Acronymes

Acronymes

3D-TCC	Three-Dimensional Turbo Convolutional Code
3G	Third Genration
3GPP2	3rd Generation Partnership Project 2
4G	Fourth Generation
ADSL	Asymmetric Digital Subscriber Line
APSK	Amplitude and Phase Shift Keying
ARQ	Automatic Repeat Request
AVC	Advanced Video Coding
AWGN	Additive White Gaussian Noise
BCH	Bose–Chaudhuri–Hocquenghem
BCJR	Bahl-Cocke-Jelinek-Raviv
BD	Blu-ray Disc
BER	Bit Error Rate
BP	Belief Propagation
BPSK	Binary Phase Shift Keying
CC	Convolutional Code
CCSDS	Consultative Committee for Space Data Systems
CD	Compact Disk
CDMA	Code Division Multiple Access
CL	Constraint Length
CRC	Cyclic Redundancy Check
DAT	Digital Audio Tape
DVB	Digital Video Broadcasting
DVD	Digital Versatile Disc
ECC	Error Correcting Code
FDX	Transmission Full-Duplex
FEC	Forward Error Correction
FER	Frame Error Rate
GF	Galois Field
GSM	Global System Mobil
HDTV	High Definition Television
HDX	Transmission Half-Duplex
LCM	Least Common Multiple
LDPC	Low Density Parity Check
LLR	Logarithm of Likelihood Ratio
LTE	Long Term Evolution
MAP	Maximum A Posteriori
MCW	Mean Column Weight
OFDM	Orthogonal Frequency Division Multiplexing
PCGC	Parallel Concatinated Gallager Code
PSK	Phase-Shift Keying

QPSK	Quadrature Phase Shift Keying
RSC	Recursive Systematic Convolutional
RSV	Reed-Solomon & Viterbi
SHF	Super High Frequency
SISO	Soft Input Soft Output
SNR	Signal to Noise Ratio
TCC	Turbo Convolutional Code
TCM	Turbo Coded Modulation
TPC	Turbo Product Code
UHF	Ultra High Frequency
UMTS	Universal Mobile Telecommunications System
VHF	Very High Frequency
WIMAX	Worldwide Interoperability for Microwave Access

Liste des figures

Liste des Figures

Liste des tableaux

Liste des tableaux

Sommaire

Sommaire

Chapitre 3

Chapitre 4

Chapitre 5

.

Introduction

Introduction

Dans la plupart des systèmes de communications sans fil, en particulier spatiales et mobiles de prochaine génération, un code puissant de correction d'erreurs est nécessaire. Ceci, permet d'améliorer la robustesse de transmission de données sur de tels canaux très hostiles.

La théorie des codes correcteurs d'erreurs est une branche de l'ingénierie et des mathématiques qui permet une transmission ou un stockage plus fiable des données. Avec l'explosion de la quantité d'information échangée depuis l'arrivée d'internet, de plus en plus de scientifiques s'intéressent à cette branche surtout que la fiabilité des supports d'information est loin d'être sûre à 100%, dans le sens où le bruit ou toute forme d'interférence entraîne souvent une distorsion des données. Pour faire face à cette situation indésirable mais inévitable, une certaine forme de redondance est incorporée dans les données originales. Avec cette redondance, même si des erreurs sont introduites (jusqu'à un certain niveau de tolérance), les informations originales peuvent être récupérées, ou au moins la présence d'erreurs peut être détectée. Jusqu'à présent, de nombreux codes de détection et de correction d'erreurs, pour différentes utilisations, ont été développés. Le lien commun entre eux est l'ajout de redondance au message original pour permettre au récepteur de détecter l'erreur et de la corriger. Ceci est crucial pour certaines applications où le ré-envoi du message n'est pas possible (par exemple, pour les communications en temps réel). Le problème crucial à résoudre est alors de concevoir un tel codeur tout en répondant à un certain nombre de critères spécifiques. En termes de performance, il y a deux critères les plus importants : la bonne convergence dans la région des cascades des courbes BER et le plancher du taux d'erreur très bas. En termes de complexité, il y a aussi deux critères à considérer : la faible complexité de calcul des algorithmes de décodage et le plus petit nombre d'itérations pour atteindre la performance désirée. Les codes les plus puissants ont été développés spécifiquement pour étudier les règles de conception de codage de canal, représentées par les critères décrits ci-dessus, afin de détecter et de corriger autant d'erreurs que possible de la manière la plus efficace et avec moins d'énergie dissipée.

Les codes correcteurs d’erreurs existants peuvent bien fonctionner dans certains cas, mais pas dans toutes les conditions et tous les environnements. Par conséquent, le besoin vital de créer de nouvelles méthodes et de réviser les anciennes techniques est bien perçu. De ce fait, nous avons proposé dans cette thèse une nouvelle méthode qui peut être une bonne alternative pour les systèmes de communication d'aujourd'hui.

Cette thèse est structurée comme suit : Dans le chapitre1, l’identification du problème de recherche et certains travaux pertinents sont présentés. Un aperçu sur la théorie de l’information est donné au chapitre 2, en décrivant les différents éléments d'un système de communication numérique. Les techniques des codes correcteurs d’erreurs étudiés dans cette thèse sont présentées dans le chapitre 3. Dans le chapitre 4, un schéma de codage et de décodage de la méthode proposée est décrit. L’analyse de la complexité de décodage en termes d'opérations mathématiques pour différentes techniques de décodage en comparaison avec notre méthode est présentée au chapitre 5. Nous présentons également, dans ce chapitre, les résultats des simulations qui montrent le compromis établi entre performance et complexité assuré par notre méthode.

Chapitre 1

Ce chapitre établit le cadre de la recherche en décrivant le contexte de l'étude et le problème qui a motivé notre recherche.

1.1 Contexte et problématique :

Claude Shannon était un ingénieur en génie électrique et mathématicien, il est considéré comme le père de la théorie de l'information. L'origine de cette théorie est la combinaison des mathématiques, des statistiques, du traitement du signal et de l'électronique, conduisant alors à la publication de son article fondateur "théorie mathématique de la communication" [1] en 1948, dans la revue interne de laboratoire Bell.

Lorsque l'information est formalisée d'une manière abstraite et mathématique, le sens du message n'aura plus d'importance. En effet, l'information peut être maintenant mesurée et quantifiée. Shannon montre dans son article que chaque canal de transmission peut transmettre d’une manière fiable autant d'information que possible. En fait, en raison du bruit latent dans les canaux de transmission, des erreurs peuvent surgir pendant la réception du message. Afin de surmonter cela, Shannon prouvait l'existence d'un système de correction d'erreur, basé sur l’ajout d’informations redondantes au message original permettant de corriger entièrement les effets de dégradation et donc de reconstituer complètement les données à la réception. Cependant, Shannon n'a pas fourni de conseils sur la façon de concevoir un tel code correcteur d’erreurs [2].

La Figure 1.1 présente un exemple d’amélioration du BER en fonction du SNR avec et sans code correcteur d’erreurs.

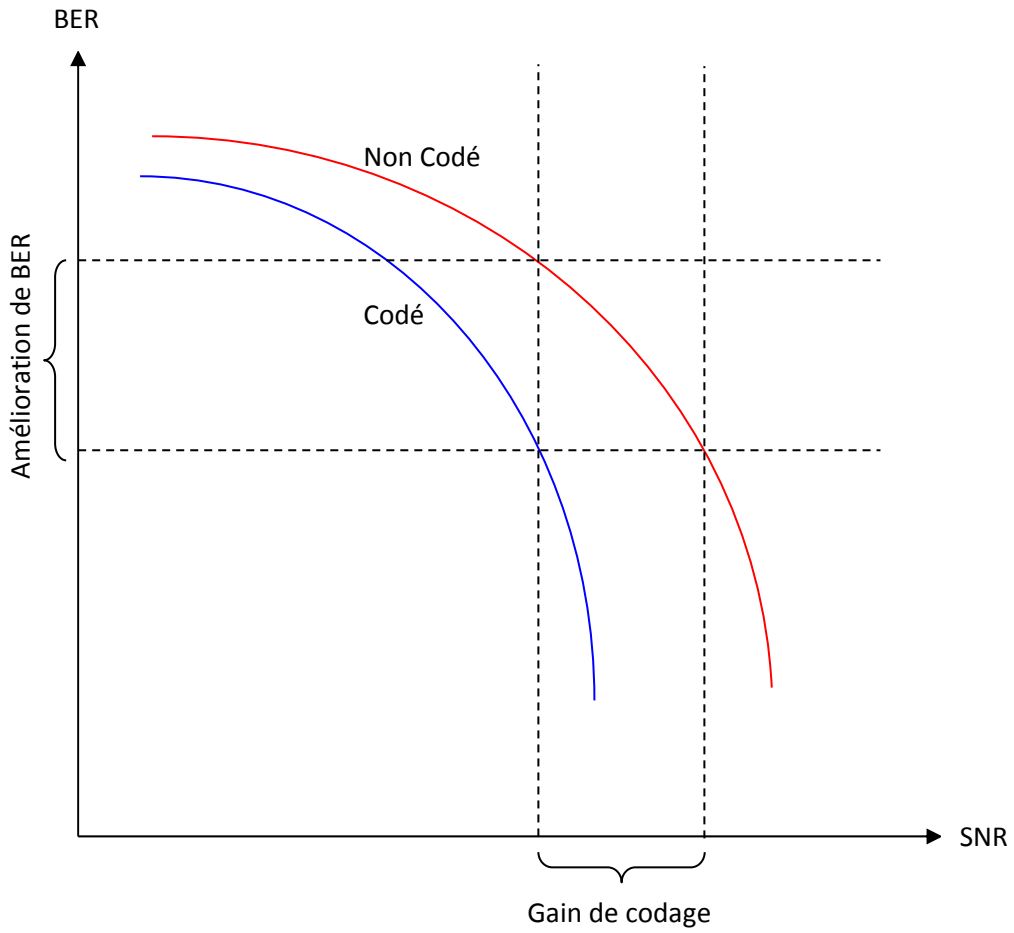


Figure 1.1 : BER des systèmes sans et avec code correcteur d'erreurs

Dans la plupart des systèmes de communication sans fil nécessitants une probabilité d'erreur très faible, par exemple dans les communications spatiales (exploration de l'univers) ou dans les communications mobiles des prochaines générations, un puissant code correcteur d'erreurs est nécessaire pour améliorer la robustesse de la transmission de données sur un canal très bruyant. Les applications de télémesure ou de contrôle-commande d'aéronef télé piloté nécessitent elles aussi des taux d'erreurs particulièrement faibles avec le moindre coût possible.

Moins de complexité, plus de performance, sont alors les deux critères spécifiques qui doivent êtres considérés lors de la conception d'un tel code correcteur d'erreurs.

1.1.1 En termes de performance :

La capacité de correction est parmi les métriques d'évaluation de performance d'un code correcteur d'erreurs. La courbe de BER (Taux d'erreur binaire) est divisée en deux régions (la région de convergence et la région de plancher d'erreur). Dans la région de convergence, un accroissement relativement faible de la qualité de transmission entraînera une amélioration significative des performances de décodage, dans le cas typique des communications sans fils du type téléphonie (3G, 3G+, 4G) des taux d'erreurs binaires (BER) maximales autour de 10^{-2} à 10^{-3} sont requis [3]. D'autre part, dans la région du plancher d'erreur, l'accroissement de la qualité de transmission n'aboutit qu'à une amélioration marginale des performances de décodage à mesure que la qualité du canal s'améliore. Par conséquent, cette région est particulièrement contraignante pour les applications nécessitant des taux d'erreurs très faibles [2]. Les taux d'erreurs ciblés peuvent être compris entre 10^{-7} et 10^{-9} selon le type d'application. Les applications utilisant de tels taux d'erreurs faibles incluent les communications par satellite ainsi que les transmissions filaires via un conducteur électrique ou une fibre optique [3].

1.1.2 En termes de complexité :

La complexité peut se représenter par la superficie du dispositif et la dissipation d'énergie. L'un des principaux défis dans la conception des dispositifs sans fil alimentés par batteries est la minimisation de leur consommation d'énergie [4,5]. On sait que les décodeurs à correction d'erreur directe (FEC) sont responsables d'une grande partie de la consommation d'énergie dans tels dispositifs (Complexité algorithmique, l'utilisation de la mémoire et le temps d'exécution) [6,7]. Etant donné qu'un décodeur puissant est un processus itératif, il a besoin d'un grand nombre d'opérations, le coût de mémoire devient de plus en plus important, entraînant ainsi une augmentation de la surface physique dédiée au dispositif décodeur.

Le critère de faible complexité de calcul peut avoir deux aspects :

• Nombre d'itérations nécessaires pour le décodage : le plus petit nombre d'itérations pour atteindre la performance désirée (c'est-à-dire un processus de décodage rapide).

• Complexité du calcul : La nouvelle mise à jour entraîne non seulement de meilleures performances d'erreur et moins d'itérations en moyenne, mais réduit également le nombre d'opérations mathématiques nécessaires pour décoder chaque bit.

Notons que le nombre total d'opérations (les opérations mathématiques et le nombre d'itérations) donne une indication sur la complexité de calcul globale pour un algorithme de décodage.

Par conséquent, les exigences de plus faibles taux d'erreurs (BER) pour une qualité de transmission constante peuvent entrainer une complexité algorithmique plus élevée D'autre part, cette solution a un impact direct sur le coût dans un contexte de communications satellitaires. Par exemple, chaque dB de gain de codage (ce qui permet de réduire la puissance, la taille des antennes…) conduit à une diminution du coût de dizaines de l'ordre de millions de dollars pour le réseau de communications avec l'espace lointain.

1.2 Travaux importants dans la conception des codes correcteurs d'erreurs :

Une grande partie de la recherche sur les codes de correction d'erreurs est consacrée soit à l'amélioration de performance, soit à réduire la complexité, soit à trouver un compromis entre les deux.

Concernant les critères décrits précédemment, des codes correcteurs d'erreurs plus puissants ont été développés afin de répondre aux règles de conception de codage de canal. Malheureusement, certains codes sont bons pour un critère, mais mauvais pour un autre, ce qui rend le problème de la conception de codes correcteurs d'erreurs efficaces très difficile.

Parmi ces codes, on retrouve le puissant Turbo code convolutionnel (TCC) de Berrou et al. [8,9], qui est le premier code à atteindre la limite énoncée par Shannon. En raison

de leur nature de décodage itératif, il est difficile d'obtenir un taux d'erreur binaire (BER) très faible dans toute la gamme de valeurs du rapport signal/bruit (SNR) en raison de son phénomène de plancher d'erreur [10].

Cependant, les chercheurs ont toujours visé à réduire le plancher d'erreur des Turbo codes. Ainsi, le turbo code tridimensionnel (3D-TCC) a été proposé dans [11,12]. Malheureusement, cela s'est fait au prix d'une légère augmentation de la complexité de décodage par rapport aux turbo codes classiques.

Dans [13,14], les auteurs ont proposé d'autres méthodes pour réduire le plancher d'erreur de turbo code classique. Cependant, la complexité du décodage, le processus d'entrelacement et la latence rendent le TCC inadapté à certaines situations.

L'étape de l'entrelacement est considérée comme le cœur du TCC, elle consiste à distancer les erreurs en rafale afin d'avoir une meilleure correction. En outre, l'entrelaceur garantit si une séquence d'information qui génère un mot codé de poids faible à l'entrée du premier codeur est entrelacée selon certains critères, ce processus aboutit à un mot codé de grand poids sous l'autre codeur. Plusieurs travaux, tels que [15,16], visaient à réduire la complexité de l'entrelacement pour obtenir un décodage à haute vitesse, réduire les besoins de stockage et diminuer la consommation d'énergie.

Pour réduire le nombre d'itérations du TCC, Hyeji. K et al [17] ont proposé une nouvelle unité de critère d'arrêt basée sur le contrôle de redondance cyclique (CRC). Concernant les applications de TCC à courte trame, des chercheurs [18] ont conçu un schéma conjoint de détection et d'arrêt précoce efficace avec une complexité réduite.

D'autre part, les progrès des codes LDPC par Mackay [19, 20] surmontent les TCC en termes de performance et d'erreur plancher dans les rangs de SNR élevés, laissant les TCC adaptés seulement pour les faibles SNR's.

Actuellement, les codes LDPC, selon leur capacité de plus en plus puissante et leur faible complexité de décodage, continuent d'attirer l'attention de la communauté scientifique sur le codage. Cependant, la mise en œuvre de tels codes s'avère très complexe en raison de la quantité importante de mémoire nécessaire pour stocker leurs matrices de contrôle de parité. Des encodeurs intelligents comme dans le travail de richardson et al [21] contribuent à réduit la complexité d'encodage et le rend presque linaire. Les codes LDPC quasi-cycliques (QC-LDPC) [22] peuvent être aussi un bon candidat pour résoudre le problème de la mémoire.

Les systèmes de communication qui ont choisi les codes LDPC au détriment des TCC ont ajouté un code de correction d'erreur externe supplémentaire pour corriger les erreurs occasionnelles qui n'ont pas pu être corrigés par le code LDPC. Par exemple, les normes récentes de diffusion vidéo numérique telles que DVB-S2 [23] utilisent un code externe BCH pour éliminer les erreurs occasionnelles de décodage LDPC.

De meilleures performances sont obtenues avec l'utilisation de TCC en tant que code externe avec le code interne LDPC. Bien que l'efficacité de cette concaténation en série ait été prouvée pour les communications dans l'espace lointain [24], elle augmente la complexité de calcul.

Plus tard, Damian A. Morero et al. [25] ont présenté une nouvelle stratégie en code de concaténation en série pour améliorer la région de plancher d'erreur, en utilisant le code LDPC et le Turbo code produit, leur approche réduit considérablement la complexité.

Il est nécessaire de mentionner le PCGC de H. Behairy et al. [26,27] qui présente une concaténation parallèle des codes LDPC avec différents poids de colonne et utilise la structure turbo sans entrelaceur. Cette approche conduit à une bonne performance malgré son délai de décodage.

De plus, Satoshi Gounai et al. [28] ont proposé un nouveau code concaténé qui combine deux codes différents (Recursive Systematic Convolutional (RSC) et LDPC)

en parallèle. En fait, leur façon de calculer et d'utiliser les informations extrinsèques entre les décodeurs n'obtient pas de meilleurs résultats.

Le code polaire est un nouveau paradigme de codage inventé par E. Arikan et al [29]. Il peut atteindre la capacité du canal pour tous les types de canaux connus avec des longueurs de code élevées. Sa construction est basée sur un phénomène de polarisation de canal. Cependant, sa performance est moins significative comparé avec le LDPC et Turbo codes. Par ailleurs, les opérations de décodage sont implémentées avec une complexité inférieure.

1.3 Conclusion

Prenant en considération les efforts précédents mentionnés ci-dessus, en particulier l'idée de Satoshi [28], cette thèse propose un code concaténé en parallèle qui combine le code LDPC et le code convolutif dans le principe turbo. L’objectif est de réaliser un compromis optimal entre la complexité et la performance. La contribution principale réside dans la méthode pour calculer les informations extrinsèques échangées entre les décodeurs d'une manière efficace permettant à chacun d'être compréhensible à l'autre [30]. Par conséquent, la complexité et les performances de calcul seront comparées au LDPC unique, au TCC conventionnel, au 3D-TCC, au PCGC et au code LDPC-BCH concaténé en série sur le canal AWGN, pour un taux de codage de 1/3 et une longueur de bloc de 1024 bits. Cette comparaison assurera la domination du code parallèle concaténé proposé par rapport aux méthodes précédentes en termes de complexité et de performance.

Finalement, lors de la conception d'un lien de communication, la sélection du code de correction d'erreurs nécessite un compromis de plusieurs paramètres. Les paramètres dominants incluent généralement l'efficacité énergétique (faible consommation d’énergie), le taux de codage (un taux de codage élevé peut être nécessaire pour respecter une contrainte de bande passante avec les modulations disponibles) et la longueur de bloc (les blocs plus courts réduisent la latence sur les liaisons à faible débit) [31].

Chapitre 2

La théorie de l'information introduite par Shannon en 1948 a subi de nombreux changements et elle a débouché sur de nouvelles applications. Dans ce chapitre, nous discutons des éléments de la théorie de l'information et illustrons ses applications pour une transmission numérique fiable. En particulier, nous présentons quelques techniques avancées de codage de canal couramment utilisées. Nous montrons aussi par ce chapitre que, pour un ensemble donné de conditions de canal, il existe une capacité de Shannon, ou un taux maximum de transmission fiable.

2.1 Eléments d'un système de communication numérique

La Figure 2.1 illustre le diagramme fonctionnel et les éléments de base d'un système de communication numérique. La sortie de la source peut être soit un signal analogique, tel qu'un signal audio ou vidéo, soit un signal numérique quelconque, qui est discret dans le temps et qui a un nombre fini de caractères de sortie [32]. Dans un système de communication numérique, les messages produits par la source sont convertis en une séquence de chiffres binaires. Idéalement, nous voulons représenter la source (message) par le moins de chiffres binaires possibles. En d'autres termes, nous recherchons une représentation efficace de la sortie de source en éliminant les redondances. Ce processus est appelé codage de source ou compression de données.

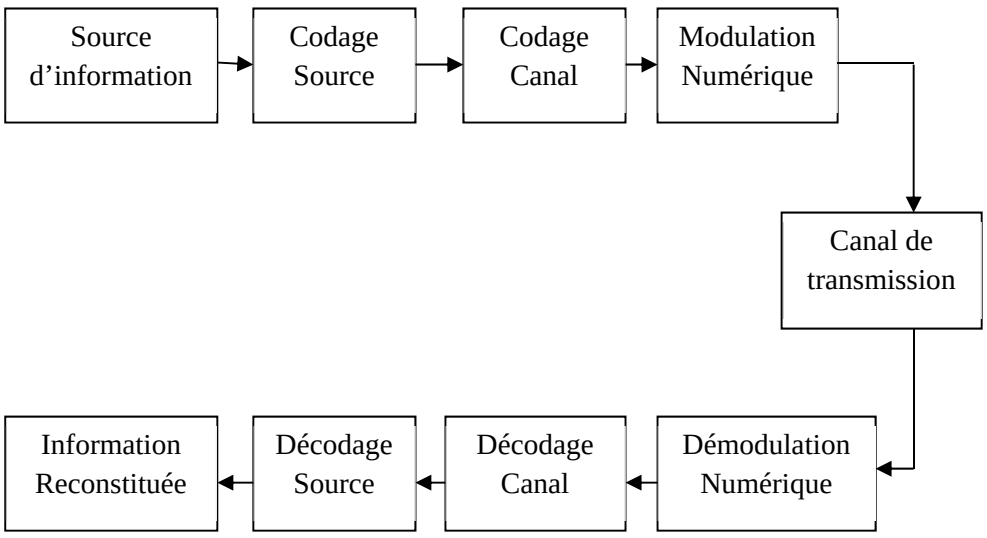


Figure 2.1 : Schéma illustratif d'une chaine de transmission typique

La séquence de chiffres binaires du codeur source, que nous appelons la séquence d'information, est transmise au codeur de canal. Il a pour but d'introduire, de manière contrôlée, une certaine redondance dans la séquence d'informations binaires qui peut être utilisée au niveau du récepteur pour surmonter les effets de bruit et d'interférence dus au canal. Ainsi, la redondance ajoutée sert à augmenter la fiabilité des données reçues et améliore la fidélité du signal reçu. En effet, la redondance dans la séquence d'information aide le récepteur à décoder la séquence d'information désirée. Par exemple, une forme (triviale) de codage de la séquence d'information binaire consiste simplement à répéter chaque chiffre binaire m fois, où m est un entier positif. Le codage sophistiqué (non trivial) implique de prendre k bits d'information à la fois et de coder chaque k séquences dans une séquence unique de n bits, appelée « mot de code ». La quantité de redondance introduite en codant les données de cette manière est mesurée par le rapport n/k. L'inverse de ce rapport, à savoir k/n, est appelé taux de codage.

