPC selon un ordonnancement par couches horizontales

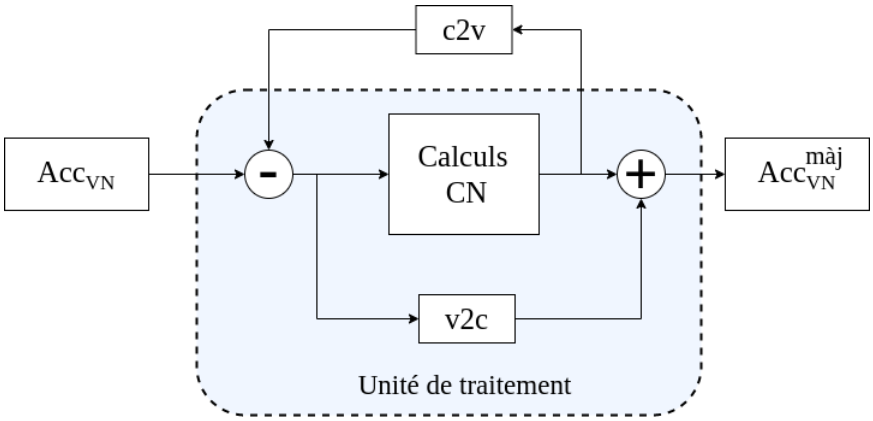

Figure 2.7 – Description d'une unité de traitement basée sur un ordonnancement par couches horizontales

partir des valeurs des accumulateurs VN et des messages $c2v$ de l'itération précédente. Ensuite les calculs des CN sont effectués pour générer les nouveaux messages $c2v$ et mettre à jour les valeurs des accumulateurs. Chaque unité de calcul est connectée à une mémoire stockant les messages $c2v$ au cours du processus de décodage. Le choix du nombre S d'unités de stockage pour les accumulateurs et T d'unités de traitement est fait en fonction des propriétés du code LDPC et des besoins applicatifs selon les critères suivants :

1. Le débit souhaité est le premier indicateur pour décider du nombre d'unités de traitement. En effet, le débit évolue linéairement avec le nombre d'unités de traitement lorsqu'elles sont utilisées efficacement (pas d'inactivé).

2. La complexité matérielle représente une contrainte qui peut limiter le nombre d'unités de traitement. En fonction de la complexité calculatoire des algorithmes, le coût matériel des unités de traitement est plus ou moins important. Ainsi, si le coût matériel du décodeur est limitant, le choix de l'algorithme et du nombre d'unités de traitement à mettre en œuvre sont deux critères primordiaux à considérer pour la conception de l'architecture du décodeur.

3. La structure de la matrice **H** dans le cadre d'un ordonnancement par couches horizontales définit les dépendances entre les différents CN. Les unités de traitement sont efficaces tant qu'il est possible d'y effectuer des calculs de CN en parallèle. Pour ce faire, les VN qui sont connectés à un CN ne doivent pas être connectés à d'autres CN en cours d'exécution. Dans le cas contraire, la valeur de l'accumulateur d'un VN peut être mis à jour dans deux unités de traitement différentes. Cela génère un conflit d'écriture en mémoire. Dans le cadre de codes QC-LDPC, une indépendance des CN par paquet de Z est assurée par la construction du code.

L'ordonnancement par couches horizontales est particulièrement intéressant lorsqu'un code LDPC quasi-cyclique (QC-LDPC) est utilisé. En effet, ces codes garantissent l'indépendance des CN par groupe de Z avec Z, le facteur d'expansion de la matrice. Cela permet de concevoir des architectures semi-parallèles efficaces lorsque Z unités de traitement et Z unités de stockage sont allouées.

Les unités de traitement peuvent être implémentées de façon séquentielle (une donnée entrante et sortante par cycle d'horloge) ou parallèle (d_c données par cycle). La méthode séquentielle est plus lente mais simplifie grandement la gestion de la mémoire. La méthode parallèle est plus rapide mais sa complexité est bien supérieure. De plus, elle aboutit à une organisation plus complexe et plus onéreuse des mémoires stockant les accumulateurs.

En résumé, les architectures semi-parallèles peuvent s'adapter aux différents domaines applicatifs. Elles permettent d'atteindre des compromis intéressant entre le coût matériel et le débit. De plus, il est possible de les rendre flexibles pour permettre le décodage de plusieurs codes LDPC. L'augmentation de la complexité qui en découle dépend de la flexibilité recherchée. Les travaux présentés dans [1] proposent une architecture semi-parallèle et générique permettant de décoder n'importe quel code LDPC à l'aide d'un ordonnancement par couches horizontales.

En plus de l'utilisation du parallélisme lié aux architectures présentées, il existe des méthodes pour augmenter les performances des décodeurs matériels. Lors de l'implantation sur circuit FPGA, il est possible de rajouter des tranches de *pipeline* afin de diminuer la durée du chemin critique et donc augmenter la fréquence de fonctionnement. Un débit

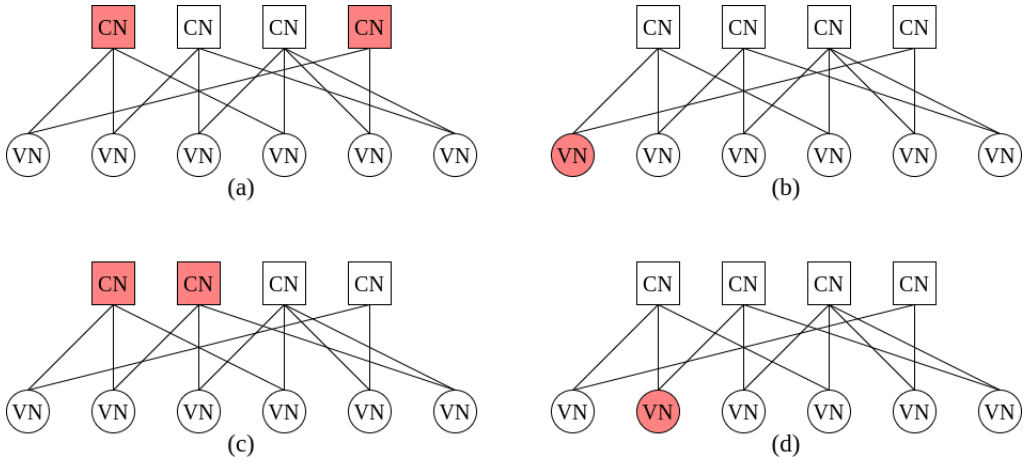

Figure 2.8 – Ordonnancement par couches verticales

de décodage plus élevé est alors obtenu. De plus, lorsque des dépendances entre les CN ne permettent pas d'utiliser efficacement une architecture semi-parallèle suivant un ordonnancement par couches horizontales, il est possible de considérer plusieurs trames en parallèle dans le décodeur afin d'avoir toujours des CN disponibles pour être exécutés. Cette organisation peut permettre une augmentation supplémentaire du débit. Cependant, ce changement impacte la latence du processus de décodage et implique une augmentation du coût mémoire.

2.1.2.2 Ordonnancement par couches verticales

Dans [55], un autre type d'ordonnancement par couches a été proposé. Cet ordonnancement est une formulation de l'algorithme avec un traitement des données par couches verticales (*vertical layered scheduling* en anglais). Les calculs sont centrés autour des nœuds de variable, comme l'illustre la figure 2.8. Cet ordonnancement est moins employé et étudié que l'ordonnancement par couches horizontales. Il peut tout de même être intéressant dans certains cas d'usage pour l'implantation matérielle de décodeur LDPC. C'est par exemple le cas dans [56] où est proposé un décodeur NMS ordonnancé par couches verticales dans le cadre d'un récepteur compatible avec le standard DVB-T2 [57].

2.1.3 Ordonnancements dynamiques

En parallèle de ces travaux sur les ordonnancements par couches, des ordonnancements dits dynamiques ont aussi été proposés [58, 59]. Dans ce cas, une sélection de nœuds VN et CN est effectuée afin de ne pas exécuter ceux qui ont peu d'impact. Une autre méthode est d'identifier les messages les moins fiables dans un ordonnancement par couches puis

de réorganiser l'ordre dans lequel sont effectués les calculs afin de les mettre à jour le plus tôt possible. Cela permet d'accélérer la diffusion des informations fiables dans le graphe. Ainsi des gains en correction d'erreurs [59] sont obtenus pour une faible augmentation de la complexité calculatoire.

2.2 Décodage sous-optimal de codes LDPC

L'algorithme SPA présenté initialement dans la littérature offre des performances de décodage de référence au prix d'une complexité calculatoire élevée liée à l'utilisation de fonctions tangentes hyperboliques ou de la fonction f. Cette complexité calculatoire est incompatible avec une montée en débit et une efficacité énergétique des architectures matérielles. En conséquence, des algorithmes de moindre complexité calculatoire et toujours basés sur la propagation de croyance ont été proposés dans la littérature. Ces algorithmes possèdent des performances en termes de pouvoir de correction proches de celles de l'algorithme BP-SPA pour une complexité calculatoire bien plus faible.

2.2.1 Algorithmes sous-optimaux

Des variantes de l'algorithme BP-SPA ont été détaillées dans la littérature afin de diminuer la complexité calculatoire du processus de décodage. Afin de les détailler, il est tout d'abord nécessaire de rappeler l'expression du calcul permettant d'estimer les messages sortants des CN dans l'algorithme BP-SPA :

$$c2v_{ji} = \underbrace{\left(\prod_{k \in N(j)-\{i\}} sign\left(v2c_{kj}\right) \right)}_{\text{signe}} \times \underbrace{f\left(\sum_{k \in N(j)-\{i\}} f\left(|v2c_{kj}|\right) \right)}_{amplitude} \tag{2.1}$$

Ce calcul peut se décomposer en deux étapes distinctes. La première est le calcul du signe du message sortant. La seconde étape correspond au calcul de son amplitude.

Tout d'abord, dans l'algorithme approximé nommé λ-min [60], le calcul des messages $c2v$ est proche de la formulation dans l'algorithme BP-SPA. La différence vient du fait que seuls les λ messages $v2c$ avec les plus faibles amplitudes sont pris en considération dans le calcul (Équation 2.1). En effet, moins l'amplitude d'un message entrant $v2c$ est importante et plus il aura d'impact sur les valeurs des messages sortants $c2v$. Nous notons $N_\lambda(j)$ l'ensemble des indices des λ messages les plus faibles entrant dans le CN_j. Ainsi le calcul des messages $c2v$ pour l'algorithme λ-min se réécrit comme suit :

$$c2v_{ji} = \left(\prod_{k \in N(j)-\{i\}} sign\left(v2c_{kj}\right) \right) \times f\left(\sum_{k \in N_\lambda(j)-\{i\}} f\left(|v2c_{kj}|\right) \right) \tag{2.2}$$

Plus la valeur de λ tend vers $d_c - 1$ et plus le pouvoir de correction de l'algorithme λ-min se rapproche de celui de l'algorithme BP-SPA. En effet, si $\lambda = d_c$ alors l'algorithme λ-min est identique au BP-SPA. A l'opposé, si $\lambda = 1$, alors l'algorithme λ-min est identique à l'algorithme Min-Sum détaillé par la suite.

En parallèle de ces travaux, l'algorithme Approximate min* (A-min*) a été présenté dans [61]. Il se démarque du BP-SPA en simplifiant les messages sortant des nœuds de parité. En effet, dans un premier temps, il identifie l'indice du VN ayant envoyé le message entrant avec l'amplitude la plus faible. Le message sortant vers ce VN est calculé avec la formule classique du BP-SPA. En revanche, les messages sortants vers les autres VN sont calculés en intégrant toutes les contributions venant des VN à l'aide de l'expression suivante :

$$c2v_{ji} = \left(\prod_{k \in N(j)-\{i\}} sign\left(v2c_{kj}\right) \right) \times f\left(\sum_{k \in N(j)} f\left(|v2c_{kj}|\right) \right) \tag{2.3}$$

Cette approximation permet de réduire la complexité calculatoire du processus de décodage.

2.2.2 Min-Sum et ses dérivés

Les approximations présentées précédemment permettaient de réduire le nombre d'opérations transcendantales exécutées. Cependant, la complexité globale reste élevée. L'algorithme Min-Sum [62] est l'approximation la plus connue de l'algorithme BP-SPA. Dans cet algorithme, les tangentes hyperboliques sont remplacées par des calculs de minimums. L'algorithme Min-Sum permet d'approximer la valeur absolue des messages $c2v$ produits par les nœuds de parité avec la formule suivante :

$$|c2v_{ji}| = \min_{i' \in N(j)-i} (|v2c_{i'j}|) \tag{2.4}$$

Cette approximation réduit radicalement la complexité calculatoire. Cependant, elle surévalue les valeurs absolues des messages $c2v$. Ce phénomène diminue les performances

de correction d'erreurs. Afin de compenser cette surévaluation de la valeur des messages, les algorithmes Offset Min-Sum et Normalized Min-Sum ont ensuite été introduits dans [63].

L'algorithme Offset Min-Sum (OMS) rajoute un terme de compensation (ou *offset* en anglais) noté α dans le calcul des informations extrinsèques. Ce terme constant durant le décodage minimise la surévaluation des messages $c2v$. Le calcul des messages $c2v$ a alors pour expression :

$$|c2v_{ji}| = max(\min_{i' \in N(j)-i}(|v2c_{i'j}|) - \alpha, 0) \tag{2.5}$$

La valeur de l'offset α est une constante positive. Une saturation est effectuée sur le résultat pour éviter d'inverser le signe quand $min(|v2c_{i'j}|) < \alpha$.

L'algorithme Normalized Min-Sum (NMS) utilise, quant à lui, un facteur de normalisation γ afin de minimiser la surévaluation des messages $c2v$. Le calcul de ces messages se fait à partir de l'expression suivante :

$$|c2v_{ji}| = \gamma \min_{i' \in N(j)-i}(|v2c_{i'j}|) \tag{2.6}$$

La valeur du facteur de normalisation γ est comprise entre 0 et 1. Son obtention et son optimisation résultent de simulation de type Monte-Carlo.

Les courbes présentées dans les figures 2.9 et 2.10 illustrent la sensibilité des performances de décodage vis à vis des paramètres α pour l'OMS et γ pour le NMS. Le code LDPC utilisé est de taille (1296, 648). Il provient du standard Wifi 802.11n.

Dans le cas de l'algorithme OMS, la valeur de l'*offset* fournissant les meilleures performances est égale à 0.15. Pour l'algorithme NMS, la valeur du facteur de normalisation produisant le meilleur pouvoir de correction est égale à 0.85.

Afin de montrer la similitude du pouvoir de correction des algorithmes BP-SPA, MS, OMS et NMS, les performances en termes de BER et de FER sont récapitulées dans la figure 2.11. Les paramètres sont $\alpha = 0.15$, $\gamma = 0.85$ et 50 itérations de décodage.

Nous observons que les performances de décodage des algorithmes BP-SPA, OMS et NMS sont similaires avec un écart maximum de 0.1 dB entre les algorithmes BP-SPA et NMS. En revanche, l'algorithme MS, qui surévalue de manière importante les messages $c2v$,

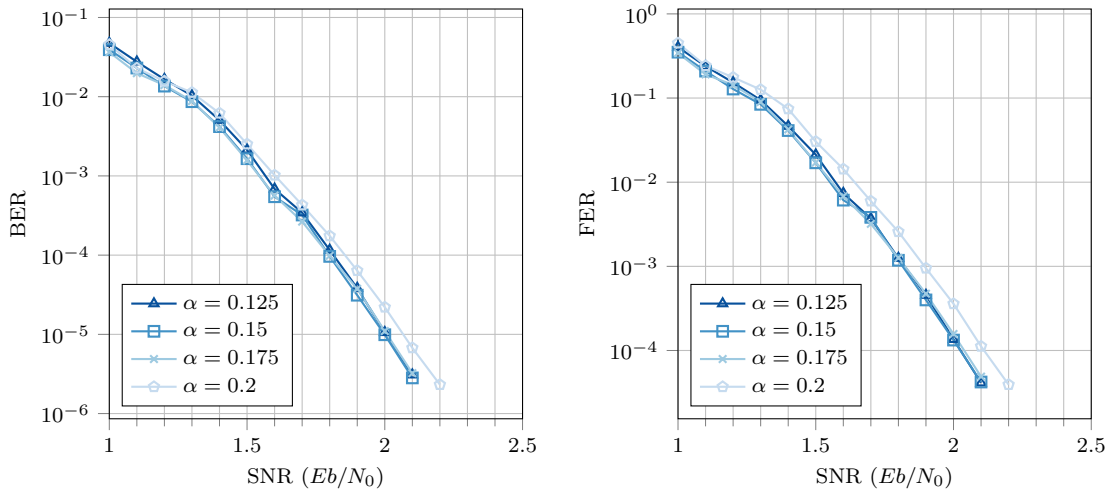

Figure 2.9 – Sensibilité des performances de décodage au paramètre α de l'algorithme OMS avec un code LDPC (1296, 648) issu du standard Wifi 802.11n (50 itérations de décodage) - Canal AWGN

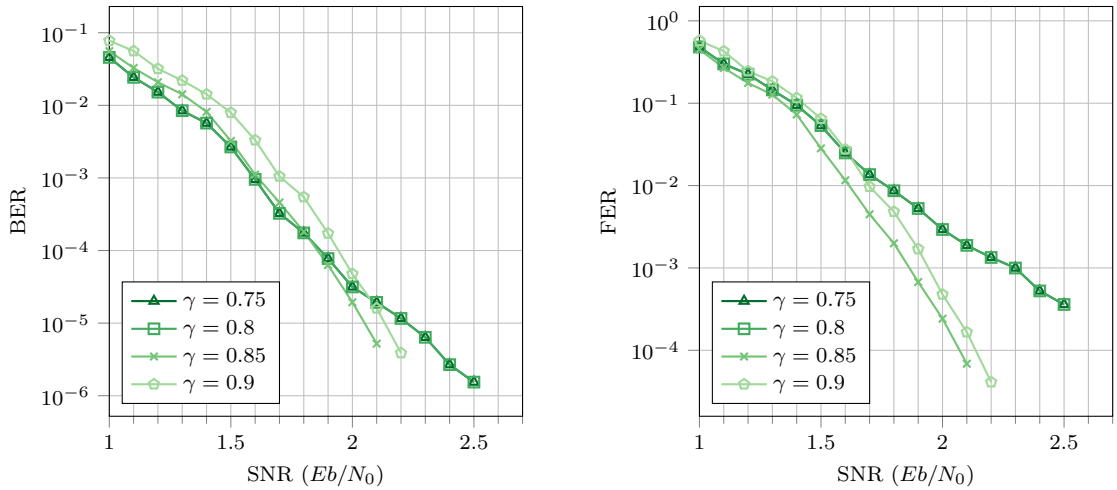

Figure 2.10 – Sensibilité des performances de décodage au paramètre γ de l'algorithme NMS avec un code LDPC (1296, 648) issu du standard Wifi 802.11n (50 itérations de décodage) - Canal AWGN

introduit une dégradation de 0.4 dB.

Afin de réduire encore la complexité calculatoire, d'autres travaux ont été menés. Par exemple, les auteurs de [64] proposent de modifier le calcul de la manière suivante :

$$|c2v_{ji}| = \min_{i' \in N(j)} (|v2c_{i'j}|) \tag{2.7}$$

Cela permet de calculer un seul minimum au lieu de deux lors de l'évaluation des CN. La

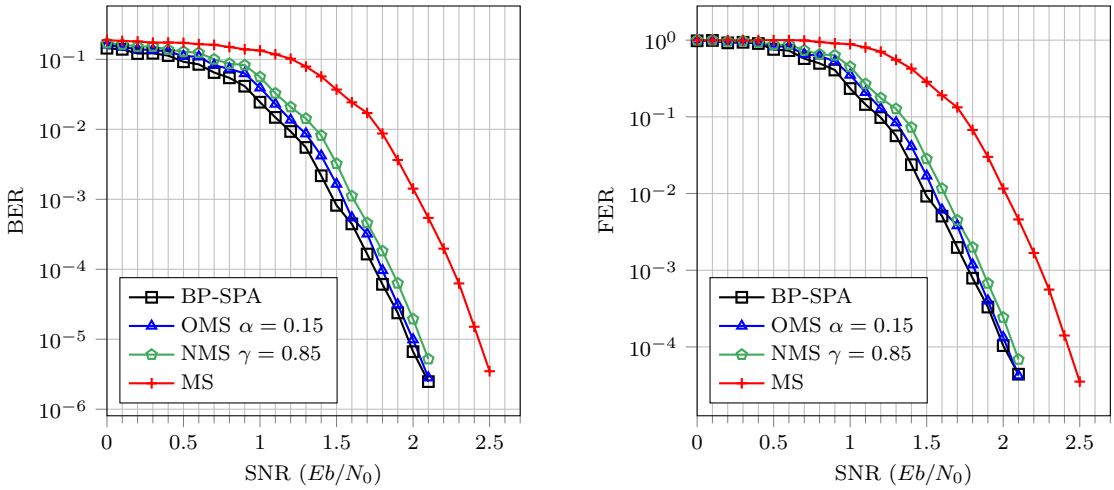

Figure 2.11 – Performances des algorithmes BP-SPA, MS, OMS et NMS pour un code LDPC (1296, 648) issu du standard Wifi 802.11n (50 itérations de décodage) - Canal AWGN

complexité calculatoire résultante diminue mais un impact notable est observé sur les performances de décodage.

D'autres travaux ont tenté de réduire l'écart de performances avec l'algorithme SPA. Par exemple, dans [65], les auteurs proposent d'utiliser deux facteurs de normalisation γ_1 et γ_2 afin de réduire de manière plus spécifique la valeur des deux minimums. Dans les travaux rapportés dans [66], les valeurs des paramètres α et γ peuvent évoluer dynamiquement en fonction des valeurs internes échangées. En parallèle, d'autres chercheurs ont proposé dans [67] de combiner les algorithmes OMS et NMS afin de corriger plus finement la surestimation des messages $c2v$.

D'un autre côté, pour limiter les commutations inutiles des CN entre deux itérations successives, le Self-Corrected Min-Sum [68] a été proposé. L'algorithme efface les messages jugés non fiables entrant dans un CN. Pour cela, il réalise un décodage de type Min-Sum mais ajoute une règle de calcul. Celle-ci consiste à mettre à zéro un message entrant dans le CN si le message provenant du même VN à l'itération précédente possède un signe différent du message de l'itération courante. Cette approche permet de légèrement améliorer les performances de décodage au prix d'une augmentation de l'empreinte mémoire et d'une réduction notable de la vitesse de convergence.

Algorithme de décodage	Calcul de $\|c2v\|$
BP-SPA	$\|c2v_{ji}\| = f\left(\sum_{k\in N(j)-\{i\}} f\left(\|v2c_{kj}\|\right)\right)$
Min-Sum	$\|c2v_{ji}\| = \min_{i'\in N(j)-i}(\|v2c_{i'j}\|)$
Offset Min-Sum	$\|c2v_{ji}\| = max(\min_{i'\in N(j)-i}(\|v2c_{i'j}\|) - \alpha, 0)$
Normalized Min-Sum	$\|c2v_{ji}\| = \gamma \min_{i'\in N(j)-i}(\|v2c_{i'j}\|)$
Relaxed Min-Sum	$\|c2v_{ji}\| = \min_{i'\in N(j)}(\|v2c_{i'j}\|)$
Self-Corrected Min-Sum	$v2c_{ij} = \begin{cases} 0, & sign(v2c_{ij}^{new}) \neq sign(v2c_{ij}^{old}) \\ sign(v2c_{ij}^{new}), & sinon \end{cases}$ $\|c2v_{ji}\| = \min_{i'\in N(j)-i}(\|v2c_{i'j}\|)$
λ-min	$\|c2v_{ji}\| = f\left(\sum_{k\in N_\lambda(j)-\{i\}} f\left(\|v2c_{kj}\|\right)\right)$
A-min*	Si $i = \underset{i'\in N(j)}{argmin}\|v2c_{i'j}\|$ $\|c2v_{ji}\| = f\left(\sum_{k\in N(j)-\{i\}} f\left(\|v2c_{kj}\|\right)\right)$ Sinon $\|c2v_{ji}\| = f\left(\sum_{k\in N(j)} f\left(\|v2c_{kj}\|\right)\right)$

Tableau 2.1 – Récapitulatif non exhaustif des différentes approximations proposées pour le calcul de l'amplitude des messages $c2v$ dans la littérature

2.2.3 Récapitulatif

Le tableau 2.1 résume les calculs des amplitudes des messages $c2v$ pour les différents algorithmes présentés dans cette partie. La figure 2.12 propose de mettre en regard respectivement leur pouvoir de correction et leur complexité calculatoire. Cette figure est inspirée de [69] et enrichie avec des algorithmes supplémentaires.

Les algorithmes présentés ici sont les plus communément utilisés dans la littérature. Ils offrent des compromis différents entre la complexité calculatoire et le pouvoir de correction. Il est à noter que les performances de certaines approximations sont sensibles à la structure du code LDPC (d_c, d_v, régularité). Cela rend impossible une comparaison fine entre les approximations sans contexte applicatif. Le choix de l'algorithme doit donc se faire en fonction des besoins applicatifs réels.

D'autres variantes existent [70, 71, 72] mais n'apportent pas de gains significatifs sauf dans des contextes applicatifs très particuliers.

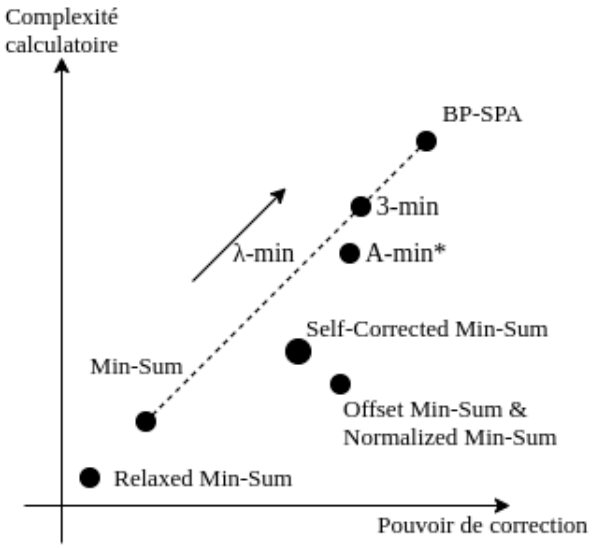

Figure 2.12 – Comparatif entre le pouvoir de correction d'erreurs et la complexité calculatoire pour les différents algorithmes de décodage de la littérature

2.2.4 Décodage à entrées dures

Les algorithmes présentés dans la section précédente ont été conçus pour effectuer un décodage à entrées souples. Ces derniers fonctionnent aussi lorsque le processus de décodage utilise des entrées dures. Cependant, pour des entrées dures, leurs pouvoirs de correction peuvent être approchés ou dépassés à l'aide d'algorithmes moins complexes car adaptés à un traitement binaire de l'information.

Gallager proposait déjà trois algorithmes pour décoder des informations dures dans [6]. Ils se nomment Gallager *algorithm* A (Gallager A), Gallager *algorithm* B (Gallager B), et Bit-Flipping *algorithm* (BF).

2.2.4.1 Algorithmes Gallager A et B

Les algorithmes Gallager A et Gallager B sont des algorithmes de type *message-passing* comme l'algorithme BP-SPA. Le processus de décodage se fait par échange de messages entre les nœuds VN et CN. Cependant, les entrées du processus de décodage ne sont pas des LLR mais des valeurs estimées des bits transmis (0 ou 1). Les valeurs échangées par les messages sont elles aussi binaires. Ainsi les messages $c2v$ sortant des CN sont calculés à partir de l'équation suivante :

$$c2v_{ji} = \bigoplus_{i' \in N(j)-i} v2c_{i'j} \tag{2.8}$$

Dans cette équation 2.8, l'opérateur $\bigoplus$ correspond à une addition modulo 2, à savoir un 'ou exclusif'.

Dans le cadre de l'algorithme Gallager A, le calcul des messages $v2c$ sortant d'un VN se fait en fonction des messages $v2c$ et de la valeur du canal y_i. Le message $v2c_{ij}$ prend la valeur du canal y_i si, au moins un message $c2v_{j'i}$ ($j' \neq j$), est identique à la valeur du canal. Sinon $v2c_{ij}$ prend la valeur complémentaire du canal $\overline{y_i}$. Il est possible de formuler le calcul de la manière suivante :

$$v2c_{ij} = \begin{cases} y_i, & \exists j' \in M(i) - j : c2v_{j'i} = y_i \\ \overline{y_i}, & sinon \end{cases} \tag{2.9}$$

L'algorithme Gallager B opère différemment. Un processus de vote est utilisé pour déterminer les valeurs des messages $v2c$ à partir des messages entrant dans un VN. S'il y a au moins t messages $c2v_{j'i}$ ($j' \neq j$) égaux à la valeur $\overline{y_i}$ alors le message $v2c_{ij}$ prend cette valeur complémentaire, sinon la valeur y_i du canal est retenue. Ce calcul se formule de la manière suivante :

$$v2c_{ij} = \begin{cases} \overline{y_i}, & |\{j' \in M(i) - j : c2v_{j'i} = \overline{y_i}\}| \geq t \\ y_i, & sinon \end{cases} \tag{2.10}$$

La fonction de calcul de la valeur optimale de t est détaillée dans [73]. Cette valeur varie en fonction des caractéristiques du code LDPC. Par exemple, pour un VN de degré $d_v = 3$, les algorithmes Gallager A et B sont identiques.

La complexité calculatoire de ces deux algorithmes binaires dédiés à manipuler des entrées dures est très faible par rapport à ceux présentés dans la section précédente. En effet, leur implantation matérielle peut se faire uniquement à partir de portes 'ou exclusives' et de comparateurs binaires.

2.2.4.2 L'algorithme Bit-Flipping

Contrairement aux algorithmes de *message passing* présentés jusqu'ici, l'algorithme de décodage nommé Bit-Flipping consiste à inverser les valeurs des bits estimés comme erronés de manière à corriger les erreurs de transmission. Son processus se décompose en trois étapes qui sont réitérées autant de fois que souhaité.

1. Dans un premier temps, le vecteur syndrome est calculé :

$$\mathbf{s} = \mathbf{y} \times \mathbf{H}^t \tag{2.11}$$

   Si le résultat est le vecteur nul alors $\mathbf{y}$ est un mot de code et le processus de décodage se termine.

2. Ensuite, pour chaque VN le nombre de CN associés non vérifiés est calculé à l'aide de l'expression suivante dans $\mathbb{N}$:

$$\mathbf{f} = \mathbf{s} \times \mathbf{H} \tag{2.12}$$

3. Puis les valeurs des bits estimés y_i des VN dont la valeur f_i dépasse le seuil θ sont inversées. Si le nombre maximal d'itérations est atteint, alors le décodage s'arrête. Sinon, le processus de décodage recommence.

La valeur du seuil θ dépend des conditions de transmission du canal. Dans [73], il est précisé que la valeur $\theta = \underset{i \in [\![0, N-1]\!]}{max} f_i$ permet à l'algorithme de s'adapter efficacement aux caractéristiques du canal.

Pour permettre une augmentation du pouvoir de correction, d'autres algorithmes basés sur le principe de *bit flipping* ont été proposés dans la littérature. C'est le cas, par exemple, du Weighted Bit Flipping (WBF), du Gradient Descent Bit Flipping (GDBF) et du Probabilistic Gradient Descent Bit Flipping (PGDBF). Ils consistent à modifier les métriques indiquant les bits à inverser afin d'améliorer les performances de décodage.

2.3 Quantification des données

Jusqu'à présent dans le document, les performances de décodage étaient obtenues à partir de simulations effectuées avec des représentations des données en virgule flottante. Cette représentation permet d'obtenir une précision quasi-infinie mais l'utilisation de ce format

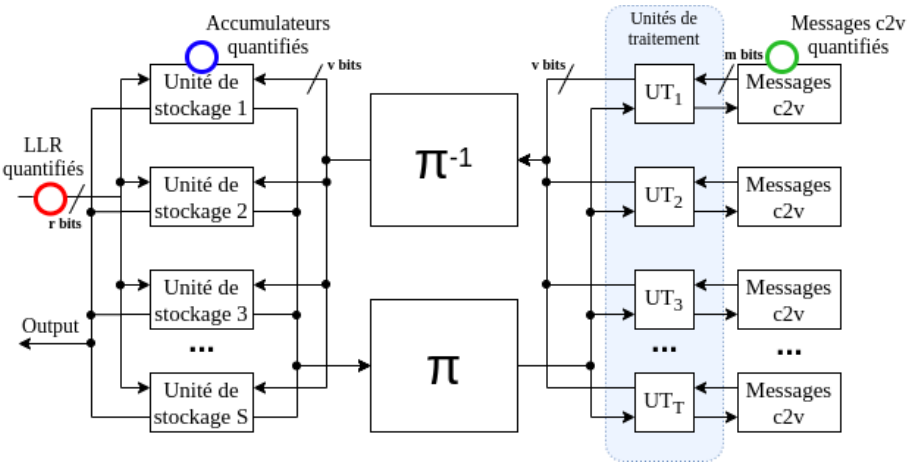

Figure 2.13 – Emplacement des valeurs quantifiées dans l'architecture semi-parallèle

de données est très coûteux en termes de ressources matérielles. Une représentation en virgule fixe est beaucoup plus efficace mais elle peut affecter la précision des calculs et donc impacter les performances de décodage.

Pour atteindre les débits attendus par les applications spatiales tout en respectant les contraintes de complexité, l'utilisation d'une représentation des données en virgule fixe est nécessaire. Ainsi, dans le cadre de la conception d'une architecture semi-parallèle exécutant un décodage par couches horizontales, trois types de données sont à quantifier. La figure 2.13 permet de visualiser le positionnement des données quantifiées. Les LLR en entrée du décodeur, indiqués par le cercle rouge sur la figure, les valeurs des accumulateurs VN, indiquées par le cercle bleu, et les valeurs des messages $c2v$, indiquées par le cercle vert, peuvent être quantifiés avec des formats différents.

Il est nécessaire d'évaluer l'impact de la quantification en virgule fixe de ces données sur les performances de décodage. Nous notons $Q_{r,v,m}$ le format de quantification. Les LLR représentés par r bits. Dans le cadre des études présentées dans ce manuscrit, la quantification des LLR en entrée du décodeur sont est uniforme. Cependant, il est possible d'utiliser une quantification non uniforme. Les accumulateurs VN sont codés sur v bits et les messages $c2v$ sur m bits. Pour limiter les saturations des accumulateurs, à l'origine de plancher d'erreurs, les valeurs v et m doivent tenir compte de la contrainte suivante :

$$v \geqslant m + log_2(d_{v,max}) \tag{2.13}$$

La figure 2.14 montre les performances obtenues pour l'algorithme NMS ($\gamma = 0.85$) en

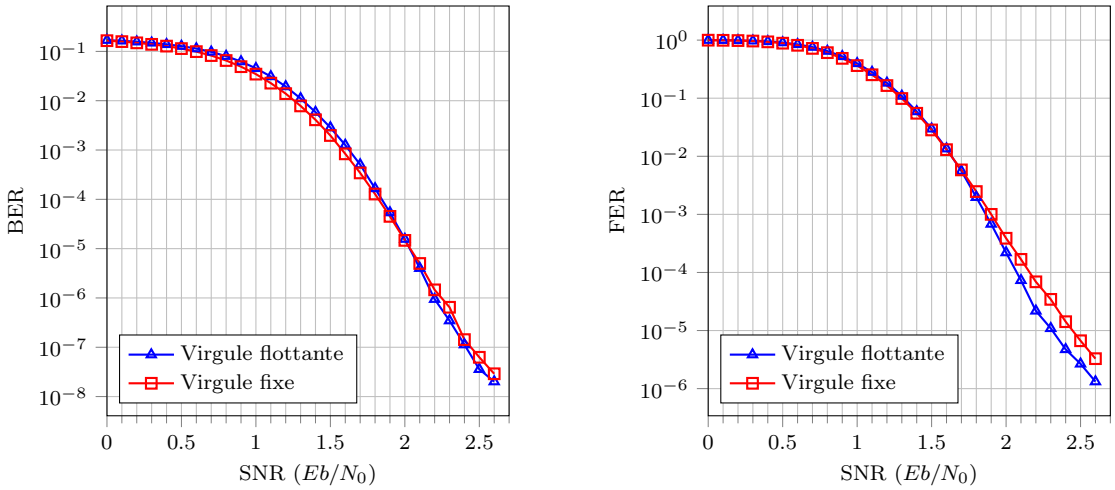

Figure 2.14 – Comparaison des performances de l'algorithme NMS entre les représentations en virgule flottante et en virgule fixe avec le code (1296, 648) du standard Wifi 802.11n (50 itérations de décodage)

virgule flottante et en virgule fixe avec le code du standard Wifi 802.11n. Le format de quantification utilisé est $Q_{6,8,5}$. Comme $d_{v,max} = 8$, le format $Q_{6,8,5}$ respecte l'équation 2.13.

Des performances similaires de décodage sont obtenues. Dans cette configuration, la quantification des données n'impacte que très peu le pouvoir de correction mais permet cependant de réduire drastiquement le coût d'implantation. Dans le chapitre 3, des quantifications avec des nombres de bits plus faibles sont étudiées. Dans certains cas, l'impact sur les performances est trop élevé pour retenir ces formats de quantification.

2.4 Plateforme ciblée

Depuis la redécouverte des codes LDPC, de nombreux travaux scientifiques ont ciblé l'implémentation en temps réel des algorithmes de décodage présentés précédemment. À l'heure actuelle, il existe un grand nombre de solutions technologiques pour implémenter des décodeurs LDPC atteignant des débits de quelques Mbit/s à plusieurs centaines de Gbit/s. Ces solutions variant de l'implantation matérielle à l'implémentation logicielle offrent toutes des compromis différents entre temps de conception et niveau de performances. Le tableau 2.2 récapitule les caractéristiques des différentes solutions technologiques.

Au cours de ces dernières années, de nombreux travaux se sont intéressés à l'implémentation logicielle de décodeurs LDPC. Ces implémentations logicielles, ciblant des technologies *multi-cores* (CPU) [51] ou *many-cores* (GPU) [74, 53], offrent des niveaux de flexibilité

élevés. Les travaux les plus récents de l'état de l'art présentent des débits de plusieurs Gbit/s pour des codes LDPC [51, 75, 74], avec des latences plus ou moins faibles [76]. Ces niveaux de performances sont obtenus au détriment d'une consommation énergétique élevée et d'un coût silicium important.

De manière plus traditionnelle, la conception de décodeurs LDPC efficaces s'est principalement tournée vers des implantations matérielles de circuits dédiés sur technologie de type ASIC [77]. La conception d'architectures dédiées de niveau RTL (Registre Transfer Level) est généralement chronophage car très dépendante du ou des codes LDPC considérés. De plus, l'utilisation des outils nécessaires à la conception de circuit ASIC, par exemple Design Compiler de Synopsys pour la synthèse logique et Encounter Digital Implementation de Cadence pour le placement et le routage, est complexe. De fait, cette approche nécessite un temps important pour la mise en production. Cependant, les décodeurs LDPC ainsi obtenus offrent un très grand niveau d'efficacité en termes de débit, de complexité matérielle et de consommation d'énergie. Afin de réduire le temps de conception de tels décodeurs, différentes approches ont été proposées. Tout d'abord, afin d'augmenter la flexibilité des architectures et de minimiser le temps de conception pour de nouveaux codes, l'utilisation d'architectures de type processeur programmable spécialisé (ASIP [78]) a été plébiscitée [79]. Ces approches pouvant être semi-automatisées, aboutissent à la génération d'architectures de type processeur offrant certains niveaux de flexibilité à la conception et à l'utilisation. Cette flexibilité s'obtient au détriment d'une légère baisse des performances des architectures ainsi conçues. Récemment, des travaux visant à l'automatisation de la génération d'architectures de décodeurs LDPC au niveau RTL ont été proposés, soit à l'aide d'outils *custom* [1], soit à l'aide de synthèse de haut niveau (HLS) [80]. D'une manière analogue, ces flots accélèrent la conception et augmente la flexibilité au coût d'une légère baisse de performances. Cependant, peu importe l'approche employée, le temps de conception d'un ASIC, allant des étapes de synthèse logique, de placement, de routage et mise en production (fonderie) reste long et onéreux.

Une dernière solution permettant d'implanter des décodeurs LDPC avec de hauts niveaux de performance consiste à cibler des circuits FPGA [81]. Ces circuits reprogrammables permettent l'implantation rapide d'architectures de niveau RTL [34]. En effet, les circuits FPGA peuvent être reprogrammés de manière à prototyper, caractériser et déployer des décodeurs LDPC. Les niveaux de performances sur ces circuits reconfigurables, à architecture RTL équivalente, seront légèrement en retrait par rapport à leurs pendants sur technologie ASIC. Toutefois, il est possible d'atteindre des débits de plusieurs Gbit/s sur de tels circuits. Lors du prototypage de nouveaux systèmes de communications ou pour la mise en production de petites séries, ces circuits FPGA sont privilégiés aux

Cible	Débit	Flexibilité	Temps de dév.	Conso. énergétique
CPU	++	+++	+++	-
GPU	++	+++	++	-
FPGA (HLS)	++	++	++	++
FPGA (ASIP)	++	++	+	++
FPGA (RTL)	+++	+	+	+++
ASIC (ASIP)	+++	+	- -	+++
ASIC (RTL)	++++	-	-	++++

Tableau 2.2 – Tableau des caractéristiques des différentes solutions technologiques pour l'implémentation de décodeur LDPC (Le nombre de '+' indique la qualité de la technologie dans une catégorie)

circuits ASIC.

Pour toutes ces raisons, nous nous sommes focalisés sur l'implantation sur circuit FPGA. Toutefois, les architectures décrites pourraient amener à la conception de décodeurs LDPC sur circuit ASIC, améliorant d'office les performances (débit, latence et énergie).

2.5 Synthèse

Dans le cadre de notre étude sur les communications optiques pour le domaine spatial, des plateformes à base de circuits FPGA sont ciblées. En effet, pour le lien descendant, un circuit FPGA de type Zynq Ultrascale + est une solution technologique intéressante couplant un nombre de ressources important et une consommation d'énergie modérée. Pour le lien montant, le décodeur LDPC doit être embarqué dans un satellite, impliquant une faible consommation énergétique, de la flexibilité et une robustesse aux radiations. Dans ce contexte, un circuit FPGA de la gamme Kintex Ultrascale a été sélectionné en accord avec ces contraintes.

À partir des éléments présentés dans les deux premiers chapitres, nous avons pu constater que la conception d'un décodeur LDPC à partir de besoins applicatifs nécessite de multiples prises de décisions. En conséquence, dans le cadre de ces travaux, la conception des décodeurs LDPC matériels sur circuit FPGA a été décomposée en plusieurs étapes disjointes mais interdépendantes. L'organisation du flot de conception qui en découle est résumée dans la figure 2.15.

Les principales étapes et leur enchaînement chronologique sont détaillés ci-dessous.

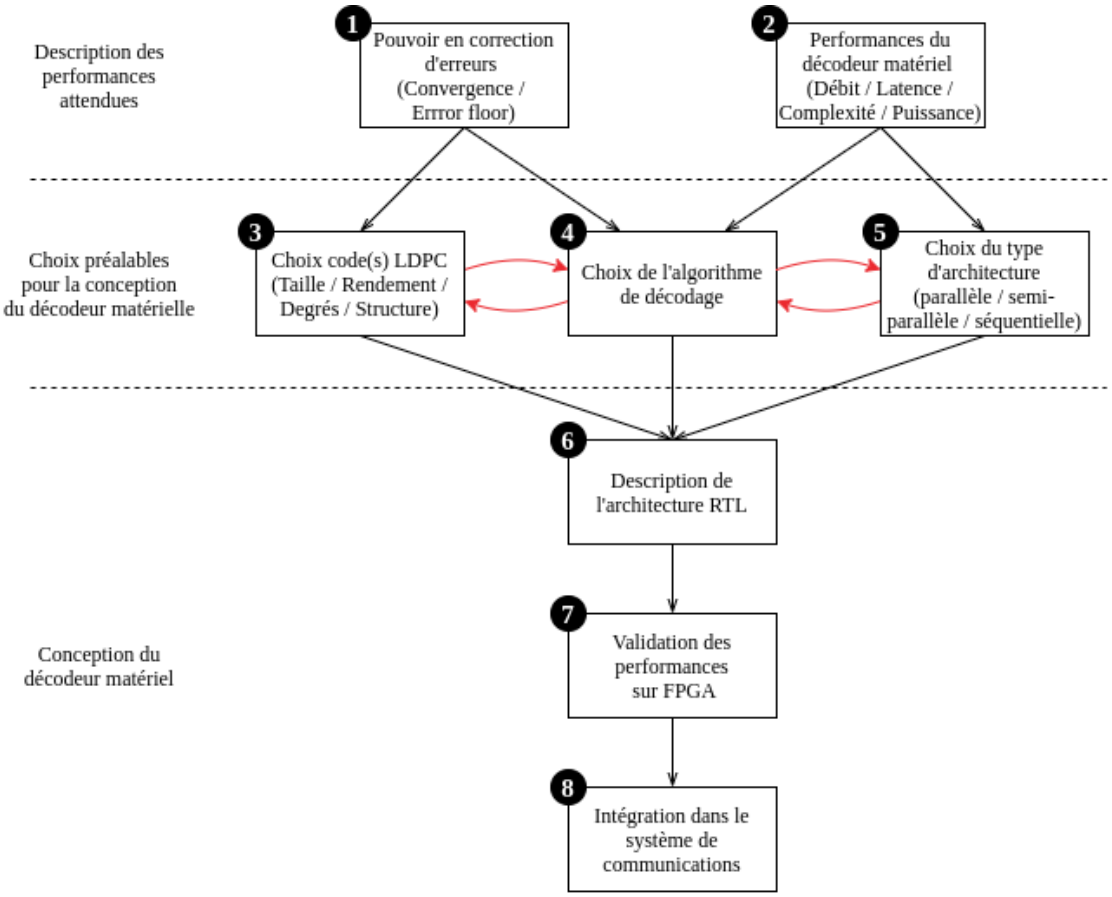

Figure 2.15 – Décomposition du flot de conception de décodeurs LDPC

❶ **Analyse du pouvoir de correction attendu** - Il s'agit d'identifier les besoins applicatifs au niveau du pouvoir de correction, notamment en positionnant la zone de convergence et l'*error floor* sur une courbe de BER/FER.

❷ **Analyse des performances matérielles attendues** - Il s'agit d'identifier les attentes en termes de débit et de latence ainsi que les contraintes de complexité matérielle et de consommation énergétique qui contraignent la conception de l'architecture.

❸ **Définition du ou des codes LDPC** - À partir des contrainte de l'analyse ❶ il est nécessaire d'identifier les contraintes à imposer sur la définition du code LDPC : la taille des trames, le rendement, etc. Cela peut aboutir à la conception de multiples codes LDPC possédant des propriétés compatibles avec les besoins exprimés. Durant cette étape, des simulations de type Monte Carlo sont réalisées afin de sélectionner les meilleurs candidats.

❹ **Définition de l'algorithme de décodage** - À partir des codes LDPC sélectionnés,

des tests sont effectués avec des algorithmes de décodage sous-optimaux. Le nombre d'itérations est restreint pour obtenir des débits compatibles avec les besoins définis dans l'étape ❷. Les performances au niveau du pouvoir de correction sont vérifiées et doivent cohérentes avec les contraintes décrites durant l'étape ❶.

❺ **Définition de l'architecture de décodage** - En fonction de la structure du code LDPC, de l'algorithme de décodage et des performances matérielles attendues, une architecture du décodeur LDPC est privilégiée. Ce choix prend en compte l'ensemble des paramètres définis au cours des étapes ❷, ❸ et ❹.

❻ **Description de l'architecture RTL** - À partir des décisions prises dans les étapes précédentes, l'architecture RTL peut être décrite. Son degré de flexibilité et sa parallélisation doivent être adaptés aux besoins. Son coût d'implantation matérielle sur FPGA doit être en accord avec les besoins énoncés à l'étape ❷.

❼ **Validation sur FPGA** - Une fois l'architecture décrite, un prototypage est effectué sur circuit FPGA afin de valider le pouvoir de correction du décodeur et les débits atteints.

❽ **Intégration** - Une fois les performances du décodeur LDPC matériel validés, l'intégration de ce décodeur LDPC dans le reste du système de communication peut être effectuée.

Les étapes ❸, ❹ et ❺ sont interdépendantes. En effet, la structure du code LDPC doit être choisie en fonction du type d'architectures pouvant atteindre le débit attendu. De plus, le choix de l'algorithme de décodage et du code LDPC doit être fait conjointement pour garantir le pouvoir de correction nécessaire au bon fonctionnement du système de communications.

Ainsi, la conception d'un décodeur LDPC matériel sur FPGA a été présentée depuis l'étape de définition des contraintes jusqu'à l'intégration du décodeur dans le système de communication.

Dans ce chapitre, nous avons mis en avant différents algorithmes de décodage utilisés régulièrement pour l'implémentation de décodeurs LDPC. Nous avons décrit plusieurs ordonnancements pour exécuter les calculs de ces algorithmes. Enfin nous avons présenté les différentes familles d'architectures de décodeurs LDPC. L'ensemble de ces éléments sont nécessaires à la conception d'un décodeur. C'est pourquoi, nous avons proposé une organisation et un flot de conception pour assurer le développement de décodeurs LDPC conformes à un cahier des charges.

3 Étude du lien descendant

Ce troisième chapitre a pour but de présenter les travaux effectués au cours de l'étude sur un lien optique descendant pour des communications spatiales. Dans un premier temps, les contraintes algorithmiques et architecturales seront mises en évidence. Ensuite, le choix de l'algorithme de décodage sera justifié par une comparaison des différents algorithmes possédant une complexité calculatoire modérée. Le code LDPC utilisé sera alors conçu suite à une étude analysant les performances de différentes techniques de construction. Puis une architecture sera proposée pour l'implantation matérielle du processus de décodage sur cible FPGA. Enfin les résultats après implantation en termes de débit et de complexité matérielle seront analysés puis comparés avec l'état de l'art pour souligner la pertinence des travaux réalisés.

3.1 Contexte

Cette première partie du chapitre a pour but de présenter les problématiques liées à la protection du lien descendant pour des satellites sur orbite terrestre. L'objectif est de mettre en exergue les contraintes issues de ce domaine applicatif. En effet, ces contraintes en termes de pouvoir de correction d'erreurs, de débit et de ressources matérielles disponibles sont contraignantes pour le code correcteur d'erreurs et le décodeur matériel associé.

3.1.1 Contraintes liées au domaine applicatif

Le système développé doit permettre de mettre en œuvre un lien descendant entre un satellite LEO d'observation et la Terre. Ce système doit atteindre un débit de l'ordre de 10 Gbit/s. Ce débit élevé est nécessaire pour permettre un transfert en temps réel des images capturées par un satellite en orbite vers une station de base.

L'utilisation de codes correcteurs d'erreurs est nécessaire pour fiabiliser la transmission. Elle contraint fortement le dimensionnement du décodeur de code correcteur d'erreurs en raison du niveau de performances attendu.

Les premières contraintes portent sur le choix du schéma de codage afin de satisfaire les besoins en correction d'erreurs de l'application.

- Les performances en correction d'erreurs doivent être similaires à celles offertes par le standard DVB-S2 afin de fiabiliser le lien optique et limiter l'impact du bruit introduit par le passage à travers le canal de transmission. Le lien étant unidirectionnel, il n'est pas possible de renvoyer une trame en cas d'erreurs. L'objectif est d'obtenir un couple code/décodeur suffisamment performant pour ne pas avoir recours à un code BCH supplémentaire comme dans le standard DVB-S2. En effet, la complexité matérielle d'un décodeur BCH à 10 Gbit/s est trop élevée pour envisager l'utilisation de cette solution dans notre système.
- À cela s'ajoute le besoin d'avoir un *error floor* se situant à un BER inférieur à 10^{-10}. Ceci est dû au lien optique qui est unidirectionnel. Il n'est donc pas envisageable de renvoyer les trames erronées et l'application nécessite un taux d'erreurs suffisamment faible pour éviter de dégrader de manière notable les images transmises.
- En outre, pour minimiser l'*overhead* de *signaling* et de *header*, et parce que les performances en termes de correction d'erreurs augmentent avec la taille du code correcteur d'erreurs, la longueur d'un mot de code est fixée à 16k bits. Ceci

représente un compromis entre l'empreinte mémoire, la complexité matérielle, la latence de décodage et le pouvoir de correction.

- Le rendement du schéma de codage est fixé à 3/4 car les performances dans le cadre du standard DVB-S2 à ce ratio de codage permettent d'envisager le fonctionnement du système pour le bilan de liaison envisagé par Airbus.

Les autres contraintes portent sur l'implantation matérielle du décodeur.

- Pour faire face aux problèmes de fading décrit dans le chapitre 1, l'étude R&T CNES a établi qu'un entrelacement de trames durant 15 ms est nécessaire. Le décodage canal doit donc introduire une latence non significative par rapport à ces 15 ms. Une contrainte supplémentaire est donc d'obtenir une latence inférieure à 100 μs.
- De plus, pour des raisons de coût et de mise en oeuvre, il est nécessaire que le décodeur canal puisse atteindre un débit codé de 10 Gbit/s à l'aide d'un unique circuit FPGA de type Xilinx Zynq UltraScale + [82]. C'est une contrainte forte sur le système à concevoir. Elle implique une maitrise fine de la complexité matérielle.

Après une étude des différents types de codage canal, il a été décidé de se focaliser sur les codes LDPC, et cela pour plusieurs raisons :

1. Leur pouvoir de correction proche de la limite de Shannon,
2. La grande diversité de codes LDPC et d'algorithmes de décodage permettant de répondre aux contraintes de cadres applicatifs variés,
3. Leurs adoptions dans les standards du domaine spatial (DVB-S2 [23], CCSDS [46]) qui ont démontré leur pertinence,
4. Les résultats d'implémentation de décodeurs récents proposent des débits supérieurs au Gpbs pour des complexités calculatoires compatibles avec le cadre de l'étude.

3.2 Choix de l'algorithme de décodage

Dans cette partie, les algorithmes de décodage à faible complexité calculatoire vont être comparés afin d'identifier le meilleur compromis entre les performances de décodage et la complexité calculatoire. L'ensemble des algorithmes considérés manipulent des entrées souples. Dans un premier temps, des algorithmes de *bit-flipping* seront présentés. Leurs

performances seront comparées à celles obtenues par des algorithmes de type Min-Sum afin de vérifier s'ils permettent ou non d'atteindre un pouvoir de correction proche des codes du standard DVB-S2.

3.2.1 Algorithmes de type Bit-Flipping

Dans un premier temps, les algorithmes de type *bit-flipping* à entrées souples ont été étudiés en raison de leur faible complexité calculatoire et leur faible empreinte mémoire.

L'algorithme WBF a été présenté la première fois dans [83]. Le fonctionnement de l'algorithme consiste à vérifier la cohérence de l'information reçue puis à inverser le bit le moins fiable durant chaque itération. En considérant un canal AWGN et une modulation BPSK, l'information reçue du canal par le décodeur est notée $\mathbf{y}$ telle que $\mathbf{y} = \mathbf{x} + \mathbf{b} = (y_1, y_2, ..., y_N)$ avec $y_i = x_i + b_i$, $\mathbf{x}$ la séquence envoyée, et $\mathbf{b}$ le bruit introduit par le canal de transmission. Dans un premier temps, pour chaque équation de parité, la valeur absolue du minimum de l'information du canal reçue par les nœuds de variable qui lui sont connectés est calculée de la manière suivante :

$$|y|_{min,j} = \min_{i \in N(j)} |y_i| \tag{3.1}$$

Ensuite, chaque itération de décodage se déroule en trois étapes successives. La première consiste à calculer les différentes composantes s_1, s_2, ..., s_M du syndrome. Puis, la valeur suivante est calculée pour chaque nœud variable :

$$E_i = \sum_{j \in M(i)} (2s_j - 1)|y|_{min,j} \tag{3.2}$$

Plus cette valeur est grande, plus le nœud variable est susceptible d'être erroné. Enfin l'itération se termine en inversant le bit du nœud variable ayant la valeur de E_i la plus élevée. Cette inversion modifiera le calcul des syndromes lors de l'itération suivante. Ces trois étapes sont répétées jusqu'à l'obtention d'un mot de code valide ou lorsque le nombre maximum d'itérations est atteint.

Afin d'améliorer le pouvoir de correction et éviter les minimums locaux, les travaux, présentés dans [84], mettent en place une approche par recuit simulé (ou *gradient descent* en anglais). L'algorithme est nommé Gradient Descent Bit Flipping (GDBF). Il s'inspire de la règle de décodage *maximum likelihood* sur un canal AWGN pour une modulation de type BPSK. Cette règle de décodage peut s'écrire sous la forme $\mathbf{s} = \underset{\mathbf{x} \in \hat{\mathbf{C}}}{\operatorname{argmax}} \sum_{i=1}^{n} x_i y_i$.

La valeur résultante correspond au mot de code le plus corrélé avec le mot reçu. De cette règle, les auteurs définissent une fonction qui correspond à l'énergie du système à maximiser de la façon suivante :

$$f(\mathbf{x}) \triangleq \sum_{i=1}^{n} x_i y_i + \prod_{i=1}^{n} \prod_{j \in N(i)} x_j \tag{3.3}$$

La fonction d'inversion est alors définie comme suit :

$$\Delta_k^{(GD)} \triangleq x_k y_k + \sum_{i \in M(k)} \prod_{j \in N(i)} x_j \tag{3.4}$$

Le single-GDBF consiste à inverser le bit qui a le $\Delta_k^{(GD)}$ minimum durant chaque itération.

Pour accélérer la convergence, l'algorithme multi-GDBF inverse tous les bits dont le $\Delta_k^{(GD)}$ est inférieur à un seuil d'inversion θ au cours d'une même itération. Si aucun bit ne vérifie cette condition, le bit ayant le $\Delta_k^{(GD)}$ minimum est tout de même inversé.

Une amélioration de l'algorithme GDBF nommée Noisy Gradient Descent Bit-Flipping (NGDBF) a été présentée dans [85]. Cette variante offre de meilleures performances de décodage en introduisant de l'aléa dans le processus de décodage. Cependant cet algorithme ne sera pas considéré ici pour les raisons suivantes. La génération de bruit, la complexité calculatoire supérieure et surtout la grande sensibilité des paramètres de cet algorithme font qu'il demande beaucoup de temps et d'ajustements précis et minutieux pour chaque code LDPC considéré. Il est nécessaire d'utiliser un algorithme flexible car un grand nombre de codes LDPC est étudié. C'est pourquoi, un meilleur couple algorithme/code est nécessaire afin d'atteindre les performances de décodage attendues.