La séquence binaire à la sortie du codeur de canal est transmise au modulateur numérique, qui sert d'interface au canal de communication. Le but principal du modulateur numérique (pour un canal de communication de type conducteur électrique) est de mapper la séquence d'informations binaires en formes d'onde de signal. Pour plus de détails, supposons que la séquence d'information codée doit être transmise en un seul bit à la fois avec un débit de Rbits par seconde (bits/s). Le modulateur numérique traduit le chiffre binaire 0 en une autre forme d'onde $s_0(t)$ et le chiffre binaire 1 en une forme d'onde $s_1(t)$. De cette manière, chaque bit du codeur de canal est transmis séparément. Nous appelons cette modulation, une modulation binaire. En variante, les symboles binaires entrants peuvent être segmentés en blocs de b-bits, le modulateur peut alors transmettre b-bits d'information codés à la fois en utilisant $M = 2^b$ formes d'onde distinctes $s_i(t)$, $i = 0, 1, \ldots, M-1$, avec une forme d'onde pour chacun des b-bits de la séquence. Nous appelons cette modulation M-ary ($M > 2$). Notez qu'une nouvelle séquence de b-bits entre dans le modulateur toutes les b/R secondes. Par conséquent, lorsque le débit binaire du canal R est fixé, la durée disponible pour transmettre l'une des M formes d'onde correspondant à une séquence de b-bits est de b fois la période de temps dans un système qui utilise une modulation binaire.

Le canal de communication est le support physique utilisé pour acheminer le signal de l'émetteur au récepteur. Il peut être sous forme de lignes filaires, de câbles de fibres optiques ou sans fil (atmosphère). Quel que soit le support physique utilisé pour la transmission de l'information, le résultat est toujours le même à savoir : le signal transmis est altéré de manière aléatoire par divers phénomènes possibles, tels que le bruit thermique additif généré par les dispositifs électroniques ; le bruit artificiel, le bruit atmosphérique, par exemple les décharges de foudre électriques pendant les orages, les champs magnétiques générés par les fils électriques, ...etc.

A l'extrémité de la réception d'un système de communication numérique, le démodulateur numérique traite la forme d'onde transmise endommagée par le canal et la traduit en une séquence de nombres représentant des estimations des symboles de données transmis (binaires ou M-ary). Cette séquence de nombres est transmise au décodeur de canal, qui tente de reconstruire la séquence d'information d'origine à partir de la connaissance du code utilisé par le codeur de canal et de la redondance contenue dans les données reçues.

Le bon fonctionnement du démodulateur et du décodeur dépend de la fréquence avec laquelle les erreurs se produisent dans la séquence décodée. Plus précisément, la probabilité moyenne d'une erreur de bit en sortie du décodeur est une mesure de la performance de la combinaison démodulateur-décodeur. En général, la probabilité d'erreurs dépend des caractéristiques du code, des types de formes d'ondes utilisées pour transmettre l'information sur le canal, de la puissance de l'émetteur, des caractéristiques du canal (c'est-à-dire, la quantité de bruit, la nature de l'interférence), et de la méthode de démodulation et de décodage.

En tant qu'étape finale, lorsqu'une sortie analogique est désirée, le décodeur de source accepte la séquence de sortie provenant du décodeur de canal et à partir de la connaissance de la méthode de codage de source utilisée, tente de reconstruire le signal original. En raison des erreurs de décodage de canal et de la distorsion éventuelle introduite par le codeur source, et éventuellement le décodeur source, le signal à la sortie du décodeur source est une approximation du signal original. La différence ou une

certaine fonction de la différence entre le signal original et le signal reconstruit est une mesure de la distorsion introduite par le système de communication numérique.

2.1.1 Modélisation de canal :

Le canal est défini comme un seul chemin pour transmettre des signaux dans une seule direction HDX ou dans les deux directions FDX. L'objectif de la modélisation de canaux sans fil est de trouver des modèles analytiques utiles pour les variations du canal.

L'inconvénient le plus important des communications sans fil est l'effet d'évanouissement généré par ces canaux d'où l'appellation canaux à évanouissement. Diverses propriétés, telles que la propagation par trajets multiples, la mobilité du terminal et l'interférence de l'utilisateur, donnent lieu à un canal à évaluer avec des paramètres variables dans le temps.

L'évanouissement dans les canaux sans fil peut être classé à grande échelle et à petite échelle. L'évanouissement à grande échelle implique la variation de la moyenne de la puissance du signal reçu sur de grandes distances par rapport à la longueur d'onde du signal. Les évanouissements à petite échelle impliquent des schémas de modulation et de démodulation robustes à ces variations. Dans les variations à petite échelle, la réflexion, la diffraction et la diffusion dans le canal de communication provoquent des variations rapides du signal reçu. Les signaux réfléchis seront reçus avec des amplitudes et des délais différents. Ce phénomène est appelé évanouissement par trajets multiples. Le mouvement relatif entre l'émetteur et le récepteur (ou vice versa) fait que la fréquence du signal reçu est décalée par rapport à celle du signal transmis. Le décalage de fréquence, ou fréquence Doppler, est proportionnel à la vitesse du récepteur et à la fréquence du signal transmis.

Un signal subit un affaiblissement lent lorsque la bande passante du signal est beaucoup plus grande que l'étalement du spectre Doppler (définie comme une mesure de l'élargissement spectral causé par la fréquence Doppler). La combinaison de l'évanouissement par trajets multiples et de ses variations temporelles provoque une

dégradation sévère du signal reçu. Cette dégradation de la qualité du signal reçu doit être compensée par diverses techniques telles que le codage de canal.

Il existe différents types de canaux - le canal AWGN, le canal Rayleigh et le canal Rician. Dans le but de concevoir et d'optimiser les structures de réception pour les systèmes de communication numérique, il est obligatoire de construire des modèles mathématiques représentant les caractéristiques typiques de ces canaux.

Généralement, les performances des systèmes de communication sur ces canaux sans fil sont analysées en utilisant la probabilité du taux d'erreur binaire (BER) en fonction du rapport signal/bruit. Le BER diminue pour un rapport signal/bruit élevé sur le récepteur. Cependant, les performances réelles du système sans fil dépendent de plusieurs problèmes de mise en œuvre et de caractéristiques du canal sans fil [33].

2.1.2 Capacité de canal :

Le modèle de canal le plus souvent utilisé est le canal AWGN (Bruit additif blanc gaussien). C'est un modèle de canal qui peut être exprimé comme une addition linéaire de bruit à large bande ou blanc avec une densité spectrale constante et une amplitude de distribution gaussienne. Tout système sans fil dans le canal AWGN peut être exprimé par $y = x + n$, où n est le bruit blanc additif gaussien, x et y sont respectivement les signaux d'entrée et de sortie. Cependant, ce canal est un modèle très utile pour de nombreux supports de communication par satellite et dans l'espace lointain. Le canal AWGN peut être illustré comme dans la Figure 2.2.

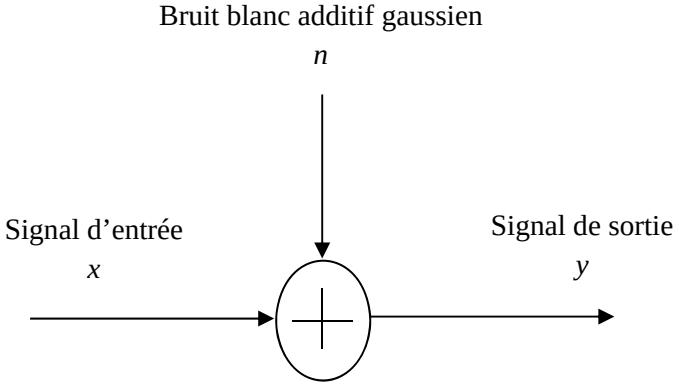


Figure 2.2 : Modèle d'un canal à bruit additif blanc gaussien

La capacité de canal est une fonction des caractéristiques de canal telles que les puissances de signal et de bruit reçues. En fait, un certain nombre de formules différentes sont couramment utilisées pour calculer la capacité du canal. Pour le canal à bruit additif blanc gaussien, la capacité de canal peut être exprimée par (2.1) [34].

$$C = W\ log_2(1 + \frac{S}{N}) \tag{2.1}$$

où W est la largeur de bande occupée par le signal porteur d'informations, S est la puissance du signal et $N = WN_0$ est la variance du bruit gaussien. En notant le débit binaire par R (en bits par seconde), l'énergie par bit par E_b (en joules) et la densité spectrale du bruit N_0 (mesurée en watts par hertz ou joules), Substituons cela dans l'équation (2.1), nous obtenons :

$$C = W\ log_2(1 + \frac{E_b R}{N_0 W}) \tag{2.2}$$

Le rapport $\tau = \frac{R}{W}$ est appelé rendement spectral ou débit spectral mesuré en bits par seconde par Hertz.

E_b et N_0 ont la même unité, donc le rapport $\frac{E_b}{N_0}$ est sans dimension; mais il est fréquemment exprimé en décibels.

D'après le théorème de codage, nous savons que pour pouvoir communiquer avec une probabilité d'erreur arbitrairement faible, le débit de transmission ne doit pas dépasser la capacité du canal, c'est-à-dire $R < C$. Appliquer cette contrainte à l'équation. (2.2), nous obtenons une borne inférieure sur le $\frac{E_b}{N_0}$ nécessaire pour une efficacité spectrale donnée :

$$\frac{R}{W} < \frac{C}{W} = log_2(1 + \frac{R}{W} \times \frac{E_b}{N_0}) \tag{2.3}$$

En résolvant l'équation (2.3) pour $\frac{E_b}{N_0}$, nous obtenons :

$$\frac{E_b}{N_0} > \frac{2^{R/W} - 1}{R/W} = \frac{2^{\tau} - 1}{\tau} \tag{2.4}$$

La Figure 2.3 montre l'efficacité spectrale réalisable pour différentes valeurs de $\frac{E_b}{N_0}$. Les points situés en dessous de la courbe de capacité indiquent la région où une communication fiable est possible [34] tandis que les points situés au-dessus de la courbe représentent la région où une communication fiable n'est pas possible. L'aspect le plus intéressant de la théorie de Shannon, tel qu'exprimé dans le théorème de codage de canal, est qu'il donne non seulement, pour toute valeur de $\frac{E_b}{N_0}$, la gamme des taux réalisables, mais indique aussi qu'on peut approcher la frontière entre les taux réalisables et irréalisables aussi près que souhaité.

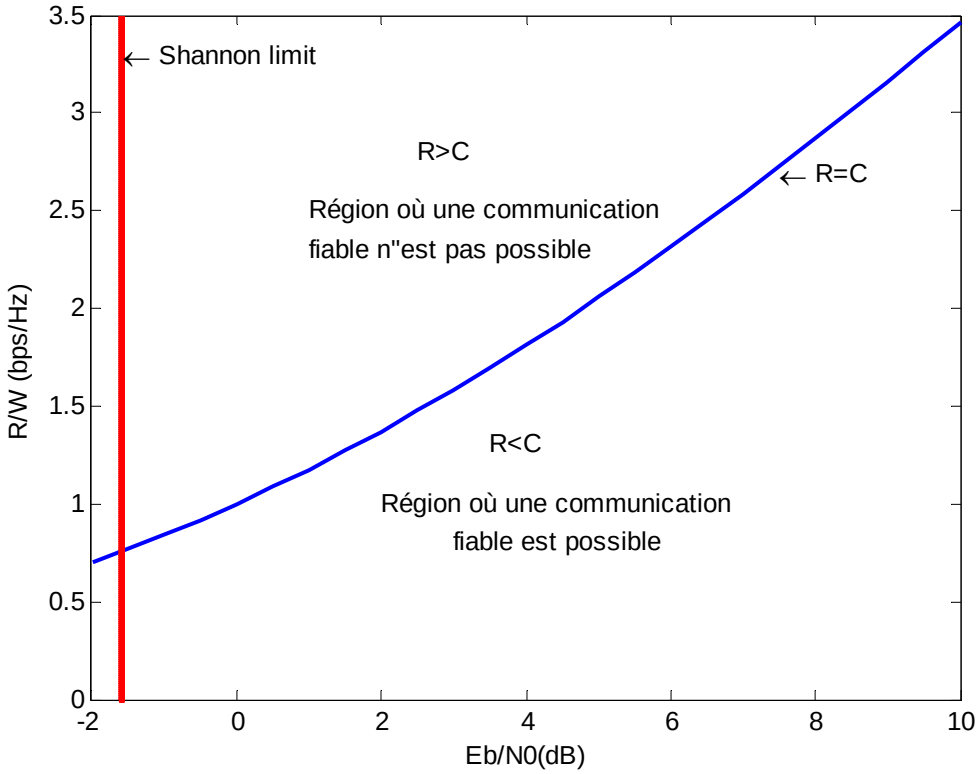


Figure 2.3 : Capacité de canal selon Shannon [34]

Un point important sur la courbe de capacité est le point correspondant à $\frac{R}{W} = 0$, c'est-à-dire lorsqu'il n'y a pas de restriction sur la bande passante. Pratiquement, cela représente la situation où des codes de contrôle d'erreurs à très faible débit sont utilisés. Pour $\tau = \frac{R}{W} \to 0$, à partir de l'équation (2.4), nous obtenons $\frac{E_b}{N_0} > \ln(2) = 0.69$ ou, de

manière équivalente, -1.6 dB. Cela signifie que, dans un canal AWGN, si $\frac{E_b}{N_0} < -1.6$ dB, une communication fiable n'est pas possible, quel que soit le nombre de bits de parité ajoutés aux bits du message.

Pour le canal AWGN, nous pouvons remarquer la relation entre $\frac{E_b}{N_0}$ et le taux de codage R, en supposant que la probabilité d'erreur est arbitrairement proche de zéro ($Pe = 10^{-5}$). Cependant, à cette valeur de la probabilité d'erreur, nous obtenons un gain de codage plus important comme le montre la Figure 2.4.

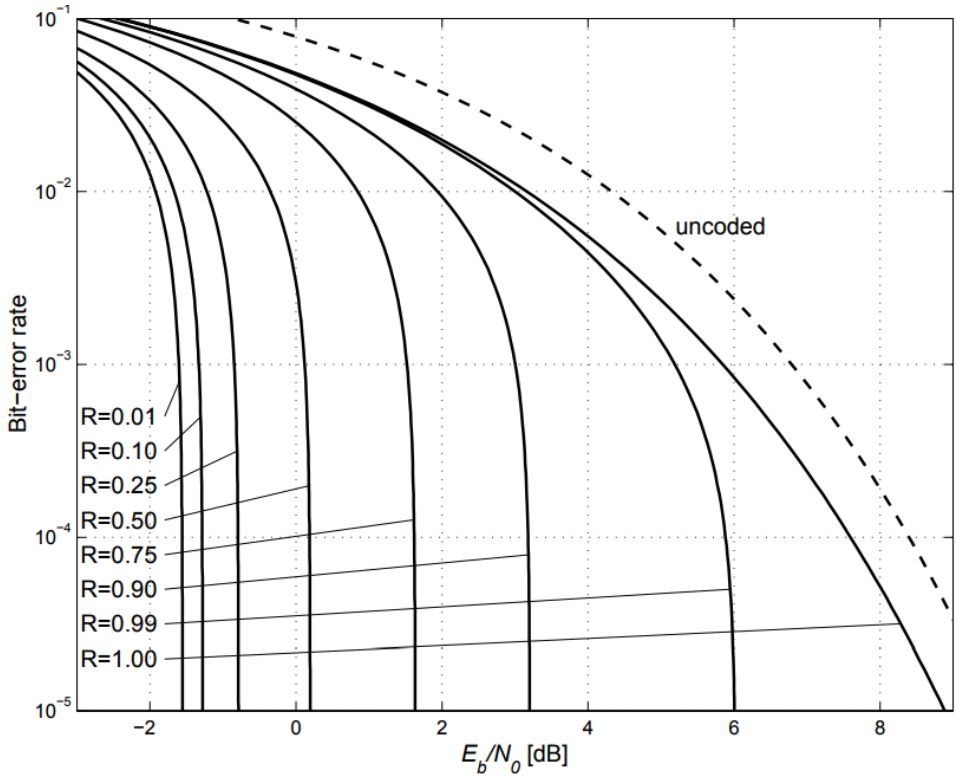


Figure 2.4 : Performances proches de la limite de Shannon en fonction de quelques valeurs de taux de codage R [35]

Dans la Figure 2.4, des limites de codage sont illustrées pour certaines valeurs de R par les courbes BER en fonction de $\frac{E_b}{N_0}$. Pour P_e très petit, les graphiques sont presque verticaux. La limite de Shannon peut être trouvée à R = 0.01 [35].

Cependant, pour étudier l'effet de la modulation M-ary pour différents schémas avec des symboles $M = 2^b$, la capacité de canal est limitée à un maximum de b bits/symbole (cette saturation se produit à des SNR élevés selon le type de modulation). Voici un tracé de toutes les techniques de modulation communes et de leurs limites de capacité Shannon vs SNR.

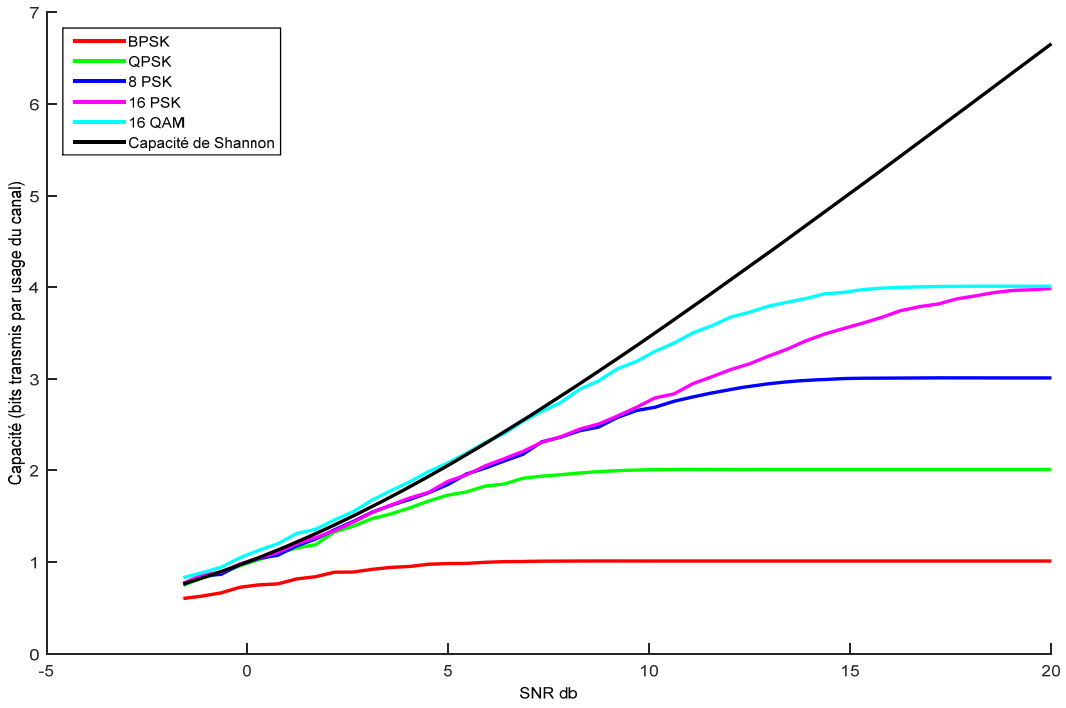


Figure 2.5 : Capacité de canal AWGN pour quelques signaux codés M-ary [36]

La Figure 2.5 illustre la capacité de canal d'un ensemble de signaux numériques M-ary les plus connus pour une probabilité d'erreur binaire $Pe = 10^{-5}$. Toutes les données de capacité sont calculées pour un rapport signal sur bruit compris entre –1.59 dB et +20 dB.

Par exemple, lorsque les conditions de canal sont très mauvaises, un schéma de modulation BPSK avec un code correcteur d'erreurs donné pourrait être utilisé pour un rapport signal/bruit requis. Lorsqu'il y a très peu d'atténuation dans le canal, un schéma de modulation à 8-PSK avec un code correcteur d'erreurs peut suffire.

2.2 Code Correcteur d'erreurs :

Le codage contrôleur ou correcteur d'erreur (ECC) est une technique de traitement du signal qui protège les informations numériques contre les erreurs de transmission et de stockage.

De nombreux canaux de communication souffrent de bruit, d'interférence ou de distorsion dus à des imperfections matérielles ou à des limitations physiques.

Le codage de canal traite des techniques de contrôle d'erreurs. Si les données à la sortie d'un système de communication ont des erreurs qui sont trop fréquentes pour l'utilisation souhaitée, les erreurs peuvent souvent être réduites en utilisant un certain nombre de techniques de contrôle d'erreurs. Le théorème de codage de canal indique qu'il existe toujours un schéma de codage permettant de coder des informations numériques de telle sorte que, même si le canal (ou le support de stockage) introduit des erreurs, le récepteur arrive les corriger avec un minimum d'erreurs. Cela n'est vrai que si le débit de données sur le canal est inférieur à la capacité du canal. Cependant, le théorème ne dit pas comment trouver le codage approprié pour une source d'information et un canal donné. Un objectif commun pour tous les systèmes de communication est d'assurer une transmission sans erreur. Le codage correcteur d'erreurs implique l'ajout systématique de données supplémentaires au message. Les bits de contrôle supplémentaires ne transmettent aucune information par eux-mêmes, mais ils permettent de détecter ou de corriger les erreurs. Mais, l'ajout de bits de contrôle supplémentaires a pour inconvénient l'élargissement de la bande passante du canal. Ce type de codage est illustré à la Figure 2.6.

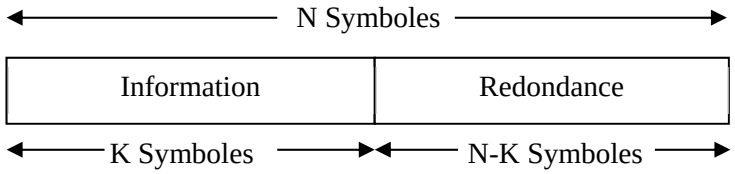


Figure 2.6 : Bloc systématique d'encodage pour un correcteur d'erreur

Les méthodes de codage de canal appartiennent à deux catégories distinctes : les codes de détection d'erreur et les codes de correction d'erreur.

Le processus de détection et de correction des erreurs au niveau du récepteur, de sorte que la retransmission n'est pas nécessaire, est appelé correction d'erreur directe (FEC : Forward Error Correction). Ce processus tente de corriger autant d'erreurs que possible sans retransmission de données. Alors qu'un système de requête automatique de retransmission (ARQ : Automatic Retransmission Query) détecte les erreurs et demande la retransmission de données erronés.

Comme la détection d'erreurs nécessite moins de redondance, ARQ est plus efficace en bande passante lorsque le canal est bon et son efficacité se détériore graduellement avec les conditions de canal. Mais l'utilisation de l'ARQ est très limitée dans les systèmes de communication où une grande partie du trafic est allouée aux applications en temps réel à large bande. Dans ce qui suit, nous nous concentrerons exclusivement sur les codes FEC.

Les codes de correction d'erreurs directes (sans voie de retour) peuvent être divisés en deux classes principales : Codes en blocs et Codes convolutionnels.

2.2.1. Correcteur d'erreurs en bloc :

2.2.1.1 Codes linéaire de Hamming :

Les codes de Hamming sont des codes de détection et de correction d'erreurs. Ils sont les plus célèbres de tous les codes correcteurs d'erreurs. Les codes de Hamming binaires, sont utilisés dans la transmission de données et peuvent détecter toutes les erreurs à un bit et à double bit et corriger toutes les erreurs à un seul bit. Pour un code C linéaire (n, k), la matrice de contrôle de parité pour un code C est la matrice génératrice P du code dual $C^{\perp}$. De plus, nous pouvons utiliser P pour déterminer les mots de code de C par tout c tel que $P \times c^T = 0$.

Un mot de code de Hamming est généré en multipliant les bits de données par une matrice génératrice g en utilisant l'arithmétique mod 2 (en mod 2 arithmétique $0 + 0 = 0,\ 0 + 1 = 1,\ 1 + 0 = 1,\ 1 + 1 = 0,\ 0 \times 0 = 0,\ 0 \times 1 = 0,\ 1 * 0 = 0\ et\ 1 \times 1 = 1$).

Le résultat de cette multiplication est appelé le vecteur de mot de code $[c_1 c_2 c_3 \ldots . c_n]$, constitué des bits de données d'origine et des bits supplémentaires utilisés pour la correction d'erreurs. La matrice génératrice g utilisée dans la construction des codes de Hamming peut s'écrire en I (la matrice d'identité) et une matrice génératrice de parité A:

$$g\ =\ [IA] \tag{2.5}$$

Un exemple de matrice génératrice de code Hamming (7,4) :

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

Si un vecteur de message de 4 bits $(d_1 d_2 d_3 d_4)$ est multiplié par g, le résultat est un mot de code de 7 bits de la forme $(d_1 d_2 d_3 d_4 p_1 p_2 p_3)$. On peut voir que chaque mot de code contient le message original et trois bits de contrôle supplémentaires, qui sont une combinaison linéaire des bits de message basés sur les colonnes de A. Valider le mot de code reçu, implique de le multiplier par une matrice de contrôle de parité, pour former s, le vecteur de contrôle de parité.

$$P\ =\ [A^T I_3] \tag{2.6}$$

On peut multiplier $r\ \times\ P^T$ ou l'équivalent $P\ \times\ r^T$.

Par exemple si $r\ =\ (1\ 0\ 0\ 1\ 0\ 0\ 1)$ alors $s\ =\ P\ *\ r^T$

$$s = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Si tous les éléments de s sont nuls, le mot de code est supposé être reçu correctement. Si s contient des éléments non nuls, le bit en erreur peut être déterminé en analysant quels bits de contrôle ont échoué, tant que l'erreur ne concerne qu'un seul bit. Par exemple, si $r = [1001001]$ alors s =$[101]^T$, ce qui correspond à la troisième colonne de Pet cela signifie que le troisième bit de r est en erreur.

En utilisant le code de Hamming (7,4), nous pouvons détecter jusqu'à deux erreurs et corriger jusqu'à une erreur, avec seulement trois bits supplémentaires pour chaque quatre bits de données.

2.2.1.2 Codes cycliques :

C'est l'une des classes de codes linéaires les plus importantes. En général, ces codes sont beaucoup plus faciles à mettre en œuvre et ont une grande valeur pratique. Les codes cycliques ont la propriété qu'il existe un générateur du code entier. Il existe deux approches pour décrire ces codes. Un code est cyclique si chaque fois que $[c_1 c_2 c_3 \ldots . c_n]$, est dans le code, le décalage $[c_n, c_1, \ldots, c_{n-1}]$ l'est aussi. Une façon plus indirecte consiste à utiliser des polynômes $g(x)$. Ce code C est appelé code cyclique et $g\ (x)$ est appelé le polynôme générateur.

2.2.1.2.1Codes de blocs cycliques BCH :

Les codes BCH (Bose-Chaudhuri-Hocquenghem) sont l'une des plus importantes classes de codes cycliques, avec de bonnes capacités de correction d'erreur et une procédure de codage et de décodage relativement rapide. Les codes BCH sont une généralisation des codes de Hamming, mais beaucoup plus efficaces puisque la distance de calcul des codes BCH peut être $d = 2$ ou plus, ce qui permet de corriger jusqu'à deux erreurs. Les codes BCH sont mieux expliqués en utilisant des polynômes, comme dans le cas des codes cycliques.

Un code BCH binaire a les paramètres suivants :

- Longueur du bloc : $n = 2^m - 1$,
- Nombre de bits de message : $k \geq n - mt$,
- Distance minimale : $d_{min} \geq 2t + 1$.

2.2.1.2.2Codes de blocs cycliques Reed-Solomon :

Les codes Reed-Solomon font partie de la famille des codes BCH non binaires. Le code de Reed Solomon fonctionne sur des symboles plutôt que sur des bits individuels. Un code Reed-Solomon corrigeant t erreurs, a les paramètres suivants :

- Longueur du bloc : $n = 2^m - 1$,
- Taille du message : k,
- Taille de la vérification de parité : $n - k = 2t$,
- Distance minimale : $d_{min} = 2t + 1$,
- Nombre d'erreurs corrigibles : $t = \frac{1}{2}(d_{min} - 1)$

2.2.1.3 Turbo Codes produits (TCP ou TPC) :

Généralement en TPC, la structure de codage est une concaténation en série de deux codes en bloc. Il y a d'abord un codeur externe puis un entrelaceur puis un codeur interne. Dans les équipements commerciaux, les codeurs interne et externe sont basés sur des codes de Hamming étendus. Les codes de Hamming étendus sont des codes Hamming avec un bit de parité à la fin. Le bit de parité augmente la distance minimale des codes de Hamming de trois à quatre. La parité utilisée est la parité paire. Certains équipements utilisent également des codeurs basés sur les codes BCH étendus et aussi le code Reed-solomon.

Supposons que deux codes de blocs linéaires $C_1(n_1, k_1, d_1)$ et $C_2(n_2, k_2, d_2)$ soient utilisés comme sous-code, où n est la longueur du code, k est la longueur de l'information, d est la distance minimale de Hamming. Un TPC à deux dimensions peut être représenté par $C_1 \times C_2$. Le processus de codage est le suivant :

- Lire séquentiellement le bloc d'information $k_1 \times k_2$ en tant que premières ligne k_1 et colonne k_2.
- Encoder la ligne k_1 avec $C_2(n_2, k_2, d_2)$.
- Encoder la colonne k_2 avec $C_1(n_1, k_1, d_1)$.

Les paramètre du TPC sont : longueur du code $n = n_1 \times n_2$, longueur de l'information $k = k_1 \times k_2$, distance minimale de Hamming $d = d_1 \times d_2$, et le taux de codage $R = R_1 \times R_2$, parmi lesquels R_1, R_2 sont les taux de codage de C_1, C_2 respectivement. À partir de la théorie de codage, si $t_1 = (d_1 - 1)/2$ et $t_2 = (d_2 - 1)/2$ les erreurs aléatoires peuvent être corrigées par C_1 et C_2 respectivement, $t = (d_1 \times d_2 - 1)/2$ les erreurs aléatoires peuvent être corrigées par $C_1 \times C_2$. Il peut également prouver que si $b <= max\,(n_1 t_2, n_2 t_1)$ les erreurs de rafale peuvent être corrigées par $C_1 \times C_2$, le paramètre b est appelé capacité de correction d'erreur en rafale. Ainsi, le TPC est un type de code avec de fortes capacités de correction d'erreur, et sa structure de codage est représentée dans la Figure 2.7.

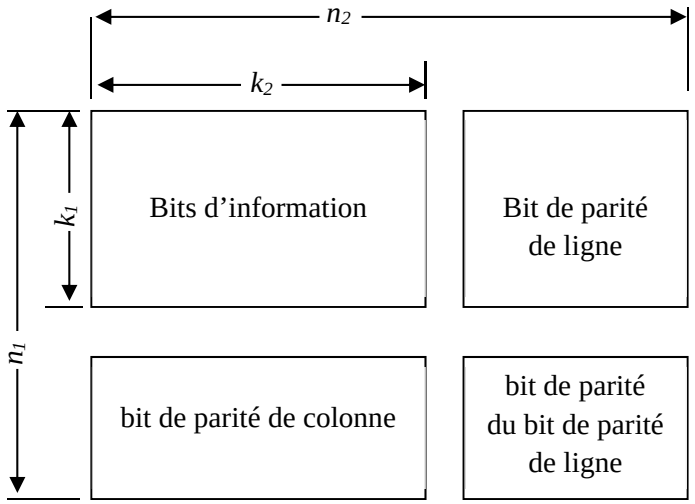


Figure 2.7 : Structure d'encodage d'un TPC

TPC est un code concaténé en série, il a de bonnes performances avec un algorithme de décodage itératif, parmi lequel, l'unité décodeur est la partie la plus importante. L'unité décodeur TPC est constituée de deux parties, l'une est un décodeur d'entrée souple et de sortie dure basé sur l'algorithme Chase2 et l'autre partie est le calcul d'informations extrinsèques qui servent à convertir l'entrée dure en sortie souple.

2.2.2 Codes correcteurs d'erreurs convolutionnel :

2.2.2.1 Code convolutionnel (CC) :

Une différence importante entre les codes de convolution et les codes de bloc c'est que le codeur convolutionnel contient de la mémoire.

Un code convolutif peut avoir plusieurs polynômes générateurs $g(D)$. Tout code de convolution peut être réalisé en utilisant un codeur de registre à décalage (feed-forward). Normalement, les codes convolutionnels sont préférés dans de nombreuses applications de télécommunication.

Un exemple de code convolutionnel est montré dans la Figure 2.8. Ce diagramme montre que les codes convolutionnels peuvent avoir au moins un polynôme générateur différent pour chaque sortie différente. Le polynôme générateur du chemin 1 est $g1 = (111)$ et le polynôme générateur du chemin 2 est $g2 = (101)$, le codeur a besoin de 2 registres pour stocker la mémoire. A partir de ce diagramme, nous pouvons obtenir 3 diagrammes différents pour représenter le codeur : l'arbre de code, le diagramme d'état et le diagramme de treillis.

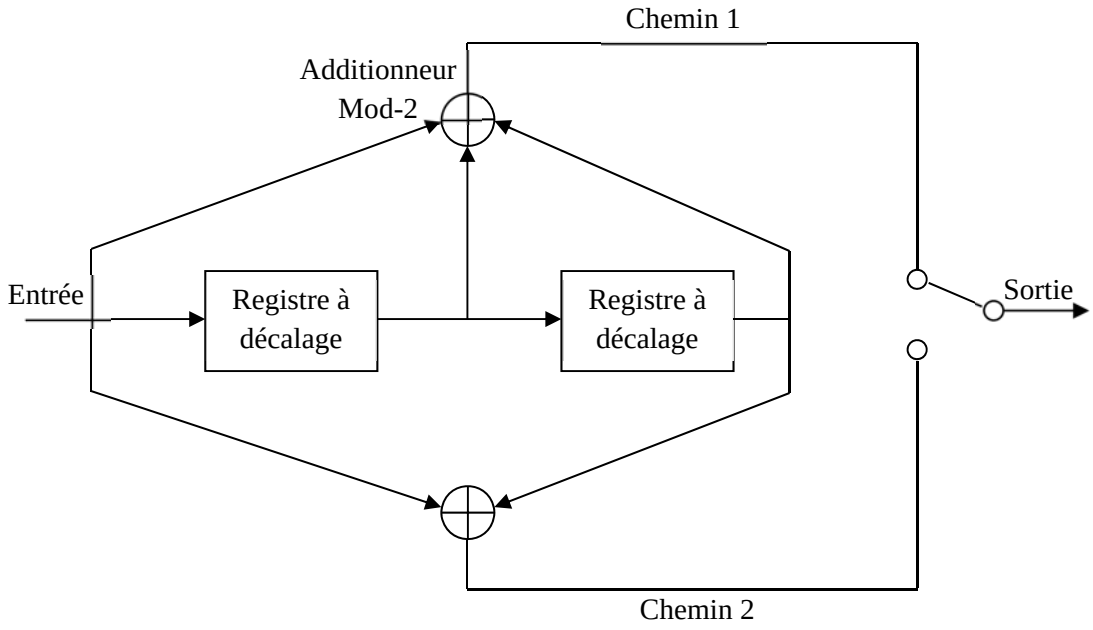


Figure 2.8 : Encodeur convolutionnel avec longueur de la contrainte *CL*=3 et taux *R*=1/2

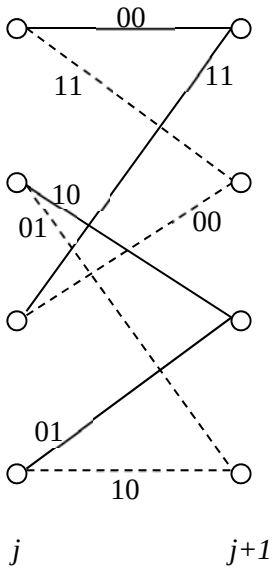


Figure 2.9 : Portion d'un diagramme de treillis pour un encodeur convolutionnel avec longueur de contrainte CL=3 et taux R=1/2

Le diagramme en treillis montre sur la Figure 2.9 que les nœuds gauches représentent les quatre états courants possibles et que les nœuds droits représentent les états suivants de la mémoire du registre. De même, l'entrée "0" est représentée par une branche pleine et l'entrée "1" est représentée par une branche en pointillé.

Les bits codés dépendent non seulement des k bits d'entrée courants, mais également des bits d'entrée antérieurs et ultérieurs.

La stratégie principale de décodage pour les codes convolutifs est basée sur l'algorithme de Viterbi largement utilisé.

En raison d'une large acceptation des codes convolutifs, de nombreux progrès ont été faits pour étendre et améliorer ce système de codage de base. Cette avancée a abouti à deux nouveaux schémas de codage, à savoir, la modulation codée en treillis (TCM) et les turbo codes. Le TCM ajoute la redondance en combinant le codage et la modulation en une seule opération (comme son nom l'indique).

2.2.2.2 Turbo code convolutionnel (TCC) :

Les Turbo codes ont été considérés comme l'une des percées (succès) technologiques les plus significatives de la théorie de l'information dans les années 90. En effet, il s'agit des premiers codes pratiques à se rapprocher de la limite de Shannon.

Ces codes ont trois idées simples : la concaténation parallèle des codes pour permettre un décodage plus simple, l'entrelacement pour une meilleure répartition du poids et un décodage souple pour augmenter les décisions du décodeur et maximiser le gain de l'interaction du décodeur.

Le codeur d'un turbo code convolutionnel (CTC) est illustré dans la Figure 2.10, où deux codeurs convolutifs systématiques récursifs (RSC) travaillent en parallèle. Un premier code le message d'origine et l'autre fonctionne sur la séquence de messages permutée.

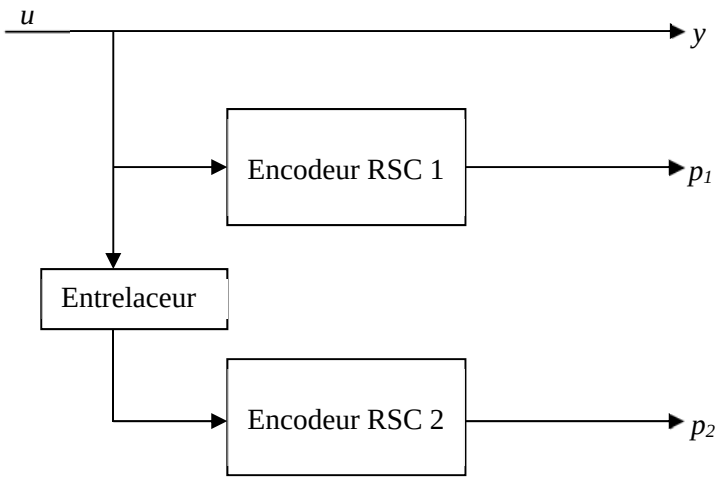


Figure 2.10 : Structure d'un turbo codeur

Le décodeur TCC est construit de la même manière que le codeur, où deux décodeurs de base sont interconnectés en parallèle comme le montre la Figure 2.11. L'algorithme de décodage optimal pour le TCC nécessite un décodage itératif, dans lequel l'information extrinsèque souple de la séquence reçue est passée d'une itération de décodage précédente à l'itération en cours. Par conséquent, DEC1 et DEC2 devraient être capables de générer des sorties souples, qui représentent la probabilité qu'un bit soit

'1' ou '0'. Chaque décodeur SISO (soft-input soft-output) utilise l'algorithme BCJR modifié, qui est également connu sous le nom d'algorithme de probabilité maximum de vraisemblance (MAP).

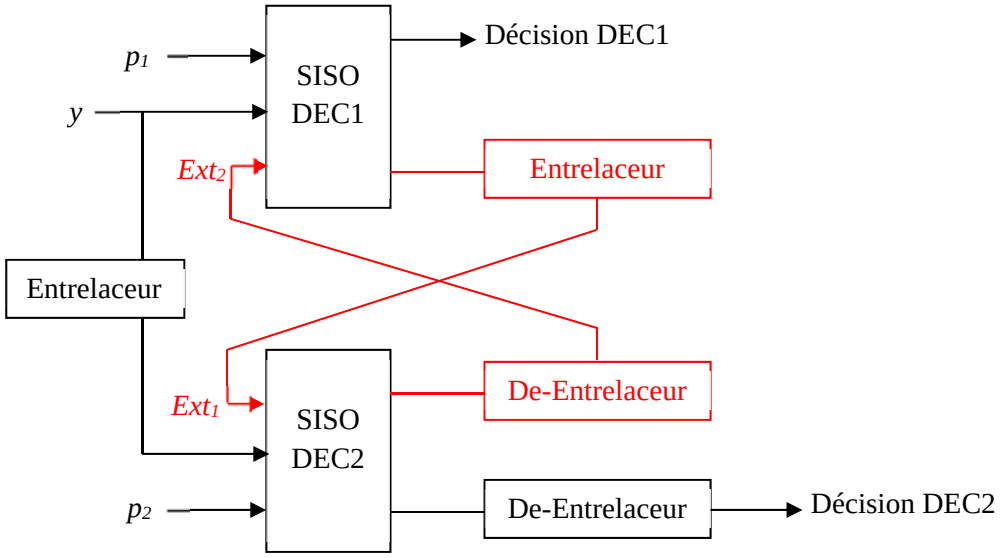


Figure 2.11 : Structure d'un turbo décodeur

2.2.3. Applications des codes correcteurs d'erreurs :

L'importance des codes correcteurs d'erreurs a augmenté avec les communications numériques, de tels codes ont été utilisés avec grand succès dans certaines applications importantes.

2.2.3.1. A l'espace lointain dans les sondes planétaires :

Les différents types de missions spatiales et orbitales effectuées suggèrent que la recherche d'un système de correction d'erreurs « favorable pour tous » sera un problème permanent pendant un certain temps. Pour les missions proches de la Terre, la nature du bruit de canal est différente de celle que subit un engin spatial dans une mission interplanétaire. De plus, lorsqu'un engin spatial s'éloigne de la Terre, le problème de la correction du bruit devient plus important.

En fait, les premières missions envoyaient leurs données non codées (Aucun code de correction d'erreur n'est utilisé). Il s'agissait des sondes Mariner, 1962-1967 (Mars, Vénus).

Plus tard, la correction d'erreur numérique a été mise en œuvre sous la forme de codes convolutifs et de codes Reed-Muller dans les sondes de Mariner [37] et Viking, 1969-1976 (Mars, Vénus).

Les missions Voyager 1 et 2 [38], qui ont débuté en 1977, ont été conçues pour fournir des images en couleurs des informations scientifiques à travers Jupiter et Saturne [39]. Cela a entraîné une augmentation des exigences de codage, et ainsi les sondes spatiales ont été supportées par des codes convolutionnels qui pouvaient être concaténés avec un code Golay externe.

Les codes concaténés sont devenus un standard pratique dans les communications par satellite et dans l'espace lointain. La mission Voyager 2 a utilisé une implémentation concaténée d'un code Reed-Solomonet Viterbi (RSV) pour une correction d'erreurs très puissante, et permettait alors un voyage prolongé de l'engin spatial vers Uranus et Neptune en 1986.

Le décodage à passage unique (Non itératif) des codes de correction d'erreurs peut produire des taux d'erreurs très faibles, mais pour les conditions de transmission à longue distance (comme l'espace lointain), un décodage itératif est recommandé.

La sonde Galileo a utilisé des codes concaténés itératifs pour compenser les conditions de taux d'erreur très élevés provoqués par une antenne.

Le CCSDS recommande actuellement l'utilisation de codes de correction d'erreurs avec des performances similaires au code Voyager 2 (RSV) au minimum. Les codes concaténés en série sont de moins en moins utilisés dans les missions spatiales, au détriment des codes plus puissants tels que les Turbo codes ou les codes LDPC.

2.2.3.2. Digital Video Broadcasting (DVB) :

La combinaison d'un code convolutionnel interne de Viterbi et d'un code extérieur de Reed-Solomon (connu sous le nom RSV) a été utilisée pour la première fois dans Voyager 2. Il est encore utilisé pour les communications par satellite, telle que la norme de diffusion de télévision numérique DVB-S. Dans la norme DVB-S2, un code LDPC hautement efficace est combiné avec un code externe algébrique afin d'éliminer toute erreur résiliente non détectée par le code LDPC interne en raison de son plancher d'erreur inhérent.

2.2.3.3. Stockage de masse numérique :

Les systèmes de stockage de données utilisant des supports amovibles s'appuient fortement sur de fortes architectures de codage de correction d'erreurs concaténées afin de garantir des taux de perte de données très faibles. En particulier, les lecteurs de bandes et les lecteurs de disques optiques (par exemple CD, DVD, disque dur et disque Blu-ray BD) utilisent des schémas ECC puissants basés sur une concaténation d'un code externe et un code interne. Ceci, leur permet d'atteindre cette performance, où un entrelacement entre les deux codes répartir les erreurs sur plusieurs paquets, facilitant ainsi la détection et la correction de ces erreurs.