Ces algorithmes présentent une faible complexité calculatoire avec des opérations mathématiques simples en comparaison avec le SPA. Cependant, il faut vérifier que leur pouvoir de correction est suffisant pour satisfaire les besoins applicatifs. C'est pourquoi une comparaison avec les algorithmes basés sur le Min-Sum est indispensable pour vérifier la pertinence du WBF ou GDBF dans notre contexte.

3.2.2 Algorithmes de type Min-Sum

Les algorithmes de décodage basés sur l'approximation Min-Sum sont des solutions dont le pouvoir de correction est proche du BP-SPA (simplement SPA ici). Les implantations matérielles de ces algorithmes sont plus onéreuses que celles des algorithmes de type

bit-flipping mais toujours moins complexes et plus pertinentes que celles de l'algorithme SPA. Comme cela a été montré dans le chapitre 2, c'est particulièrement vrai quand ils sont utilisés avec un ordonancement par couches horizontales sur des codes de type QC-LDPC avec de grands facteurs d'expansion Z. Beaucoup de travaux ont utilisé ces algorithmes. Nous pouvons par exemple citer les travaux présentés dans [86, 87] qui utilisent des variantes du Min-Sum pour décoder la code LDPC du standard DVB-S2.

Les performances de décodage des algorithmes de type Min-Sum sont proches de celles obtenues à l'aide de l'algorithme SPA. Dans la figure 3.1, les résultats de simulation d'un code de taille N = 16384 sur un canal BI-AWGN sont rapportés. Les performances des algorithmes de type Min-Sum se situent à environ 0.2 dB de l'algorithme SPA. Cette faible pénalité fait de ces algorithmes de bons candidats pour obtenir les performances de correction souhaitées avec un faible coût de mise en œuvre. Cependant afin de valider ce postulat, il est nécessaire de comparer l'OMS et le NMS avec les algorithmes de type *bit-flipping* afin de positionner leurs performances respectives.

3.2.3 Performances des algorithmes de décodage

Les performances nécessaires à l'application sont similaires à celles du standard DVB-S2. Cependant, le décodage doit se passer de l'utilisation d'un code supplémentaire de type BCH en raison des contraintes matérielles. C'est pourquoi le couple algorithme/code LDPC sélectionné doit avant tout respecter les performances attendues puis, dans un second temps posséder la complexité calculatoire la plus faible possible afin d'atteindre un débit de 10 Gbit/s.

Les courbes de la figure 3.1 ont été obtenues en considérant une modulation BPSK sur canal AWGN avec un code LDPC de taille (16384, 4096). Les algorithmes de *bit-flipping* ont été simulés pour 100 itérations de décodage tandis que ceux basés sur le Min-Sum n'exécutent que 10 itérations. Cette différence entre les nombres d'itérations vient du fait qu'une itération d'un algorithme de type *bit-flipping* est beaucoup moins complexe qu'une itération de Min-Sum. Cependant, les algorithmes de type *bit-flipping* nécessitent un plus grand nombre d'itérations pour converger vers des mots de code. Enfin, le SPA est présenté avec 50 itérations de décodage pour définir la limite supérieur atteignable. Pour les courbes issus du standard DVB-S2, le rendement sélectionné est de 11/15 avec la concaténation d'un code LDPC et d'un code BCH pour une taille totale de trame N = 16200. Les algorithmes ont été décrits avec une représentation flottante des données.

Dans le cadre de l'étude, les courbes démontrent que les performances des algorithmes de type *bit-flipping* sont en retrait du point de vue du pouvoir de correction par rapport à

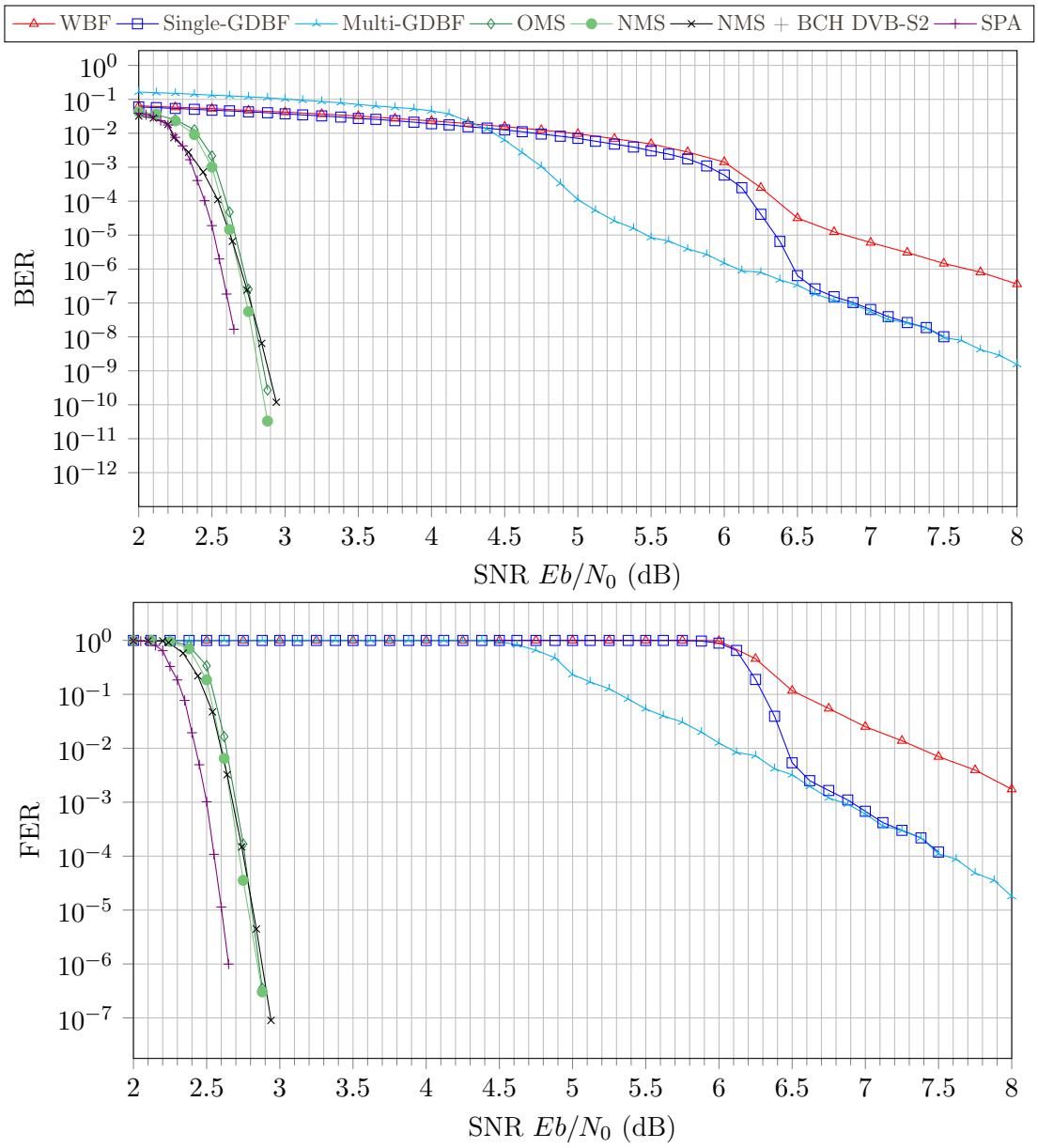

Figure 3.1 – Performances des différents algorithmes étudiés pour un code LDPC $N = 16384$ et $R = 3/4$ comparés au standard DVB-S2 sur un canal BI-AWGN

ce qui peut être obtenu en décodant à l'aide d'un algorithme de type Min-Sum. Comme les performances attendues du système doivent être similaires à celles présentées dans le standard DVB-S2 à taille et à rendement équivalents, les algorithmes de type *bit-flipping* ne sont pas des solutions acceptables. En effet, pour un FER de 10^{-4}, il y a environ 4.5 dB d'écart entre la courbe du multi-GDBF et celle de l'OMS. Ainsi, seuls les algorithmes dérivés du Min-Sum atteignent ici un pouvoir de correction satisfaisant pour l'application ciblée.

D'autres algorithmes sous-optimaux dérivés du BP-SPA comme le λ-min ($\lambda > 2$) ou le A-min* présentent des complexités calculatoires supérieures à celle du Min-Sum. Ils ne sont donc pas considérés dans le reste de l'étude car les besoins en correction d'erreurs sont déjà satisfaits par les algorithmes OMS et NMS.

Maintenant que le choix de la famille d'algorithmes de décodage a été fait, il est nécessaire de concevoir un code LDPC permettant de maintenir les performances tout en présentant des attributs facilitant l'implantation matérielle du décodeur associé.

3.3 Étude des différentes familles de codes LDPC

La première partie du chapitre a mis en avant les différentes contraintes liées au domaine applicatif. Certaines nécessitent de faire des choix importants dès l'étape de construction du code LDPC. En effet, la taille des trames doit être égale à 16k comme indiqué en début de chapitre. Cette contrainte sur la taille des trames écarte les architectures entièrement parallèles et déroulées basées sur un ordonnancement par inondation [36], présentées dans le chapitre 2. En effet, la complexité matérielle est trop élevée et l'implémentation sur circuit FPGA est difficile voir impossible à réaliser en raison de la congestion au niveau du routage. Par contre, l'utilisation d'un code QC-LDPC avec une architecture semi-parallèle basée sur un ordonnancement par couches horizontales semble opportun. Ce type d'architecture a déjà été utilisé avec succès pour le décodage de codes LDPC de grande taille [54, 88, 89]. C'est pourquoi les codes LDPC retenus seront quasi-cycliques et de taille 16k bits avec un facteur d'expansion $Z = 256$. La valeur $Z = 256$ permet de simplifier l'architecture tout en offrant un niveau de parallélisme élevé. En effet, l'utilisation de facteurs d'expansion plus grands est difficile car le pouvoir de correction d'un code LDPC diminue quand Z atteint des valeurs trop élevées au regard de N. Par ailleurs, d'autres paramètres comme la régularité ou la densité de la matrice peuvent varier afin de converger vers un bon compromis entre la complexité calculatoire et le pouvoir de correction. Afin d'identifier le meilleur candidat, trois types de construction ont été employés : IRA, ARJA et FG. Beaucoup de codes LDPC ont été construits dans le cadre de l'étude. Seuls ceux ayant un bon compromis entre performances de décodage et complexité calculatoire du décodage sont détaillés ci-dessous.

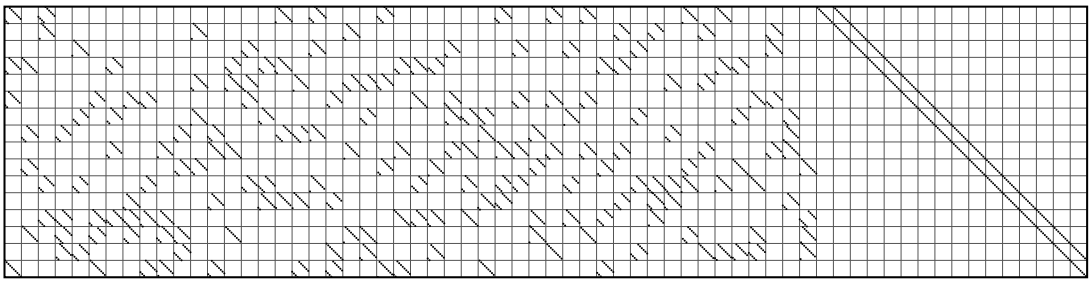

Figure 3.2 – Matrice **H** du code IRA construit avec $N = 16384$ et $R = 3/4$

3.3.1 Construction de codes LDPC

3.3.1.1 Codes LDPC Irregular Repeat-Accumulate

Les codes LDPC basés sur une construction IRA présentent de bons pouvoirs de correction et des attributs facilitant l'encodage. Cette construction est une évolution du Repeat-Accumulate (RA) [90] qui consiste initialement en une concaténation série, à travers un entrelaceur, d'un code à répétition avec un accumulateur ayant la fonction de transfert 1/(1+D). Dans le cas des codes LDPC IRA [91], le rendement de la répétition peut varier pour chaque bit d'information. Ceci permet de construire des codes avec des rendements élevés et s'approchant de la capacité du canal. Les codes LDPC IRA peuvent être systématiques et ainsi permettre un codage efficace de l'information à transmettre.

Les codes LDPC IRA sont présents dans plusieurs standards de communication actuels. C'est le cas du standard DVB-S2 [23], utilisé dans le cadre des communications par satellites, qui associe un code LDPC IRA, $N = 16200$ ou 64800, à un code BCH pour améliorer l'error floor. Cette famille de codes LDPC est aussi présente dans les standards Wifi IEEE 802.11n [25] et WiMax 802.16e [24].

La méthode de génération de codes LDPC IRA utilisée dans le cadre de notre étude est la combinaison des algorithmes de construction PEG avec la contrainte ACE. L'algorithme Progressive-Edge-Growth (PEG) [92] a pour but de générer un code LDPC en créant un graphe de Tanner point par point afin de maximiser les distances de Hamming locales. L'algorithme Approximate Cycle Extrinsic message degree (ACE) [93] permet, quant à lui, de maximiser la taille des *stopping-set*.

Un code IRA a été sélectionné pour l'étude de performances. Il possède les caractéristiques suivantes : $N = 16384$, $R = 3/4$, irrégulier et le degré moyen des nœuds variable est $d_{v,avg} = 3.48$. La forme de sa matrice **H** est donnée dans la figure 3.2.

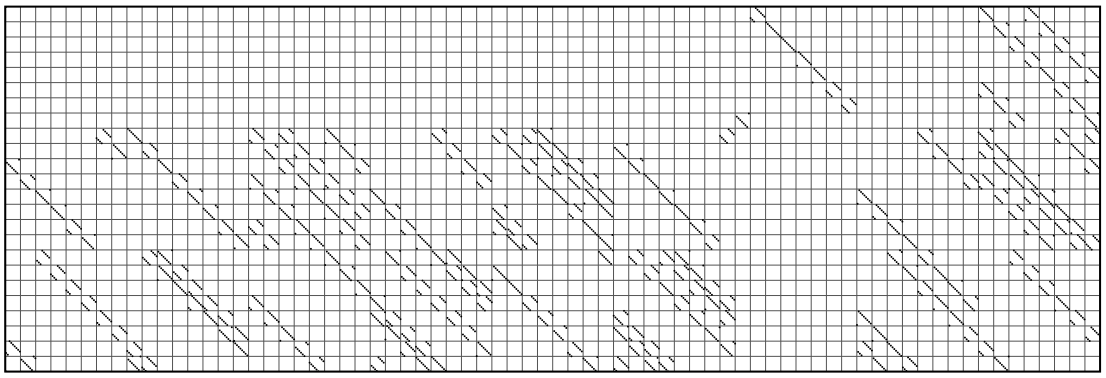

Figure 3.3 – Matrice **H** du code ARJA construit avec $N = 16384$ et $R = 3/4$

3.3.1.2 Codes LDPC Accumulate-Repeat-Jagged-Accumulate

Afin d'améliorer les performances de décodage, un accumulateur supplémentaire peut être ajouté à un code IRA afin de pré-coder une sous-partie des bits d'informations [94, 95]. Cet ajout permet principalement d'améliorer le seuil de convergence. Cependant cela introduit des nœuds de variable poinçonnés qui, certes ne demandent pas la transmission de bits supplémentaires mais impliquent un agrandissement de la matrice **H**. Cela impacte donc la complexité calculatoire du décodage. Le code LDPC obtenu est appelé Accumulate-Repeat-Accumulate (ARA) [94, 95].

Des travaux complémentaires dans [96, 97] montrent que les performances de décodage peuvent être améliorées en réduisant certaines connections dans le protographe d'un code ARA. Le protographe ainsi créé est appelé Accumulate-Repeat-"Jagged"-Accumulate (ARJA) en référence à son accumulateur qui se retrouve "déchiqueté" ("jagged" en anglais). C'est ce type de code LDPC basé sur les protographes ARJA qui a été standardisé pour les communications *near-Earth* par l'organisme CCSDS [46].

Le code ARJA sélectionné pour l'étude possède les caractéristiques suivantes : $N = 16384$ et $R = 3/4$ après poinçonnage, irrégulier, et le degré moyen des nœuds variable est $d_{v,avg} = 3.44$. La forme de sa matrice **H** est donnée dans la figure 3.3.

3.3.1.3 Codes LDPC Finite-Geometry

En complément des travaux menés sur les topologies IRA et ARJA, des travaux connexes ont démontré que les codes LDPC peuvent aussi être construits de manière algébrique par géométrie finie. Le principe de cette méthode de construction a été introduit par Kou, Lin et Fossorier [83] en 2000. Ils ont montré que la géométrie finie peut être utilisée pour créer des codes LDPC performants. Ces derniers ont un pouvoir de correction proche de

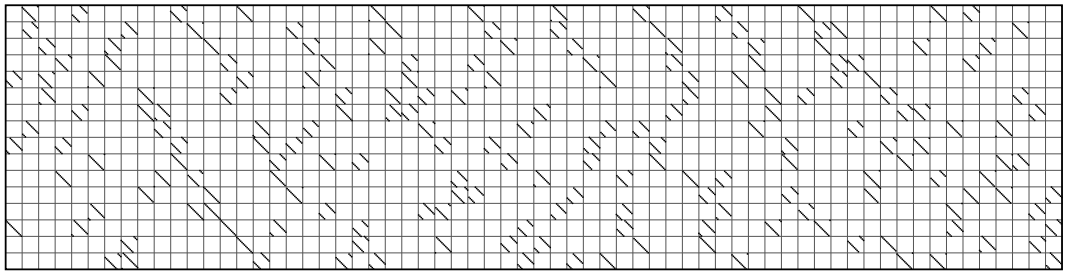

Figure 3.4 – Matrice **H** du code FG de rendement 0.75 construit avec $N = 16384$ et $R = 3/4$

Code	Type	N	Z	$d_{v,avg}$	$d_{c,avg}$	R	Régulier	Densité
IRA	IRA	16384	256	3.48	13.3	0.75	non	0.09 %
FG_1	FG	16384	256	3.00	12.0	0.75	oui	0.07 %
FG_2	FG	16384	256	2.98	14.7	0.8	non	0.09 %
FG_3	FG	16384	256	3.00	16.0	0.8125	oui	0.10 %
ARJA	ARJA	16384 [3]	256	3.44	10.3	0.75 [3]	non	0.06 %
DVB_1 [1]	IRA	16200	360	2.93	11.0	0.733	non	0.07 %
DVB_2 [2]	IRA	16200	360	2.93	11.0	0.723	non	0.07 %
DVB_3 [1]	IRA	16200	360	2.78	12.5	0.778	non	0.06 %
DVB_4 [2]	IRA	16200	360	2.78	12.5	0.767	non	0.06 %

[1] : DVB-S2 avec LDPC, [2] : DVB-S2 avec LDPC+BCH, [3] : après poinçonnage

Tableau 3.1 – Propriétés des codes LDPC étudiés

la limite théorique de Shannon lorsqu'un décodage itératif basé sur la propagation de croyance est utilisé. Ces codes sont appelés codes Finite-Geometry (FG)-LDPC.

Trois codes FG-LDPC ont été sélectionnés pour l'étude avec différents rendements : 0.75, 0.8 et 0.8125. Ils ont pour taille $N = 16384$ et leur degré de nœud variable est environ égal à 3. La structure de la matrice **H** du code de rendement 0.75 est donnée dans la figure 3.4.

3.3.2 Performances des différents codes LDPC

Le but de ces travaux étant d'obtenir un code LDPC possédant des performances au moins similaires à celles obtenues dans le standard DVB-S2 à rendement équivalent, cette sous-partie a pour objectif de comparer les performances des différents codes LDPC que nous avons conçus. Le tableau 3.1 répertorie les caractéristiques des codes LDPC étudiés.

Tous les codes LDPC ont une taille de mot de code d'environ 16k bits. Le degré des VN est compris entre 2.75 et 3.5 tandis que les rendements varient entre 0.72 et 0.82. Parmi

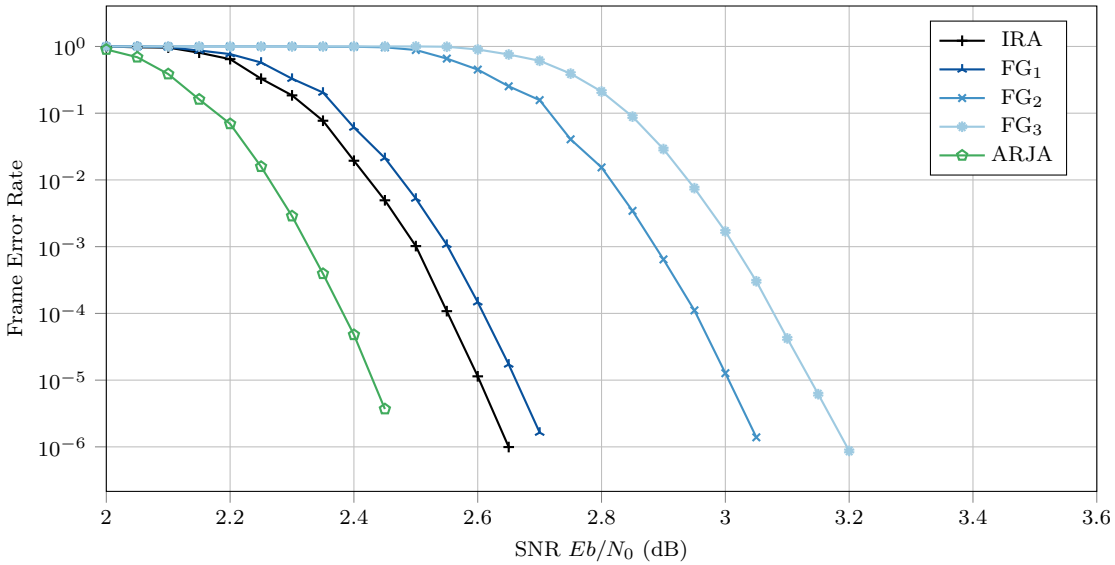

Figure 3.5 – Performances des codes LDPC considérés avec l'algorithme SPA pour 50 itérations de décodage

les candidats retenus, certains sont irréguliers. Ils permettront d'étudier l'influence de la régularité sur les coûts d'implantation matérielle du décodeur. Enfin, ces codes LDPC ont des densités (nombre de '1' dans la matrice rapporté au nombre d'éléments) variant de 0.06% à 0.1%. La densité est un facteur important car elle a un impact direct sur la complexité calculatoire du processus de décodage ainsi que sur l'empreinte mémoire du décodeur.

Les performances de décodage des codes LDPC répertoriés dans le tableau 3.1 sont présentées dans les figures 3.5 et 3.6. L'algorithme de décodage retenu pour la figure 3.5 est le SPA avec 50 itérations. Pour la figure 3.6, l'algorithme NMS ($\gamma = 0.75$) est a été appliqué avec 10 itérations de décodage. Le canal considéré est l'AWGN et la modulation est une BPSK. Les données sont en représentation flottante.

Les courbes obtenues avec l'algorithme SPA, sur la figure 3.5, montrent les performances optimales des codes LDPC construits. Comme attendu, le passage de l'algorithme SPA au NMS accompagné d'une baisse du nombre d'itérations impacte le pouvoir de correction. Le code LDPC ARJA présente le meilleur pouvoir de correction avec un gain de 0.2 dB par rapport au code LDPC IRA avec l'algorithme SPA. Cependant, ses performances sont moins bonnes que celles des codes LDPC IRA et FG_1 lorsque l'algorithme NMS est employé avec seulement 10 itérations de décodage. Ceci est principalement dû au poinçonnage qui fait que les codes LDPC ARJA nécessitent un nombre d'itérations plus important afin de converger. Sur la figure 3.6, il apparait que les codes LDPC du standard

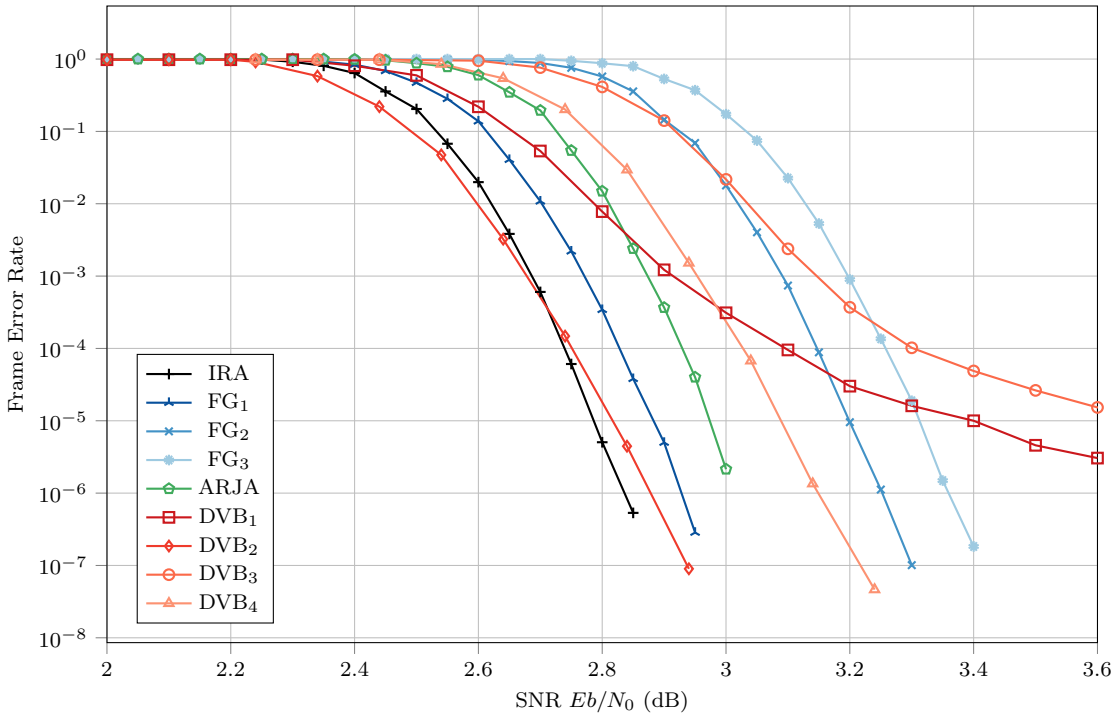

Figure 3.6 – Performances des codes LDPC considérés avec l'algorithme NMS pour 10 itérations

DVB-S2 sans la concaténation série avec le code BCH (DVB_1 et DVB_3) présentent des *error floors* trop élevés par rapport au cadre de l'étude. En revanche, la courbe DVB_2 intégrant le code BCH correspond aux performances souhaitées pour l'application. Il est notable que le code IRA dépasse les performances du DVB-S2 à partir d'un SNR de 2.7 dB et celà malgré un rendement plus élevé. Ce code IRA seul permet d'obtenir des performances similaires à celles du code LDPC du standard DVB-S2 concaténé à un code BCH. Ensuite, le code FG_1 se positionne à moins de 0.1 dB du code IRA. Le code ARJA arrive environ 0.1 dB après le code FG_1. En ce qui concerne les codes FG de rendements plus élevés, les performances en termes de correction d'erreurs ne sont pas assez élevées pour l'application cible. En effet, leur zone de convergence apparait à des SNR supérieurs à 2,5 dB.

En résumé, les codes IRA et FG_1, présentant les meilleurs niveaux de performances, nous les avons retenus pour l'implantation matérielle. Leurs différences en termes de régularité et de densité sont les principaux éléments à prendre en compte pour l'implantation matérielle du décodeur.

3.4 Implantation matérielle du décodeur LDPC

Dans les parties précédentes, les algorithmes OMS et NMS ainsi que certains codes LDPC se sont démarqués comme de bons candidats pour l'implantation matérielle d'un décodeur sur circuit FPGA. Dans cette partie, la conception du décodeur est abordée à travers la présentation de l'architecture et et l'étude des performances d'implantation sur circuit FPGA.

3.4.1 Présentation de l'architecture

Afin d'atteindre des débits élevées, il est nécessaire de paralléliser le processus de décodage. Nous nous sommes donc focalisés sur une architecture semi-parallèle afin de gérer le compromis complexité/débit. Les travaux se sont basés sur un modèle architectural de type Application Specific Instruction set Processor (ASIP) [78]. Cette architecture de décodeur implante un processus de décodage selon les algorithmes OMS et NMS avec un ordonnancement par couches horizontales. L'utilisation d'une architecture type No-Instruction-Set-Computer (NISC) ou ASIP permet d'obtenir une grande flexibilité à l'exécution en permettant le décodage de n'importe quel code QC-LDPC. Cette philosophie de conception permet d'évaluer facilement les performances d'implantations matérielles pour différents codes LDPC et ainsi de sélectionner la solution la plus pertinente.

3.4.1.1 Description de l'architecture

L'architecture qui a servi de base à notre implantation matérielle sur circuit FPGA est issue des travaux présentés dans [53, 79, 1]. Cette architecture matérielle est résumée dans la figure 3.7. Elle est générique et permet de traiter tous les types de codes LDPC (structurés ou non). Les différentes ressources matérielles (mémoires, éléments de calcul, entrelaceurs) mises en œuvre sont gérées par un contrôleur qui active les ressources nécessaires lors de chaque étape du décodage. Son fonctionnement se déroule comme suit :

- En entrée, les LLR sont stockés dans des unités mémoires. Il est alors possible durant un même cycle d'horloge d'accéder à autant de LLR qu'il y a d'unités de stockage dans l'architecture. Cependant les LLR stockés dans une même unité ne sont pas accessibles en même temps.