2.2.3.4. Communications sans fil numériques :

Des codes correcteurs d'erreurs ont été introduits dans les normes définissant les systèmes sans fil de troisième génération (3G). Turbo codes ont été introduits à la fin des années 1990 dans la spécification UMTS, édité par 3GPP. Ils ont également été inclus dans la norme cdma2000, supportée par 3GPP2 [40].

En charge de la définition d'une norme UMTS améliorée, basée sur des spécifications existantes. Les codes avancés de correction d'erreurs ont été l'une des améliorations examinées et étudiées. Le LTE (Long Term Evolution) défini par le consortium 3GPP, est une évolution des normes de téléphonie mobile GSM, CDMA2000, et UMTS.

2.3 Conclusion

Au début des années 1940, on pensait qu'il était impossible d'envoyer des informations à travers un canal de communication non parfait et avec une probabilité d'erreur négligeable. Shannon a surpris la communauté de la théorie de la communication en prouvant que la probabilité d'erreur pouvait être presque nulle en utilisant des codes ayant un taux R inférieur à la capacité du canal C. Ces codes possèdent aujourd'hui de nombreuses applications dans notre vie.

Chapitre 3

Ce chapitre commence par rappeler le principe du codage concaténé série et parallèle. Ceci comprend également l'entrelacement utilisé dans la concaténation et le problème de plancher d'erreur dans le décodage itératif avancé. Ensuite, il décrit les codes de contrôle de parité à faible densité (LDPC). Le chapitre traite aussi les codes convolutifs concaténés parallèles ou les Turbo codes et la structure du décodeur itératif associé. Il explique également d'autres classes de codes concaténés tels que les codes de blocs concaténés en série (LDPC-BCH) et la concaténation parallèle des codes LDPC (PCGC). Enfin, ce chapitre donne un aperçu succinct sur la concaténation hybride qui est une combinaison entre la concaténation parallèle et série appelée 3D Turbo code. Tous ces codes sont comparés par la suite avec la méthode proposée dans le chapitre 5.

3.1 Concaténation des codes :

La concaténation des codes a pour but de combiner l'amélioration de performances de deux codes ou plus. Cette technique peut suivre une architecture série ou parallèle. Il existe de nombreuses méthodes de concaténation de codes réussies qui sont plus sophistiquées qu'un système de codage simple. Une probabilité d'erreur de bit améliorée peut survenir à partir d'une combinaison de deux ou plusieurs codes formant un code puissant de longueur de bloc plus élevée avec un entrelaceur [41], permettant ainsi une transmission fiable à des débits proches de la capacité. Cependant, un tel décodeur serait probablement complexe.

3.1.1 Concaténation en série :

La Figure 3.1 illustre la concaténation en série, qui utilise deux codes appelés interne et externe. Le code interne peut être de type quelconque et le code externe tente de fournir une amélioration supplémentaire. L'entrelaceur réorganise l'ordre des bits transmis, tandis que le désentrelaceur correspondant restaure l'ordre d'origine. Le désentrelacement peut ainsi disperser une rafale d'erreurs associée à des erreurs isolées de sorte que ces erreurs individuelles peuvent être plus faciles à corriger par le décodeur externe.

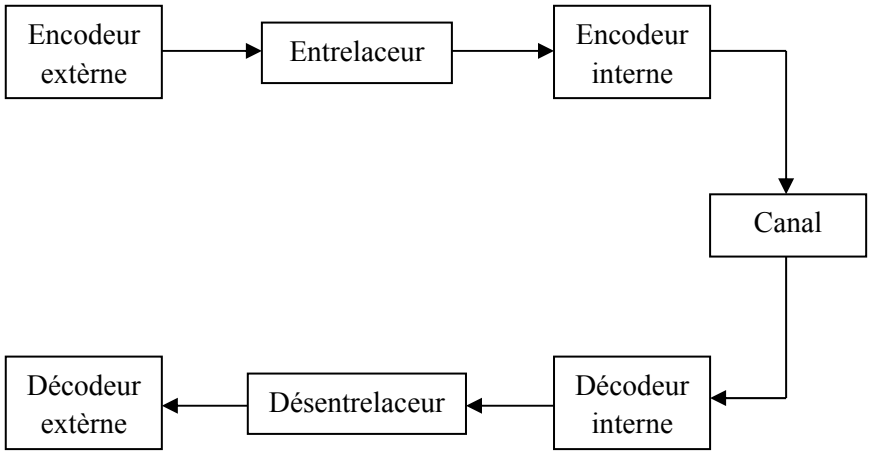


Figure 3.1 : Concaténation série

3.1.2 Concaténation en parallèle :

La Figure 3.2 illustre une concaténation parallèle pour deux codes. La même séquence d'information est codée par deux codeurs différents. Cependant, l'un des codeurs agit sur une version entrelacée de la séquence d'entrée.

La dénomination de "code interne" et "code externe" est un peu vague pour une concaténation parallèle, de sorte que les codes sont souvent appelés premier et deuxième code. Toutes les sorties d'un code peuvent être transmises, ou certains sous-ensembles de ces sorties peuvent être transmis par effacement régulier (poinçonnage) des sorties du codeur.

La concaténation parallèle peut être étendue à plus de deux codes en ajoutant des entrelaceurs et des codeurs supplémentaires d'une manière parallèle sur la Figure 3.2.

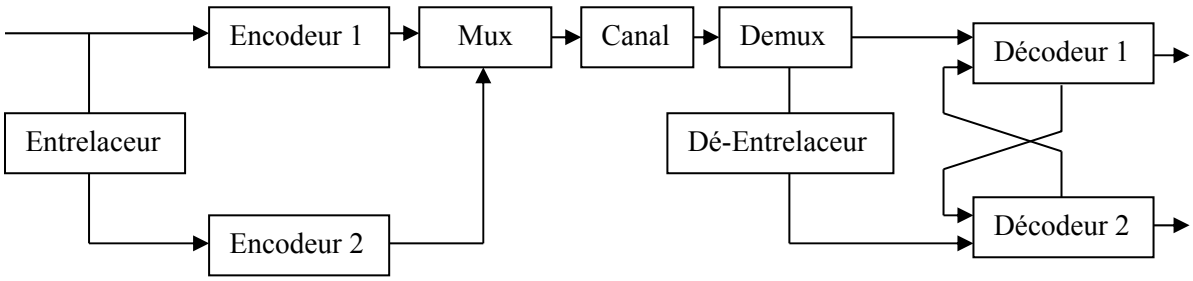


Figure 3.2 : Concaténation parallèle

Le Tableau 3.1 montre une comparaison des taux de codage R pour une concaténation en parallèle et en série.

Parallèle	Série
$R_p = \frac{R_1 R_2}{1-(1-R_1)(1-R_2)}$	$R_s = R_1 R_2$

Tableau 3.1 : Taux de codage pour les deux types de concaténation

3.1.3 Entrelacement :

L'entrelacement (Interleaving en Anglais) est un réarrangement périodique et réversible de blocs de K bits transmis [41]. Les bits sont réorganisés en conséquence par désentrelacement dans le récepteur. L'entrelacement est utilisé pour disperser les rafales d'erreurs qui peuvent survenir en raison d'un bruit de canal non stationnaire qui peut être localisé à quelques dimensions telles qu'une rafale de bruit impulsif dans le temps ou une perte d'une bande de fréquence étroite dans l'OFDM [42]. Des rafales d'erreurs peuvent également se produire à cause de la décision incorrecte d'un décodeur quelconque concaténé sur les Figures 3.1 ou 3.2.

Les bits erronés provoqués par la correction des erreurs de code interne s'étendent généralement à l'ensemble du mot de code incorrect, ce qui entraîne une rafale de bits erronés. Si ces rafales sont séparées par un intervalle long par rapport à la période d'entrelacement, elles peuvent être réparties plus uniformément dans le temps par le désentrelaceur dans le récepteur.

3.1.4. Poinçonnage :

Les bits redondants dans le codage diminuent l'efficacité de la bande passante (si le nombre des bits augmente, l'efficacité de la bande passante diminue). Le poinçonnage (Puncturing en anglais) est le processus de suppression de certains bits de parité (redondants) du mot de code en fonction d'une matrice de poinçonnage [43]. Ceci va augmenter le taux de codage des codes concaténés en parallèle de 1/3 à 1/2 ou plus sans augmenter la complexité, mais au prix de la diminution de la distance libre

du code. Cette opération fait donc le compromis entre le taux de codage et la performance.

3.1.5. Le phénomène de plancher d'erreur dans le codage itératif :

Pour tout code de longueur finie typique avec un décodage itératif, la courbe BER vs SNR se compose de deux régions distinctes :

La région de cascade 'waterfall' et la région de plancher d'erreur 'error floor'. Dans la région des cascades (qui est la région à faible SNR), le taux d'erreur diminue significativement avec l'augmentation du SNR, ce qui fait que la courbe ressemble à une cascade. D'autre part, dans la région de plancher d'erreur (qui est la région de SNR élevé), la diminution du taux d'erreur ralentit considérablement et la courbe a tendance à s'aplatir en se transformant en un plancher d'erreur. Le taux d'erreur est illustré en échelle logarithmique sur la Figure 3.3.

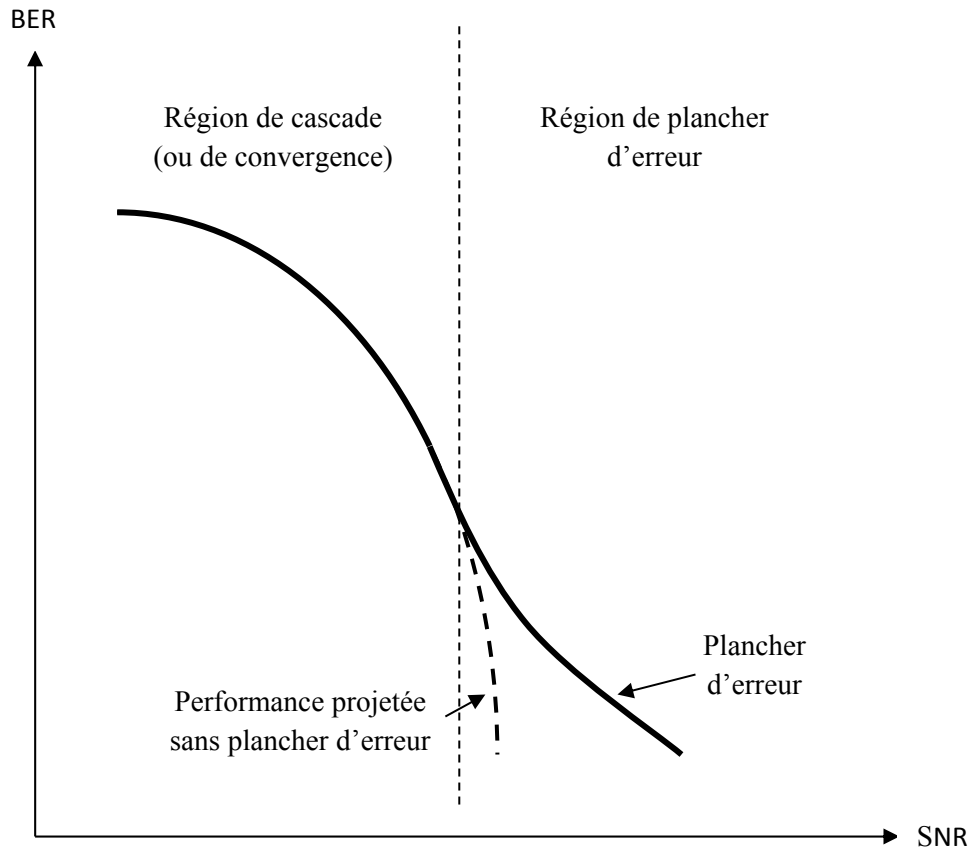


Figure 3.3 : Les deux régions de la courbe BER vs SNR

Le plancher d'erreur peut provenir de différentes causes, généralement en relation avec la distance minimale du code ou des formes d'erreurs particulières qui, d'une certaine manière, piègent le décodeur itératif. Il convient de noter qu'il existe une sorte de compromis entre les performances dans les régions de cascade et de plancher d'erreur : les codes conçus pour présenter des capacités de correction d'erreur à des faibles SNR présentent généralement des niveaux de plancher d'erreur relativement élevés. Alors que les codes atteignant des taux d'erreurs très faibles sans aucun signe d'atténuation des pentes, sont généralement caractérisés par des gains de codage plus faibles dans la région des cascades [44].

Résoudre le problème du plancher d'erreur était un problème critique au cours de la dernière décennie. Il est très difficile d'étudier les niveaux d'erreur d'un codeur itératif pratique, car les simulations logicielles pour des taux d'erreurs très faibles prennent souvent des jours voire des semaines.

3.2. Code de vérification de parité de faible densité (LDPC : Low Density Parity Check Code) :

Les codes LDPC longs avec décodage itératif basé sur la propagation de croyances (BP) réalise une performance d'erreur presque d'une fraction de décibel loin de la limite de Shannon [19]. Cette découverte rend les codes LDPC des concurrents puissants aux turbo codes dans de nombreux systèmes de communication nécessitants une grande fiabilité.

3.2.1 Encodeur LDPC:

Le code LDPC est une technique de codage linéaire de message défini par un ensemble de deux matrices binaires ; une matrice de contrôle de parité H très clairsemée et sa matrice génératrice G tel que :

$$GH^T = 0 \tag{3.1}$$

Où la matrice H est une combinaison de 0 et de 1. Elle est caractérisée, aussi, par le poids moyen de la colonne (MCW), qui est le poids moyen sur toutes les N colonnes de la matrice, il est défini par :

$$MCW = \sum_{q=1}^{b} \frac{d_q \lambda_q}{N} \tag{3.2}$$

Où $\sum_{q=1:b} \lambda_q = N$,$d_q$ est le poids d'une colonne (nombre de '1' dans une colonne), λ_q est le nombre de colonnes qui ont le poids d_q, b est le nombre des différents poids dans la matrice de contrôle de parité $H(M \times N)$.

La matrice (3.3) représente la matrice de contrôle de parité $H(4 \times 8)$ pour $MCW = 3$. Le mélange des différents poids de colonnes (poids : $d_q = 3\ et\ 2$) peut construire un code de $MCW = 2.625$ ($\frac{1}{3}$ des colonnes sont de poids 2 et $\frac{2}{3}$ sont de poids 3).

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{3.3}$$

La matrice de contrôle de parité $H(M \times N)$ définit le taux $r = K/N, K = N - M$

(pour $r = 1/2, K = M$).

H peut être réduite à une forme systématique $H_{sys} = [I_M | A_{M \times K}]$ par l'élimination de Gauss-Jordan. Ceci détermine la matrice génératrice $G = [-A^T | I]$ (pour le mode binaire $-A = A$), Où $A(M \times K)$ est une matrice arbitraire et $I(M \times M)$ est une matrice identité.

$$H_{sys} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.4}$$

$$G = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

Soit $s = [s_1, s_2, \dots s_M]$ la séquence d'information à coder et x le mot code obtenu par :

$$x = s \times G \tag{3.6}$$

Le codage est le processus pour obtenir le code x à l'aide de la matrice G et la séquence d'information s. Ce mot code est composé de deux parties, les bits de contrôle de parité p et les bits systématiques s:

$$x = [p, s] = [p_1, p_2, \dots p_K, s_1, s_2, \dots s_M].$$

Le mot code est valable seulement s'il satisfait le calcul du syndrome c:

$$c = H.x^T = 0 \tag{3.7}$$

3.2.2 Décodeur LDPC :

Considérons la matrice de contrôle de parité $H(M \times N)$ avec $MCW = 3$ en dessous. Le problème de décodage est de trouver le vecteur x le plus probable de telle sorte que $H.x^T = 0$. Supposons que nous avons le mot code x suivant:

$$x = [x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$$

Où x_1, x_2, x_3, x_4 sont les bits de parité et x_5, x_6, x_7, x_8 sont les bits systématiques, où quel que soit un mot code donné x satisfait l'ensemble des équations de parité telle que :

$$H.x^T = 0.$$

$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \qquad \begin{aligned} x_1 \oplus x_2 \oplus x_3 \oplus x_6 \oplus x_7 \oplus x_8 &= 0 \\ x_1 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_7 \oplus x_8 &= 0 \\ x_2 \oplus x_3 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_8 &= 0 \\ x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_6 \oplus x_7 &= 0 \end{aligned} \tag{3.8}$$

La matrice H a une petite densité de 1 et chaque ligne de H induit un contrôle de parité sur le mot code x. Chaque colonne de la matrice représente un bit de mot code tandis

que chaque ligne représente une équation de contrôle de parité. Dans notre exemple, chaque x_i peut être un '0' ou '1' et le mot code a maintenant 4 équations de contrôle de parité.

Une autre représentation pour la matrice de contrôle de parité H est le graphe de Tanner [45] qui se compose de deux types de nœuds. N nœuds de variables et M nœuds de contrôle interconnectés par de nombreux chemins transportant des métriques Q_{ij}(nœud de variable au nœud de contrôle) et R_{ij} (nœud de contrôle au nœud de variable) comme indiqué sur la Figure 3.4. Dans le graphe de Tanner le $j^{ème}$ nœud de contrôle est connecté au $i^{ème}$nœud de variable si et seulement si le $j^{ème}$ élément dans le $i^{ème}$ligne h_{ij}dans la matrice de contrôle de parité H est un '1'. Où :

$j = (1, \dots N)$ et $i = (1, \dots, M)$.

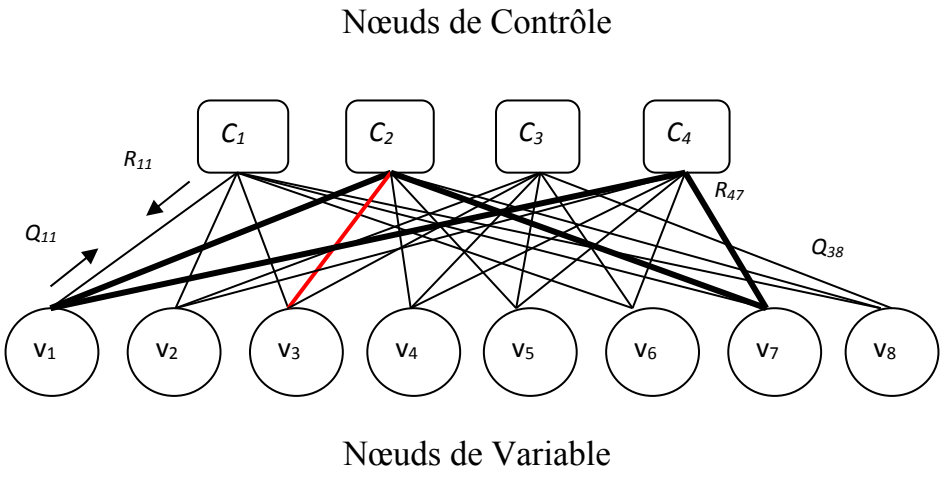


Figure 3.4 : Le graphe de Tanner pour le Code LDPC

Un chemin qui commence et finit avec le même nœud s'appelle un cycle. Si un graphe de Tanner contient des cycles, ces cycles ont des longueurs paires. La longueur du plus court cycle dans le graphe s'appelle le périmètre du graphe.

Le chemin marqué en gras ($C_2 \rightarrow V_1 \rightarrow C_4 \rightarrow V_7 \rightarrow C_2$) est un exemple pour le cycle court de longueur 4.

Le décodage peut être fait par SPA (Algorithme de Somme-produit), un algorithme efficace qui fonctionne sur des probabilités [19,20]. Il est basé sur la propagation de croyances (BP) aussi connu par l'algorithme de passage de messages sur le graphe de Tanner entre les nœuds adjacents (nœud de variable j et nœud de contrôle i) connectés comme le montre la Figure 3.4.

L'algorithme Somme-produit est décrit principalement en deux étapes, horizontale et verticale respectivement [19,20]. L'étape horizontale calcule les messages déplacés à partir de chaque nœud de contrôle i au nœud de variable j, qui indiquent la probabilité du nœud de contrôle R_{ij}. De même façon, l'étape verticale calcule les messages envoyés à partir du nœud de variable j au nœud de contrôle i. Ce processus met à jour la probabilité de nœud de variable Q_{ij}(également appelée information extrinsèque).

Soit $f_j^{u_j}$ la probabilité de u, la probabilité que le vecteur message u est 1 à l'instant j.
Alors :

$$f_j^1 = P(u_j = +1 | y_j) = 1/1 + e^{(-2y_j/\sigma^2)} \quad (3.9)$$

Où :

u_j est un symbole transmis modulé en BPSK, avec une valeur donnée par:

$$u_j = \begin{cases} +1 & quand \quad x_j = 1 \\ -1 & quand \quad x_j = 0 \end{cases}$$

x_j est un bit transmis du mot code, $x_j \epsilon \{0,1\}$.

y_j est un symbole reçu à l'instant j, $y_j = u_j + n_j$,

n_j est un bruit blanc gaussien additif (AWGN) de variance σ^2.

La probabilité pour que le vecteur message u est -1 à l'instant j peut être déduite par :

$$f_j^0 = 1 - f_j^1 \quad (3.10)$$

La norme de SPA est décrite comme suit :

Initialisation :

Initialement, les probabilités $f_i^{u_i}$ sont envoyées des nœuds de variables aux nœuds de contrôle comme Q_{ij} i.e. fixer $Q_{ij}^1 = f_j^1$ et $Q_{ij}^0 = f_j^0$ pour tout i, j, pour que $H_{ij} = 1$.

Etape Horizontale :

Les probabilités connues $\{f_1^b, f_2^b, \dots, f_N^b\}$ correspondant aux bits $\{y_1, y_2, \dots, y_N\}$ sont utilisées dans la première itération pour trouver la distribution de probabilité sur la somme binaire :

$C_j = y_1 \oplus y_2 \oplus \dots \oplus y_N$.

$$P(C_j = 0) = \frac{1}{2}\left(1 + \prod_{j=1}^{N}(f_j^0 - f_j^1)\right) \quad (3.11)$$

Avec la correspondance $Q_{ij}^b = f_j^b$, nous avons :

$$R_{ij}^0 = \frac{1}{2}\left(1 + \prod_{j' \in Z_j \setminus j}(Q_{ij'}^0 - Q_{ij'}^1)\right) \quad (3.12)$$

Notons par Z_j l'ensemble des bits j qui participent au contrôleur i (cela signifie tous les nœuds de variable connectés au nœud de contrôle C_j).

Soit : $\delta Q_{ij'} = Q_{ij'}^0 - Q_{ij'}^1$

$$R_{ij}^0 = \frac{1}{2}\left(1 + \prod_{j' \in Z_j \setminus j} \delta Q_{ij'}\right) \quad (3.13)$$

Et:

$$DR_{ij} = \prod_{j' \in Z_j \setminus j} \delta Q_{ij'} \quad (3.14)$$

De toute évidence :

$$R_{ij}^0 = \frac{1}{2}\left(1 + DR_{ij}\right) \quad (3.15)$$

Et :

$$R_{ij}^1 = \frac{1}{2}\left(1 - DR_{ij}\right) \quad (3.16)$$

Etape Verticale :

Maintenant, pour chaque i, j et pour $u = b, b \in \{0,1\}$ nous mettons à jour $Q_{ij}(b)$ par considération de $R_{ij}(b)$ précédemment calculée.

$$Q_{ij}^0 = f_j^0 \prod_{i' \epsilon Z_j \backslash i} R_{i'j}^0 = f_j^0 \prod_{i' \epsilon Z_i \backslash i} R_{i'j}^0 \quad (3.17)$$

Définissons l'ensemble des contrôleurs dans lesquels le bit j participe à Z_i.

$$Q_{ij}^1 = f_j^1 \prod_{i' \epsilon Z_i \backslash i} R_{i'j}^1 \quad (3.18)$$

Les Q_{ij}^b calculées à la fin de chaque itération sont utilisées pour l'itération suivante.