- Ensuite, le décodage commence et les LLR sont transmis dans un premier entrelaceur implanté sous la forme d'un réseau de Benes. Un réseau de Benes [98] est un schéma

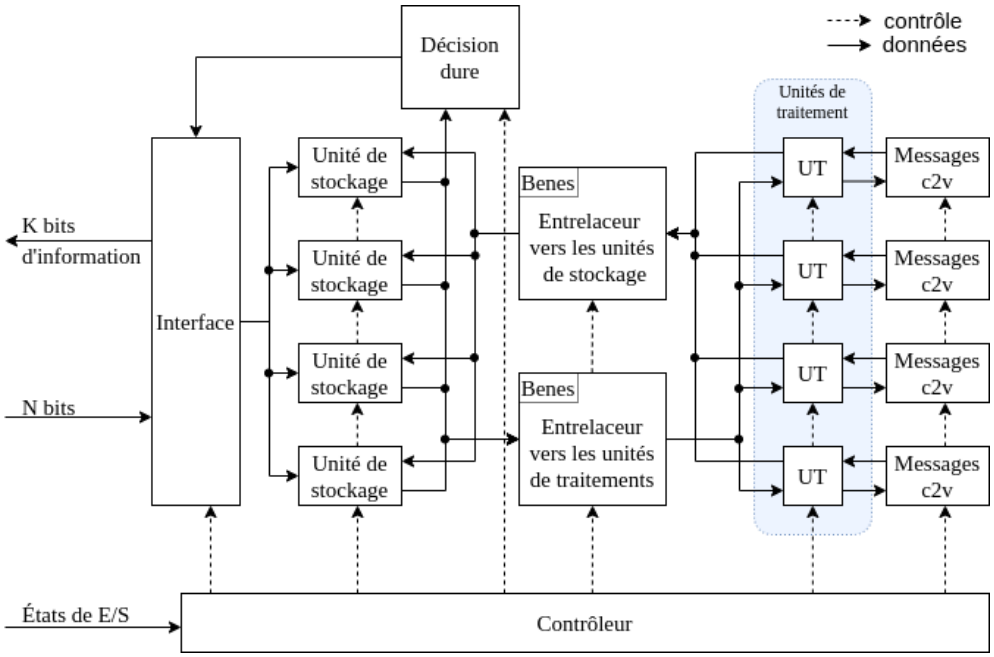

Figure 3.7 – Architecture du décodeur LDPC présenté dans [1]

de routage permettant de router une permutation arbitraire.

- Les données sont alors réordonnées de manière à être exploitées par les différentes unités de traitement (UT) selon les équations de parité du code LDPC considéré. Ces unités de traitement permettent de décoder la trame selon les algorithmes OMS ou NMS.

- Les valeurs des LLR mises à jour dans les UT sont alors transmises dans un second réseau de Benes pour rétablir l'organisation naturelle des données et ainsi pouvoir les stocker dans les unités mémoire.

- Après avoir atteint le nombre maximal d'itérations, les valeurs a posteriori sont soumises à une décision dure qui donne la valeur finale des bits d'information décodés par le décodeur. Ces derniers sont ensuite envoyés en dehors de l'architecture.

Cette architecture totalement générique permet le décodage de tous types de codes LDPC. Cette flexibilité implique un surcoût matériel qui s'avère inutile lorsque des codes QC-LDPC sont ciblés. Pour décoder efficacement les codes QC-LDPC, cette architecture a été retravaillée afin de tirer bénéfice de la structure quasi-cyclique. L'architecture résultante est présentée sur la figure 3.8. L'interface du décodeur a été optimisée pour enregistrer et renvoyer les données par paquet de Z éléments. Ceci est possible grâce à la refonte de la mémoire stockant les accumulateurs des VN. En effet, grâce à la structure quasi-cyclique

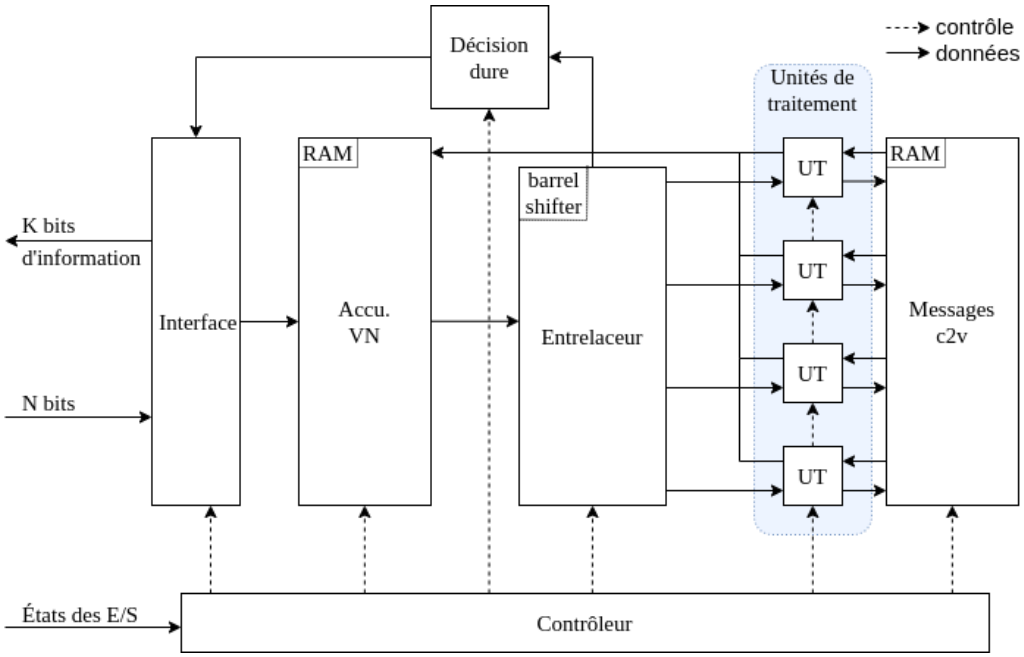

Figure 3.8 – Architecture globale du décodeur QC-LDPC

du code LDPC, une mémoire RAM unique, comme décrit dans la figure 3.9, permet de stocker l'ensemble de ces valeurs. Ensuite le processus itératif est appliqué et les LLR sont envoyés par paquets de Z dans l'entrelaceur qui est implanté sous la forme d'un *barrel shifter*. Ce *barrel shifter* remplace les réseaux de Benes et son utilisation est rendu possible par la structure des codes QC-LDPC. Une fois entrelacés, les LLR sont répartis dans les Z unités de traitement dans lesquelles sont opérées les opérations de l'OMS ou du NMS. Ces unités de traitement renvoient les valeurs mises à jour des LLR et stockent les messages CN vers VN dans la RAM associée pour la prochaine itération. Les mémoires stockant ces messages sont regroupées en une unique RAM qui les stocke par groupe de Z. Puis les LLR mis à jour sont de nouveau stockés dans leur mémoire RAM tout en gardant leur ordre entrelacé. Cet ordre est pris en compte lorsqu'un paquet de Z LLR est à nouveau envoyé dans l'entrelaceur. Ce dernier décale les données de manière à ce qu'elles arrivent dans le bon ordre dans les unités de traitement. L'utilisation d'un unique entrelaceur est possible grâce à la structure quasi-cyclique des codes QC-LDPC. Une fois toutes les itérations effectuées, les LLR repassent dans l'entrelaceur pour retrouver leur ordre naturel. Enfin, des décisions dures sont prises sur les LLR a posteriori pour obtenir les K bits d'information décodés.

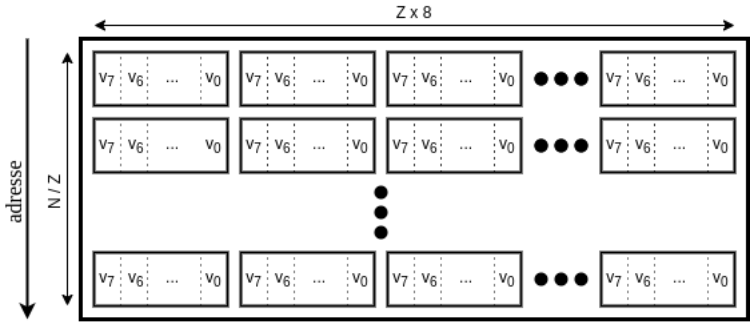

Figure 3.9 – Organisation des données dans la mémoire RAM des accumulateurs VN

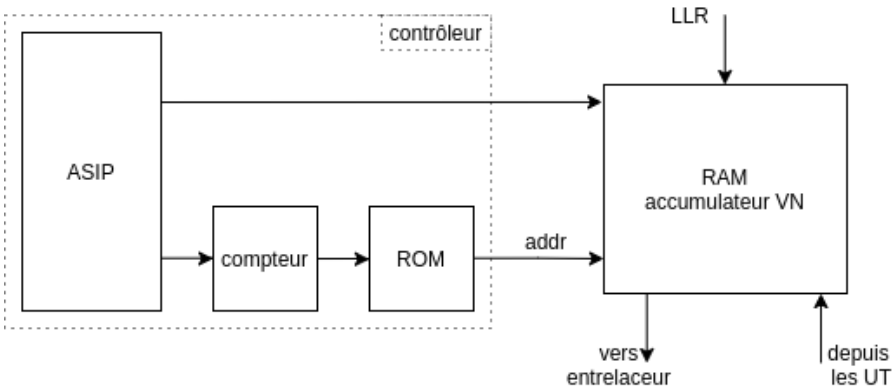

Figure 3.10 – Gestion de l'adressage de la mémoire RAM stockant les LLR dans l'architecture du décodeur

3.4.1.2 Contrôle des mémoires et de l'entrelaceur

Le contrôle de la mémoire RAM stockant les LLR se base sur l'utilisation de Read-Only Memory (ROM) où sont stockées les adresses d'accès à la mémoire dans l'ordre qui correspond au besoin du décodage. Cette mémoire ROM est pilotée par un compteur qui s'incrémente ou se réinitialise selon les instructions traitées par le contrôleur. Ce fonctionnement est détaillé dans la figure 3.10. Pour la mémoire RAM contenant les messages, l'adresse d'accès est gérée par un compteur qui se remet à zéro à chaque début d'itération.

En ce qui concerne l'entrelaceur, le contrôle de son comportement est similaire à celui de la mémoire RAM contenant les LLR. En effet, les valeurs qui définissent les rotations des données durant chaque cycle d'horloge sont stockées de manière linéaire dans une mémoire ROM. De même, cette dernière est pilotée par un compteur d'adresse contrôlé par les instructions exécutées par le contrôleur. Ce fonctionnement est illustré par la figure 3.11.

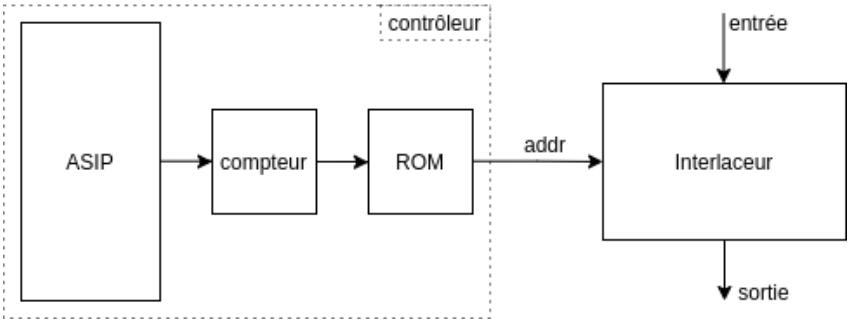

Figure 3.11 – Gestion du contrôle de l'entrelaceur

3.4.1.3 Unités de traitement (UT)

L'architecture du décodeur LDPC implante les algorithmes OMS ou NMS avec ordonnancement par couches horizontales. Ainsi, avec cet ordonnancement, l'algorithme OMS peut être décrit comme présenté dans l'algorithme 2. Cette description de l'algorithme de décodage centré sur les CNs permet de concentrer les opérations arithmétiques dans les UT.

Ainsi chaque UT se compose de deux étages distincts qui correspondent aux deux étapes élémentaires S1 et S2. La première étape (S1) est chargée du calcul des messages des VN vers les CN puis de la mise à jour des calculs de parité à travers le traitement du signe et des minima. Ensuite, la seconde étape (S2) est en charge du calcul des messages sortant des CN vers les VN et de la mise à jour des valeurs des LLR V_i.

Dans l'architecture matérielle, le traitement des étapes S1 et S2 est séquentiel en raison des dépendances de données entre les calculs. Ces deux étapes distinctes sont cependant *pipelinées* et permettent de traiter deux CN en parallèle. Pour optimiser ce traitement, les degrés de deux CN traités en parallèle doivent être identiques. Dans le cas contraire, un des étages de l'architecture sera inexploitable. La figure 3.13 montre un exemple de remplissage du *pipeline* en fonction du degré des nœuds CN consécutifs. CN_j et CN_k sont tous deux de degré 4. Quand CN_j passe à l'étape S2, CN_k entre dans l'étape S1. Comme ces nœuds ont le même degré d_c, ils terminent chacun leur traitement au même moment. Cependant, lorsque les degrés sont différents comme c'est le cas avec CN_k et CN_h dans la figure 3.13, l'étape traitant le CN avec le plus petit degré doit attendre que l'autre étage termine son exécution. Cela implique l'insertion d'un ou plusieurs états d'*idle*.

L'architecture ainsi conçue est une solution flexible et efficace pour traiter les codes QC-LDPC. De plus, une utilisation optimale des UT est atteinte pour des codes réguliers.

Algorithm 2: Horizontal-layered Offset Min-Sum

▷ Input (LLR reçu) : $\boldsymbol{y} = (y_1, y_1, ..., y_N)$
init
$t = 0$, $Y_i = \frac{2y_i}{\sigma^2}$ and $V_i = 0$ for $i \in [1, .., N]$
repeat
▷ Ordonnancement par couches horizontales
for all j ∈ [1, .., M] **do**
$C_j^{(t)} = +\infty$
▷ Stage S1
for all i ∈ N(j) **do**
▷ Calcul des messages $v2c_{ij}^{(t)}$

$$v2c_{ij} = \begin{cases} V_i, & t = 0 \\ V_i - c2v_{ji}^{(t-1)}, & \text{otherwise} \end{cases}$$

▷ Calcul de nœud de parité$C_j^{(t)}$

$$|C_j^{(t)}| = min(|v2c_{ij}|, |C_j^{(t)}|)$$
$$sign(C_j^{(t)}) = \prod_{(i)} sign(v2c_{ij})$$

end for
▷ Stage S2
for all i ∈ N(j) **do**
▷ Calcul des messages $c2v_{ji}^{(t+1)}$

$$\Delta = \begin{cases} |C_j^{(t)}|, & |C_j^{(t)}| \neq |v2c_{ij}| \\ \min\limits_{i' \in N(j) \setminus i} (|v2c_{i'j}|), & \text{otherwise} \end{cases}$$

$$|c2v_{ji}^{(t)}| = max(\Delta - \alpha, 0)$$
$$sign(c2v_{ji}^{(t)}) = sign(C_j^{(t)}) \times sign(v2c_{ij})$$

▷ Mise à jours des accumulateurs V_i

$$V_i = v2c_{ij} + c2v_{ji}^{(t+1)}$$

end for
end for
$t = t + 1$
until $t \leq t_{max}$
▷ Output (mot décodé) : $\boldsymbol{x} = (x_1, x_2, ..., x_N) \in \{0, 1\}^N$ with $x_i = sign(V_i)$

3.4.2 Flot de conception associé au décodeur

Afin de générer rapidement des décodeurs LDPC au niveau RTL, un flot de conception automatisé a été mis en place. Tout d'abord l'architecture matérielle présentée dans la sous-partie précédente et décrite en langage VHDL est générique. En effet, elle a été décrite en utilisant la clause *generic* pour définir les paramètres architecturaux qui dépendent du code QC-LDPC considéré ainsi que des choix de l'utilisateur comme la quantification des données dans les unités de traitement et dans les mémoires. À ce modèle

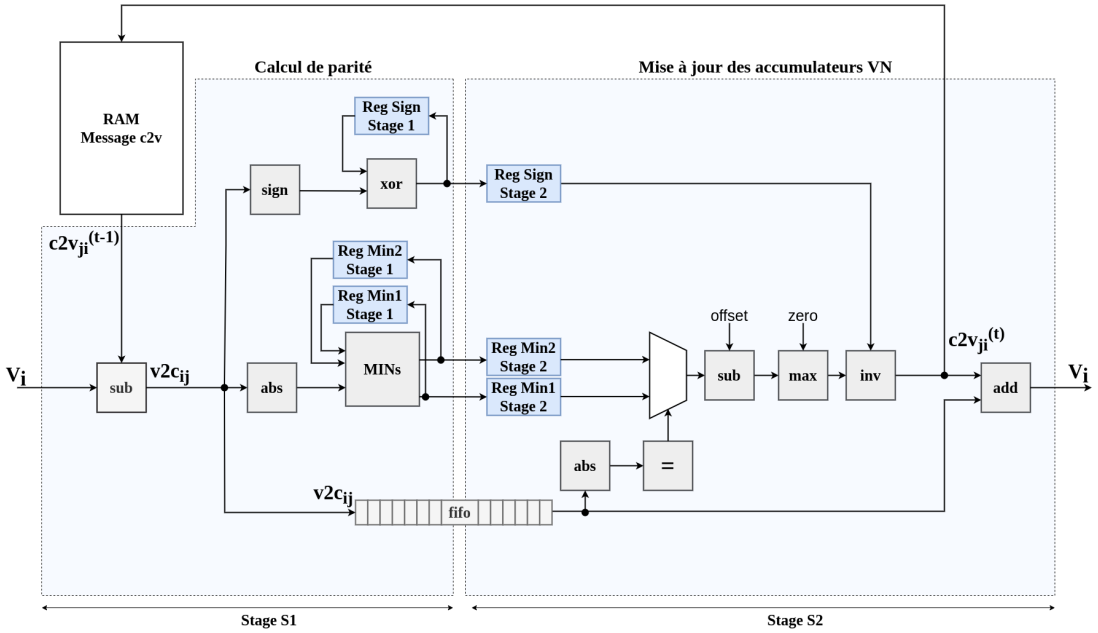

Figure 3.12 – Unité de traitement associé à l'algorithme de type Offset Min-Sum avec un ordonnancement par couches horizontales

architectural est associé un programme en Python. Ce dernier crée tous les fichiers non génériques mais indispensables au fonctionnement du décodeur. C'est le cas par exemple de l'entrelaceur, des valeurs des *generics*, de la mémoire ROM stockant les instructions et des mémoires ROM pilotant l'entrelaceur et les mémoires RAM. Via une interface graphique, l'utilisateur sélectionne le fichier contenant la matrice **H** du code QC-LDPC, le nombre d'itérations, la quantification des données et la stratégie d'ordonnancement afin de configurer le décodeur LDPC. L'architecture RTL spécifique est ainsi générée. Elle peut ensuite être utilisée en tant qu'IP. Ce flot de conception est récapitulé dans la figure 3.14.

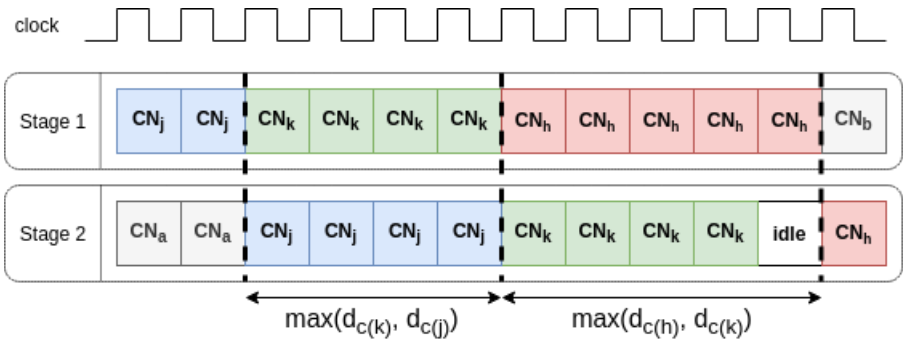

Figure 3.13 – Ordre d'exécution des CN dans les unités de traitement

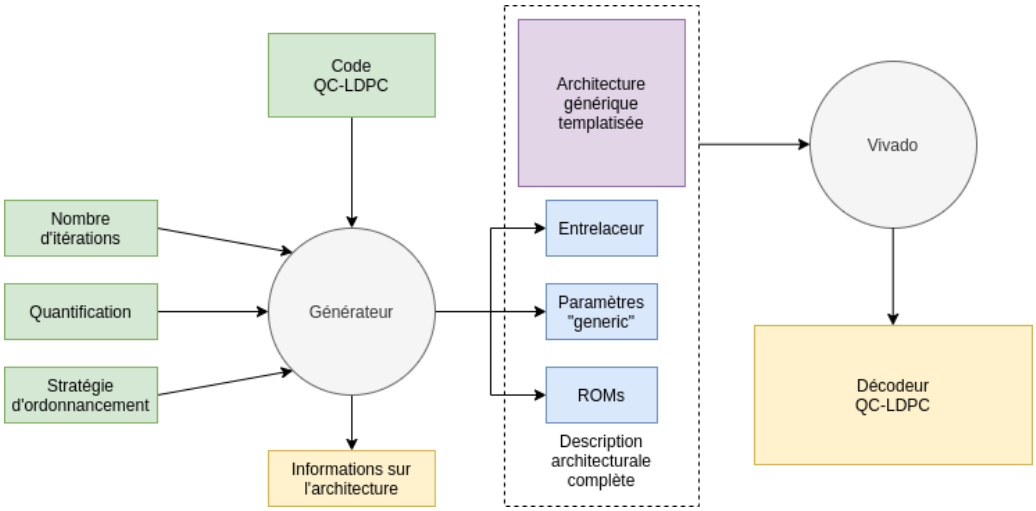

Figure 3.14 – Flot de conception de génération de décodeurs LDPC

L'ordonnancement des différents calculs de CN est une étape importante afin d'assigner efficacement les différents blocs du décodeur. Le générateur gère trois cas possibles :

1. Dans le meilleur cas possible, la matrice **H** est telle qu'il est possible de trouver un ordre d'exécution des CNs qui ne génère aucun conflit d'accès mémoire. C'est ce cas qui est illustré par la figure 3.13. Les différents CN s'enchaînent naturellement les uns après les autres.

2. Si cela n'est pas possible, l'utilisateur a la possibilité de laisser des cycles creux (*idle*) pendant lesquels les unités de traitement ne fonctionnent pas. Ces cycles supplémentaires permettent d'éviter les conflits d'accès mémoire. Ainsi, il faut attendre que des données des accumulateurs VN soient disponibles pour effectuer les calculs d'un CN complet.

3. La dernière solution consiste à traiter plusieurs trames en parallèle afin que des données soient toujours disponibles pour les unités de traitement. Cette solution est plus coûteuse au niveau de la mémoire mais permet d'atteindre un taux d'utilisation des UT proche ou égal à 100%. Cependant, sur cible FPGA, les mémoires BRAM disponibles ne sont pas forcément totalement occupées avec une trame. Le traitement simultané de plusieurs trames peut, dans certains cas, ne pas augmenter la complexité mémoire. Par exemple, pour n'importe quel code QC-LDPC régulier, le décodage simultané de trois trames permet d'obtenir un taux d'utilisation optimal.

Une fois l'architecture RTL générée, le programme en Python fournit quelques informations à l'utilisateur. Il indique, par exemple, le débit estimé ainsi que les tailles des mémoires

ROM générées et celles des mémoires RAM instanciées dans l'architecture.

Ce flot de conception permet de générer rapidement une architecture de décodeur dédiée à un code QC-LDPC spécifique. Cela rend possible une exploration matérielle rapide pour chacune des différentes solutions issues de la partie 3.3.

3.4.3 Performances des décodeurs

Afin d'évaluer les performances matérielles des décodeurs utilisant les codes QC-LDPC proposés dans la section 3.3, des expérimentations ont été réalisées. Le but est d'une part de valider les performances en termes de correction d'erreurs lorsqu'un format en virgule fixe est utilisé, et d'autre part, de garantir que le décodeur matériel conçu permet d'atteindre le débit souhaité sur le circuit FPGA de l'étude.

3.4.3.1 Impact de la représentation des données en virgule fixe

Dans un premier temps, il est important de quantifier l'impact d'une représentation des données en virgule fixe sur les performances de décodage.

L'impact de la quantification des données fluctue selon l'algorithme et la structure des codes LDPC employés. Tout d'abord, les paramètres des algorithmes OMS et NMS, soient l'offset α et le facteur de normalisation γ, sont aussi quantifiés dans une implémentation à virgule fixe. Ceci limite le nombre de valeurs possibles pour ces paramètres. Les liens optiques terrestres fournissent en entrée du décodeur des données quantifiées sur 6 bits dont 3 pour la partie décimale. Par ailleurs, le paramètre α de l'algorithme OMS étant inférieurs à 1, seuls trois bits sont utiles pour le représenter dans un format à virgule fixe. La figure 3.15 récapitule les différences de performances entre les algorithmes NMS et OMS en représentation flottante et en représentation fixe. Le code LDPC FG_1 de rendement 3/4 est utilisé en considérant 8 bits pour les accumulateurs des VN et 6 bits pour les messages. La valeur de l'offset de l'OMS est fixé à $\alpha = 0.125$ tandis que le facteur de normalisation du NMS est fixé à $\gamma = 0.75$.

Il apparaît que le passage en virgule fixe des décodeurs dégrade les performances du décodeur NMS de 0.2 dB. En revanche, dans le cas du décodage avec l'algorithme OMS, la dégradation est de moins de 0.05 dB. Les deux algorithmes ayant des complexités calculatoires similaires, l'algorithme OMS se démarque par ses performances en virgule fixe pour le code LDPC retenu.

L'influence de la quantification est différente selon le code LDPC. La figure 3.16 présente

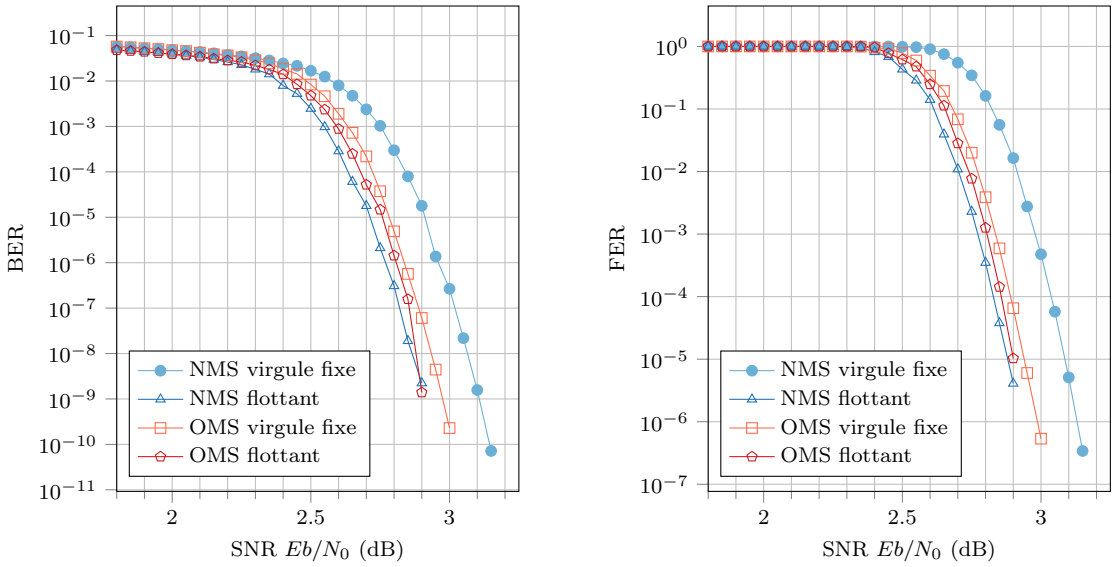

Figure 3.15 – Comparatif des performances entre les algorithmes OMS et NMS en fonction du format de quantification des données pour le code LDPC FG_1

les performances de l'algorithme OMS pour différents formats de quantification et pour les codes FG et IRA sélectionnés lors de la construction des codes LDPC. Le format de quantification est noté $Q_{v,m}$ avec v bits pour les accumulateurs et m bits pour les messages. Le nombre de bits pour la représentation des entrées souples est constant et fixé à 6 bits. La quantification des entrées souples est uniforme. Pour le code LDPC FG_1, les quantifications $Q_{8,6}$ et $Q_{7,5}$ ne font pas apparaitre d'*error floor* lorsque le BER est supérieur à 10^{-10}. Les besoins du cadre applicatif sont alors respectés. Quant au code LDPC IRA, l'*error floor* apparait dès que le BER atteint 10^{-9} pour une quantification est $Q_{7,5}$. Seul le format $Q_{8,6}$ est satisfaisant pour le code LDPC basé sur une construction IRA.

3.4.3.2 Performances et coûts matériels

Une fois les formats de quantification et leur impact sur les performances de décodage validés, il est important de s'assurer que l'implantation matérielle des décodeurs LDPC proposés présente un coût matériel compatible avec la contrainte de débit de 10 Gbit/s sur le FPGA xczu9eg-2ffvb1156e retenu dans le contexte de l'étude.

Pour cela, nous avons évalué les performances matérielles des décodeurs. Le tableau 3.2 résume le coût matériel et les performances en termes de débit et de latence de décodeurs LDPC obtenus pour différents formats de quantification. Les résultats présentés ont été obtenus sur Vivado version 2018.2 après synthèse, placement et routage avec une

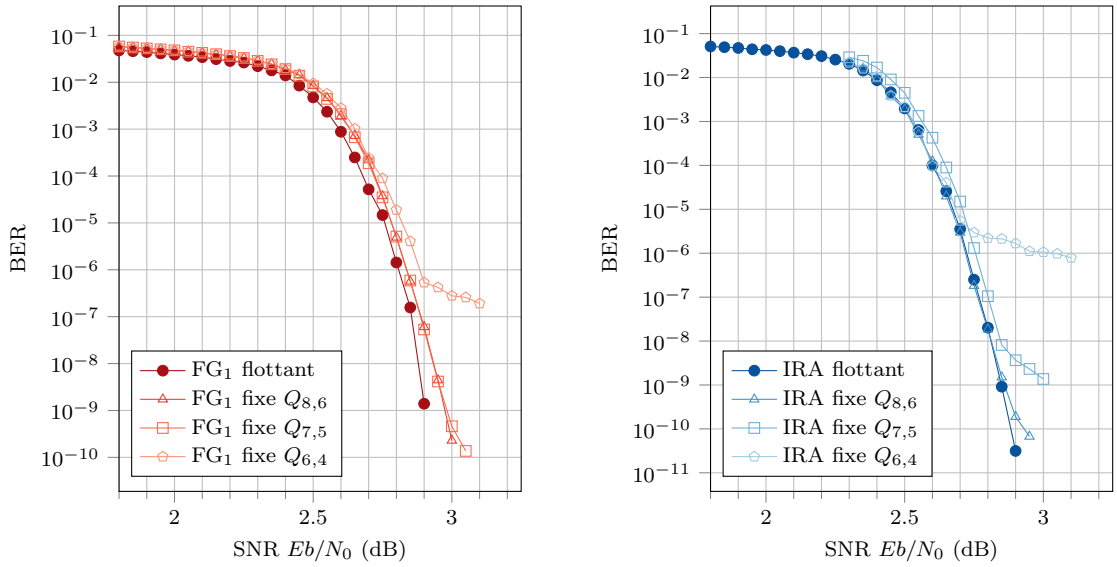

Figure 3.16 – Comparatif des performances en virgule fixe pour différents codes LDPC et différents formats de quantification avec l'algorithme OMS

Code LDPC	Quantif. ($Q_{v,m}$)	Complexité matérielle LUTs	FFs	RAM18K	Performances d'un décodeur Débit	Latence	Débit maximum sur xczu9eg-2ffvb1156e FPGA	
IRA	$(8,6)$	49121 17.9%	26903 4.9%	50 5.5%	1.64 Gbit/s	10.96 μs	6.6 Gbit/s	(4 décodeurs)
FG	$(8,6)$	49121 17.9%	26903 4.9%	50 5.5%	2.13 Gbit/s	8.44 μs	8.5 Gbit/s	(4 décodeurs)
FG	$(7,5)$	38913 14.2%	23483 4.3%	43 4.7%	2.13 Gbit/s	8.44 μs	10.6 Gbit/s	(5 décodeurs)

Tableau 3.2 – Performances des décodeurs matériels pour les deux codes LDPC retenus sur cible Zynq Ultrascale + avec $F = 250$ MHz

fréquence de fonctionnement contrainte à 250 MHz. Ces résultats ont été publiés dans le cadre de la conférence ISTC 2018 [34].

Pour atteindre ce débit de 10 Gbit/s, une quantification $Q_{7,5}$ est nécessaire afin de pouvoir instancier 5 décodeurs LDPC dans le FPGA. Le décodeur final sélectionné est donc associé au code LDPC FG$_1$ avec $N = 16384$ et $R = 3/4$ avec un format de quantification $Q_{7,5}$. L'irrégularité et la plus grande densité du code IRA font que le débit atteint par le décodeur pour ce code LDPC est inférieur à celui obtenu par le décodeur avec le code FG. De plus, la quantification $Q_{7,5}$ avec le code IRA s'est avérée insuffisante pour atteindre un *error floor* satisfaisant. En résumé, le code LDPC IRA ne permet pas d'atteindre les 10 Gbit/s sur le FPGA ciblé dans le cadre de l'étude.

Un décodeur LDPC décodant le code FG$_1$ utilise environ 15% du circuit FPGA en termes

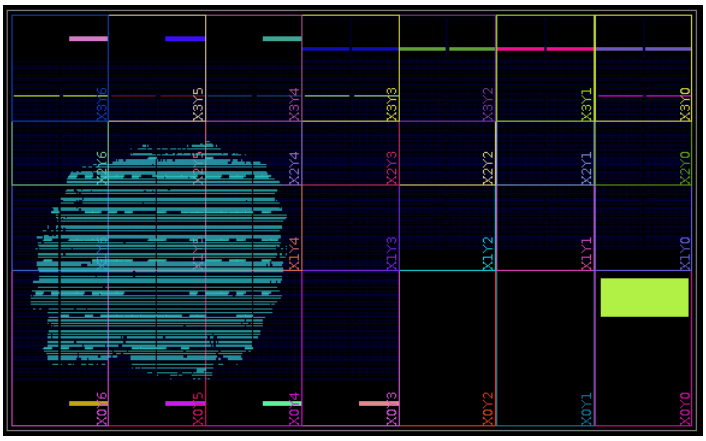

Figure 3.17 – Floor-plan du circuit FPGA pour un décodeur LDPC

de LUTs. Il s'agit des ressources du circuit FPGA qui sont les plus utilisées. Afin de maintenir une fréquence de fonctionnement élevée, le FPGA ne doit pas être occupé à plus de 80% pour faciliter, voir simplement permettre, le placement et le routage. Le nombre de décodeurs matériels est limité à 5. Nous remarquons que la durée du chemin critique n'est pas augmentée et que la fréquence de fonctionnement reste élevée avec 5 instances de décodeur. Le taux d'utilisation du FPGA avoisine alors 75% des LUTs. Les figures 3.17 et 3.18 montrent les *floor-plans* du FPGA dans différentes configurations. Cela permet de visualiser comment l'occupation du circuit FPGA est impactée (affichage en couleur cyan sur les figures).

La fréquence de fonctionnement du FPGA peut atteindre des valeurs plus élevées, ce qui augmente le débit de façon proportionnelle. En effet, grâce à l'utilisation de couches de *pipeline* supplémentaires aux travaux présentés dans [34], et notamment dans le *barrel shifter*, le décodeur peut fonctionner à plus de 250 MHz sur le FPGA de l'étude. Le décodeur peut monter à des fréquences encore plus élevées sur des FPGA récents. Cependant cette optimisation a un coût matériel en termes de bascule Flip-Flops (FF). L'évolution des performances de l'architecture, depuis sa forme initiale (NISC [1]) jusqu'à sa forme optimisée actuelle avec les tranches de pipeline dans l'entrelaceur, la suppression des 'reset' inutiles et la simplification de l'entrelaceur est mise en évidence dans le tableau 3.3.

La spécialisation de l'architecture du NISC pour les codes QC-LDPC (NISC -> ISTC) a permis de diviser par 2 le nombre de LUTs utilisées et par 5 le nombre de BRAMs. La diminution des LUTs vient de l'utilisation d'un unique *barrel shifter* à la place de deux réseaux de Benes et de la simplification du contrôleur des mémoires. Un réseau de Benes de taille 256 avec des entrées sur 8 bits dans l'architecture présentée dans [1]

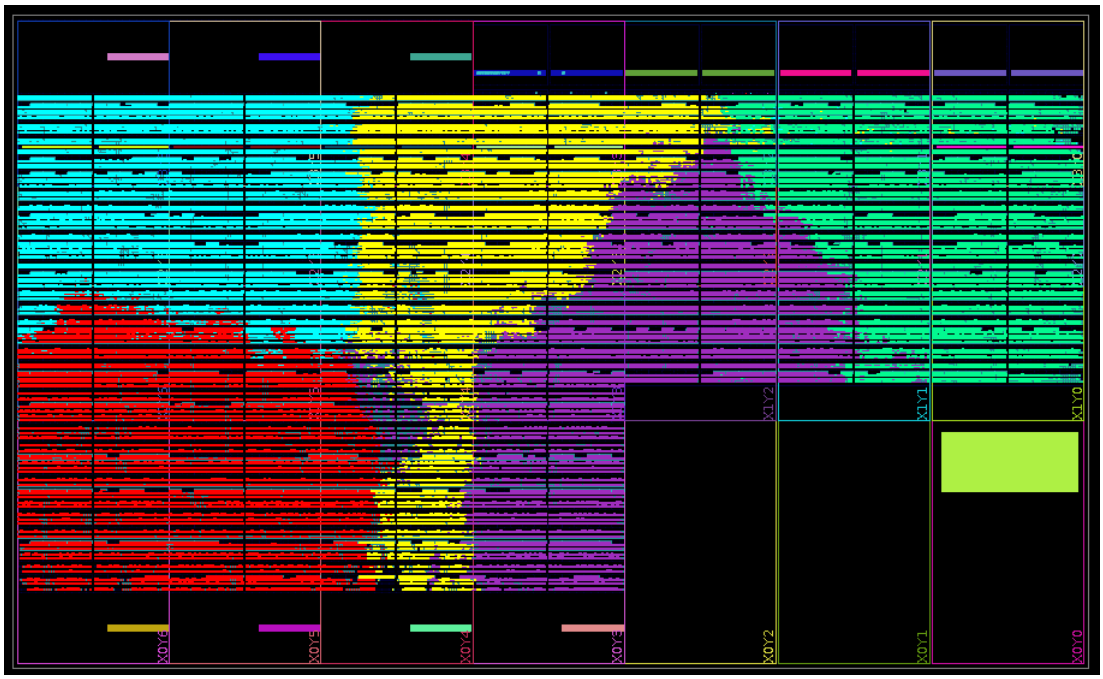

Figure 3.18 – Floor-plan du circuit FPGA pour 5 décodeurs LDPC

utilisent environ 10 000 LUTs avec 15 000 LUTs et 24 BRAMs supplémentaires pour le contrôle des entrelaceurs. En comparaison, l'utilisation d'un unique *barell shifter* consomme 8 000 LUTs. Il permet également la diminution du coût du contrôleur qui utilise alors uniquement des FFs à la place des BRAMs et 16 LUTs au lieu des 15 000 avec les Benes. La simplification des mémoires grâce à la structure quasi-cyclique permet aussi un gain significatif d'environ 20 000 LUTs pour leur contrôle. La diminution du nombre de BRAMs vient principalement du regroupement des unités de stockage des accumulateurs et des messages implantés dans différents BRAMs en une unique unité mettant pleinement à profit la capacité des BRAMs. Le chemin critique étant situé dans l'entrelaceur, la simplification de ce dernier a permis une augmentation de la fréquence maximum de fonctionnement qui est passée de 158 MHz à 252 MHz.

Les optimisations qui ont été faites après la valorisation [34] et l'ajout de tranches de pipeline dans l'entrelaceur a permis d'atteindre une fréquence de fonctionnement de 274 MHz, de diminuer de 20% le nombre de LUTs en échange d'une augmentation d'environ 3500 FFs. Il est alors possible d'assigner un 5ème décodeur manipulant des données au format $Q_{8,6}$ dans le FPGA de l'étude pour atteindre plus de 10 Gbit/s (11.7 Gbit/s). Le débit cible a donc été atteint avec une quantification $Q_{8,6}$ au lieu de $Q_{7,5}$. Cette amélioration a permis de limiter l'apparition de planchers d'erreurs.

La répartition des ressources LUTs et FFs utilisées par un décodeur dans sa version

Archi.	Coût d'un décodeur			Performances d'un décodeur			Performances max sur le Zynq US+	
	LUTs	FFs	RAM18K	Débit	Latence	Fmax	Débit	# dec.
NISC [1]	96078 35.0 %	28949 5.3 %	267 29.3 %	1.34 Gbit/s	13.46 μs	158 MHz	2.67 Gbit/s	2
ISTC [34]	49121 17.9 %	26903 4.9 %	50 5.5 %	2.13 Gbit/s	8.44 μs	252 MHz	8.5 Gbit/s	4
Opti.	40416 14.7 %	30385 5.5 %	50 5.5 %	2.34 Gbit/s	7.71 μs	274 MHz	11.7 Gbit/s	5

Tableau 3.3 – Évolution des performances d'implantation des décodeurs matériels sur une cible Zynq Ultrascale+ avec le code FG_1 en $Q_{8,6}$ pour 10 itérations de décodage

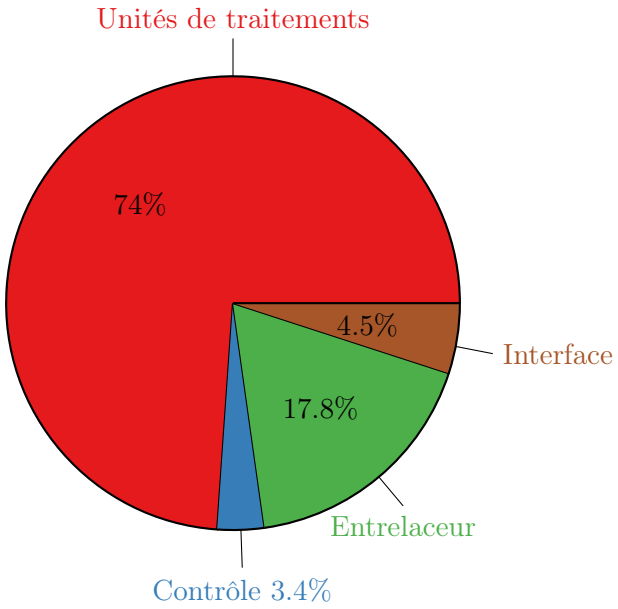

Figure 3.19 – Répartition des LUTs utilisées par le décodeur avec le code FG_1

définitive est présentée dans les figures 3.19 et 3.20. Il apparaît que les unités de traitement utilisent la majorité des ressources matérielles. Les mémoires de type BRAM sont utilisées pour la mémoire des accumulateurs des VN et la mémoire des messages.

L'intérêt de concevoir un code LDPC avec des propriétés permettant une implantation matérielle efficace est confortée par la comparaison des résultats présentés dans le tableau 3.4. Si nous considérons les codes LDPC présents dans les standards du domaine spatial à partir de notre architecture matérielle, des débits moins élevés sont atteints. Dans le tableau 3.4, des décodeurs issus des codes LDPC standardisés sont comparés avec celui issu du code LDPC FG_1. Il est possible d'instancier seulement 3 décodeurs issus des codes LDPC DVB-S2X. Le débit maximal obtenu sur le FPGA cible est plus faible avec ces codes LDPC car le nombre de messages échangés est plus élevé. Les résultats présentés dans le tableau sont donnés pour le FPGA xczu9eg-3ffvb1156e.

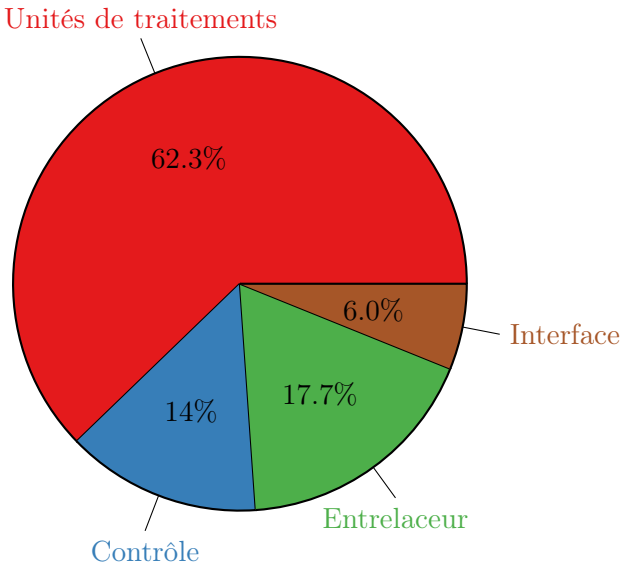

Figure 3.20 – Répartition des FFs utilisées par le décodeur avec le code FG_1

Code LDPC	# dec.	Occup. FPGA	Fréquence maximale	Débit	Latence	Puissance
(16384, 4096) FG_1	1	15%	274 MHz	2.34 Gbit/s	7.7 μs	3.29 W
(16384, 4096) FG_1	5	75%	274 MHz	11.65 Gbit/s	7.7 μs	13.80 W
(64800, 29160) DVB-S2X	1	22%	271 MHz	1.36 Gbit/s	47.5 μs	4.74 W
(64800, 29160) DVB-S2X	3	68%	270 MHz	4.08 Gbit/s	47.7 μs	12.92 W
(22528, 6144) CCSDS	1	31%	273 MHz	4.02 Gbit/s	6.1 μs	3.63 W
(22528, 6144) CCSDS	2	61%	273 MHz	8.04 Gbit/s	6.1 μs	6.62 W

Tableau 3.4 – Résultats d'implantation matérielle pour FG_1 comparés à des codes LDPC standardisés pour des applications spatiales

Le schéma de codage canal proposé, associé à l'implantation matérielle du décodeur LDPC, permet de satisfaire tous les besoins applicatifs énoncés en début de chapitre. Les performances en termes de pouvoir de correction d'erreurs ont été validées. Le débit attendu est obtenu sur la cible FPGA sélectionnée et la latence du décodeur est inférieure à 10 μs.

3.5 Comparaison avec l'état de l'art

Dans ce chapitre, nous avons présenté une solution pour le lien descendant d'une communication numérique optique par satellite. Il est intéressant de comparer la solution proposée avec d'autres solutions présentes dans la littérature.

Le tableau 3.5 regroupe les performances des différentes solutions. Les travaux proposés

	DVB-S2 [99]	Notre solution sur FPGA Intel	CCSDS [100]	Notre solution sur FPGA Xilinx
N	64800	16384	8176	16384
R	3/4	3/4	7/8	3/4
FPGA	Stratix 5 5sgxea7	Stratix 5 5sgxea7	Virtex 5 xc5vlx330	Zynq xczu9eg
Algorithme	3-min	OMS	NMS	OMS
Fréquence	250 MHz	150 MHz	250 MHz	250 MHz
Nombre d'itérations	35(max)	10	14	10
Format de quantification	8 bits VN 6 bits MSG	8 bits VN 6 bits MSG	6 bits	8 bits VN 6 bits MSG
# LUTs (Xilinx) # ALMs (Intel)	63694	40954	56778	40416
# FFs	75372	39713	86942	30385
Mémoires	3 Mbits	2.86 Mbits	132 BRAM	50 BRAM
Débit	1056 Mbit/s	1278 Mbit/s	2 Gbit/s	2.34 Gbit/s

Tableau 3.5 – Tableau comparatif des travaux avec la littérature

dans [99] détaillent un décodeur DVB-S2. L'architecture proposée atteint un débit de 1056 Mbit/s pour une complexité supérieure au décodeur présenté dans ce chapitre. L'algorithme employé est le *3-min* qui correspond à l'algorithme λ-min avec $\lambda = 3$. Comme vu dans le chapitre 2, la complexité calculatoire est supérieure à celle du Min-Sum, ce qui explique la différence d'ALMs utilisés sur FPGA Intel. Les performances matérielles sont données sur une cible Stratix 5 d'Intel. Dans le but de présenter une comparaison équitable, nous avons aussi évalué les performances matérielles de notre décodeur avec le code FG_1 sur la même cible Stratix 5 d'Intel. Pour les FPGA d'Intel, les opérations logiques se font dans les Adaptive Logic Module (ALM). La fréquence maximale de fonctionnement du décodeur est réduite sur cette cible, ce qui diminue le débit.

Ensuite, les performances d'un décodeur pour le standard CCSDS sont détaillées dans [100]. Le décodeur atteint un débit de 2 Gbit/s. En revanche, son empreinte mémoire est bien supérieure en termes de BRAM et de Flip-Flops.

Enfin, les travaux récents sont de plus en plus souvent présentés pour des cibles de type ASIC. Il est donc difficile de faire une comparaison directe avec nos travaux.

Il ressort de cette comparaison que le décodeur proposé atteint un débit élevé et une complexité matérielle relativement faible par rapport aux autres décodeurs décrits dans la littérature.

Solution	Code LDPC	# cœurs	Fréquence	Débit	Latence	Puissance	Énergie
Matérielle	(16384, 4096)	5	274 MHz	11.65 Gbit/s	7.7 μs	13.80 W	1.19 nJ/bit
	(64800, 29160)	3	270 MHz	4.08 Gbit/s	47.7 μs	12.92 W	3.2 nJ/bit
	(22528, 6144)	2	273 MHz	8.04 Gbit/s	6.1 μs	6.62 W	0.83 nJ/bit
Logicielle	(16384, 4096)	40	2194 MHz	11.25 Gbit/s	60 μs	343 W	30.49 nJ/bit
	(64800, 29160)	40	2194 MHz	7.60 Gbit/s	345 μs	370 W	48.69 nJ/bit
	(22528, 6144)	40	2194 MHz	7.55 Gbit/s	122 μs	345 W	45.70 nJ/bit

Tableau 3.6 – Comparaison des solutions logicielles et matérielles

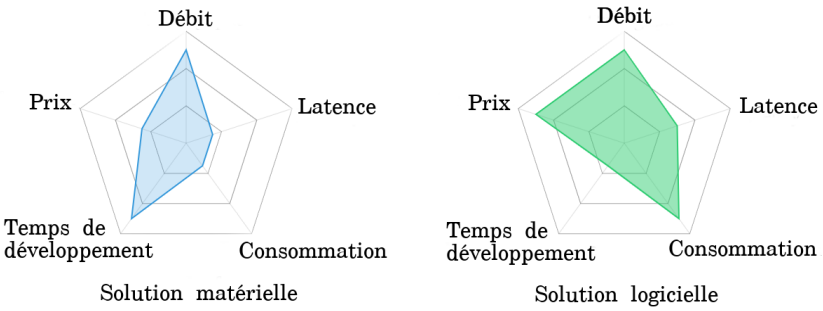

Figure 3.21 – Comparaison des caractéristiques des approches matérielles et logicielles

Il est aussi possible d'implémenter le décodeur du récepteur sur une cible CPU. Nous parlons alors de SDR (Software Defined Radio). En effet, le décodeur se trouvant dans la station de base sur Terre, une implémentation logicielle est envisageable. Une comparaison entre les solutions logicielles et matérielles est proposée dans le tableau 3.6. La cible CPU est un Intel Xeon Gold 6148 et le FPGA est le Xilinx Ultrascale+ xczu9eg-3ffvb1156e. Le prix de ces cibles est respectivement d'environ 7500 $ et 5200 $.

Il est à noter que pour atteindre 11.2 Gbit/s sur le serveur de calcul avec 40 cœurs de processeur actifs, la puissance consommée est de 343 W avec le code logiciel issu de [51]. Pour l'implantation sur FPGA, 13.8 W sont consommés pour un débit de 11.65 Gbit/s avec le même code LDPC FG_1. Cela correspond à un facteur 25 en termes d'énergie consommée par bit décodé. Nous pouvons aussi noter que la solution matérielle propose des latences 7x à 20x inférieures à celles obtenues avec la solution logicielle. La figure 3.21 résume les caractéristiques des différentes implémentations. Par exemple, il y apparait que le temps de développement sur FPGA est plus long que pour un développement logiciel. Cette étude comparative a été l'objet d'une publication lors de la conférence ICECS 2020 [75].

3.6 Conclusion

Dans le but de fiabiliser au maximum le lien descendant d'une communication optique par satellite, l'utilisation d'un schéma de codage canal est nécessaire. La première partie du chapitre présente les problématiques liées au domaine applicatif. Cette partie permet de mettre en exergue les contraintes qui doivent être satisfaites par le décodeur LDPC.

Dans la deuxième partie du chapitre, une étude de différents algorithmes de décodage LDPC a été réalisée. Tout d'abord, les algorithmes de *bit-flipping* ont été mis en avant pour leur faible complexité calculatoire. Cependant leur pouvoir en correction d'erreurs est trop faible pour satisfaire les besoins du domaine applicatif. C'est pourquoi, l'algorithme Min-Sum et ses variantes NMS et OMS ont été sélectionnés en raison de leur fort pouvoir de correction associé à une complexité calculatoire conforme aux attentes.

Dans un troisième temps, différentes techniques de construction de codes LDPC ont été considérées afin de construire un code qui permet de faciliter l'implantation matérielle d'un décodeur OMS avec un ordonnancement par couches horizontales. Ainsi des codes QC-LDPC, permettant l'utilisation d'architectures semi-parallèles, ont été comparés et deux codes ont été retenus : un code IRA, irrégulier, et un code FG, régulier.

La quatrième partie détaille les différents aspects de l'architecture conçue à partir des choix et contraintes mis en avant dans les parties précédentes. Le décodeur final permet d'atteindre le débit de 10 Gbit/s pour le lien descendant tout en atteignant les performances souhaitées en termes de pouvoir de correction d'erreurs.

La dernière partie propose une comparaison du décodeur matériel développé avec les décodeurs présents dans la littérature qui se basent sur les standards de communications numériques spatiales. Une comparaison entre notre solution matérielle et une implémentation sur CPU a ensuite été proposée.

Les travaux présentés dans ce chapitre ont été valorisés à travers une publication à la conférence ISTC 2018 à Hong Kong [34], une publication à la conférence ICECS 2020 à Glasgow [75] ainsi que deux présentations lors des journées "Codes correcteurs d'erreurs pour les communications spatiales" du COMET (Comité d'experts du CNES) à Toulouse en 2019 et "Codage, modulation et traitement du signal pour les communications optiques" du GdR ISIS (Groupement de Recherche Information, Signal, Image et viSion)à Paris en 2019.

Le décodeur proposé est générique et supporte tous types de codes QC-LDPC. Cependant, il est actuellement nécessaire de faire la synthèse, le placement et le routage à chaque

changement de code LDPC. Un axe de travail serait de remplacer les différentes mémoires ROM par des mémoires de type RAM dont les données peuvent être modifiées à la volée en fonction de l'évolution du contexte. Par exemple, il peut être intéressant de changer de code LDPC pour augmenter le rendement lorsque les conditions du canal font que le SNR croît et que les besoins en correction d'erreurs sont moins importants.

4 Étude du lien montant

Ce quatrième chapitre a pour but de présenter les travaux effectués au cours de l'étude du lien montant pour des communications spatiales optiques. Dans un premier temps, les spécifications liées à cette nouvelle application sont présentées. En effet, dans ce cadre applicatif, le décodeur se trouve intégré dans un FPGA présent à l'intérieur du satellite. Cela induit des contraintes sévères, notamment au niveau du décodage qui doit se faire à partir d'entrées dures à cause de contraintes technologiques. Dans ce contexte, le choix de l'algorithme de décodage a été réalisé après une étude comparative des algorithmes à entrées dures présents dans l'état de l'art. Une nouvelle formulation de l'algorithme Gallager E permettant une implantation matérielle efficace est alors proposée. Puis une architecture de décodage basée sur cet algorithme est détaillée. Enfin les performances en termes de débit, de complexité et de consommation d'énergie sont analysées et comparées avec l'état de l'art pour démontrer l'intérêt de la solution architecturale retenue.

4.1 Contexte

4.1.1 Domaine applicatif

Dans le chapitre précédent, l'étude du schéma de codage canal et de l'implantation matérielle du décodeur associé a été menée pour un lien optique descendant pour des communications numériques par satellite. Dans ce chapitre 4, les travaux portent sur le lien montant. Cela implique que le décodeur doit être embarqué dans un circuit FPGA présent dans un satellite. Ce positionnement du décodeur induit de fortes contraintes en matière de complexité matérielle et de consommation d'énergie.

L'intérêt des liens optiques pour les communications numériques Terre-Satellite a déjà été présenté dans les chapitres précédents. Le domaine applicatif envisagé pour ce lien montant est un *feeder* optique. Ce dernier doit, là encore, supporter un débit de 10 Gbit/s.

La figure 4.1 contient un exemple de *feeder* optique pour fournir une connexion par satellite vers des stations de base. Dans cet exemple, les satellites sont connectés à une station de base au sol via un lien optique. Ensuite, les clients peuvent se connecter à ces satellites via un lien hertzien, sur la bande Ka située entre 26,5 et 40 GHz en suivant la norme DVB-S2X par exemple. Il est important d'obtenir un débit aussi grand que possible sur le lien optique entre les satellites et les stations de base afin de proposer de très hauts débits aux clients.

Cette étude s'inscrit dans le cadre d'un second projet R&T CNES qui traite du lien optique montant.

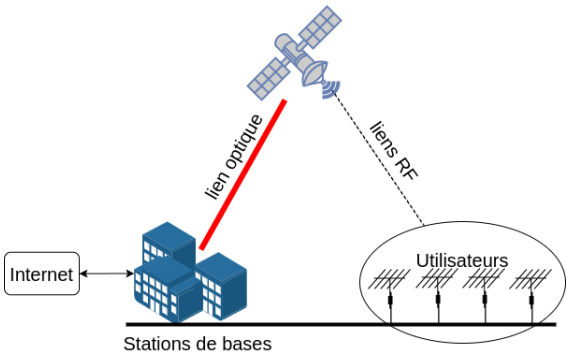

Figure 4.1 – Application *feeder* optique envisagée

4.1.2 Contraintes liées à l'application

Le lien montant introduit des problématiques supplémentaires par rapport au lien descendant. La raison principale provient de son intégration dans le satellite. Ces contraintes sont de différentes natures :

- La première contrainte est liée au récepteur optique embarqué. En effet, peu de récepteurs optiques ont été certifiés pour le spatial pour des débits de 10 Gbit/s ou plus. Les récepteurs certifiés sont basés sur des convertisseurs analogique-numérique sur 1 bit. L'information entrant dans le décodeur est donc obligatoirement binaire. Le processus de décodage peut uniquement être appliqué avec des entrées dures.
- Les FPGA utilisables pour des applications spatiales sont limités à un ensemble de puces certifiées. En effet, ces derniers doivent être durcis pour résister aux radiations. Le Kintex Ultrascale KU060 xqrku060 est actuellement en voie de recevoir la certification QML-Y [101] pour une utilisation spatiale. Le xcku060-ffva1156-1 [82] est un FPGA *low speed grade* et *pin compatible* avec le xqrku060. Les deux FPGA possèdent le même nombre de ressources matérielles. Les résultats obtenus dans ce chapitre avec le xcku060 sont donc transposables à ce qui pourrait être obtenu avec le xqrku060.
- Dans cette étude, le décodeur de codes correcteurs d'erreurs et l'entrelaceur doivent se partager les ressources présentes dans le circuit FPGA. Ainsi seulement 30% des ressources disponibles sont assignées à la fonction de décodage. Il est donc nécessaire de cibler des algorithmes de décodage de faible complexité calculatoire.
- La fréquence de fonctionnement maximum du FPGA est fixée à 250 MHz en raison de la dissipation thermique et de la consommation énergétique.
- Le débit retenu est de 10 Gbit/s.
- Le pouvoir de correction ciblé est moins important que pour le lien descendant. Le lien est bidirectionnel avec possibilité de renvoi de trames erronées. Les éventuels planchers d'erreurs sont donc moins problématiques. Toutefois, un objectif de plancher d'erreurs en dessous d'un FER de 10^{-6} a été choisi pour minimiser le renvoi de trames fausses.