Afin de fixer la valeur du nœud de variable, nous calculons : Q_j^b:

$$Q_j^0 = f_j^0 \prod_{i' \epsilon Z_i} R_{i'j}^0 \quad (3.19)$$

$$Q_j^1 = f_j^1 \prod_{i' \epsilon Z_i} R_{i'j}^1 \quad (3.20)$$

Q_{ij}^b et Q_j^b sont normalisées comme suit:

$$Q_{ij}^b = \frac{Q_{ij}^b}{Q_{ij}^0 + Q_{ij}^1} \quad (3.21)$$

$$Q_j^b = \frac{Q_j^b}{Q_j^0 + Q_j^1} \quad (3.22)$$

Pour réduire le nombre de multiplication, Kozintsev [46] remplace les multiplications dans les termes $\prod \delta Q_{ij'}$ dans (3.14) et $\prod R_{i'j}^b$ dans (3.17), (3.18), (3.19) et (3.20) en additions par l'utilisation du domaine logarithmique. Les équations (3.15), (3.16), (3.17), (3.18), (3.19) et (3.20) deviennent :

$$R_{ij}^0 = \frac{1}{2}(1 + Pdq') \quad (3.23)$$

$$R_{ij}^1 = \frac{1}{2}(1 - Pdq') \quad (3.24)$$

$$Pdq' = \frac{Pdq}{\delta Q_{ij}} \quad (3.25)$$

Où:

$$Pdq = e^{log \sum_{j \epsilon Z_j} \delta Q_{ij}} \quad (3.26)$$

Et:

$$Q_{ij}^b = \frac{Pr_j^b f_j^b}{R_{ij}^b} \quad (3.27)$$

$$Q_j^b = \sum_1^{d_i - 1} f_j^b Pr_j^b \quad (3.28)$$

Où :

$$Pr_j^b = e^{log \sum_{i \epsilon Z_i} R_{ij}^b} \quad (3.29)$$

d_i est le nombre de '1' par colonne.

La décision est obtenue par :

Pour $j = 1,2, \dots N$

$$L_j^+ = Q_j^1 - Q_j^0 \quad (3.30)$$

$$\widehat{u_j} = \begin{cases} 1 \text{ , } si\ sign(L_j^+ = Q_j^1 - Q_j^0) > 0 \\ 0, \quad sinon \end{cases} \quad (3.31)$$

Le décodage itératif est arrêté si la séquence décodée $\widehat{u_j}$ vérifie toutes les équations de contrôle de parité du code : $H.\widehat{u_j}^t = 0$, ou le nombre maximum d'itérations est atteint.

3.3 Turbo Code Convolutionnel (TCC : Turbo Convolutional Code) :

Le turbo code ou le codeur convolutionnel en concaténation parallèle inventé par Berrou et Glavieux [8,9] est constitué de la concaténation parallèle de deux codeurs convolutifs systématiques récursifs (RSC) séparés par un entrelaceur.

3.3.1 Encodeur TCC:

Le schéma de structure du turbo code est représenté sur la Figure 3.5. Le taux de codage du codeur Turbo est 1/3.

Comme nous l'avons déjà signalé, les Turbo codes profitent d'un algorithme d'entrelacement intégré qui pourrait fournir des améliorations importantes à la communication. L'entrelaceur permute la séquence d'entrée *x* pour obtenir une séquence différente, par conséquent, le poids de ce code sera plus fort avec la combinaison des deux encodeurs.

La fonction de transfert de la longueur de contrainte de code (CL) est :

$$G(D) = \left[1, \frac{g_1(D)}{g_0(D)}\right] \tag{3.32}$$

Où :

$g_0(D) = 1 + D2,$ Pour : $CL = 3,$

$g_1(D) = 1 + D + D2,$

Les valeurs initiales des registres à décalage des codeurs constitutifs à 2^{CL-1} états doivent être toute mises à zéro lorsque le codage de bits d'entrée commence. La sortie du turbo codeur est :

$$x_1, p_1, p'_1, x_2, p_2, p'_2, \dots, x_K, p_K, p'_K$$

Où $x_1, x_2, \dots, x_K$ sont les bits d'entrée, les bits entrelacés sont désignés par $x'_1, x'_2, \dots, x'_K$, K est le nombre de bits dans la séquence d'entrée, $p_1, p_2, \dots, p_K$ et $p'_1, p'_2, \dots, p'_K$ sont les bits de parité du premier et deuxième codeur constitutif à 2^{CL-1} états, respectivement.

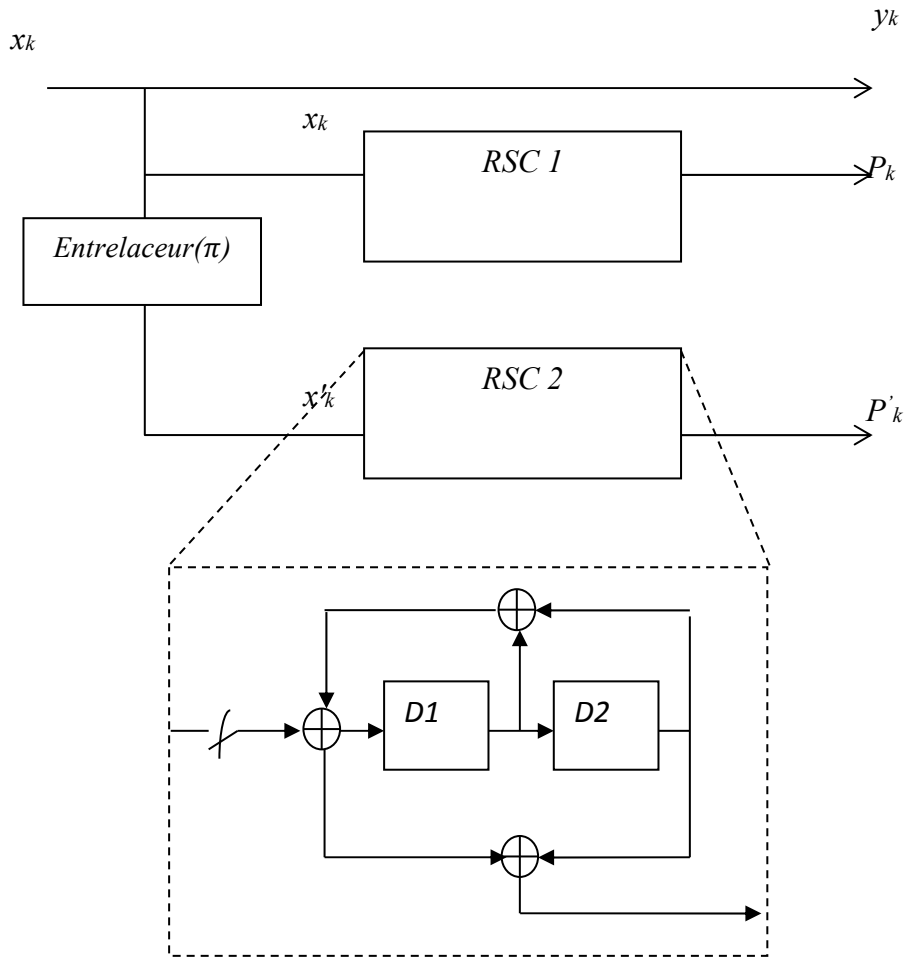


Figure 3.5: Structure d'un Turbo codeur de taux $\mathbf{1/3}$ ($\boldsymbol{CL = 3}$)

La fermeture de treillis « bits de queue ou tail-biting » est un problème pour les turbo codes. En effet, m bits supplémentaires doivent être envoyés à travers le canal, pour forcer l'encodeur à un état connu à la fin du bloc. La fermeture du treillis garantit donc que l'état de départ du codeur est le même que l'état de fin, si le treillis se termine dans un état particulier s0, cela sera a priori connu par le décodeur pour lequel il fixera α (s0) = 1 et α (s) = 0 pour s0 ≠ s. Par conséquent, sans connaissance des états de début et de fin, des erreurs supplémentaires se produisent en raison d'une mauvaise fin du bloc de données résultant.

Pour un codeur convolutionnel de longueur de contrainte $CL = 3$, $m = 2$ et $R = 1/3$, les bits de queue de la séquence systématique sont $\{x_{K+1}, x_{K+2}\}$ correspondant au premier encodeur. En plus de ces 2 bits de queue, 4 bits de queue de parité correspondants $\{p_{K+1}, p_{K+2}\}$ et $\{p'_{K+1}, p'_{K+2}\}$ pour le premier et le deuxième encodeur

respectivement sont également transmis. Les bits de queue sont ensuite transmis à la fin de la trame de données codée. La séquence supplémentaire des bits de queue est donc :

$$x_{K+1}, p_{K+1}, x_{K+2}, p_{K+2}, p'_{K+1}, p'_{K+2}$$

La longueur totale de la séquence de bits codée devient maintenant $3K + 6$, $3K$ étant les bits de données codés et 6 étant les bits de queue. Le taux de codage du codeur est donc $R = K/(3K + 6)$. Cependant, pour une grande taille de K, la perte fractionnaire du taux de codage due aux bits de queue est négligeable et ainsi, le taux de codage est approximé à $1/3$.

Un code convolutif peut être converti dans la description en treillis comme dans la Figure 3.6:

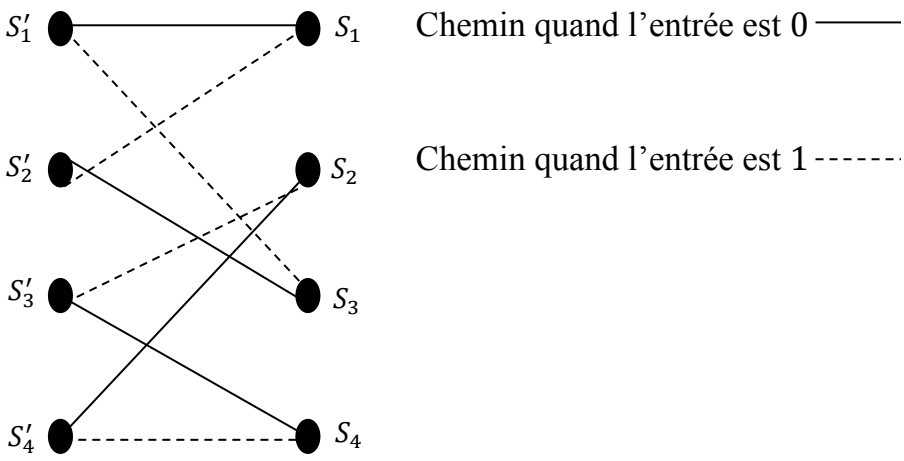


Figure 3.6 : Treillis pour $CL = 3$

Le diagramme de Treillis est composé par la matrice des nœuds (les cercles remplis noirs indiqués ci-dessus). Chaque colonne des nœuds représente tous les états possibles et leurs précédentes et prochaines sorties.

3.3.2 Décodeur TCC :

Soit (y, p, p') la séquence reçue de longueur $3 \times K$, où y représente les bits systématiques, p et p'représentent les bits de parité reçus. Le décodeur 1 utilise les informations (y, p) pour calculer l'information a postériori $L^{+}(y)$ en se basant sur l'information extrinsèque $L^{e}(y')$ provenant à partir du décodeur 2 et calculer l'information extrinsèque $L^{e}(y)$ qui doit être utilisée d'une façon similaire par le décodeur 2 pour le calcul de l'information a postériori $L^{+}(y')$ en utilisant l'information (y', p'), où y' représente les bits systématiques entrelacés. Durant les itérations, les deux décodeurs échangent l'information extrinsèque pour améliorer la décision sur les bits de message x_K en se basant sur les valeurs l'information a postériori du deuxième décodeur. L'algorithme du Turbo décodeur est résumé dans la Figure 3.7.

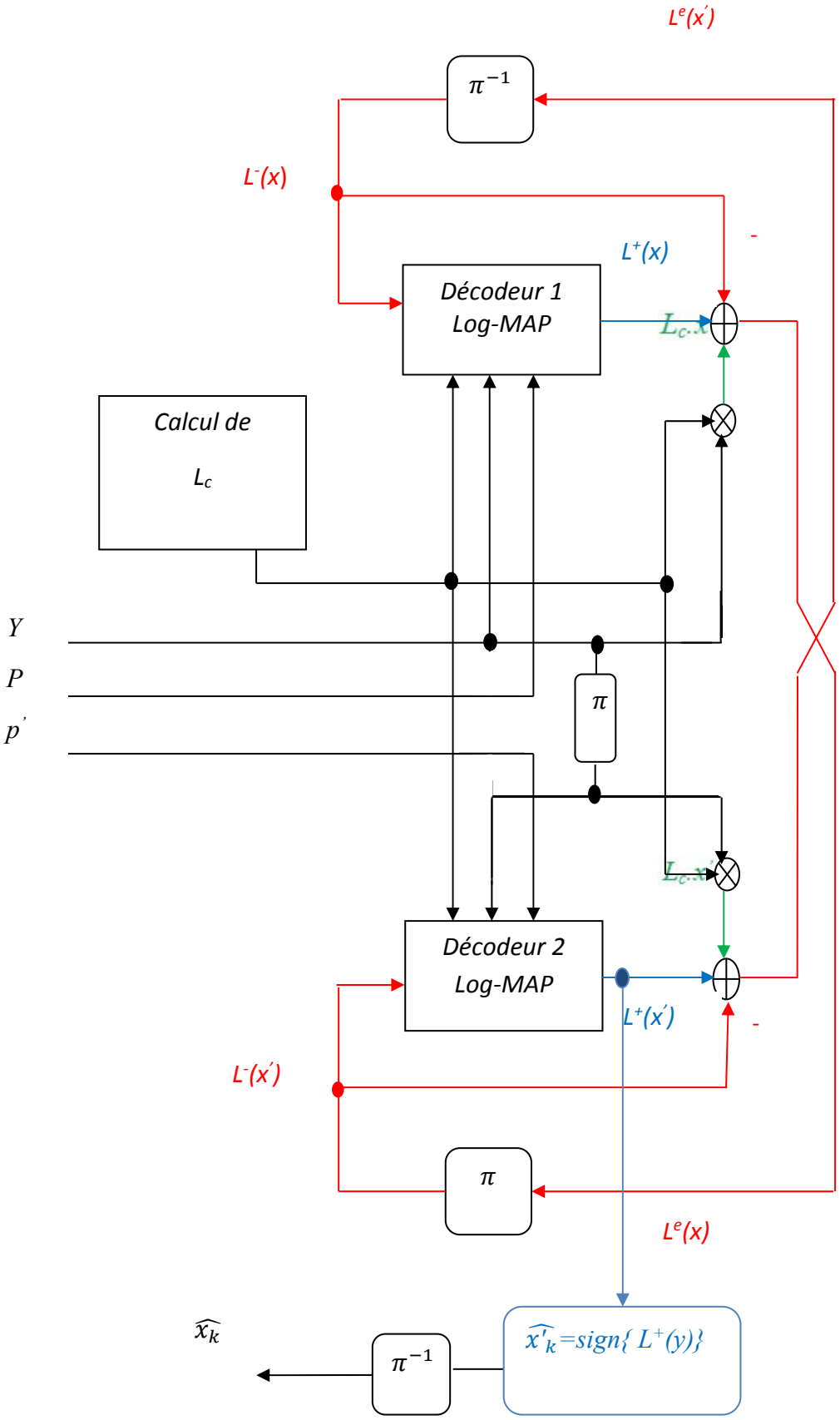


Figure 3.7: Turbo décodage itératif

L'information a postériori pour le k-*ème* bit de donnée x_k est calculée en utilisant le rapport logarithmique de vraisemblance (LLR). Bahl et al [47] ont proposé l'algorithme MAP également nommé BCJR pour le décodage convolutif afin de calculer le LLR conditionné d'un bit décodé x_k, à l'instant k, conditionné par le signal reçu y^K pour une séquence de K bits :

$$L^+(x_k, y_1^K) = log\left(\frac{P(x_k=1, y_1^K)}{P(x_k=-1, y_1^K)}\right) \tag{3.33}$$

Nous pouvons remplacer $x_k = \pm 1$ dans (3.33) par deux états s' (état initial) et s (état final), et (3.33) devient :

$$L^+(x_k, y_1^K) = log\left(\frac{\sum_+ P(s', s, y_1^K)}{\sum_- P(s', s, y_1^K)}\right) \tag{3.34}$$

Où $\sum_{\pm}$ est la somme sur toutes les paires possibles de la branche de transition (s', s) dans le treillis, à l'instant k pour $x_k = \pm 1$.

Nous prenons les K bits de la séquence y et nous les séparons en trois parties, de 1 à $k-1$, le k-*éme* bit, puis à partir de $k+1$ au K-*éme* bit :

$$y_1^K = y_1^{K-1}, y_k, y_{k+1}^K$$

Cela signifie, les points du passé, le point actuel et les futurs points :

$$y_1^K = y_p, y_c, y_f$$

Alors :

$$P(s', s, y_1^K) = P(s', s, y_p, y_c, y_f)$$
$$= P(y_f | s', s, y_p, y_c) P(s', s, y_p, y_c) \tag{3.35}$$

Le terme y_f représente les futurs points qui sont indépendants du passé, du point

actuel et de l'état précédents'. Ils dépendent seulement du présent état s. Nous pouvons donc éliminer ces dépendances :

$$P(s', s, y_1^K) = P(y_f|s)P(s', s, y_p, y_c)$$
$$= P(y_f|s)P(s, y_c|s', y_p)P(s', y_p) \quad (3.36)$$

Notons :

$$\beta_k(s) = P(y_f|s) \quad (3.37)$$

$$\gamma_{k-1}(s', s) = P(s, y_c|s', y_p) \quad (3.38)$$

$$\alpha_{k-1}(s') = P(s', y_p) \quad (3.39)$$

L'équation (3.3.3) peut être écrite comme suit :

$$L^+(x_k) = log\left(\frac{\sum_{x_k=+1} \alpha_{k-1}(s')\beta_k(s)\gamma_{k-1}(s',s)}{\sum_{x_k=-1} \alpha_{k-1}(s')\beta_k(s)\gamma_{k-1}(s',s)}\right) \quad (3.40)$$

Les termes α, β et γ sont la métrique récurrente aller, la métrique récurrente retour et la métrique de transition respectivement que nous allons calculer par le décodeur MAP:

Calcul de Alpha :

$$\alpha_k(s) = \sum_{alls'} P(s, s', y_p, y_c)$$
$$= \sum_{j=0}^{1} \gamma_{k-1}^{j}(s', s)\alpha_{k-1}(s') \quad (3.41)$$

γ est calculée dans la direction aller de treillis.

Pour le décodeur MAP, la condition initiale pour calculer alpha est que, lorsque nous commençons toujours à l'état 1 au 1^{er} bit. Toutes les probabilités sont à ce point égales à 1 ou 0 :

$$\alpha_1(1) = \begin{cases} 1 & si\ s = 1\ et\ k = 1 \\ 0 & si\ non \end{cases} \quad (3.42)$$

Calcul de Beta :

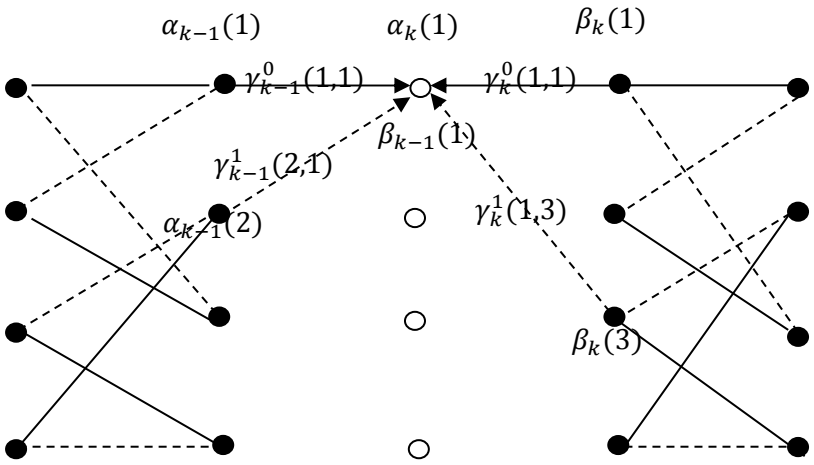
$$\beta_{k-1}(s') = \sum_{j=0}^{1} \beta_k \gamma_k^j\ (s', s) \quad (3.43)$$

γ est calculée dans la direction retour de treillis.

Comme pour les valeurs initiales de l'alpha, nous supposons toujours que le treillis se termine à l'état 1 au K-ième bit. Toutes les probabilités sont à ce point égales à 1 ou 0 :

$$\beta_N(1) = \begin{cases} 1 & si\ s = 1\ et\ k = K \\ 0 & si\ non \end{cases} \quad (3.44)$$

Un exemple pour calculer les métriques α et β est montré sur la Figure 3.8.



$$\alpha_k(1) = \alpha_{k-1}(1).\gamma_{k-1}^0(1,1) + \alpha_{k-1}(2).\gamma_{k-1}^1(2,1)$$

$$\beta_{k-1}(1) = \beta_k(1).\gamma_k^0(1,1) + \beta_k(3).\gamma_k^1(1,3)$$

Figure 3.8 : Exemple pour calculer les métriques aller et retour

Calcul de Gamma :

Les probabilités de la branche de transition en direction aller sont données par [48]:

$$\gamma_{k-1}^0 = \exp\left[\frac{1}{2}L_c(-y_k + p_k Out_0^{ant})\right] \times \frac{1}{1+\exp(L^-(x_k))} \quad (3.45)$$

$$\gamma_{k-1}^1 = \exp\left[\frac{1}{2}L_c(+y_k + p_k Out_1^{ant})\right] \times \frac{\exp(L^-(x_k))}{1+\exp(L^-(x_k))} \quad (3.46)$$

Les probabilités de la branche de transition en direction retour sont données par [48]:

$$\gamma_k^0 = \exp\left[\frac{1}{2}L_c(-y_{k+1} + p_{k+1}Out_0^{ult})\right] \times \frac{1}{1+\exp(L^-(x_{k+1}))} \quad (3.47)$$

$$\gamma_k^1 = \exp\left[\frac{1}{2}L_c(+y_{k+1} + p_{k+1}Out_1^{ult})\right] \times \frac{\exp(L^-(x_{k+1}))}{1+\exp(L^-(x_{k+1}))} \quad (3.48)$$

Où :

Out_0^{ant} : est la valeur antérieure de sortie dans le treillis quand l'entée $=$ 0

Out_1^{ant} : est la valeur antérieure de sortie dans le treillis quand l'entée $=$ 1

Out_0^{ult} : est la valeur ultérieure de sortie dans le treillis quand l'entée $=$ 0

Out_1^{ult} : est la valeur ultérieure de sortie dans le treillis quand l'entée $=$ 1

$$L_c = 4.a.r.\frac{E_b}{N_0} \quad (3.49)$$

L_c représente la valeur de fiabilité du canal de transmission. E_b est l'énergie par bit d'information, N_0 est la densité spectrale de puissance du bruit, a est le fading (variation ou atténuation d'amplitude du signal), pour le canal AWGN non-fading $a = 1$, et r c'est le taux de codage.

En raison de sa complexité croissante, cet algorithme a été simplifié pour Turbo code au Log-MAP qui utilise l'algorithme MAP dans le domaine logarithmique [49].

$$\mathrm{A_k(s')} = \log\alpha_k(s') = \log\sum_{j=0}^{1} exp\left(\gamma_{k-1}^j(s',s) + \alpha_{k-1}(s')\right) \quad (3.50)$$

$$\mathrm{B_{k-1}(s)} = \log\beta_{k-1}(s) = \log\sum_{j=0}^{1} exp\left(\gamma_k^j(s',s) + \beta_k(s)\right) \quad (3.51)$$

$$G_k(s',s) = log\gamma_k(s',s) \quad (3.52)$$

$$L^+(x_k) = log\sum_{x_k=+1} exp(A_{k-1}(s') + B_k(s) + G_{k-1}(s',s)) - log\sum_{x_k=-1} exp(A_{k-1}(s') + B_k(s) + G_{k-1}(s',s)) \quad (3.53)$$

En outre, le Log-MAP peut être exploité en invoquant l'approximation par l'utilisation du logarithme jacobien [50] :

$$\max^*(x,y) \triangleq \log(e^x + e^y) = \max(x,y) + \log\left(1 + e^{-|y-x|}\right) \quad (3.54)$$

Alors, les équations (3.50) (3.51) et (3.52) deviennent :

$$\alpha_k(s_j) = \max^*\{(\alpha_{k-1}(s_{i1}) + \gamma^0_{k-1}(s_{i1}, s_j)), (\alpha_{k-1}(s_{i2}) + \gamma^1_{k-1}(s_{i2}, s_j))\} \quad (3.55)$$

Où s_{i1} et s_{i2} sont deux états au point $k-1$ qui sont reliés à l'état s_j au point k.

$$\beta_{k-1}(s_j) = \max^*\{(\beta_k(s_{ii1}) + \gamma^0_k(s_j, s_{ii1})), (\beta_k(s_{ii2}) + \gamma^1_k(s_j, s_{ii2}))\} \quad (3.56)$$

Où s_{ii1} et s_{ii2} sont deux états au point k qui sont reliés à l'état s_j au point $k-1$.

La probabilité de la branche de transition en direction aller est :

$$G^0_{k-1} = \frac{1}{2}L_c(-y_k + p_k Out_0^{ant}) - log(1 + \exp(L^-(x_k))) \quad (3.57)$$

$$G^1_{k-1} = \frac{1}{2}L_c(+y_k + p_k Out_1^{ant}) + L^-(x_k) - log(1 + \exp(L^-(x_k))) \quad (3.58)$$

La probabilité de la branche de transition en direction retour est :

$$G^0_k = \frac{1}{2}L_c(-y_{k+1} + p_{k+1} Out_0^{ult}) - log(1 + \exp(L^-(x_{k+1}))) \quad (3.59)$$

$$G^1_k = \frac{1}{2}L_c(+y_{k+1} + p_{k+1} Out_1^{ult}) + L^-(x_{k+1}) - log(1 + \exp(L^-(x_{k+1}))) \quad (3.60)$$

Ensuite, l'information extrinsèque est calculée comme suit :

$$L^e(x_k) = L^+(x_k) - L^-(x_k) - L_c y_k \quad (3.61)$$

Le signe de LLR $\{L^+(x_k)\}$ indique si le bit x_k est plus susceptible d'être 1ou 0. Il est calculé à partir du décodeur 2 à chaque itération :

$$\widehat{x_k} = sign\{L^+(x_k)\} \quad (3.62)$$

3.4 Concaténation série des codes LDPC et BCH :

Les codes de vérification de parité à faible densité, également connus sous le nom de LDPC (Low-Density-Parity-Check), sont les codes les plus puissants aujourd'hui. Ils atteignent une très bonne performance de correction d'erreur à des taux de codage élevés, mais ils souffrent souvent de ce que l'on appelle un plancher d'erreur à des rapports signal à bruit élevés pour des faibles MCW. Pour éliminer le plancher d'erreur, il est courant d'utiliser un code algébrique, par exemple le code BCH (Bose-Chaudhuri-Hocquenghem) comme code externe.

La Figure 3.9 montre un codage et décodage commun de canal. Du côté du codage, les données provenant de la source sont codées avec un code BCH (externe), la première séquence de bits résultant est codée avec un code LDPC (interne). La seconde séquence de bits résultant est envoyée à travers le canal. Le décodeur interne, qui effectue un décodage LDPC en utilisant un décodage souple, une décision dure convertit alors l'information douce du décodage LDPC à une séquence binaire dure, car le décodeur externe (BCH) suivant a besoin de bits durs. En d'autres termes, le décodeur BCH est alimenté avec des bits durs dérivés de la sortie du décodeur LDPC, d'où une grande partie de l'information douce disponible après le décodage LDPC est perdue. A cet effet, c'est un inconvénient dans le système global.

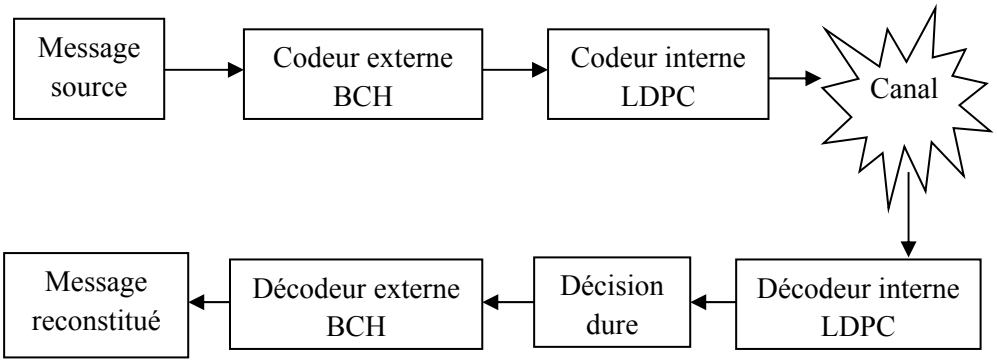


Figure 3.9 : Codes BCH et LDPC Concaténés en série.

Lors de la conception d'un système de communication, l'utilisation du décodage de décision souple fournit des degrés de liberté supplémentaires. Les informations supplémentaires fournies par une décision souple dans la plupart des cas peuvent conduire à un gain de codage supplémentaire ou une performance de correction

d'erreur. Le décodage turbo, consiste à transmettre des décisions souples ou des informations de fiabilité d'un décodeur à l'entrée du décodeur suivant et itérer à travers ce processus.

3.4.1 Le code BCH :

Pour tout entier positif m ($m \geq 3$) et t ($t < 2m - 1$), il existe un code BCH avec les paramètres suivants [51] :

- Block length : $n = 2^m - 1$
- Number of parity-check digits : $n - k \leq mt$
- Minimum distance : $d_{min} \geq 2t + 1$

Ces caractéristiques impliquent que ce code est capable de corriger toute combinaison de t ou quelques nombres d'erreurs dans un code de bloc $n = 2^m - 1$ bits. Le polynôme générateur de ce code est spécifié en termes de ses racines à partir du champ Galois GF (2^m). Si α est l'élément primitif de GF (2^m), alors le polynôme générateur $g(x)$ du code BCH de longueur $2^m - 1$ est le polynôme de degré le plus bas avec les racines comme $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2t}$ et aussi leurs conjugués [ie, $g(\alpha^i) = 0$, pour $1 \leq i \leq 2t$]. Si $\Phi_i(x)$ a la racine comme α^i, $g(x)$ est le plus petit commun multiple de $\Phi_1(x), \Phi_2(x), \ldots, \Phi_{2t}(x)$, c'est-à-dire :

$$g(x) = LCM\{\Phi_1(x), \Phi_2(x), \Phi_3(x), \ldots, \Phi_{2t}(x)\} \quad (3.63)$$

Où LCMsignifie (Least Common Multiple).

Si i est pair, cela peut être exprimé par $i = k2^l$, où k est un entier impair et $l \geq 1$. Alors α^i a un conjugué α^k, et donc α^iet α^k ont le même polynôme. Par conséquent, $g(x)$ peut être exprimé comme suit :

$$g(x) = LCM\{\Phi_1(x), \Phi_3(x), \Phi_5(x), \ldots, \Phi_{2t-1}(x)\} \quad (3.64)$$

Comme le degré de chaque polynôme est $<= m$, le degré de $g(x)$ est au plus de $m.t$. De l'équation (3.64), le code de correction d'erreur unique peut être généré par $g(x) = \Phi 1\ (x)$.

Si $\Phi(x)$ est le polynôme avec les coefficients du champ de Galois GF (2^m), qui est irréductible, on l'appelle polynôme minimal. Il a le degré $\leq m$ et il divise $x^{2m} + x$.

Par exemple, les polynômes minimaux de $\gamma = \alpha^7$ dans GF (2^4) est :

$\phi(x) = 1 + x^3 + x^4$.

Les polynômes minimaux sont obtenus en résolvant l'équation :

$\Phi(\mathrm{x}) = \alpha_0 + \alpha_1 \mathrm{x} + \alpha_2 \mathrm{x}^2 + \alpha_3 \mathrm{x}^3 + \mathrm{x}^4$

où x est substitué par $\gamma = \alpha^7$ et $m = 4$ pour GF (2^4). Les polynômes minimaux générés par $p(x) = x^4 + x + 1$ sont donnés au Tableau 3.2.

Racines conjuguées	Polynôme Minimal
0	x
1	$x+1$
$\alpha, \alpha^2, \alpha^4, \alpha^8$	$x^4 + x + 1$
$\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$	$x^4 + x^3 + x^2 + x + 1$
α^5, α^{10}	$x^2 + x + 1$
$\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}$	$x^4 + x^3 + 1$

Tableau 3.2 : Polynômes minimaux dans GF (2^4) Généré par $p(x) = x^4 + x + 1$

En utilisant les polynômes minimaux, un champ de Galois d'ordre quelconque peut être construit et des polynômes générateurs peuvent être sélectionnés en fonction de la capacité de correction d'erreur. Le polynôme générateur *g(x)* du code BCH à t correction d'erreurs, de longueur de code $n = 2^m - 1$ peut-être généré si chaque code polynomial a les racines $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2t}$ et leurs conjugués. Maintenant, si $v(x)$ est le polynôme du code de GF(2), il est divisible par les polynômes minimaux $\Phi_1(x), \Phi_2(x), \ldots, \Phi_{2t}(x)$. Par conséquent, $v(x)$ est également divisible par générateur polynomial $g(x)$, où $g(x) = LCM\{\Phi_1(x) \cdot \Phi_2(x) \cdot \Phi_3(x) \ldots \Phi_{2t}(x)\}$. Une liste de polynômes générateurs de codes BCH binaires d'une longueur de bloc allant jusqu'à $2^5 - 1$ est donnée dans le Tableau 3.3.

n	k	t	Les générateurs polynomiales
7	4	1	$1 + x + x^3$
15	11	1	$1 + x + x^4$
15	7	2	$1 + x^4 + x^6 + x^7 + x^8$
15	5	3	$1 + x + x^2 + x^4 + x^5 + x^8 + x^{10}$

31	26	1	$1+x^2+x^5$
31	21	2	$1+x^3+x^5+x^6+x^8+x^9+x^{10}$
31	16	3	$1+x+x^2+x^3+x^5+x^7+x^8+x^9+x^{10}+x^{11}+x^{15}$
31	11	5	$1+x^2+x^4+x^6+x^7+x^9+x^{10}+x^{13}+x^{17}+x^{18}+x^{20}$
31	6	7	$1+x+x^2+x^5+x^9+x^{11}+x^{13}+x^{14}+x^{15}+x^{16}+x^{18}+x^{19}+x^{21}+x^{24}+x^{25}$

Tableau 3.3 : Polynômes génératrices du code BCH

Si le code polynomial BCH est : $v(x) = v_0 + v_1x + v_2x^2 + \ldots + v_{n-1}x^{n-1}$ de longueur du code $2^m - 1$ a la racine α^i, alors :

$$v(\alpha^i) = v_0 + v_1\alpha^i + v_2\alpha^{2i} + \ldots + v_{n-1}\alpha^{(n-1)i} = 0 \tag{3.65}$$

Si on considère une matrice H constituée des éléments primitifs qui sont les racines du vecteur de code comme suit :

$$H = \begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \ldots & \alpha^{n-1} \\ 1 & \alpha^2 & (\alpha^2)^2 & (\alpha^2)^3 & \ldots & (\alpha^2)^{n-1} \\ 1 & \alpha^3 & (\alpha^3)^2 & (\alpha^3)^3 & \ldots & (\alpha^3)^{n-1} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ 1 & \alpha^{2i} & (\alpha^{2i})^2 & (\alpha^{2i})^3 & \ldots & (\alpha^{2i})^{n-1} \end{bmatrix} \tag{3.66}$$

Comme v est un vecteur de code, alors :

$$v \cdot H^T = 0 \tag{3.67}$$

L'équation (3.67) donne la matrice de contrôle de parité H, alors si α^i est une racine de l'un des polynômes minimaux de l'équation (3.64), alors ses conjugués sont aussi des racines. Ces conjugués apparaissent dans les rangées paires de Eq. (3.67), donc, ils peuvent être enlevés de H.

$$H = \begin{bmatrix} 1 & \alpha & \alpha^2 & \alpha^3 & \ldots & \alpha^{n-1} \\ 1 & \alpha^3 & (\alpha^3)^2 & (\alpha^3)^3 & \ldots & (\alpha^3)^{n-1} \\ 1 & \alpha^5 & (\alpha^5)^2 & (\alpha^5)^3 & \ldots & (\alpha^5)^{n-1} \\ . & . & . & . & \ldots & . \\ . & . & . & . & \ldots & . \\ 1 & \alpha^{2i-1} & (\alpha^{2i-1})^2 & (\alpha^{2i-1})^3 & \ldots & (\alpha^{2i-1})^{n-1} \end{bmatrix} \tag{3.68}$$

où Chaque élément de H est un élément GF (2^m) qui peut être représenté par un vecteur de m bits.

Le code BCH avec une distance minimale d'au moins d_0 n'a pas plus de $m(d_0 - 1)$ bits de contrôle de parité et il est capable de corriger ou moins $(d_0 - 1)/2$ d'erreurs. La limite inférieure de la distance minimale est appelée la limite de BCH.

$m = 2$	1 (0 1 2)	$m = 6$	1 (0 1 6) 3 (0 1 2 4 6) 5 (0 1 2 5 6) 7 (0 3 6) 9 (0 2 3) 11 (0 2 3 5 6) 13 (0 1 3 4 6) 15 (0 2 4 5 6) 21 (0 1 2) 23 (0 1 4 5 6) 27 (0 1 3) 31 (0 5 6)
$m = 3$	1 (0 1 3) 3 (0 2 3)		
$m = 4$	1 (0 1 4) 3 (0 1 2 3 4) 5 (0 1 2) 7 (0 3 4)		
$m = 5$	1 (0 2 5) 3 (0 2 3 4 5) 5 (0 1 2 4 5) 7 (0 1 2 3 5) 11 (0 1 3 4 5) 15 (0 3 5)		

Tableau 3.4 : Éléments primitifs des polynômes minimaux pour $1 < m \leq 6$

Exemple [51]:

Si α est l'élément primitif du champ de Galois GF (2^4), avec $1 + \alpha + \alpha^4 = 0$, les polynômes minimaux sont :

$$\Phi_1(\mathrm{x}) = 1 + \mathrm{x} + \mathrm{x}^4 \tag{3.69}$$

$$\Phi_3(\mathrm{x}) = 1 + \mathrm{x} + x^2 + x^3 + \mathrm{x}^4 \tag{3.70}$$

$$\Phi_5(\mathrm{x}) = 1 + \mathrm{x} + \mathrm{x}^2 \tag{3.71}$$

Le code BCH de correction d'erreur double de longueur $n = 2^4 - 1 = 15$ est généré par :

$$f(x) = LCM\{\Phi_1(x).\Phi_3(x)\} \tag{3.72}$$

$$f(x) = (1 + \mathrm{x} + \mathrm{x}^4)(1 + \mathrm{x} + x^2 + x^3 + \mathrm{x}^4) = 1 + x^4 + x^6 + \mathrm{x}^7 + \mathrm{x}^8 \tag{3.73}$$

Ainsi, ce code est un code cyclique (15, 7) de $d_{min} = 5$.

De même, le code de correction d'erreur triple (15, 5) avec $d_{min} = 7$ peut être généré par le polynôme générateur :

$$f(x) = LCM\{\Phi_1(x).\Phi_3(x).\Phi_5(x)\} \quad (3.74)$$

$$f(x) = (1 + \mathrm{x} + \mathrm{x}^4)(1 + \mathrm{x} + x^2 + x^3 + \mathrm{x}^4)(1 + \mathrm{x} + \mathrm{x}^2) \quad (3.75)$$

$$f(x) = 1 + x + x^2 + x^4 + x^5 + \mathrm{x}^8 + \mathrm{x}^{10} \quad (3.76)$$

Considérons que le vecteur de code transmis $v(x) = v_0 + v_1x + v_2x^2 + \ldots + v_{n-1}x^{n-1}$ est reçu comme: $r(x) = r_0 + r_1x + r_2x^2 + \ldots + r_{n-1}x^{n-1}$. Si $e(x)$ est l'erreur introduite alors :

$$r(x) = v(x) + e(x) \quad (3.77)$$

Le vecteur du syndrome peut être calculé comme :

$$S = r \cdot H^T \quad (3.78)$$

où *H* est donné en (3.68). La *i*^{ème} composante du syndrome est $S_i = r(\alpha^i)$, c'est-à-dire pour $1 \leq i \leq 2t$:

$$S_i = r_0 + r_1\alpha^i + r_2\alpha^{2i} + \cdots + r_{n-1}\alpha^{(n-1)i} \quad (3.79)$$

On peut remarquer que les composants du syndrome sont les éléments du champ GF(2^m). En divisant $r(x)$ par le polynôme minimal $\Phi_i(x)$ de α^i, on aura :

$$r(x) = a_i(x)\,\Phi_i(x) + b_i(x) \quad (3.80)$$

où $b_i(x)$ est le reste avec un degré inférieur à $\Phi_i(x)$. Maintenant $\Phi_i(\alpha^i) = 0$, donc :

$$S_i = r(\alpha^i) = b_i(\alpha^i) \quad (3.81)$$

Ainsi, les composantes du syndrome peuvent être obtenues en évaluant $b_i(x)$ avec $x = \alpha^i$. Puisque α^i est la racine de chaque code polynomial et $v(\alpha^i) = 0$. Des équations (3.77) et (3.81), pour $1 \leq i \leq 2t$, nous pouvons relier les composantes du syndrome avec le modèle d'erreur comme suit :

$$S_i = r(\alpha^i) = b_i(\alpha^i) = e(\alpha^i) \quad (3.82)$$

On peut remarquer que les composantes du syndrome S_i dépendent du modèle d'erreur. Supposons qu'il existe p nombre d'erreurs dans le modèle d'erreur $e(x)$ aux emplacements $x^{j1}, x^{j2}, \ldots, x^{jp}$, nous pouvons écrire :

$$e(x) = x^{j1} + x^{j2} + x^{j3} + \cdots + x^{jp} \tag{3.83}$$

où $0 < j1 < j2 < \ldots < jp < n$. De l'équation (3.82), nous pouvons déduire l'ensemble d'équations suivant pour S_i.

$$S_1 = \alpha^{j1} + \alpha^{j2} + \alpha^{j3} + \cdots + \alpha^{jp}$$
$$S_2 = (\alpha^{j1})^2 + (\alpha^{j2})^2 + (\alpha^{j3})^2 + \cdots + (\alpha^{jp})^2$$
$$S_3 = (\alpha^{j1})^3 + (\alpha^{j2})^3 + (\alpha^{j3})^3 + \cdots + (\alpha^{jp})^3$$
$$\vdots$$
$$S_{2t} = (\alpha^{j1})^{2t} + (\alpha^{j2})^{2t} + (\alpha^{j3})^{2t} + \cdots + (\alpha^{jp})^{2t}$$

Dans l'ensemble des équations ci-dessus, les racines α^{j1}, α^{j2}, α^{j3},..., α^{jp} sont inconnues. En résolvant ces équations, nous pouvons trouver les emplacements d'erreur. Toute méthode pour résoudre ces équations est un algorithme de décodage pour les codes BCH, parmi ces algorithmes on trouve le Berlekamp-Massey (BM).

3.4.2 LDPC et BCH codes dans le standard DVB-S2 :

Comme mentionné ci-dessus, il est courant d'utiliser le code LDPC comme code interne et le code BCH comme code externe.

Le standard DVB-S2 (deuxième génération de télévision numérique par satellite) [23] a été défini autour de trois concepts clés :

- La meilleure performance de transmission.
- La flexibilité totale.
- La complexité raisonnable du récepteur.

Ces concepts sont atteints grâce à la nouvelle méthode de codage source MPEG-4 AVC (Advanced Video Coding), H264/AVC et à l'utilisation de la concaténation en série des codes LDPC et BCH. La configuration du système DVB-S2 permet d'utiliser

4 types de modulations PSK, concrètement QPSK, 8PSK, 16APSK et 32APSK. De nos jours, la norme DVB-S2 est orientée vers le support de la diffusion TV en qualité HDTV (Télévision Haute Définition). Selon le scénario de transmission actuel, le système DVB-S2 permet de larges possibilités de paramétrage :

• 11 taux de codage différents (1/4, 1/3, 2/5, 1/2, 3/5, 2/3, 3/4, 4/5, 5/6, 8/9, 9/10).

• 2 séquences possibles (séquence normale (64800) et séquence courte (16200)).

Ce système doit effectuer un codage externe (BCH), un codage interne (LDPC) et un entrelacement de bits. La séquence d'entrée doit être composée de K_{bch} bits et la séquence de sortie de N_{ldpc} bits.

Chaque K_{bch} bits doivent être traités par le système de codage FEC, pour générer N_{ldpc} bits. Les bits de contrôle de parité (BCH FEC) du code externe BCH systématique doivent être ajoutés après les K_{bch} bits, et les bits de contrôle de parité (LDPC FEC) du codeur LDPC interne doivent être ajoutés après BCH FEC, comme indiqué dans la figure 3.10.