Dans cette seconde étude, le choix du code LDPC a été fait au préalable. En effet, l'efficacité des codes FG-LDPC [83] démontrée dans le chapitre précédent fait qu'ils représentent une solution attractive dans le domaine des communications spatiales. De

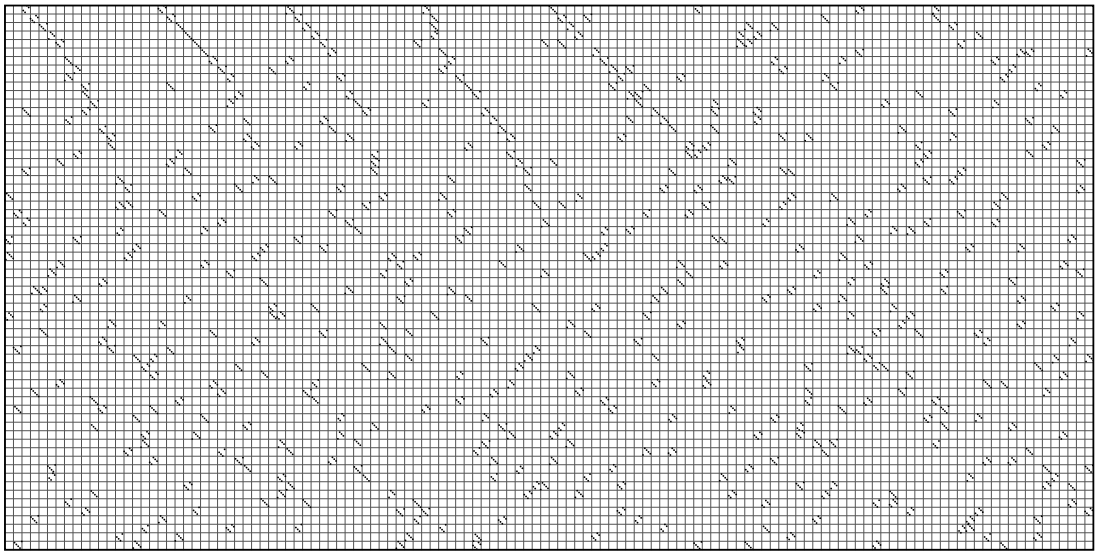

Figure 4.2 – Structure de la matrice **H** du code FG de rendement 1/2

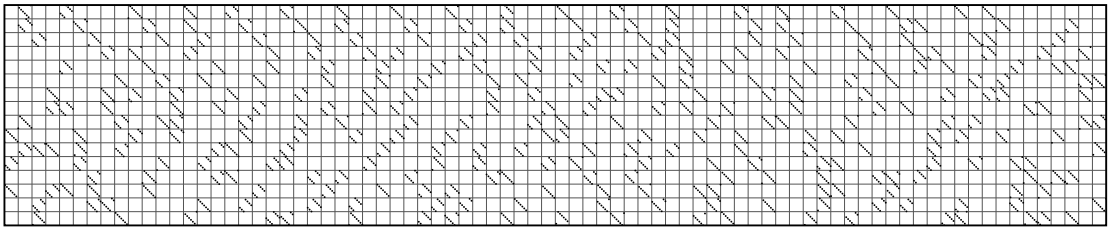

Figure 4.3 – Structure de la matrice **H** du code FG de rendement 4/5

plus, ils ont aussi été utilisés dans la littérature, conjointement avec des algorithmes à faible complexité calculatoire [102, 103]. Ainsi, deux codes FG-LDPC ont été conçus. Ces derniers possèdent des rendements 1/2 et 4/5. La taille des trames est fixée par K = 16384 pour les mêmes raisons de fiabilisation du lien explicitées dans le chapitre précédent. Les matrices **H** de ces codes LDPC sont représentées dans les figures 4.2 et 4.3.

L'objectif de l'étude est de trouver la solution qui apporte le meilleur pouvoir de correction tout en présentant un coût matériel satisfaisant l'ensemble des contraintes. En effet, la contrainte principale porte sur la complexité matérielle et le décodage haut débit à entrées dures dans l'espace, un domaine qui, à ce jour, a été peu exploré.

4.2 Choix de l'algorithme de décodage

Le cadre applicatif étant défini, il faut maintenant choisir l'algorithme de décodage à entrées dures possédant une faible complexité calculatoire et un bon pouvoir de correction

afin d'identifier celui qui sera utilisé pour l'implantation matérielle.

4.2.1 Gallager B étendu : Gallager E

4.2.1.1 Présentation de l'algorithme

L'algorithme Gallager B [6] est basé sur l'échange de messages binaires entre les VN et CN. Il a ensuite été étendu dans les travaux présentés dans [104]. Nommé Gallager *Algorithm E* dans [105] ou simplement Gallager E, son but est d'augmenter la dynamique des messages internes au décodeur afin d'améliorer le pouvoir de correction. Cet objectif est atteint grâce à l'introduction d'une troisième valeur possible. Ainsi les messages ne sont plus binaires mais ternaires.

Pour rappel, avec un ordonnancement par inondation, les messages échangés au cours d'un décodage avec l'algorithme Gallager B sont calculés de la manière suivante. Nous définissons premièrement la fonction *sign*, comme il suit,

$$sign(x) = \begin{cases} +1, & x \geqslant 0 \\ 0, & x = 0 \\ -1, & \text{otherwise} \end{cases} \tag{4.1}$$

Ensuite, le calcul des messages des VN vers les CN se fait en deux étapes. Tout d'abord, la somme s_{ij} des messages entrants dans le VN_i avec sa valeur initiale est évaluée pour déterminer la valeur du message à envoyer au CN_j.

$$s_{ij} = y_i + \sum_{\mathrm{CN}_k \in M(i) - \{\mathrm{CN}_j\}} c2v_{ki} \tag{4.2}$$

Si la valeur de cette somme est nulle, alors cela indique que le vote à majorité n'est pas fiable. En conséquence, le VN_i transmet l'information reçue du canal a priori y_i au CN_j comme cela est formalisé dans l'équation 4.3.

$$v2c_{ij} = \begin{cases} y_i, & |s_{ij}| = 0 \\ sign(s_{ij}), & \text{otherwise} \end{cases} \tag{4.3}$$

Une fois les messages $v2c_{ij}$ calculés, l'estimation des messages $c2v_{ji}$ peut être réalisée. Cette estimation se fait par un calcul de type 'ou exclusif', équivalent à l'addition dans

GF(2), formalisé dans l'équation 4.4 par un produit. Ceci est lié au fait que les valeurs renvoyées vers les messages $v2c_{ij}$ sont représentées par les valeurs -1 et +1.

$$c2v_{ji} = \prod_{\mathrm{VN}_k \in N(j) - \{\mathrm{VN}_i\}} v2c_{ij} \tag{4.4}$$

Afin d'améliorer les performances du processus de décodage, l'idée introduite par le Gallager E consiste à considérer un état non confiant indiquant l'incertitude. Cela permet de nuancer les messages échangés entre les VN et les CN. Afin d'implanter ce comportement, les valeurs échangées deviennent ternaires {-1, 0 , +1}, le zéro codant l'incertitude. Les calculs des messages issus des VN sont modifiés pour prendre en compte cette possibilité supplémentaire.

$$v2c_{ij} = sign\left(\omega^{(t)} y_i + \sum_{\mathrm{CN}_k \in M(i) - \{\mathrm{CN}_j\}} c2v_{ki}\right) \tag{4.5}$$

Le vote à majorité est mis à jour avec l'utilisation de la valeur '0' quand aucune majorité ne se démarque. Une autre évolution par rapport au Gallager B est l'ajout d'un poids $\omega^{(t)}$. Ce paramètre permet d'augmenter l'importance du canal en début de décodage. Cela entraîne une amélioration notable au niveau du pouvoir de correction d'erreurs dans la zone de convergence. Pour les messages $c2v$, le calcul reste identique. Il est à noter que le résultat est donc nul si l'un des messages entrant dans le CN est nul.

4.2.1.2 Ordonnancement par couches horizontales

Dans la littérature, l'algorithme Gallager E a été étudié lorsqu'un ordonnancement par inondation est employé. Cependant, le Gallager E peut être décrit de manière à utiliser un ordonnancement par couches horizontales. Cet ordonnancement traditionnellement utilisé dans les décodeurs à entrées souples permet de réduire le nombre d'itérations de décodage pour un niveau de performances équivalent et de minimiser la complexité mémoire. De plus, cet ordonnancement permet la conception d'architectures semi-parallèles efficaces comme expliqué dans le chapitre précédent. Nous avons donc repensé la formulation de l'algorithme de manière à faire apparaître les différentes étapes d'une unité de traitement de la même manière que dans le chapitre précédent.

Cependant le calcul des messages $c2v$ en provenance des nœuds de parité pose problème car la multiplication n'est pas inversible dès qu'il y a une valeur nulle. En effet, dans

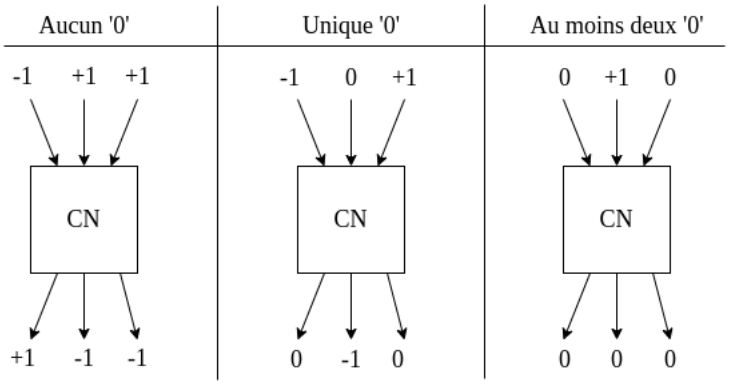

Figure 4.4 – Différents cas possibles en fonction du nombre de '0' entrant dans un CN pour le Gallager E

un ordonnancement par couches horizontales, le but est d'accumuler une valeur totale à laquelle est retranchée la contribution du VN de destination. Or, dans le cas du Gallager E, le message $c2v$ est obtenu à l'aide d'une multiplication. Si un message entrant dans le CN est nul, alors le produit est nul. Ainsi il devient impossible de reconstruire le message sortant en enlevant la contribution du '0'. La solution proposée ici est présentée dans l'algorithme 3. Elle consiste à compter le nombre de '0' entrant dans le CN et à en déduire les messages sortants. Les différents cas possibles sont récapitulés dans la figure 4.4.

Cette description de l'algorithme permet le même type d'implémentation que celle de l'algorithme OMS avec l'ensemble des calculs concentré dans des unités de traitement avec deux tranches de *pipeline*. Nous notons, par la suite, Ψ le nombre d'itérations pendant lesquelles $\omega^{(t)} = 2$, donc $\forall t \in [\![0; \Psi]\!], \omega^{(t)} = 2$ et $\forall t \in [\![\Psi + 1; t_{max} - 1]\!], \omega^{(t)} = 1$.

4.2.1.3 Étude des performances en correction d'erreurs

L'amélioration du pouvoir de correction, entre l'algorithme Gallager B et l'algorithme Gallager E, ainsi que l'utilisation de l'ordonnancement par couches horizontales, sont illustrées dans la figure 4.5.

Le passage de l'algorithme Gallager B à l'algorithme Gallager E introduit un gain en performance de décodage important et une convergence plus rapide. Pour un FER de 10^{-3}, il y a un gain d'un facteur 4 en probabilité d'erreurs liées au canal. La complexité calculatoire est cependant plus importante, notamment avec l'augmentation de la dynamique dans le décodeur. En effet, les messages sont maintenant représentés sur deux bits au lieu d'un seul.

La courbe correspondant au Gallager E avec un ordonnancement par inondation a été

Algorithm 3: Gallager E avec un ordonnancement par couches horizontales

▷ Input (bits reçus) : $\boldsymbol{y} = (y_1, y_1, ..., y_N) \in \{0,1\}^N$

init

$t = 0$, $Y_i = 1 - 2y_i$ and $V_i = 0$ for $i \in [1, .., N]$

repeat

▷ Ordonnancement par couches horizontales

for all j $\in [1, .., M]$ **do**

$C_j^{(t)} = 1$, $Sum0 = 0$

▷ Étape S1

for all i $\in$ N(j) **do**

▷ Calcul des messages $v2c_{ij}^{(t)}$

$$M_{ij} = \begin{cases} V_i, & t = 0 \\ V_i - c2v_{ji}^{(t-1)}, & \text{otherwise} \end{cases}$$

$$v2c_{ij}^{(t)} = sign(M_{ij} + \omega^{(t)} * Y_i)$$

▷ Calcul du nœud de parité j $C_j^{(t)}$

$$C_j^{(t)} = \begin{cases} C_j^{(t)}, & M_{ij} \geqslant 0 \\ -C_j^{(t)}, & \text{otherwise} \end{cases}$$

if $v2c_{ij}^{(t)} = 0$ **then** $Sum0 = Sum0 + 1$ **end if**

end for

▷ Étape S2

for all i $\in$ N(j) **do**

▷ Calcul des messages $c2v_{ji}^{(t+1)}$

$$c2v_{ji}^{(t)} = \begin{cases} 0, & Sum0 \geqslant 2 \\ 0, & Sum0 = 1 \text{ and } v2c_{ij}^{(t)} \neq 0 \\ C_j^{(t)}, & Sum0 = 1 \text{ and } v2c_{ij}^{(t)} = 0 \\ C_j^{(t)} \times v2c_{ij}^{(t)}, & \text{otherwise} \end{cases}$$

▷ Mise à jour des accumulateurs V_i

$$V_i = M_{ij} + c2v_{ji}^{(t+1)}$$

end for

end for

$t = t + 1$

until $t \leq t_{max}$

$$\forall i \in [1, .., N],\ x_i = \begin{cases} y_i, & Y_i + V_i = 0 \\ 0, & Y_i + V_i > 0 \\ 1, & Y_i + V_i < 0 \end{cases}$$

▷ Output (mot décodé) : $\boldsymbol{x} = (x_1, x_1, ..., x_N) \in \{0,1\}^N$

tracée avec 20 itérations de décodage avec $\Psi = 8$. Ces paramètres sont fixés à 10 itérations et $\Psi = 4$ pour l'ordonnancement par couches horizontales. Cette dernière est fournie en figure 4.5. Nous observons que les pouvoirs de correction de ces deux formulations sont similaires. Ceci est lié à la convergence qui est environ deux fois plus rapide avec un ordonnancement par couches horizontales [47]. Ainsi, l'importance accordée à

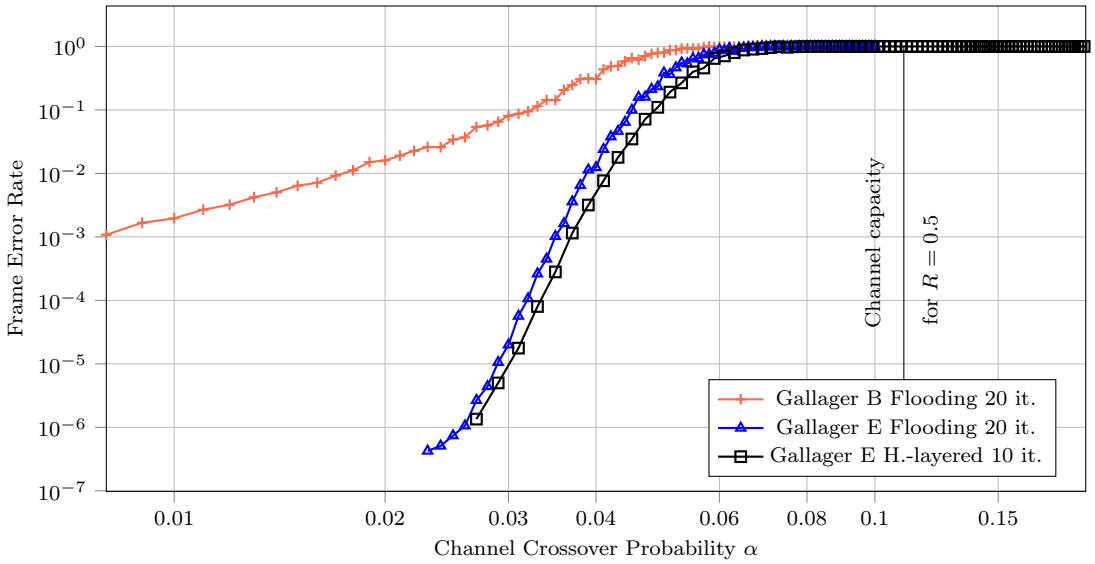

Figure 4.5 – Gain apporté par l'algorithme Gallager E avec un ordonnancement par couches horizontales sur un canal BSC avec le code QC-LDPC (1296,648)

l'information du canal est plus grande en considérant la même valeur de Ψ. Cela provient du fait que 4 itérations avec un ordonnancement par couches horizontales permettent d'obtenir des performances similaires à celles de 8 itérations avec un ordonnancement par inondation.

Ensuite l'influence du nombre d'itérations sur le pouvoir de correction est illustrée par la figure 4.6. Il apparait qu'à partir de 10 itérations de décodage (courbe en noir), l'utilisation d'itérations supplémentaires n'augmente plus les performances de manière significative pour $\Psi = 4$.

Enfin, les valeurs de $\omega^{(t)}$ ont une influence importante sur le pouvoir de correction. Dans les travaux présentés dans [105], il a été démontré que des valeurs pertinentes de $\omega^{(t)}$ sont 2 pour les 2 premières itérations puis 1 pour les autres itérations pour les codes LDPC réguliers de rendement 1/2 avec $d_v = 3$ et $d_c = 6$. La figure 4.7 montre l'influence de cette valeur en fonction du nombre maximal d'itérations pour une probabilité d'erreur de 0.035 sur BSC pour le code QC-LDPC régulier de taille (1296, 648) présenté dans [106] avec $Z = 54$, $d_v = 4$ et $d_c = 8$. Il apparaît que le Ψ optimal varie en fonction du nombre maximum d'itérations. Par exemple, pour 7 itérations, la valeur optimale de Ψ est 3 alors que de 8 à 10 itérations, c'est 4.

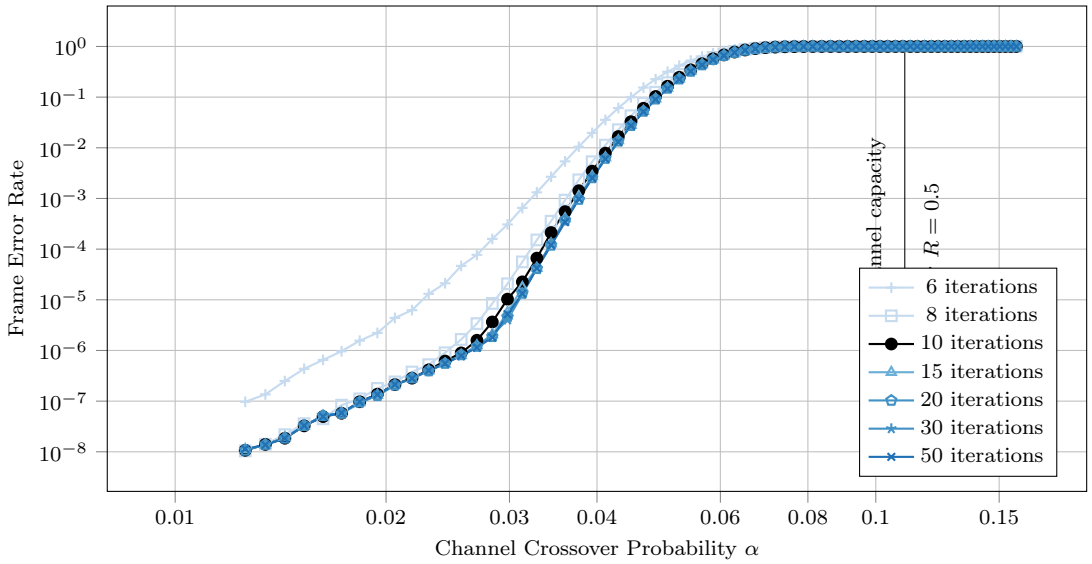

Figure 4.6 – Évaluation du pouvoir de correction de l'algorithme Gallager E en fonction du nombre d'itérations effectuées

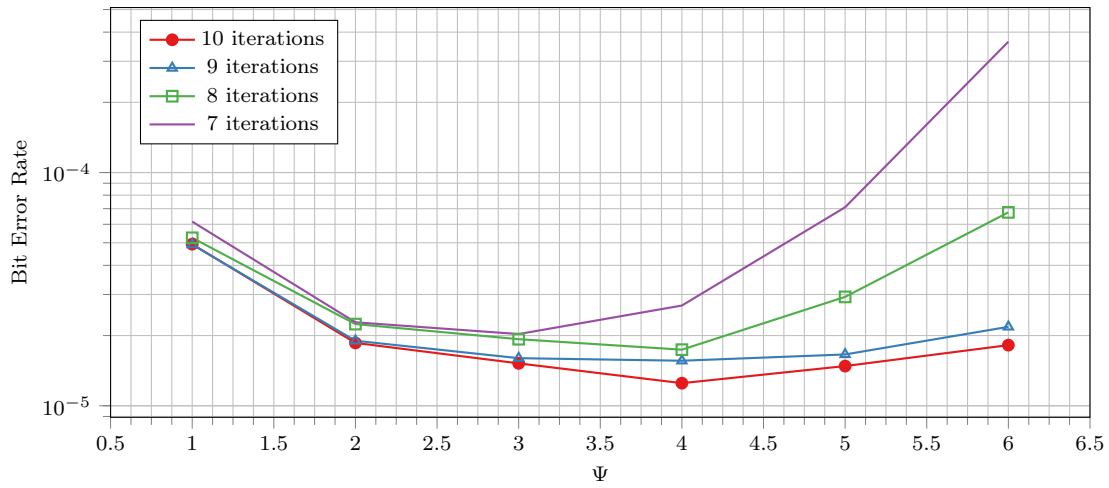

Figure 4.7 – Influence de Ψ et du nombre d'itérations sur les performances de décodage pour une probabilité d'erreur de 0.035 sur BSC

4.2.2 Comparaison des performances

L'algorithme Gallager E possède un meilleur pouvoir de correction que l'algorithme Gallager B tout en conservant une complexité calculatoire raisonnable. Il s'avère donc être un candidat attractif pour l'application ciblée sur satellite. La comparaison qui suit avec d'autres algorithmes récents de la littérature permet de confirmer le choix de l'algorithme Gallager E en tant que compromis satisfaisant les spécifications imposées par le cadre applicatif.

Ainsi l'algorithme Gallager E est comparé aux algorithmes Bit-Flipping (BF) [6], Gradient Descent Bit-Flipping (GDBF) [84], Probabilistic GDBF (PGDBF) [107], Gallager B (GaB) [6], Probabilistic Gallager B (PGaB) [108], Min-Sum (MS) [109] et Offset Min-Sum (OMS) [109]. Le code sélectionné reste le code régulier QC-LDPC (1296,648) avec $Z = 54$, $d_v = 4$ et $d_c = 8$.

Par ailleurs, les Finite Alphabet Iterative Decoders (FAID) [110] peuvent permettre un décodage très efficace pour la correction d'erreurs sur un canal BSC. Cependant, leurs performances dépendent des codes LDPC utilisés. En effet, les FAID ont été considérés pour des codes LDPC de petite taille pour lesquels il est possible d'énumérer les *trapping* et *stopping sets* afin de personnaliser le décodeur en conséquence. Un décodeur utilisant ce type d'algorithme ne peut donc pas être flexible et gérer les deux longs codes LDPC retenus dans le cadre de l'étude. C'est pourquoi les décodeurs FAID ne sont pas considérés dans cette comparaison algorithmique.

L'algorithme PGDBF [107] est une évolution de l'algorithme GDBF [84] présenté dans le chapitre précédent. L'ajout d'un effet aléatoire qui peut empêcher un bit $\Delta_k^{(GD)}$ qui est inférieur à un seuil d'inversion θ d'être inversé permet de s'extraire de certains minimums locaux et ainsi d'accroître le pouvoir de correction.

Dans un même registre, le principe du PGaB [108] est d'introduire de l'aléatoire dans le calcul des messages $v2c$ pour éviter les *trapping set*. En effet, dans le vote à majorité, un bit supplémentaire, choisi aléatoirement selon une distribution de Bernoulli, peut être considéré.

Les algorithmes MS et OMS sont trop complexes pour une implantation de décodeur à 10 Gbit/s. Ils sont tout de même présents dans cette comparaison en tant que borne supérieure au niveau du pouvoir de correction d'erreurs. En effet, le décodeur OMS du chapitre précédent utiliserait environ 75% des LUTs disponibles sur le FPGA pour atteindre 10 Gbit/s. Dans notre nouveau contexte, seulement 30% des ressources peuvent être allouées au décodeur. Le nombre d'itérations est fixé à 20 et la quantification est de 6 bits pour les LLR a posteriori et 4 bits pour les messages comme présenté dans [106]. Il est à noter que la courbe en vert pour l'algorithme OMS a été simulée sous AFF3CT pour 10 itérations de décodage pour pouvoir effectuer une comparaison à nombre d'itérations équivalent.

L'algorithme Gallager E utilise une valeur de Ψ de 4 avec 10 itérations pour un ordonnancement par couches horizontales. Enfin, tous les autres algorithmes de décodage sont simulés avec 300 itérations de décodage.

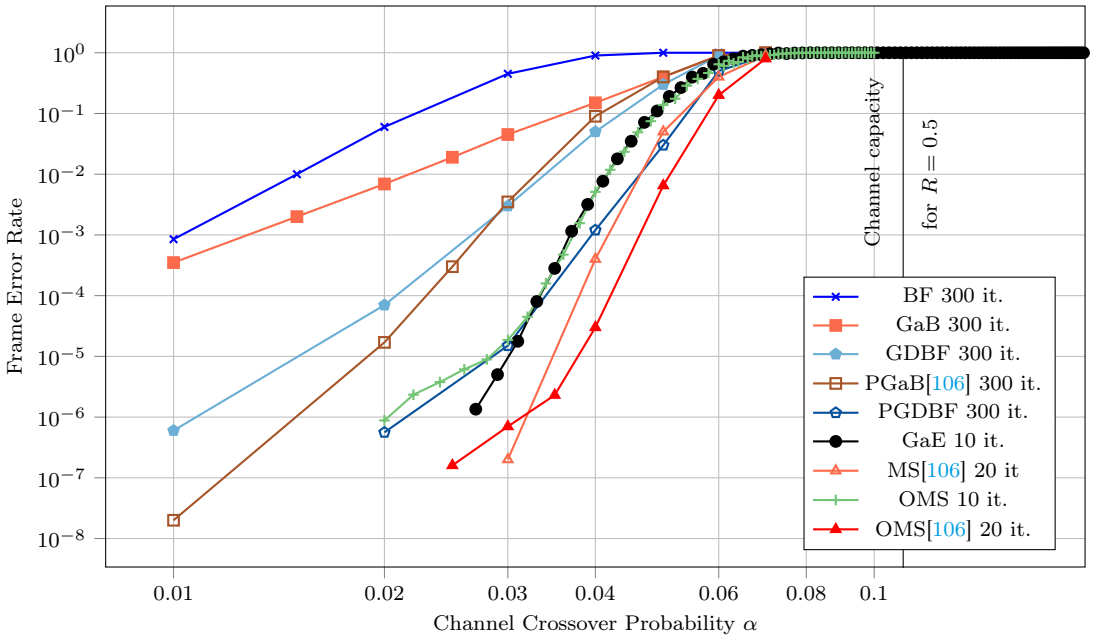

Figure 4.8 – Comparaison d'algorithmes de décodage sur canal BSC pour le code QC-LDPC (1296,648)

La figure 4.8 montre les performances en termes de taux d'erreur trame pour les différents algorithmes considérés. L'algorithme Gallager E avec 10 itérations de décodage n'atteint pas le pouvoir de correction du Min-Sum et de l'OMS avec 20 itérations de décodage comme présentés dans [106]. Cependant, pour un nombre d'itérations similaire, l'algorithme Gallager E est plus performant que l'OMS simulé avec AFF3CT au niveau de l'*error floor*. En outre, l'algorithme Gallager E présente une meilleure capacité à corriger les erreurs que le Gallager B et sa version probabiliste, ainsi que le BF et GDBF. Le PGDBF commence à converger avant l'algorithme Gallager E mais à partir d'une probabilité d'erreurs de $\alpha = 0.032$, l'algorithme PGDBF présente un taux d'erreurs plus élevé. L'algorithme Gallager E ressort alors comme l'algorithme permettant d'obtenir de bonnes performances en matière de pouvoir de correction en comparaison avec les autres algorithmes de décodage à entrées dures présents dans la littérature.

La partie suivante introduit une architecture matérielle que nous avons conçue pour implanter l'algorithme Gallager E sur cible FPGA. Cette dernière permet d'atteindre le débit attendu avec les ressources disponibles.

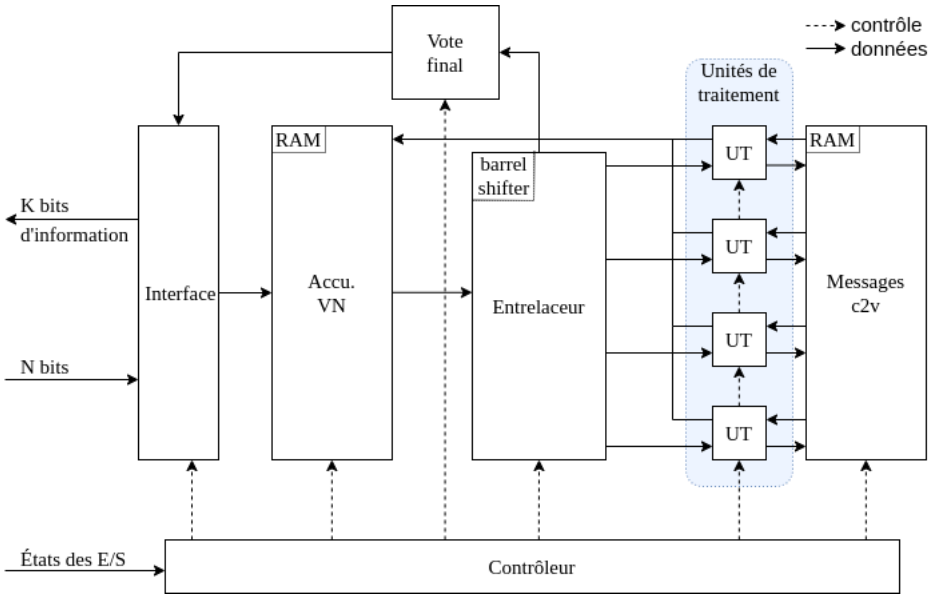

Figure 4.9 – Architecture globale du décodeur LDPC

4.3 Implantation matérielle du décodeur

Comme nous venons de le voir, l'algorithme Gallager E offre des performances de décodage intéressantes ainsi qu'une complexité calculatoire relativement faible. Afin de valider et caractériser ces atouts, un décodeur matériel a été développé. Cette partie présente l'architecture matérielle du décodeur qui a été conçue pour implanter efficacement l'algorithme Gallager E sur cible FPGA. L'architecture globale du décodeur est similaire à celle traitant du lien descendant, à l'exception des unités de traitement, du dimensionnement des mémoires et de leur utilisation. Le bloc de détermination des bits en sortie du décodeur est également différent. Ensuite, son coût matériel ainsi que ses performances de décodage sont étudiés pour confirmer ce choix dans le cadre de l'étude R&T CNES.

4.3.1 Architecture du décodeur

L'architecture globale du décodeur est détaillée dans la figure 4.9. Toutes les ressources sont contrôlées par le même ASIP que l'architecture du chapitre précédent pour l'algorithme OMS.

Le premier changement architectural consiste à modifier l'interface du décodeur afin d'exploiter une information binaire et de la placer correctement en mémoire. Contrairement au décodeur Offset Min-Sum l'information binaire du canal doit être accessible en même

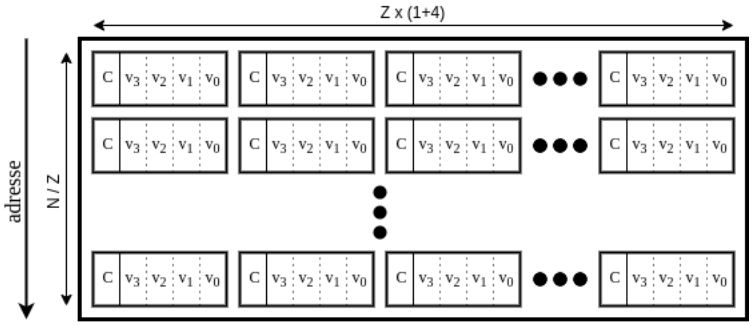

Figure 4.10 – Organisation de la RAM des accumulateurs VN

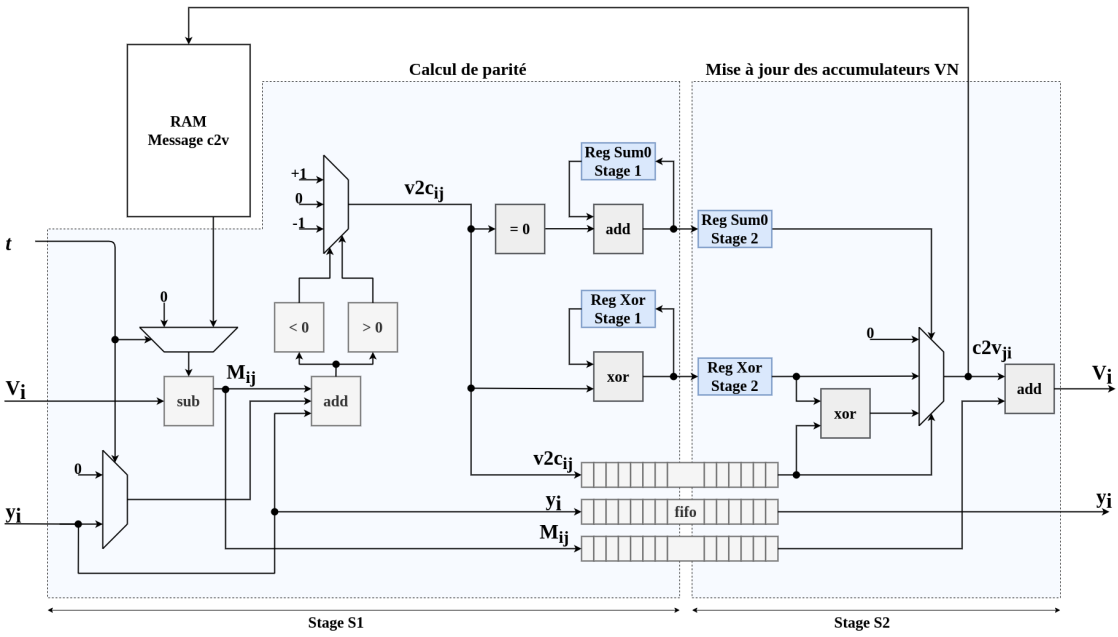

Figure 4.11 – Unité de traitement dédiée à l'algorithme Gallager E

temps que son accumulateur de VN associé. Ces deux valeurs sont mémorisées ensemble dans la mémoire RAM des accumulateurs VN. La figure 4.10 montre comment cette mémoire est organisée. Le bit C correspond au bit reçu en entrée du décodeur et les bits v_0, v_1, v_2 et v_3 à l'accumulateur. Ces valeurs se déplacent dans la RAM pour éviter l'utilisation d'un second entrelaceur. Il est donc indispensable de les associer au sein d'un même emplacement mémoire.

Ensuite, le bloc en sortie de décodage doit prendre une décision dure dans le cadre de l'algorithme OMS. Dans le cas présent, c'est un vote de majorité qui est nécessaire pour décider de la valeur du bit en sortie du décodeur comme décrit dans l'algorithme 3.

Les unités de traitement ont été repensées afin d'effectuer les opérations de l'algorithme

Code LDPC	Z	Complexité matérielle			Performances d'un décodeur	
		# LUT	# FF	# BRAM	Débit	Latence
(1296, 648)	54	3 239	2 382	8	0.31 Gbit/s	4.160 μs
(32768, 16384)	256	13 979	10 573	42	1.53 Gbit/s	21.42 μs
(20480, 4096)	256	13 979	10 573	34	1.51 Gbit/s	13.55 μs

Tableau 4.1 – Résultats d'implantation sur FPGA du décodeur Gallager E (après place & route, fréquence = 250 MHz) pour 10 itérations de décodage

Gallager E. Elles sont détaillées dans la figure 4.11. Tout d'abord, par rapport aux unités de traitement du décodeur OMS, des *fifos* ont été ajoutées pour pouvoir transmettre à la fois les valeurs du canal mais aussi les messages entrants. Pour la valeur du canal, cela est nécessaire pour pouvoir l'enregistrer en même temps que la valeur de l'accumulateur et donc de maintenir une cohérence au niveau du stockage des données. En ce qui concerne la valeur du message entrant, elle est nécessaire dans l'étape S2 pour générer le message de sortie.

Le contrôle des différentes ressources des 2 étapes S1 et S2 reste identique d'un point de vue temporel. Ceci permet d'utiliser le flot de conception développé pour la génération du décodeur OMS. Cela nécessite, tout de même, une option permettant de configurer une quantification de l'information compatible avec l'algorithme Gallager E.

4.3.2 Caractéristiques du décodeur et performances

Afin d'évaluer les performances de l'architecture et de valider sa possible utilisation dans le contexte de l'étude, nous l'avons caractérisée pour la cible FPGA KU 060.

L'architecture conçue doit avoir un coût matériel satisfaisant les spécifications de l'étude tout en offrant le pouvoir de correction attendu.

Tout d'abord, le coût matériel de l'architecture est détaillé dans le tableau 4.1 pour les différents codes LDPC étudiés. Les résultats ont été obtenus avec Vivado 2018.2 après synthèse, placement et routage sur le FPGA Xilinx xcku060-ffva1156-1 pour une fréquence de fonctionnement fixée à 250 MHz. Pour les codes LDPC ($K = 16384$) envisagés dans le cadre de l'étude, 13 979 LUT et 10 573 FF sont nécessaires dans le circuit FPGA. Il apparaît qu'un décodeur élémentaire atteint environ 1.5 Gbit/s lorsque 10 itérations de décodage sont exécutées. La latence de décodage est de l'ordre de la dizaine de μs. Ainsi, elle reste négligeable face à la durée d'entrelacement de plus de 100 ms. La fréquence de fonctionnement a été fixée à 250 MHz mais le décodeur proposé peut atteindre 309

MHz sur ce FPGA *low speed grade*. Cela permet d'accéder en pratique à un débit de 1.87 Gbit/s si le circuit fonctionne à sa fréquence nominale.

Pour rappel, Z unités de traitement sont utilisées. Cela engendre un plus gros impact en termes de ressources logiques. En ce qui concerne la mémoire, le nombre de blocs RAM utilisés peut être calculé d'après les expressions suivantes.

$$N_{RAM_{VN}} = \left\lceil \frac{\lceil log_2(d_v) + 2 \rceil \times Z}{RAM_{width}} \right\rceil \times \left\lceil \frac{T \times N}{Z \times RAM_{depth}} \right\rceil \tag{4.6}$$

$$N_{RAM_{MSG}} = \left\lceil \frac{2 \times Z}{RAM_{width}} \right\rceil \times \left\lceil \frac{T \times d_v \times N}{Z \times RAM_{depth}} \right\rceil \tag{4.7}$$

Ainsi la taille de la mémoire pour les accumulateurs des VN dépend de N, Z et de d_v. De plus, lorsque l'ordonnancement des CN se fait avec l'utilisation de trames supplémentaires en parallèle, comme indiqué dans le chapitre précédant, la valeur de T n'est plus égale à 1 et l'empreinte mémoire peut augmenter. Cela est visible avec le deuxième opérande de la multiplication dans l'équation. En effet, si $T \times N < Z \times RAM_{depth}$, alors cette valeur est égale à 1. Pour un FPGA Xilinx de la gamme Ultrascale, $RAM_{depth} = 512$. Ainsi, l'inégalité précédente est vérifiée jusqu'à $T = 6$ pour le code LDPC (20480, 4096). Comme l'architecture est utilisée de façon optimale dès $T = 3$, le nombre de blocs RAM utilisés pour cette mémoire ne dépend pas de T pour ce code LDPC.

Pour la mémoire contenant les messages, le résultat est différent puisque la valeur de d_v est considérée dans ce calcul. Ici $d_v = 4$, donc l'inégalité $T \times d_v \times N < Z \times RAM_{depth}$ n'est vérifiée que pour T égal à 1 ou 2 avec le même code LDPC (20480, 4096).

Le nombre de blocs RAM utilisés par l'architecture dépend donc de la technologie du FPGA cible, des caractéristiques du code LDPC et du choix d'utiliser ou non des trames en parallèle.

Enfin, il est important de situer le pouvoir de correction du décodeur matériel. La figure 4.12 permet de comparer les performances du décodeur pour les différents codes lorsque $\Psi = 4$. Les courbes avec un ordonnancement par couches horizontales ont été obtenues via l'implantation matérielle du décodeur sur FPGA. En revanche, les courbes avec un ordonnancement par inondation ont été obtenues par simulation logicielle sous AFF3CT.

Comme constaté en début de chapitre, à Ψ équivalent, l'ordonnancement par couches horizontales présente un meilleur pouvoir de correction. En outre, les longs codes LDPC présentent une convergence plus rapide que le code LDPC (1296, 648). De plus, leur

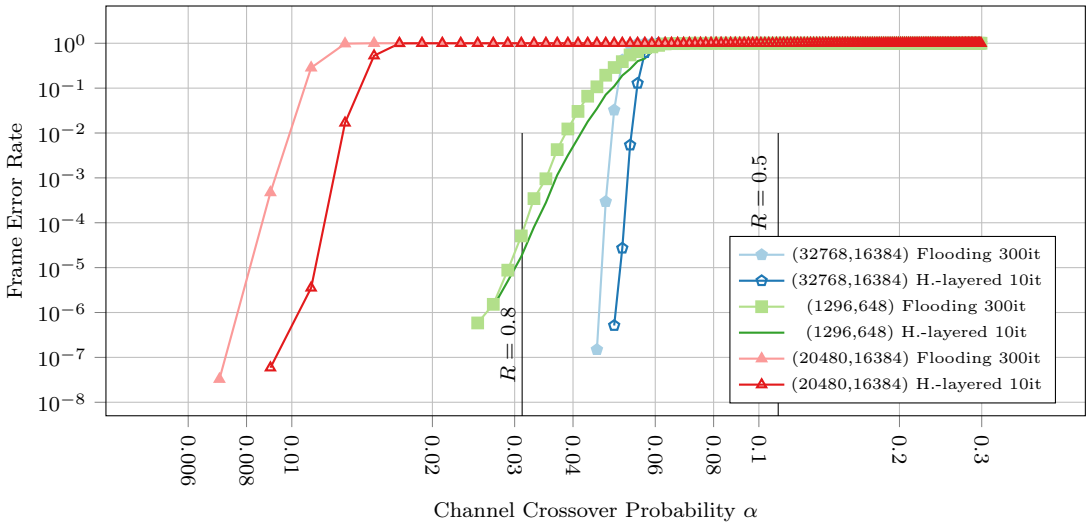

Figure 4.12 – Performances de correction pour différents codes LDPC en considérant différents ordonnancements et nombres d'itérations

pouvoir de correction s'approche de la capacité du canal. Par exemple, un simple facteur 2 est obtenu au niveau de la probabilité d'erreurs entre les performances du décodeur avec le code LDPC (20480, 4096) et la capacité du canal. Enfin, aucun *error floor* n'apparaît au-dessus d'un FER de 10^{-6}, ce qui satisfait les besoins applicatifs.

4.4 Interprétation des résultats et comparaisons

4.4.1 Interprétation des résultats

Le but de cette partie est de montrer la faisabilité d'une solution dans le cadre de l'étude à partir des résultats d'implantation acquis précédemment.

La place allouée au décodage canal sur le FPGA cible correspond à 100 000 LUTs et FFs et à 504 blocs RAM. Ainsi, à partir des résultats expérimentaux fournis dans le tableau 4.1, il est possible d'assigner jusqu'à 7 décodeurs élémentaires. Chacun de ces décodeurs élémentaires fournit un débit de 1.5 Gbit/s lorsqu'il fonctionne à 250 MHz. Cela permet d'atteindre un débit total de plus de 10 Gbit/s. Cette solution à base de 7 décodeurs consomme 98% des LUTs et 7% des blocs RAM alloués au décodage canal comme le montre le tableau 4.2. Les ressources utilisées y sont présentées en pourcentage par rapport aux ressources totales allouées au décodage canal.

Le point bloquant pour ajouter plus de décodeurs se situe au niveau des LUTs. Il est aussi possible de placer 6 décodeurs et de fixer le nombre d'itérations à 9 pour atteindre un débit

# dec.	% LUTs	% BRAMs	Fréquence	# iter.	Débit
1	14	7	250 MHz	10	1.51 Gbit/s
7	98	47	250 MHz	10	10.57 Gbit/s
6	84	40	250 MHz	9	10.07 Gbit/s
1	14	7	309 MHz	10	1.87 Gbit/s
6	84	40	309 MHz	10	11.20 Gbit/s
5	70	34	309 MHz	9	10.37 Gbit/s

Tableau 4.2 – Coûts matériels et débits de différentes solutions architecturales en fonction du nombre de décodeurs, de la fréquence de fonctionnement et du nombre d'itérations pour le code (20480,16384)

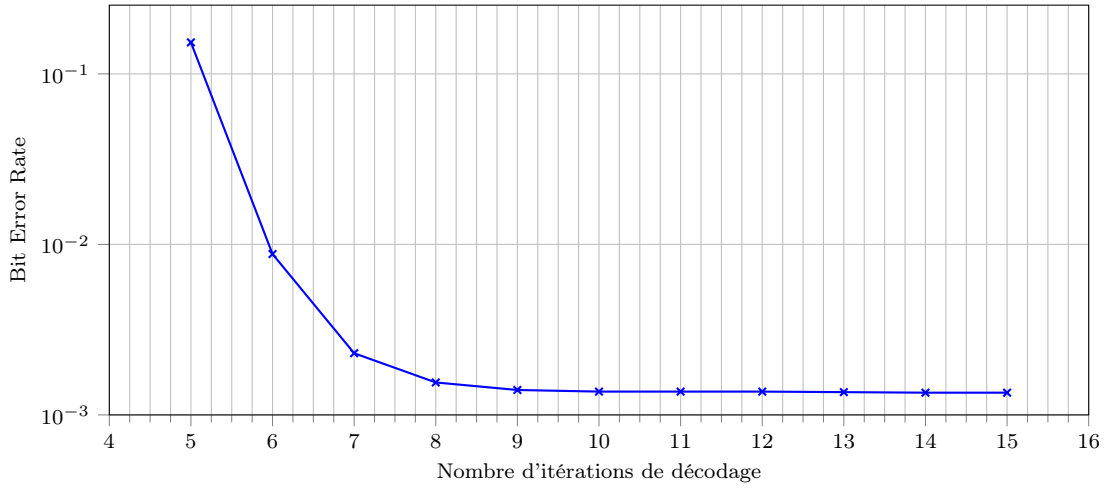

Figure 4.13 – Performances en BER de l'algorithme Gallager E pour le code LDPC (20480, 4096) avec $\alpha = 0.01224$

de 10 Gbit/s. Cette approche peut permettre de laisser plus de marge pour les autres parties du système en échange d'une infime perte au niveau de la correction d'erreurs comme le montre la figure 4.13 pour le code LDPC (20480, 4096). Cette configuration occuperait en effet 84% des LUTs allouées au décodage canal. Cela permettrait de relâcher les contraintes globales du système. Lorsque nous utilisons la fréquence de fonctionnement nominale du décodeur (309MHz), il est possible de supprimer un décodeur supplémentaire tout en maintenant un débit supérieur à 10 Gbit/s.

Ainsi, l'objectif de l'étude est atteint. Un décodeur a été conçu à partir d'une description innovante de l'algorithme Gallager E avec un ordonnancement par couches horizontales. Il fournit le débit souhaité et il respecte les contraintes imposées par le domaine applicatif.

	Gallager B	Gallager E	OMS
# LUTs	7 833 (8%)	13 979 (14%)	49 137 (49%)
# FFs	8 379 (8%)	10 573 (11%)	26 919 (27%)
# BRAM	23 (5%)	34 (7%)	73 (14%)
Quantification	4 bits VN 1 bit MSG	5 bits VN 2 bits MSG	8 bits VN 6 bits MSG
Fréquence max.	347 MHz	309 MHz	156 MHz
Débit	2.10 Gbit/s	1.87 Gbit/s	0.94 Gbit/s

Tableau 4.3 – Complexité matérielle d'un décodeur pour différents algorithmes avec le code LDPC (20480, 4096) et 10 itérations de décodage sur le circuit FPGA KU 060

4.4.2 Comparaison avec l'état de l'art

Dans un premier temps, il est intéressant de positionner le coût matériel de cette architecture par rapport à des architectures similaires pour les algorithmes Gallager B et OMS. C'est ce qui est proposé dans le tableau 4.3. Ces résultats ont été obtenus avec Vivado après synthèse, placement et routage sur le FPGA Xilinx xcku060-ffva1156-1 pour le code LDPC (20480, 4096). Une architecture semi-parallèle de Gallager B a été spécialement conçue pour cette comparaison. Il est à noter que le coût matériel en termes de LUTs, FFs et blocs RAM augmente avec la complexité calculatoire et la quantification des données. La fréquence maximale est d'autant plus faible que l'algorithme de décodage est complexe.

L'algorithme OMS présente un pouvoir de correction similaire à l'algorithme Gallager E pour le même nombre d'itérations. Pourtant, le débit souhaité ne peut être atteint avec l'algorithme OMS. En effet, 540% des ressources disponibles en termes de LUTs sont nécessaires pour atteindre 10 Gbit/s avec 11 décodeurs élémentaires pour l'algorithme OMS. Quant à l'algorithme Gallager B, il est certes 2.5 fois plus efficace en termes de complexité matérielle mais le pouvoir de correction atteint n'est pas satisfaisant.

Il est important de rappeler que le circuit FPGA est un *low speed grade*. Les fréquences de fonctionnement maximales atteignables sont donc plus faibles que pour le circuit FPGA utilisé dans le chapitre 3. Ainsi pour d'autres FPGA, des montées en fréquence sont possibles, par exemple, sur le Xilinx Zynq UltraScale+ xczu9eg-ffvb1156-3-e. Pour ce dernier, le décodeur Gallager E atteint une fréquence de fonctionnement de plus de 500 MHz et donc un débit de 3 Gbit/s par décodeur élémentaire.

Dans un second temps, notre solution est comparée aux architectures de la littérature pour un canal BSC. Des architectures récentes et efficaces implantées sur FPGA pour les algorithmes PGaB et PGDBF ont respectivement été présentées dans [106] et [111].

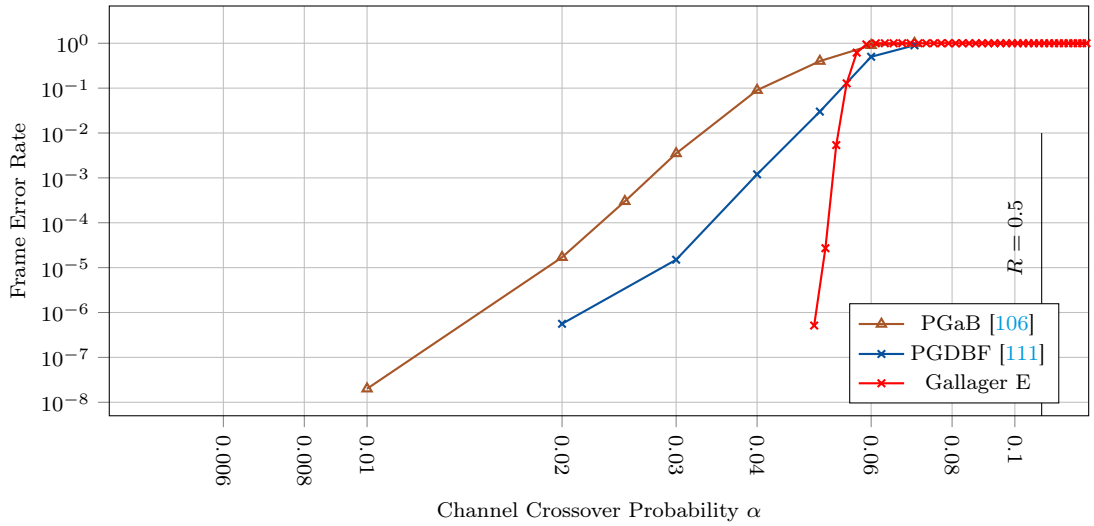

Figure 4.14 – Performances de notre solution et des implémentations de l'état de l'art

Ces deux décodeurs sont comparés avec notre solution dans le tableau 4.4. Pour notre architecture, nous considérons le code LDPC (32768, 16384) et non le code LDPC de taille (1296, 648) pour avoir des coûts matériels et un rendement équivalents. Cela permet une comparaison équitable au niveau du débit et du pouvoir de correction.

La figure 4.14 présente les performances des différentes implémentations avec les codes LDPC considérés dans le tableau de comparaison. Notre solution est la meilleure au niveau du pouvoir de correction d'erreurs. Un écart similaire entre le décodeur PGDBF et notre architecture qu'entre nos travaux et la capacité limite du canal est observé pour un FER de 10^{-6}.

En ce qui concerne les débits, les travaux sur le PGDBF permettent d'atteindre un débit de 5.36 Gbit/s pour une probabilité d'erreur $\alpha = 0.01$. C'est supérieur de 2.25 Gbit/s à ce qui est atteint par notre architecture. Pour ce niveau faible de bruit, le FER obtenu est inférieur à 10^{-6} et peu d'itérations, et même parfois aucune, sont nécessaires pour trouver un mot de code. Cela permet d'atteindre des débits très élevés. Il aurait été intéressant de connaître le débit pour des probabilités d'erreurs plus élevées et d'évaluer l'évolution du débit en fonction du bruit. En ce qui concerne l'implantation matérielle du Probabilistic Gallager B, elle atteint un très haut débit de 37.9 Gbit/s en moyenne mais présente des performances au niveau du pouvoir de correction éloignées d'un facteur 10 par rapport à la capacité du canal pour un FER d'environ 3.10^{-8}. L'objectif étant d'implanter un décodeur avec un bon pouvoir de correction vis-à-vis des ressources disponibles, l'architecture proposée pour le PGaB n'est pas envisageable.

Enfin, comme pour le chapitre précédent, une étude comparative des implémentations logi-

	[111]	[106]	Notre solution
N	1296	1296	32768
R	0.5	0.5	0.5
FPGA	Virtex-6 xc6vlx240	Virtex-6 xc6vlx240	Kintex xcku060 / Zynq zu9eg (max)
Algorithme	PGDBF	PGaB	Gallager E
Rang en pouvoir de correction	2nd	3ème	1er
Fréquence	Non communiquée	146 MHz	250 MHz 509 MHz (max)
# LUT	13 973	14 605	13 979
# FF	2 971	9 141	10 573
# BRAM	0	0	42
Débit	5.36 Gbit/s (pour $\alpha = 0.01$)	37.9 Gbit/s (moyenne)	1.53 Gbit/s 3.11 Gbit/s (max) (garanti)

Tableau 4.4 – Tableau comparatif avec des travaux de la littérature

cielles et matérielles a été faite. C'est ce qui est détaillée dans le tableau 4.5. L'utilisation d'un serveur de calcul embarqué dans un satellite n'est pas envisageable en raison des contraintes de consommation et de dissipation. Cependant, cette étude est pertinente dans le cas d'un décodage à entrées dures pour des besoins importants au niveau de la correction d'erreurs. Les cibles matérielles sont les mêmes que celles du chapitre précédent. La cible CPU est un Intel Xeon Gold 6148. Le circuit FPGA est le Xilinx Ultrascale+ xczu9eg-3ffvb1156e. Les prix de ces cibles sont respectivement d'environ 7500 $ et 5200 $. Les résultats pour l'implantation matérielle sont donnés pour 15 décodeurs sur le circuit FPGA ciblé avec une fréquence de 500 MHz.

L'implémentation logicielle du Gallager E introduit des problématiques supplémentaires par rapport à l'implémentation de l'algorithme OMS. En effet, l'association des accumulateurs des VN avec les valeurs du canal et les différences au niveau des calculs des messages font que l'implémentation du Gallager E est moins efficace que celle de l'OMS sur CPU. Nous arrivons tout de même à atteindre un débit de 5.4 Gb/s sur la cible Xeon. Cependant l'implantation matérielle présente de meilleures performances sur tous les points. L'énergie consommée par bit est en moyenne 80 fois plus faible. Le débit atteint sur circuit FPGA est 10 à 12 fois plus élevé que sur le CPU.

	Code LDPC	# déc.	Déb. (Mbit/s)	Lat. (μs)	P (Watts)	E (nJ/bit)
CPU	(32768, 16384)	1	180	113	172	956
	(20480, 4096)	1	247	132	169	685
	(32768, 16384)	40	4298	191	340	80
	(20480, 4096)	40	5446	248	341	63
FPGA	(32768, 16384)	1	3060	10.7	3.5	1.15
	(20480, 4096)	1	3020	6.8	3.4	1.13
	(32768, 16384)	15	45900	10.7	44.4	0.97
	(20480, 4096)	15	45300	6.8	40.0	0.89

Tableau 4.5 – Comparaison de performances entre l'implémentation logicielle sur cible CPU Xeon et l'implémentation matérielle sur cible FPGA Zynq Ultrascale + : Les résultats sont donnés pour des décodeurs LDPC effectuant 10 itérations de décodage

Ainsi, les diminutions conjointes de la complexité calculatoire et de l'empreinte mémoire du Gallager E par rapport à l'OMS permet d'augmenter les performances de l'implantation matérielle. Cependant ces changements ne sont pas directement exploitables pour une implémentation logicielle.