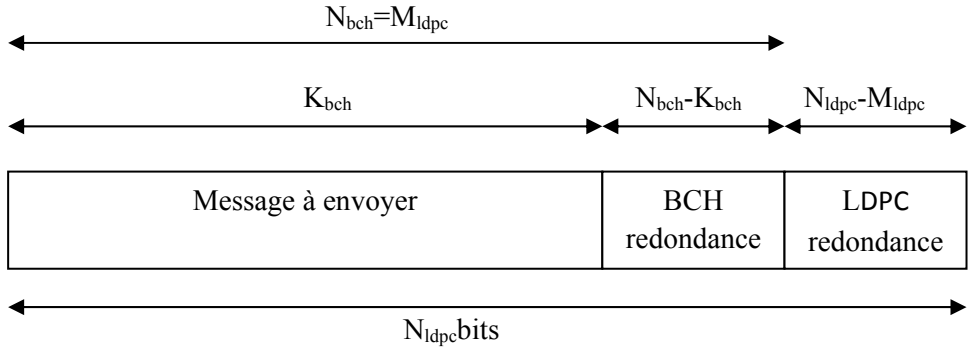


Figure 3.10 : Format de donnée après l'encodage et sans entrelacement

NB : N_{ldpc}=64800 bits pour une longueur de trame normal, N_{ldpc} =16200 bits pour une longueur de trame court.

Le Tableau 3.5 donne les paramètres de codage FEC pour le N_{ldpc} bits normal (N_{ldpc} = 64800 bits) et le Tableau 3.6 pour le court N_{ldpc} bits (N_{ldpc} = 16 200 bits).

Taux de codage LDPC	BCH : Bloc non codé K_{bch}	BCH : Bloc codé N_{bch} LDPC : Bloc non codé M_{ldpc}	BCH : capabilité de correction d'erreur t	LDPC : Bloc codé N_{ldpc}
1/4	16008	16200	12	64800
1/3	21408	21600	12	64800
2/5	25728	25920	12	64800
1/2	32208	32400	12	64800
3/5	38688	38880	12	64800
2/3	43040	43200	10	64800
3/4	48408	48600	12	64800
4/5	51648	51840	12	64800
5/6	53840	54000	10	64800
8/9	57472	57600	8	64800
9/10	58192	58320	8	64800

Tableau 3.5 : Paramètres de codage pour la longueur de trame normale long N_{ldpc}=64800

Taux de codage LDPC	BCH : Bloc non codé K_{bch}	BCH : Bloc codé N_{bch} LDPC : Bloc non codé M_{ldpc}	BCH : capabilité de correction d'erreur t	Taux de codage effectif LDPC M_{ldpc}/16200	LDPC : Bloc codé N_{ldpc}
1/4	3072	3240	12	1/5	16200
1/3	5232	5400	12	1/3	16200
2/5	6312	6480	12	2/5	16200
1/2	7032	7200	12	4/9	16200
3/5	9552	9720	12	3/5	16200
2/3	10632	10800	12	2/3	16200
3/4	11712	11880	12	11/15	16200
4/5	12432	12600	12	7/9	16200
5/6	13152	13320	12	37/45	16200
8/9	14232	14400	12	8/9	16200
9/10	NA	NA	NA	NA	NA

Tableau 3.6 : Paramètres de codage pour la longueur de trame normale courte N_{ldpc}=16200

3.4.2.1 Encodage BCH externe :

Un code BCH (N_{bch}, K_{bch}) de t corrections d'erreur doit être appliqué au message à envoyer de longueur (K_{bch}) pour générer un paquet protégé contre les erreurs. Les paramètres du code BCH pour N_{ldpc} = 64800 sont donnés dans le Tableau 3.5 et pour N_{ldpc} = 16200 dans le Tableau 3.6.

Le polynôme générateur du codeur BCH corrigeant t erreurs est obtenu en multipliant les t premiers polynômes dans le Tableau 3.7 pour N_{ldpc} = 64800 et dans le Tableau 3.8 pour N_{ldpc} = 16200.

$g_1(x)$	$1 + x^2 + x^3 + x^5 + x^{16}$
$g_2(x)$	$1 + x + x^4 + x^5 + x^6 + x^8 + x^{16}$
$g_3(x)$	$1 + x^2 + x^3 + x^4 + x^5 + x^7 + x^8 + x^9 + x^{10} + x^{11} + x^{16}$
$g_4(x)$	$1 + x^2 + x^4 + x^6 + x^9 + x^{11} + x^{12} + x^{14} + x^{16}$
$g_5(x)$	$1 + x + x^2 + x^3 + x^5 + x^8 + x^9 + x^{10} + x^{11} + x^{12} + x^{16}$
$g_6(x)$	$1 + x^2 + x^4 + x^5 + x^7 + x^8 + x^9 + +x^{10} + x^{12} + x^{13} + x^{14} + x^{15} + x^{16}$
$g_7(x)$	$1 + x^2 + x^5 + x^6 + x^8 + x^9 + x^{10} + x^{11} + x^{13} + x^{14} + x^{16}$
$g_8(x)$	$1 + x + x^2 + x^5 + x^6 + x^8 + x^9 + x^{12} + x^{13} + x^{14} + x^{16}$
$g_9(x)$	$1 + x^5 + x^7 + x^9 + x^{10} + x^{11} + x^{16}$
$g_{10}(x)$	$1 + x + x^2 + x^5 + x^7 + x^8 + x^{10} + x^{12} + x^{13} + x^{14} + x^{16}$
$g_{11}(x)$	$1 + x^2 + x^3 + x^5 + x^9 + x^{11} + x^{12} + x^{13} + x^{16}$
$g_{12}(x)$	$1 + x + x^5 + x^6 + x^7 + x^9 + x^{11} + x^{12} + x^{16}$

Tableau 3.7 : Polynômes génératrices du code BCH (Pour la trame normale N_{ldpc}=64800)

$g_1(x)$	$1 + x + x^3 + x^5 + x^{14}$
$g_2(x)$	$1 + x^6 + x^8 + x^{11} + x^{14}$
$g_3(x)$	$1 + x + x^2 + x^6 + x^9 + x^{10} + x^{14}$
$g_4(x)$	$1 + x^4 + x^7 + x^8 + x^{10} + x^{12} + x^{14}$
$g_5(x)$	$1 + x^2 + x^4 + x^6 + x^8 + x^9 + x^{11} + x^{13} + x^{14}$
$g_6(x)$	$1 + x^3 + x^7 + x^8 + x^9 + x^{13} + x^{14}$
$g_7(x)$	$1 + x^2 + x^5 + x^6 + x^7 + x^{10} + x^{11} + x^{13} + x^{14}$
$g_8(x)$	$1 + x^5 + x^8 + x^9 + x^{10} + x^{11} + x^{14}$
$g_9(x)$	$1 + x + x^2 + x^3 + x^9 + x^{10} + x^{14}$
$g_{10}(x)$	$1 + x^3 + x^6 + x^9 + x^{11} + x^{12} + x^{14}$
$g_{11}(x)$	$1 + x^4 + x^{11} + x^{12} + x^{14}$
$g_{12}(x)$	$1 + x + x^2 + x^3 + x^5 + x^6 + x^7 + x^8 + x^{10} + x^{13} + x^{14}$

Tableau 3.8 : Polynômes génératrices du code BCH (Pour la trame courte N_{ldpc}=16200)

3.4.2.2 Encodage LDPC interne :

L'encodage est basé sur la matrice de contrôle de parité H, donc il faut calculer la matrice H code LDPC avec un taux de codage R selon la norme DVB-S2. Par exemple, la matrice H par défaut de vérification de parité (32400x64800) correspond à un code LDPC irrégulier avec la structure indiquée dans le tableau suivant :

Ligne	Nombre de 1 par ligne	Colonne	Nombre de 1 par colonne
1	6	1 à 12960	8
2 à 32400	7	12961 à 32400	3
-	-	32401 à 64799	2
-	-	64800	1

Tableau 3.9 : Nombre de '1' dans les lignes et les colonnes de la matrice H

Les colonnes de 32401 à 64800 forment une matrice triangulaire inférieure. Seuls les éléments sur sa diagonale principale et la sous-diagonale juste en bas sont 1's.

Donc la matrice de contrôle de parité va prendre la forme suivante :

$$H = [H_1 H_2] \tag{3.84}$$

Où H_1 est une matrice de 32400×32400 et H_2 prend la forme :

$$H_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & . & . & 0 \\ 1 & 1 & 0 & 0 & . & . & 0 \\ 0 & 1 & 1 & 0 & . & . & 0 \\ 0 & 0 & . & . & . & . & . \\ . & . & . & . & . & . & . \\ . & . & . & . & 1 & 1 & 0 \\ 0 & 0 & . & . & 0 & 1 & 1 \end{bmatrix} \tag{3.85}$$

Et la matrice génératrice est sous une forme systématique :

$$G = [I\ P] \tag{3.86}$$

$$P = H_1^T H_2^{-T} \tag{3.87}$$

Notons que nous allons utiliser les codes LDPC et BCH pour nos comparaisons dans le chapitre 5. Cependant, pour respecter la longueur du code très court (1024 bits), pour tous les schémas de codage, nous n'allons pas les utiliser sous le format décrit dans le standard DVB-S2.

3.5 Concaténation parallèle des codes Gallager (PCGC) :

Une bonne performance des codes LDPC nécessite généralement une longueur de code élevée, ce qui entraîne un long délai de transmission des données et une complexité de décodage accrue. A cet égard, l'utilisation de codes PCGC (Parallel Concatenated Gallager Codes) [26] a été proposée afin de réduire le délai de transmission de données et la complexité de décodage des codes LDPC. Ce type de codage utilise des codes LDPC concaténés parallèlement dans la structure de turbo code pour rompre en étapes le codage et le décodage assez complexes d'un code long.

3.5.1 Encodeur PCGC :

Un exemple d'un codeur PCGC est montré dans la Figure 3.11. Le paramètre x représente les bits d'informations systémiques, c'est-à-dire les données à coder parles deux codeurs LDPC disposés en parallèle. Le premier codeur LDPC1 génère des bits de parité P_1 et le second codeur LDPC2 génère des bits de parité P_2. En tant que tel, la sortie du codeur PCGC est constituée des bits d'information systématiques x et des bits de parité P1 et P2 pour définir un taux de codage R=1/3.

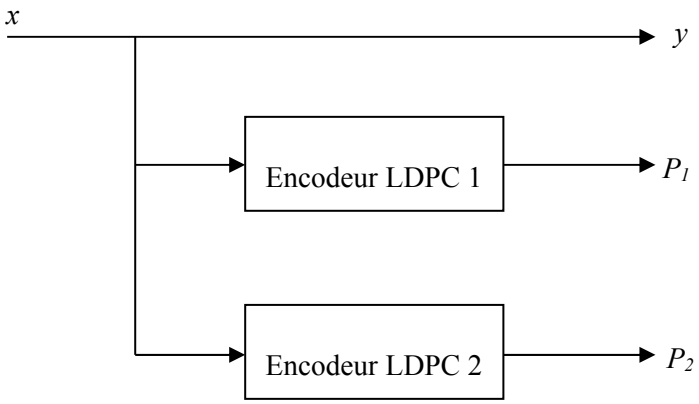
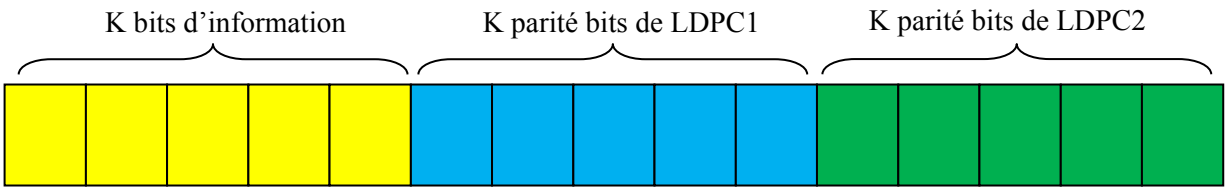


Figure 3.11 : Encodeur PCGC

Dans ce qu'est montré dans la Figure 3.11, un entrelaceur n'est pas utilisé entre les deux codeurs LDPC. Les deux codeurs LDPC implémentent différents codes LDPC ayant des performances d'erreur différentes. A un faible rapport signal à bruit (SNR), le premier codeur LDPC1 présente de meilleures performances que celui implémenté par le deuxième codeur. Cependant, à un SNR élevé, le code LDPC2 implémenté par le deuxième codeur présente une meilleure convergence de décodage et fonctionne mieux que le premier codeur LDPC1 ; en raison d'un nombre plus petit d'erreurs non détectées. En d'autres termes, chaque composant du codeur PCGC a un poids moyen de colonnes (MCW : le poids moyen sur toutes les colonnes de la matrice de parité H) différent pour obtenir différents gains de codage. Cette combinaison de poids différents équivaut à un code LDPC irrégulier qui peut être représenté dans la forme bipartite classique de Tanner-graph, un exemple de Tanner-graphe bipartite de 5x15 PCGC et R=1/3 est représenté sur la Figure 3.12.



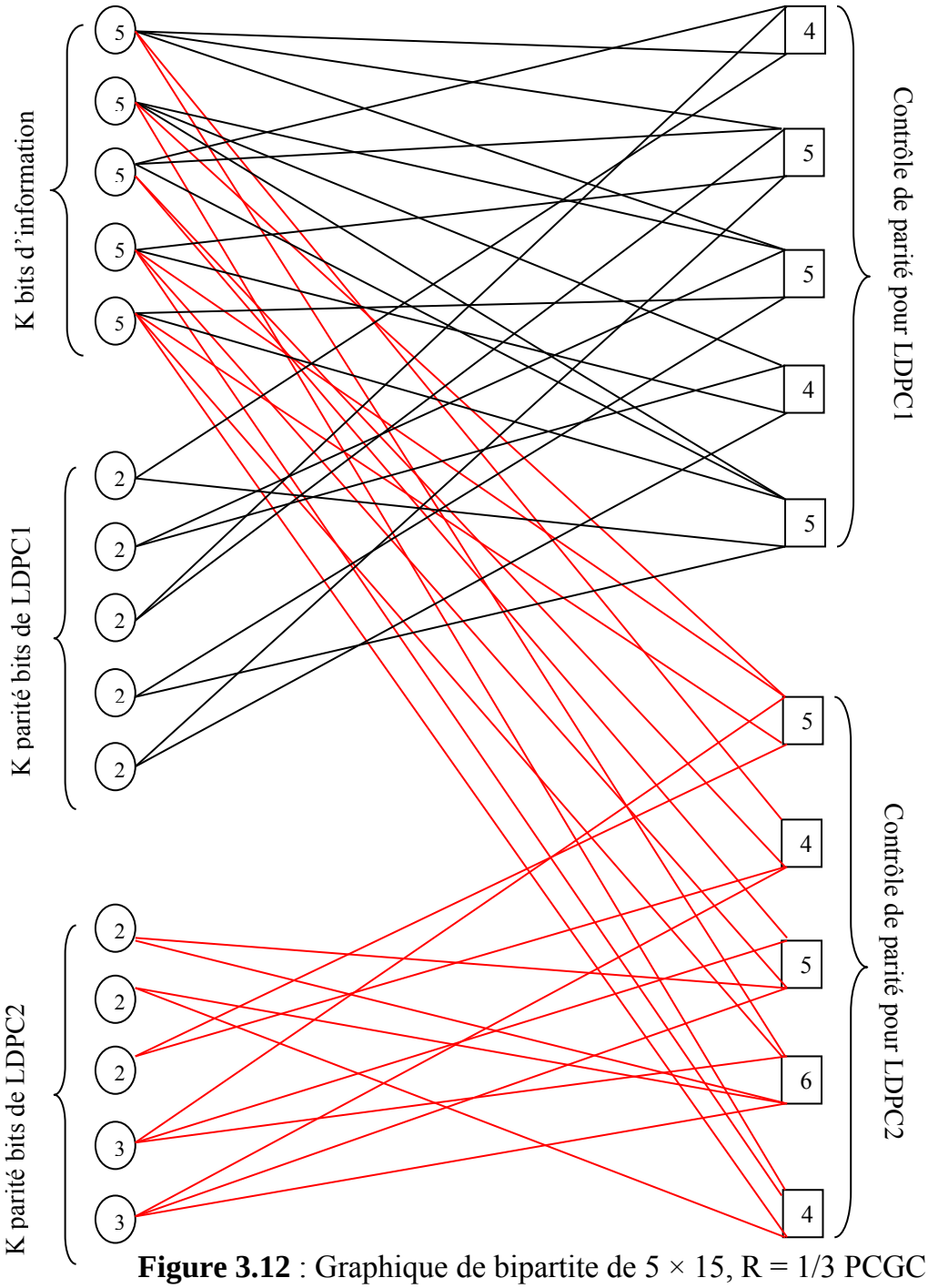


Figure 3.12 : Graphique de bipartite de 5 × 15, R = 1/3 PCGC (MCW1=2,MCW2=2,667)

3.5.2 Codes LDPC réguliers et irréguliers :

Les conditions à satisfaire dans la construction de la matrice de contrôle de parité H d'un code LDPC régulier binaire sont :

• La matrice de contrôle de parité correspondante H doit avoir un poids de colonnes wc et de lignes wr fixe.

• Le nombre de ‘l’ entre deux colonnes adjacents n'est pas supérieur à 1.

• wc et wr doivent être des petits nombres comparés à la longueur du code N et au nombre de lignes dans H.

Cependant, un code LDPC irrégulier a une matrice de contrôle de parité H qui a une variable wc ou wr. En général, la performance du taux d'erreurs binaires (BER) des codes LDPC irréguliers est meilleure que celle des codes LDPC réguliers [52].

3.5.3 Décodeur PCGC :

Comme déjà mentionné, avec de faibles rapports Signal/Bruit (SNR), les codes LDPC construits avec un faible MCW surpassent les codes avec un MCW plus élevé. Cependant, les codes avec un MCW plus élevé montrent une forte performance à des SNR modérés à élevés. Donc, ce mélange de différents poids de colonnes obtenus en combinant deux codes LDPC plus l'algorithme de décodage itératif constituent deux éléments essentiels à l'amélioration des performances obtenues par cette classe de codes correcteurs d’erreurs.

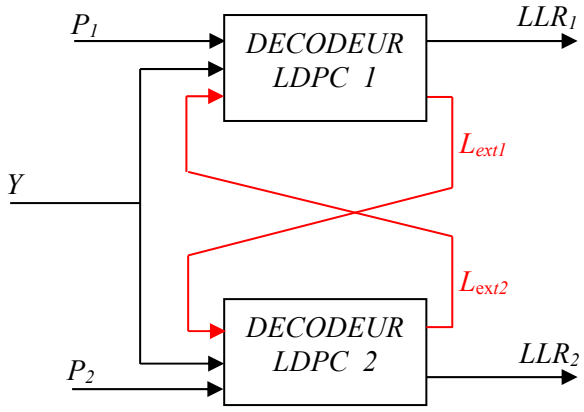


Figure 3.13 : Décodeur PCGC

Un décodeur PCGC est représenté sur la Figure3.13. De manière complémentaire au codeur PCGC de la Figure3.11, le décodeur PCGC comprend un premier décodeur LDPC1 et un second décodeur LDPC2. Le signal reçu est composé de Y, P_1 et $P2$; les bits systématiques et de parité respectivement. Le premier décodeur LDPC1 traite Y et P_1 et le second décodeur LDPC2 traite Y et P_2. Une super itération est définie comme le processus d'échange d'informations entre les deux décodeurs LDPC. Dans chaque super-itération, chaque composant décodeur effectue un certain nombre d'itérations locales avant de transmettre toute information à l'autre composant décodeur. Dans la première super-itération, le décodeur LDPC1 calcule la valeur a postériori LLR1 (rapport de vraisemblance logarithmique) et l'information extrinsèque Le_1 des bits codés correspondants en utilisant les séquences reçues Y et P_1 sans information extrinsèque de deuxième décodeur puisque la valeur de Le_2 est initialisée à 0. Le deuxième décodeur LDPC2 calcule alors la valeur a posteriori de probabilité LLR2 et l'information extrinsèque Le_2 en utilisant les séquences reçues Y et P_2 avec les informations extrinsèques Le_1 disponibles auprès du premier décodeur. Lors des itérations suivantes, les décodeurs LDPC utilisent les informations extrinsèques générées par l'autre décodeur pour calculer les valeurs a postériori LLR et mise à jours les informations extrinsèques. Le processus d'échange des informations entre les composants décodeurs se poursuivent jusqu'à ce que les deux décodeurs convergent vers des mots de code valides, ou qu'un nombre maximal de super-itérations soit atteint.

3.6 Turbo code à trois dimensions (3D-TCC) :

Dans les futures générations de systèmes de communication numérique, un BER inférieur, avec un plancher d'erreur pratiquement minimal, est peut-être nécessaire pour des applications en temps réel plus exigeantes, telles que la communication avec l'espace profond, la télédiffusion HD-HQ ou la vidéo conférence HD-HQ…etc. Les applications commerciales actuelles des Turbo codes tels que le WiMax, DVB-RCS,…etc, sont généralement basées sur des codeurs à 8 états. Cependant, la performance d'un Turbo code peut ne pas être suffisamment grande pour offrir une telle correction d'erreur à des SNR élevés [53]. Il existe plusieurs façons d'améliorer les performances et de minimiser le phénomène de plancher d'erreur. L'utilisation des composants plus forts des codes à 16 états par exemple peuvent diminuer le plancher d'erreur du Turbo code, mais au prix de doubler la complexité du décodage. La concaténation des composants codeurs en série plutôt qu'en parallèle [54] est une autre façon largement utilisée pour améliorer la convergence du BER aux SNR élevés.

Les codes concaténés en série donnent des meilleures convergences aux SNR élevés que la concaténation en parallèle, et le contraire est vrai pour les bas SNR, ce qui pourrait être inacceptable pour certaines applications.

Parmi les dernières alternatives pour améliorer le compromis des performances des Turbo codes on retrouve la structure de concaténation hybride combinant à la fois une concaténation parallèle et sérielle basée sur un code à trois dimensions (3D-TCC). L'approche proposée est dérivée du Turbo code classique en ajoutant une troisième dimension représentée par un codeur à taux partiel. Ce post-codeur est concaténé à la sortie du Turbo code parent, codant seulement une fraction λ des bits de parité. Le 3D-TCC que nous décrivons par la suite est un 3D-TCC binaire.

3.6.1 Encodeur 3D-TCC :

Un schéma de principe de la 3D-TCC est représenté sur la Figure 3.14. La séquence de données d'informations *x* de longueur *K* bits est codée par un turbo codeur conventionnel binaire parent constitué d'une concaténation parallèle de deux codeurs notés ENC1et ENC2 séparés par un entrelaceur π1. Ici ENC1 et ENC2 sont des codeurs convolutifs récursifs à 8 états. Le taux de code global avant l'opération de poinçonnage (puncturing) est de 1/3. Un post-codeur d'un taux unitaire est concaténé à la sortie du turbo code parent, codant seulement une fraction λ des bits de parité délivrés de ce dernier après un deuxième entrelaceur π2 dont la sortie est notée w. Un codeur convolutif récursif binaire à 4 états est utilisé en pratique en tant que post-codeur du 3D-TCC. La fraction λ est appelée taux de perméabilité.