4.5 Conclusion

La contribution détaillée dans ce quatrième chapitre avait pour objectif de fiabiliser le lien montant de communications numériques optiques par satellite. La première partie du chapitre présente le domaine applicatif de l'étude ainsi que les différentes problématiques. Elle formalise les contraintes à satisfaire par le décodeur LDPC en termes de complexité matérielle, de débit et de fréquence de fonctionnement.

La deuxième partie du chapitre présente une nouvelle façon de décrire l'algorithme Gallager E avec un ordonnancement par couches horizontales qui s'avère être une solution attractive pour concevoir des décodeurs à entrées dures. De plus, une comparaison de l'algorithme proposé avec d'autres algorithmes récents de la littérature est faite afin de conforter notre choix.

Dans la troisième partie, une architecture semi-parallèle conçue pour décoder des codes QC-LDPC selon l'algorithme Gallager E est détaillée. Son coût matériel et son pouvoir de correction sont ensuite analysés.

La dernière partie propose une analyse des résultats obtenus avec cette architecture.

Elle vise à démontrer que le décodeur ainsi conçu permet de répondre à tous les besoins applicatifs du lien montant. De plus, une comparaison avec les décodeurs de la littérature pour un canal BSC est réalisée.

Les travaux présentés dans ce chapitre ont été valorisés à travers une publication à la conférence ICECS 2020 [112].

Le modèle architectural proposé est générique et supporte tous les codes QC-LDPC. Cependant il n'exploite pas totalement les deux bits disponibles pour les messages car seulement trois valeurs sur quatre sont exploitées. Des tentatives d'utilisation de cette valeur ont été faites mais rien de concluant n'a, jusqu'à présent, été trouvé. Il serait intéressant d'approfondir la réflexion dans ce sens afin d'augmenter le pouvoir de correction sans augmenter significativement le coût matériel de l'architecture résultante.

Conclusions et perspectives

Conclusions

Les travaux de recherche effectués au cours de cette thèse se sont focalisés sur l'étude et l'implantation matérielle de décodeurs LDPC pour des communications numériques satellitaires exploitant des liens optiques.

Les codes correcteurs d'erreurs permettent la correction des erreurs de transmission sur des canaux de communication peu fiables. Différentes familles de codes correcteurs ont été proposées dans la littérature et certaines d'entre elles fournissent des performances proches de la limite théorique de Shannon. Les familles de codes correcteurs les plus utilisées de nos jours sont présentées dans le chapitre 1. Ce rappel permet de préciser les concepts nécessaires à la bonne compréhension des différentes contributions faites dans le cadre de la thèse. Ce premier chapitre aborde aussi l'utilisation des codes correcteurs d'erreurs dans un domaine particulier, à savoir le domaine des communications spatiales.

Les algorithmes usuels employés pour le décodage de codes LDPC sont décrits dans le chapitre 2. Les différents ordonnancements des calculs et les différentes architectures de décodeurs associées sont détaillés et analysés. En effet, ces dernières présentent différents niveaux de parallélisme et différents niveaux de performances. Afin d'atteindre de hauts débits tout en maîtrisant la complexité matérielle, l'utilisation d'algorithmes sous-optimaux s'avère nécessaire. De plus, l'impact d'un format de quantification en virgule fixe primordiale pour l'implantation matérielle de décodeurs LDPC est mis en évidence. À partir de l'ensemble de ces éléments, un flot de conception spécifique est proposé afin de permettre l'implantation matérielle de décodeurs LDPC sur circuit FPGA en fonction de contraintes applicatives.

Une première étude est détaillée dans le chapitre 3. Il s'agit de la définition d'un schéma de codage pour fiabiliser un lien optique descendant. Cette définition est contrainte par l'implantation matérielle du décodeur LDPC à 10 Gbit/s. Nous avons donc proposé une

étude conjointe des codes LDPC et des algorithmes de décodage à entrées souples afin d'obtenir un couple code/algorithme fournissant un compromis efficace entre performances et complexité calculatoire. L'objectif était d'atteindre un pouvoir de correction similaire à celui proposé par le standard DVB-S2 qui utilise un code LDPC concaténé avec un code BCH. Nous rappelons que le débit visé est de 10 Gbit/s. À partir des couples codes LDPC et algorithmes de décodage permettant d'atteindre les performances de correction souhaitées sans l'utilisation du code BCH, nous avons développé une architecture de décodage semi-parallèle. Cette architecture, implantée sur circuit FPGA, est capable de décoder n'importe quel code QC-LDPC tout en atteignant de hauts débits. Après une phase d'expérimentations sur circuit FPGA, les performances mesurées montrent qu'une association de 5 décodeurs élémentaires permet d'atteindre 10 Gbit/s tout en maintenant un pouvoir de correction satisfaisant l'application ciblée. Ce schéma de codage à base de codes LDPC spécifiques permet de positionner notre implantation matérielle de décodeurs LDPC matériels parmi les plus efficaces de la littérature pour des applications spatiales.

Une seconde étude s'est focalisée sur le lien optique montant. Cette dernière est détaillée dans le chapitre 4. Dans ce nouveau cadre applicatif, le décodeur LDPC matériel est intégré dans un circuit FPGA embarqué dans le satellite. Le processus de décodage doit traiter des entrées binaires, dites entrées dures, pour des raisons technologiques. Afin d'assurer un pouvoir de correction élevé avec une complexité matérielle faible, une reformulation de l'algorithme Gallager E est proposée pour traiter ces entrées dures. Contrairement à ce qui a été préalablement proposé dans la littérature, cette reformulation repose sur un ordonnancement par couches horizontales. Les performances obtenues avec l'algorithme de décodage ont été comparées avec l'état de l'art. Elles s'avèrent, dans notre cadre applicatif, être le meilleur compromis entre complexité calculatoire et pouvoir de correction. Une architecture matérielle intégrant cet algorithme de décodage a été implantée sur circuit FPGA. L'étude des performances de ces décodeurs matériels démontre qu'ils permettent d'atteindre 10 Gbit/s tout en respectant des contraintes applicatives sévères.

Perspectives

À la suite de ces travaux de thèse, plusieurs perspectives de recherche, à court et moyen termes, se dégagent :

1. Dans le cadre des travaux concernant le lien optique montant, le décodeur LDPC est embarqué dans le satellite. Cela impose de forte contraintes en termes de complexité matérielle et de consommation énergétique. Une solution à court terme

pour améliorer les performances du décodeur (débit, complexité et consommation), est de porter l'architecture numérique sur une cible ASIC. Cela peut être effectué rapidement à l'aide de la description matérielle générique faite dans le langage VHDL. Il est alors possible d'envisager soit d'augmenter le nombre d'itérations et la valeur de Ψ afin d'améliorer le pouvoir de correction tout en conservant un débit de 10 Gbit/s, soit d'augmenter le débit. Ces améliorations pourraient permettre d'envisager de nouveaux cadres applicatifs, par exemple, des communications optiques satellitaires à 100 Gbit/s.

2. À moyen terme, il serait intéressant d'augmenter le pouvoir de correction du décodeur embarqué dans l'espace en développant un algorithme de décodage plus efficace que le Gallager E. Deux approches semblent prometteuses. La première consiste à utiliser un algorithme proche du Min-Sum avec un nombre très limité de bits de quantification. Des travaux ont déjà été proposés à ce sujet mais ils considèrent un processus de décodage à entrées souples. Une seconde méthode consisterait à exploiter la quatrième valeur possible lors du décodage selon l'algorithme Gallager E. En effet, des premières expérimentations faites à ce propos montrent que les performances d'un tel algorithme varient beaucoup en fonction des calculs des messages. L'idée serait donc d'établir un algorithme plus performant que le Gallager E tout en limitant la complexité calculatoire.

3. À moyen terme, une autre étude intéressante serait de modifier les unités de traitement OMS et Gallager E. Cela impliquerait de passer d'un fonctionnement séquentiel des unités de calculs à un fonctionnement parallèle. Cela permettrait de diminuer le nombre de cycles d'horloge nécessaires à l'exécution d'un nœud de parité de 2 d_c cycles à seulement 2 cycles d'horloge. Cependant, ce changement implique aussi de revoir entièrement l'organisation des données en mémoire. Cela aboutirait à une augmentation du nombre de BRAM à assigner.

4. À moyen terme, la conception d'un environnement de développement automatisé de décodeurs LDPC matériels permettrait de grandement simplifier le travail du concepteur. L'idée consisterait à développer un outil qui, à partir d'un code LDPC, génèrerait différentes architectures de décodeurs exploitant conjointement parallélisation intra trames et inter trames. Ces derniers offriraient de nouveaux compromis entre complexité matérielle, empreinte mémoire, débit et latence. Ainsi, en fonction de l'application ciblée, le concepteur pourrait sélectionner le décodeur qui représente le meilleur compromis par rapport à ses contraintes applicatives. Un tel outil permettrait à des personnes avec moins de connaissance en conception sur circuit FPGA d'évaluer la faisabilité de certaines applications en termes de complexité matérielle et de débit.

Bibliography

[1] B. Le Gal, C. Jego, and C. Leroux, "A flexible NISC-based LDPC decoder," *IEEE Transactions on Signal Processing*, vol. 62, no. 10, pp. 2469–2479, 2014.

[2] L. Canuet, "Fiabilisation des transmissions optiques satellite-sol," Thèse, 2018.

[3] K. E. Wilson and J. R. Lesh, "An overview of the galileo optical experiment (GOPEX)," 1993.

[4] K. Wilson, J. Lesh, K. Araki, and Y. Arimoto, "Overview of the ground-to-orbit lasercom demonstration (GOLD)," *Space Communications*, vol. 2990, 1998.

[5] G. D. Fletcher, T. R. Hicks, and B. Laurent, "The silex optical interorbit link experiment," *Electronics Communication Engineering Journal*, vol. 3, no. 6, pp. 273–279, 1991.

[6] R. Gallager, "Low-density parity-check codes," *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962.

[7] D. J. C. MacKay and R. M. Neal, "Near shannon limit performance of low density parity check codes," *Electronics Letters*, vol. 33, no. 6, pp. 457–458, 1997.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *Proceedings of the IEEE International Conference on Communications (ICC)*, vol. 2, 1993, pp. 1064–1070 vol.2.

[9] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[10] P. Elias, "Coding for noisy channels," *IRE convention Record*, vol. 3, pp. 37–46, 1955.

[11] C. Berrou, *Codes et turbocodes*. Springer Science & Business Media, 2007.

[12] R. Johannesson and K. S. Zigangirov, *Fundamentals of Convolutional Coding, Second Edition*. Wiley - IEEE Press, 2015.

[13] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transaction on Information Theory*, vol. IT-13, pp. 260–269, 1967.

[14] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, vol. 20, no. 2, pp. 284–287, 1974.

[15] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27, no. 3, pp. 379–423, 1948.

[16] 3GPP, "Multiplexing and channel coding," 2009.

[17] R. Garzón-Bohórquez, R. Klaimi, C. A. Nour, and C. Douillard, "Mitigating correlation problems in turbo decoders," in *Proceedings of the IEEE 10th International Symposium on Turbo Codes Iterative Information Processing (ISTC)*, 2018, pp. 1–5.

[18] 3GPP, *Study on New Radio (NR) Access Technology Physical Layer Aspects*. Technical Report TR-38.802, 2016.

[19] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[20] O. Afisiadis, A. Balatsoukas-Stimming, and A. Burg, "A low-complexity improved successive cancellation decoder for polar codes," in *Proceedings of the 48th Asilomar Conference on Signals, Systems and Computers*, 2014, pp. 2116–2120.

[21] K. Niu and K. Chen, "Stack decoding of polar codes," *Electronics Letters*, vol. 48, no. 12, pp. 695 –697, 2012.

[22] C. Xia, J. Chen, Y. Fan, C. Tsui, J. Jin, H. Shen, and B. Li, "A high-throughput architecture of list successive cancellation polar codes decoder with large list size," *IEEE Transactions on Signal Processing*, vol. 66, no. 14, pp. 3859–3874, 2018.

[23] ETSI, "Second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications (DVB-S2)," in *EN 302 307*, 2009.

[24] IEEE, "IEEE Standard for Local and Metropolitan Area Networks - Part 16: Air Interface for Fixed Broadband Wireless Access Systems," *IEEE Std 802.16-2001*, pp. 1–1200, 2002.

[25] ——, "IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," *IEEE Std 802.11n-2009*, pp. 1–565, 2009.

[26] R. Tanner, "A recursive approach to low complexity codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, 1981.

[27] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 599–618, 2001.

[28] T. J. Richardson, M. A. Shokrollahi, and R. L. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 619–637, 2001.

[29] Sae-Young Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes within 0.0045 db of the shannon limit," *IEEE Communications Letters*, vol. 5, no. 2, pp. 58–60, 2001.

[30] M. Fossorier, "Quasicyclic low density parity check codes," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2003, pp. 150–.

[31] Z. Wang and Z. Cui, "Low-complexity high-speed decoder design for quasi-cyclic LDPC codes," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 1, pp. 104–114, 2007.

[32] W. Zhang, S. Chen, X. Bai, and D. Zhou, "A full layer parallel QC-LDPC decoder for wimax and wi-fi," in *Proceedings of the IEEE 11th International Conference onPC(ASICON)*, 2015, pp. 1–4.

[33] S. Li, Q. Zhang, Y. Chen, and X. Zeng, "A high-throughput QC-LDPC decoder for Near-Earth Application," in *Proceedings of the IEEE 23rd International Conference on Digital Signal Processing (DSP)*, 2018, pp. 1–4.

[34] V. Pignoly, B. Le Gal, C. Jego, and B. Gadat, "High data rate and flexible hardware QC-LDPC decoder for satellite optical communications," in *Proceedings of the IEEE 10th International Symposium on Turbo Codes Iterative Information Processing (ISTC)*, 2018, pp. 1–5.

[35] I. B. Djordjevic, S. Sankaranarayanan, and B. V. Vasic, "Projective-plane iteratively decodable block codes for wdm high-speed long-haul transmission systems," *Journal of Lightwave Technology*, vol. 22, no. 3, pp. 695–702, 2004.

[36] F. R. Kschischang and B. J. Frey, "Iterative decoding of compound codes by probability propagation in graphical models," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 219–230, 1998.

[37] A. Cassagne, O. Hartmann, M. Léonardon, K. He, C. Leroux, R. Tajan, O. Aumage, D. Barthou, T. Tonnellier, V. Pignoly, B. Le Gal, and C. Jégo, "AFF3CT: A Fast Forward Error Correction Toolbox!" *SoftwareX*, vol. 10, p. 100345, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352711019300457

[38] D. J. MacKay and M. S. Postol, "Weaknesses of margulis and ramanujan-margulis low-density parity-check codes," *Electronic Notes in Theoretical Computer Science*, vol. 74, pp. 97–104, 2003.

[39] G. D. Forney, R. Koetter, F. R. Kschischang, and A. Reznik, *On the Effective Weights of Pseudocodewords for Codes Defined on Graphs with Cycles*, B. Marcus and J. Rosenthal, Eds. New York, NY: Springer New York, 2001.

[40] Changyan Di, D. Proietti, I. E. Telatar, T. J. Richardson, and R. L. Urbanke, "Finite-length analysis of low-density parity-check codes on the binary erasure channel," *IEEE Transactions on Information Theory*, vol. 48, no. 6, pp. 1570–1579, 2002.

[41] "OFC 2018: Post-show report," Available at : https://www.ofcconference.org/library/images/ofc/2018/Documents/OFC-2018-Show-Report.pdf [Online].

[42] F. Steiner, E. Ben Yacoub, B. Matuz, G. Liva, and A. G. i. Amat, "One and two bit message passing for SC-LDPC codes with higher-order modulation," *Journal of Lightwave Technology*, vol. 37, no. 23, pp. 5914–5925, 2019.

[43] ETSI, "DVB-RCS," in *EN 301 790*, 1999.

[44] CCSDS, *CCSDS 131.0-B-3 Recommandation for Space Data System Standards; TM Synchronization and Channel Coding.* CCSDS, 2017.

[45] ETSI, "Digital video broadcasting DVB; second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications, part 2: S2-extensions DVB-S2X," in *EN 301 307-2*, 2014.

[46] CCSDS, *CCSDS 131.0-B-2 Recommandation for Space Data System Standards; TM Synchronization and Channel Coding.* CCSDS, 2011.

[47] M. M. Mansour and N. R. Shanbhag, "High-throughput LDPC decoders," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 11, no. 6, pp. 976–996, 2003.

[48] E. Sharon, S. Litsyn, and J. Goldberger, "Efficient serial message-passing schedules for LDPC decoding," *IEEE Transactions on Information Theory*, vol. 53, no. 11, pp. 4076–4091, 2007.

[49] G. Wang and et al, "High throughput low latency LDPC decoding on GPU for SDR systems," in *Proceedings of the IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2013.

[50] B. Le Gal and C. Jego, "Low-latency software LDPC decoders," in *Proceedings of the IEEE Workshop On Signal Processing Systems (SiPS)*, 2017.

[51] B. Le Gal and C. Jego, "High-throughput multi-core LDPC decoders based on x86 processor," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1373–1386, 2016.

[52] B. Le Gal and C. Jego, "High-throughput LDPC decoder on low-power embedded processors," *IEEE Communication Letters*, 2015.

[53] B. Le Gal, C. Jego, and J. Crenne, "A high throughput efficient approach for decoding LDPC codes onto gpu devices," *IEEE Embedded Systems Letters*, vol. 6, no. 2, pp. 29–32, 2014.

[54] O. Boncalo and A. Amaricai, "Ultra high throughput unrolled layered architecture for QC-LDPC decoders," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, 2017, pp. 225–230.

[55] Juntan Zhang and M. P. C. Fossorier, "Shuffled iterative decoding," *IEEE Transactions on Communications*, vol. 53, no. 2, pp. 209–213, 2005.

[56] M. Li, C. A. Nour, C. Jégo, and C. Douillard, "Design and FPGA prototyping of a bit-interleaved coded modulation receiver for the DVB-T2 standard," in *Proceedings of the IEEE Workshop On Signal Processing Systems (SiPS)*, 2010, pp. 162–167.

[57] ETSI, "Implementation guidelines for a second generation digital terrestrial television broadcasting system (DVB-T2)," 2009.

[58] A. I. V. Casado, M. Griot, and R. D. Wesel, "LDPC decoders with informed dynamic scheduling," *IEEE Transactions on Communications*, vol. 58, no. 12, pp. 3470–3479, 2010.

[59] X. Liu, Y. Zhang, and R. Cui, "Variable-node-based dynamic scheduling strategy for belief-propagation decoding of LDPC codes," *IEEE Communications Letters*, vol. 19, no. 2, pp. 147–150, 2015.

[60] F. Guilloud, E. Boutillon, and J.-L. Danger, "Lambda-Min Decoding Algorithm of Regular and Irregular LDPC Codes," in *Proceedings of the International Symposium on Turbo Codes and Related Topics (ISTC)*, Brest, France, 2003, pp. 451–454.

[61] C. Jones, E. Valles, M. Smith, and J. Villasenor, "Approximate-min constraint node updating for LDPC code decoding," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, vol. 1, 2003, pp. 157–162 Vol.1.

[62] M. P. C. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," *IEEE Transactions on Communications*, vol. 47, no. 5, pp. 673–680, 1999.

[63] J. Chen and M. P. C. Fossorier, "Density evolution for two improved bp-based decoding algorithms of LDPC codes," *IEEE Communications Letters*, vol. 6, no. 5, pp. 208–210, 2002.

[64] S. Hemati, F. Leduc-Primeau, and W. J. Gross, "A relaxed min-sum LDPC decoder with simplified check nodes," *IEEE Communications Letters*, vol. 20, no. 3, pp. 422–425, 2016.

[65] M. Roberts and R. Jayabalan, "A modified normalized min - sum decoding algorithm for irregular LDPC codes," *International Journal of Engineering and Technology*, vol. 5, pp. 4881–4893, 2013.

[66] Juntan Zhang, M. Fossorier, Daqing Gu, and Jinyun Zhang, "Improved min-sum decoding of LDPC codes using 2-dimensional normalization," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 3, 2005, pp. 6 pp.–.

[67] M. Shaghaghi, Y. L. Guan, K. Cai, and Z. Qin, "Combined normalized and offset min-sum decoding over partial response channels," in *Proceedings of the 7th International Conference on Information, Communications and Signal Processing (ICICS)*, 2009, pp. 1–4.

[68] V. Savin, "Self-corrected min-sum decoding of LDPC codes," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2008, pp. 146–150.

[69] J.-B. Doré, "Optimisation conjointe de codes LDPC (Low Density Parity Check) et de leurs architectures de décodage et mise en oeuvre sur FPGA (Field Programmable Gate Array)," Thèse, 2007.

[70] F. Tadros, S. Eisa, H. H. Issa, and K. Shehata, "Modified scaled min sum LDPC decoder for DVB-S2/S2X/T2," in *Proceedings of the 30th International Conference on Microelectronics (ICM)*, 2018, pp. 172–175.

[71] Meng Xu, Jianhui Wu, and Meng Zhang, "A modified offset min-sum decoding algorithm for LDPC codes," in *Proceedings of the 3rd International Conference on Computer Science and Information Technology*, vol. 3, 2010, pp. 19–22.

[72] A. Atashbar-Tehrani and S. Ghazi-Maghrebi, "A new efficient adaptive normalized min-sum decoder for irregular LDPC codes in OFDM," in *Proceedings of the International Symposium on Networks, Computers and Communications (ISNCC)*, 2018, pp. 1–4.

[73] W. Ryan and S. Lin, *Channel Codes: Classical and Modern.* Cambridge University Press, 2009.

[74] J. Andrade, G. Falcao, V. Silva, and L. Sousa, "A survey on programmable LDPC decoders," *IEEE Access*, vol. 4, pp. 6704–6718, 2016.

[75] V. Pignoly, B. Le Gal, L. Barthe, B. Gadat, and C. Jego, "Fair Comparison of Hardware and Software LDPC Decoder Implementations for SDR Space Links," in *Proceedings of the IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2020.

[76] B. Le Gal and C. Jego, "Low-latency software LDPC decoders for x86 multi-core devices," in *Proceedings of the IEEE International Workshop on Signal Processing Systems (SiPS)*, 2017, pp. 1–6.

[77] R. Maunder, "Survey of ASIC implementations of LDPC decoders," 2016. [Online]. Available: https://eprints.soton.ac.uk/399259/

[78] M. K. Jain, M. Balakrishnan, and A. Kumar, "Asip design methodologies: survey and issues," in *Proceedings of the Fourteenth International Conference on VLSI Design*, 2001, pp. 76–81.

[79] B. Le Gal and C. Jego, "Fpga prototyping of an asip LDPC decoder for the dvb-t2 standard," in *Proceedings of the 2012 Conference on Design and Architectures for Signal and Image Processing*, 2012, pp. 1–2.

[80] Y. Delomier, B. Le Gal, J. Crenne, and C. Jego, "Model-based design of efficient LDPC decoder architectures," in *Proceedings of the IEEE 10th International Symposium on Turbo Codes Iterative Information Processing (ISTC)*, 2018, pp. 1–5.

[81] P. Hailes, L. Xu, R. G. Maunder, B. M. Al-Hashimi, and L. Hanzo, "A survey of FPGA-based LDPC decoders," *IEEE Communications Surveys Tutorials*, vol. 18, no. 2, pp. 1098–1122, 2016.

[82] Xilinx, *UltraScale Architecture and Product Data Sheet: Overview*, 2020.

[83] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity-check codes based on finite geometries: a rediscovery and new results," *IEEE Transactions on Information Theory*, vol. 47, no. 7, pp. 2711–2736, 2001.

[84] T. Wadayama, K. Nakamura, M. Yagita, Y. Funahashi, S. Usami, and I. Takumi, "Gradient descent bit flipping algorithms for decoding LDPC codes," *IEEE Transactions on Communications*, vol. 58, no. 6, pp. 1610–1614, 2010.

[85] G. Sundararajan, C. Winstead, and E. Boutillon, "Noisy gradient descent bit-flip decoding for LDPC codes," *IEEE Transactions on Communications*, vol. 62, no. 10, pp. 3385–3400, 2014.

[86] G. Falcao, J. Andrade, V. Silva, and L. Sousa, "Real-time DVB-S2 LDPC decoding on many-core GPU accelerators," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 1685–1688.

[87] B. Zhang, H. Liu, X. Chen, D. Liu, and X. Yi, "Low complexity DVB-S2 LDPC decoder," in *Proceedings of the IEEE 69th Vehicular Technology Conference (VTC)*, 2009, pp. 1–5.

[88] S. Mhaske, H. Kee, T. Ly, A. Aziz, and P. Spasojevic, "High-throughput fpga-based QC-LDPC decoder architecture," in *Proceedings of the IEEE 82nd Vehicular Technology Conference (VTC)*, 2015, pp. 1–5.

[89] N. Jiang, K. Peng, J. Song, C. Pan, and Z. Yang, "High-throughput QC-LDPC decoders," *IEEE Transactions on Broadcasting*, vol. 55, no. 2, pp. 251–259, 2009.

[90] R. M. D. Divsalar, H. Jin, "Coding theorems for turbo-like codes," in *Proceedings of the 36th Annual Allerton Conf. on Communication, Control, and Computing*, 1998.

[91] H. Jin, A. Khandekar, R. McEliece *et al.*, "Irregular repeat-accumulate codes," in *Proceedings of the 2nd International Symposium on Turbo codes and related topics (ISTC)*, 2000, pp. 1–8.

[92] Xiao-Yu Hu, E. Eleftheriou, and D. Arnold, "Progressive edge-growth tanner graphs," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 2, 2001, pp. 995–1001 vol.2.

[93] Tao Tian, C. Jones, J. D. Villasenor, and R. D. Wesel, "Construction of irregular LDPC codes with low error floors," in *Proceedings of the IEEE International Conference on Communications (ICC)*, vol. 5, 2003, pp. 3125–3129 vol.5.

[94] "Accumulate repeat accumulate codes," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 1, 2004, pp. 509–513 Vol.1.

[95] "Accumulate repeat accumulate codes," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 1, 2004, pp. 509–513 Vol.1.

[96] D. Divsalar, C. Jones, S. Dolinar, and J. Thorpe, "Protograph based LDPC codes with minimum distance linearly growing with block size," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 3, 2005, pp. 5 pp.–.

[97] D. Divsalar, S. Dolinar, and C. Jones, "Construction of protograph LDPC codes with linear minimum distance," in *Proceedings of the IEEE International Symposium on Information Theory (ISIT)*, 2006, pp. 664–668.

[98] V. Beneš, "Mathematical theory of connecting networks and telephone traffic," ser. Mathematics in Science and Engineering. Elsevier, 1965, vol. 17.

[99] C. Marchand and E. Boutillon, "LDPC decoder architecture for DVB-S2 and DVB-S2X standards," in *Proceedings of the IEEE Workshop On Signal Processing Systems (SiPS)*, 2015, pp. 1–5.

[100] T. Xie, B. Li, M. Yang, and Z. Yan, "Memory compact high-speed QC-LDPC decoder," in *Proceedings of the IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC)*, 2017.

[101] M. D. Elftmann. (2018) Xilinx On-Orbit Reconfigurable Kintex UltraScale FPGA Technology for Space. [Online]. Available: https://indico.esa.int/event/232/contributions/2161/

[102] T. C. . Chang and Y. T. Su, "Dynamic weighted bit-flipping decoding algorithms for LDPC codes," *IEEE Transactions on Communications*, vol. 63, no. 11, pp. 3950–3963, 2015.

[103] K. Ma, Y. Li, and H. Zhang, "Fast weighted bit flipping algorithm for higher-speed decoding of low-density parity-check codes," *China Communications*, vol. 10, no. 9, pp. 114–119, 2013.

[104] M. Mitzenmacher, "A note on low density parity check codes for erasures and errors," 1998.

[105] T. J. Richardson and R. L. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Information Theory*, vol. 47, pp. 599–618, 2001.

[106] F. Ghaffari, B. Unal, A. Akoglu, K. Le, D. Declercq, and B. Vasić, "Efficient FPGA implementation of probabilistic Gallager B LDPC decoder," in *Proceedings of the 24th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2017, pp. 178–181.

[107] O. A. Rasheed, P. Ivaniš, and B. Vasić, "Fault-tolerant probabilistic gradient-descent bit flipping decoder," *IEEE Communications Letters*, vol. 18, no. 9, pp. 1487–1490, 2014.

[108] B. Unal, F. Ghaffari, A. Akoglu, D. Declercq, and B. Vasić, "Analysis and implementation of resource efficient probabilistic Gallager B LDPC decoder," in *Proceedings of the 15th IEEE International New Circuits and Systems Conference (NEWCAS)*, 2017, pp. 333–336.

[109] Jinghu Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and Xiao-Yu Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Transactions on Communications*, vol. 53, no. 8, pp. 1288–1299, 2005.

[110] B. Vasic, D. Declercq, and S. K. Planjery, "Decoder diversity architectures for finite alphabet iterative decoders," in *Proceedings of the 48th Asilomar Conference on Signals, Systems and Computers*, 2014, pp. 131–135.

[111] F. Ghaffari and B. Vasic, "Probabilistic gradient descent bit-flipping decoders for flash memory channels," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, 2018, pp. 1–5.

[112] V. Pignoly, B. Le Gal, L. Barthe, B. Gadat, and C. Jego, "High Speed LDPC Decoding for Optical Space Link," in *Proceedings of the IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, 2020.