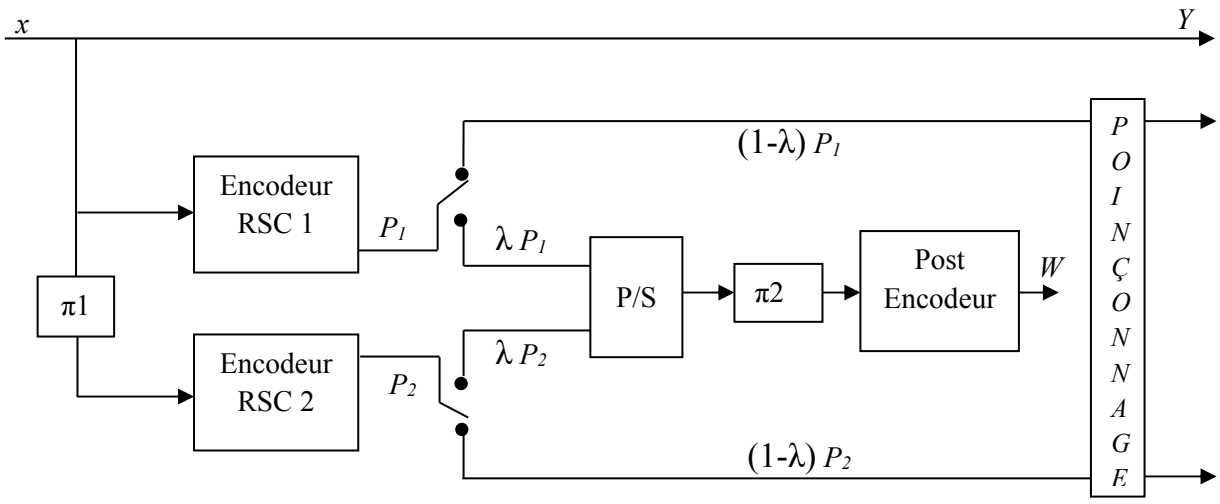


Figure 3.14 : 3D Turbo encodeur

Habituellement, des modèles de perméabilité régulière très simples sont appliqués. Par exemple, si λ = 1/4, les bits à post-coder sont choisis de façon régulière {1000} pour les codeurs ENC1 et ENC2. Ensuite, 1 bit sur 4 est sélectionné régulièrement dans les deux séquences de parité bits (P1 et P2) en commençant par le premier bit de chaque séquence. Notez que le taux de perméabilité a un effet sur les performances du 3D-TCC.

Les courbes du BER d'un turbo code peuvent être divisées en deux régions : la région cascade, où le taux d'erreur diminue rapidement, et la région de plancher d'erreur, où une variation de la courbe apparaît lorsque le code atteint son gain asymptotique. La valeur de λ peut être utilisée pour compenser la performance dans la région de cascade avec la performance dans la région de plancher d'erreur. Si nous choisissons une grande valeur de λ, la distance minimale est significativement augmentée. Cependant, la performance dans la région des cascades est dégradée. Le taux de code global *R* de la 3D-TCC est donnée par :

$$R = \frac{bits\ de\ données}{bits\ systématique + bits\ de\ parité\ post\ encodés + \ autres\ bits\ de\ parité}$$

$$R = \frac{1}{1+2\lambda+2(1-\lambda)\rho} \tag{3.81}$$

Où $0 \leq \rho \leq 1$ est la fraction des bits survivants dans P_1 et P_2 après le poinçonnage. En fait, nous avons *K* bits systématiques, $P = 2 \times \lambda \times K$ bits de parité post-codés et $2 \times (1 - \lambda) \times \rho \times K$ bits de parité survivant après le poinçonnage.

La 3D-TCC est donc caractérisée par deux permutations désignées par π_1 et π_2, comme le montre la Figure 3.14. En théorie, les deux permutations doivent être optimisées conjointement. Cependant, π_1 est la permutation interne du TCC, π_2 est utilisée pour répartir une fraction λ des bits de parité avant de les transmettre au post-codeur. En d'autres termes, π_2 est utilisée pour répartir $P = 2 \times \lambda \times K$ bits de parité à la sortie du TCC avant le post-codage. Le rôle principal de la permutation π_2 est d'éviter que le pré-décodeur renvoie des paquets d'erreurs à l'entrée du décodeur principal. Des recherches ont montrées que le 3D-TCC fonctionne mieux avec un π_2 régulier qu'avec un entrelaceur aléatoire [55] ; la permutation régulière étant un entrelaceur réalisant une répartition de $\sqrt{2P}$, où P est la taille de la séquence de post-encoder.

3.6.2 Décodeur 3D-TCC :

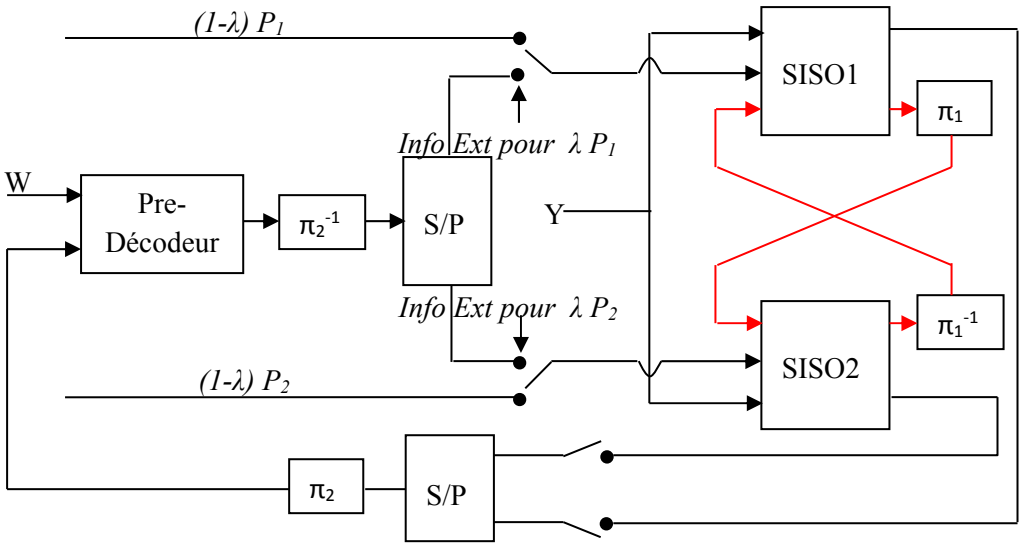


Figure 3.15 : 3D Turbo décodeur

Comme représenté sur la Figure 3.15, le 3D Turbo décodeur comprend un Turbo décodeur classique et un pré-décodeur correspondant au post-codeur et tous échangent des informations extrinsèques. Le principe de turbo décodage classique itératif est toujours utilisé pour le 3D Turbo décodeur. Premièrement, le pré-décodeur à 4 états est activé pour alimenter le décodeur Turbo classique à 8 états avec des informations extrinsèques sur les bits de parité post-codés.

Les deux décodeurs SISO (Soft Input/Soft Output) du Turbo décodeur classique échangent des informations extrinsèques sur les bits systématiques. Ils fournissent également au pré-décodeur des informations extrinsèques sur les bits de parité post-codés. Le processus de décodage se poursuit de manière itérative jusqu'à ce qu'un nombre maximum d'itérations ait été effectué.

Par rapport à un turbo décodeur classique, la complexité supplémentaire est principalement due à la mise en œuvre du décodeur binaire à 4 états mais également au calcul des informations extrinsèques sur les bits de parité post-codés.

3.7 Conclusion :

Dans ce chapitre, nous avons étudié deux méthodes conventionnelles (LDPC et TCC), ainsi qu'un certain nombre de méthodes de correction d'erreurs avancées basées essentiellement sur les codes LDPC et TCC. Ceci, en explorant les différents types de concaténation (Concaténation en série, en parallèle et hybride) cherchant le meilleur compromis possible entre la performance et la complexité adéquate pour chacun des deux codes conventionnels (LDPC et TCC). Également, nous allons présenter dans le chapitre suivant notre contribution dans ce contexte. Par conséquent, tous ces codes sont utilisés dans nos comparaisons dans le chapitre 5.

Chapitre 4

Le système de correction d'erreur avancé LDPC dépasse Turbo code en termes de plancher d'erreur pour les hautes valeurs du SNR. En revanche, le Turbo code présente un plancher d'erreur significative, mais aussi présente une meilleure convergence dans la zone de cascade de la courbe, laissant Turbo code donc mieux adapté pour les bas SNR uniquement.

L'idée clé de notre approche est de bénéficier plus efficacement des avantages du code LDPC et du code convolutif dans la structure du turbo code, et ce pour assurer d'une manière significative un plancher erreur réduit avec une zone de cascade éventuellement escarpée [30]. Cela peut être un bon choix pour la communication dans l'espace profond et la communication mobile des prochaines générations nécessitant des services multimédias de haute qualité.

4.1 Encodeur de la méthode proposée:

La méthode de codage à taux $^1/_3$ consiste à combiner deux encodeurs différents en concaténation parallèle. Pour cela, nous proposons un code LDPC de taux $^1/_2$ comme premier composant encodeur et le code RSC à taux $^1/_2$ en tant que second composant encodeur. Les deux encodeurs travaillent sur la même séquence de bits sans entrelacement, et comme le code RSC a la propriété d'ajouter des bits de queue (tail bits) à la fin de la séquence de bits u, ces bits supplémentaires doivent être pris en compte dans le codeur LDPC comme illustré à la Figure 4.1. Enfin, le codeur transmet le mot code suivant:

$S = \{x_1, x_2, \dots x_K, p_1, p_2, \dots p_M, p'_1, p'_2, \dots p'_K\}$.

Le mot de code contient la parité, les bits systématiques du codeur LDPC et les bits de parité du codeur RSC. Sa longueur est :$K = N - M$, où M et N sont les nombres de lignes et de colonnes de la matrice de contrôle de parité H. Pour le code LDPC de taux $R = 1/2$, on a $M = K = L + m$, où m est la longueur des bits de queue du codeur RSC et L est la longueur du message original u.

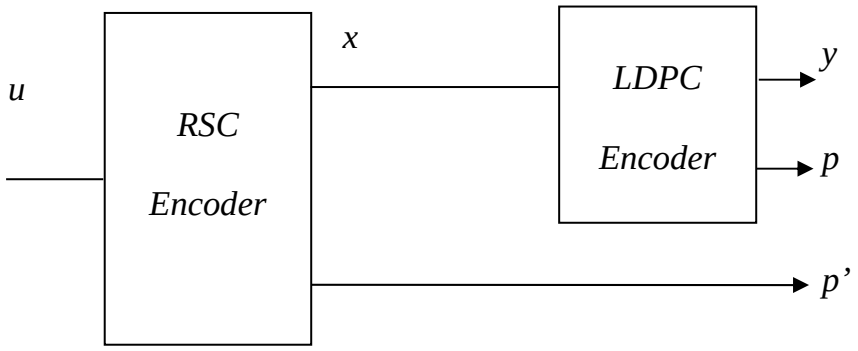


Figure 4.1: Schéma synoptique de l'encodeur pour l'approche proposée

4.2 Décodeur de la méthode proposée:

La Figure 4.2 illustre le schéma de principe du décodeur proposé; le décodage est une collaboration entre deux décodeurs différents montés en parallèle. Pour chaque décodeur constitutif, un algorithme de décodage détermine les valeurs de vraisemblance et extrinsèque pour les symboles d'information. La séquence de valeurs extrinsèques obtenue doit être adaptée est ensuite transmise au décodeur constitutif suivant qui fera de même. De cette manière, chaque décodeur profite des informations fournies par le décodeur précédent comme dans [8,9]. Soit y les bits systématiques reçus après transmission, le mot de code reçu : $C = \{y_1, y_2, \dots y_K, p_1, p_2, \dots p_M, p'_1, p'_2, \dots p'_K\}$ est composé de trois parties ; les bits systématiques r, les bits de parité p du codeur LDPC et les bits de parité p' du codeur RSC respectivement. Le premier sous-mot de code $w = \{p, y\}$ est décodé par le décodeur LDPC et le sous-mot de code $z = \{p', y\}$ est décodé par le décodeur Log-MAP.

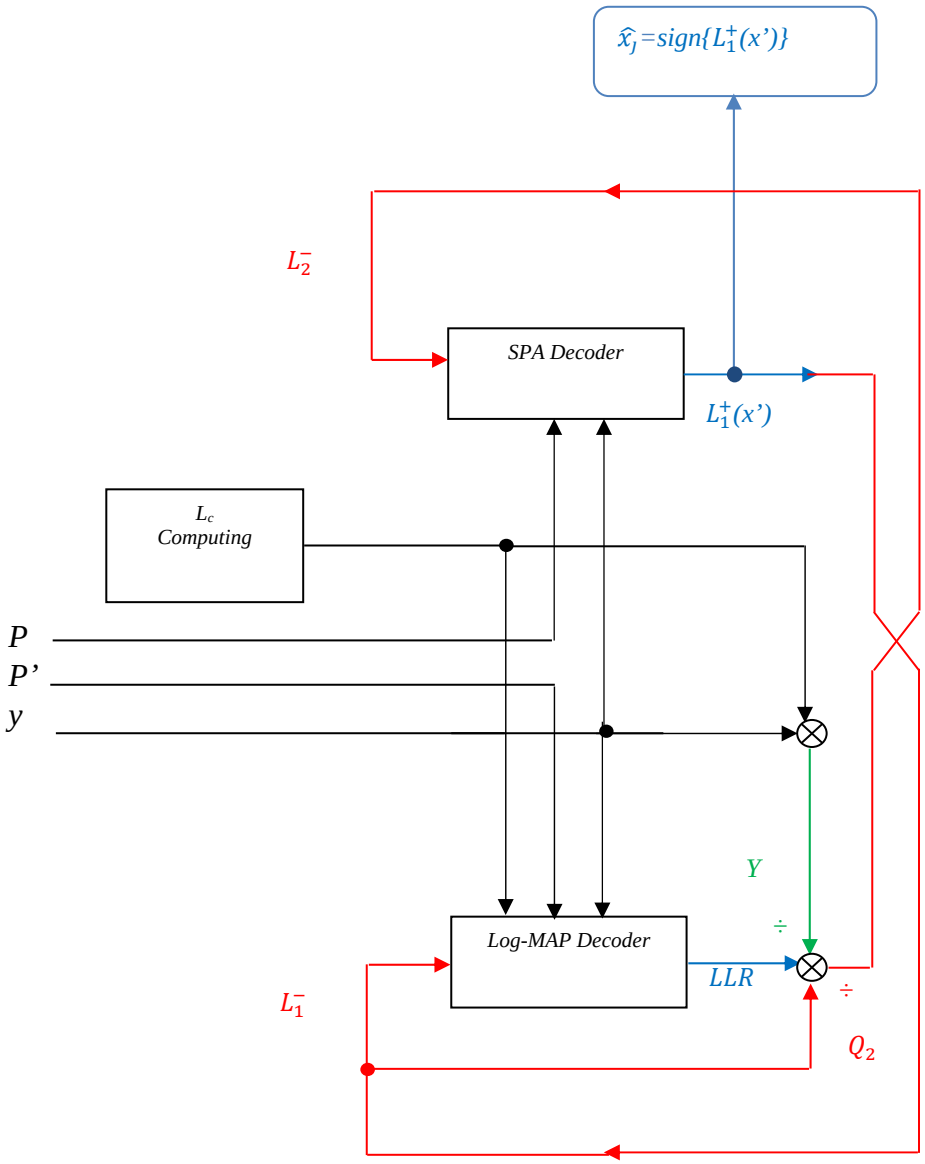


Figure 4.2: Schéma synoptique de décodeur pour l'approche proposée

Le processus de décodage est comme dans Mackay [19] pour DEC1 et comme dans Berrou [9] pour DEC2, à l'exception de quelques modifications. Nous visons à utiliser le décodeur LDPC comme premier décodeur et le décodeur Log-MAP comme second. Donc, nous avons deux niveaux d'itérations; Le décodage LDPC nécessite, à l'intérieur de certaines itérations, des auto-itérations avant de transmettre des informations extrinsèques au second décodeur à la super-itération I.

Dans notre approche, les métriques récurrentes aller-retour de décodeur Log-MAP sont initialisées comme suit:

$$\alpha_i(s) = \begin{cases} 0 & si\ s = 1\ et\ i = 1 \\ -inf & ailleurs \end{cases} \quad (4.1)$$

$$\beta_i(s) = 0 \quad partout$$

Les deux algorithmes SPA et Log-MAP diffèrent en ce que le premier trouve des probabilités marginales dans un arbre, et le second trouve l'état de probabilité maximale du treillis.

Exemple:

Pour l'algorithme Log-MAP:

Nous calculons les deux probabilités de '0' et '1' en utilisant les métriques α, β et γ. C'est un processus récursif, la métrique de la branche aller α est la somme de la métrique aller précédente et de la métrique de la branche γ.

La raison pour laquelle nous appelons ces probabilités des « métriques » est que ce ne sont pas vraiment des probabilités au sens strict, mais juste une mesure de probabilité ou, en terminologie plus précise, "amplitude" car elles peuvent être des valeurs positives ou négatives, comme représenter le Tableau 4.1.

Signal reçu	0.9352	-0.4999	0.1971	0.7910	1.2167	-0.9646
L_{ext}	-10.9842	-1.7728	5.3660	17.1762	-12.2312	6.9623
P(x=1)	1.6073e-8	2.0565e-5	6.2380	0.8692	7.8704e-7	5.8748
P(x=0)	3.9955	0.3803	4.2116e-8	8.676e-12	2.6498	4.0444e-7
L^+	-19.3312	-9.8250	18.8134	25.3303	-15.0294	16.4914
Decision	0	0	1	1	0	1

Tableau 4.1: Exemple de probabilités calculées par l'algorithme Log-MAP à la $5^{ème}$itération

La décision est prise en fonction du signe des valeurs LLR.

$$LLR = log\ (P\ (x = 1)\ /\ P\ (x = 0))$$

Pour l'algorithme SPA:

Le Tableaux 4.2 affiche quelques échantillons réels de simulation, les valeurs de probabilité de Q_1 et Q_0 sont quantifiées par des nombres compris entre 0 et 1 avec $Q_1 + Q_0 = 1$.

Signal reçu	0.8415	1.6546	-1.1545	2.4546	-3.6546	3.1544
L_{ext}=Ap^0/Ap^1	1.1545	0.5565	1.7546	0.2554	31.6544	0.0396
Q1	0.46	0.69	0.37	0.79	0.032	0.97
Q0	0.54	0.31	0.63	0.21	0.968	0.03
L^+	-0.08	0.38	-0.26	0.58	-0.936	0.94
Decision	0	1	0	1	0	1

Tableau 4.2: Exemple de probabilités calculées par l'algorithme SPA

La décision est prise aussi en fonction du signe des valeurs LLR.

$LLR = Q_1 - Q_0$

Par conséquent, les informations échangées entre les décodeurs doivent être adaptées les unes aux autres. L'information extrinsèque présentée dans les probabilités marginales de DEC1 doit être convertie en métrique pour DEC2, et l'information extrinsèque présentée en métrique à partir de DEC2 doit être convertie en probabilité pour DEC1.

A cet effet, l'algorithme SPA est utilisé dans DEC1. La probabilité a priori est calculée comme dans [26,27] par:

$$f_j^1 = P\left(x_j = +1 \middle| y_j\right) = 1/(1 + L_{j,2}^- \, e^{(-2y_j/\sigma^2)}) \quad (4.2)$$

Où : L_2^- est l'information extrinsèque à partir de $DEC2$.

$$f_j^0 = 1 - f_j^1 \quad (4.3)$$

A chaque itération T, on définit l'information extrinsèque L_1^- de longueur K qui est transmise à DEC2. Elle peut être obtenue directement à partir des probabilités finales Q_j^0 et Q_j^1 déjà calculées comme dans [19], divisées par la probabilité a priori, de sorte que seulement l'information extrinsèque du bit systématique est conservée [26,27]:

$$Q_{j,2}^0 = Q_j^0 / f_j^0 \quad (4.4)$$

$$Q_{j,2}^{1} = Q_{j}^{1} / f_{j}^{1} \quad (4.5)$$

$Q_{j,2}^{0}$ et $Q_{j,2}^{1}$ doivent êtres normalisées comme suit:

$$Q_{j,2}^{0} = Q_{j,2}^{0} / Q_{j,2}^{0} + Q_{j,2}^{1} \quad (4.6)$$

$$Q_{j,2}^{1} = Q_{j,2}^{1} / Q_{j,2}^{0} + Q_{j,2}^{1} \quad (4.7)$$

Soit l'équation (4.8) qui calcule la probabilité du bit '0' à partir de la métrique *x*.

$$q^{0} = 1/(1 + \mathrm{e}^{x}) \quad (4.8)$$

Nous allons extraire la valeur de *x* de (4.8) en utilisant l'information de probabilité q^{0}.

$$-\log(q^{0}) = \log(1 + \mathrm{e}^{x}) \quad (4.9)$$

En utilisant les propriétés exponentielles, nous obtenons:

$$\mathrm{e}^{-\log(q^{0})} = 1 + \mathrm{e}^{x} \quad (4.10)$$

$$\mathrm{e}^{x} = \mathrm{e}^{-\log(q^{0})} - 1 \quad (4.11)$$

$$x = \log(\mathrm{e}^{-\log(q^{0})} - 1) \quad (4.12)$$

Ainsi, l'information extrinsèque L_{1}^{-} peut être calculée à partir de la probabilité finale $Q_{j,2}^{0}$ de la partie systématique obtenue par (4.6):

$$L_{1}^{-} = \log(\mathrm{e}^{-\log\left(Q_{j,2}^{0}\right)} - 1) \quad (4.13)$$

Clairement, cette information pertinente sera utilisée comme dans [8,9].

De même, le DEC2 calcule l'information extrinsèque L_{2}^{-} puis la transmet à DEC1. Le Log-MAP travaille sur l'amplitude du signal pour fournir les sorties souples LLR qui indiquent la fiabilité de la décision. Pour que L_{2}^{-} soit compréhensible au DEC1, il doit être converti en probabilité.

Définissons les dépendances suivantes:

$$LLR^{0}(k) = 1/\left(1 + \mathrm{e}^{L_{2}^{+}(k)}\right) \quad (4.14)$$

$$LLR^{1}(k) = 1 - LLR^{0}(k) \quad (4.15)$$

$$Y^{0}(k) = 1/\left(1 + \mathrm{e}^{y(k)}\right) \quad (4.16)$$

$$Y^{1}(k) = 1 - Y^{0}(k) \quad (4.17)$$

Où L_{2}^{+} et y sont des informations a postériori de DEC2 et des informations a priori du bit reçu respectivement.

L'information extrinsèque fournit une mesure de la fiabilité de l'information a postériori. En TCC, c'est le résultat de la soustraction d'une information a postériori à l'information préalable et à l'information extrinsèque de l'autre décodeur. Cette soustraction vise à éviter les phénomènes de propagation d'erreurs lors des itérations.

Dans notre méthode, le calcul des informations extrinsèques est obtenu en divisant la probabilité d'une information a postériori par une information a priori puis par l'information extrinsèque de DEC1. De cette manière, nous obtenons:

$$Ap^0 = LLR^0/Y^0/Q_2^0 \quad (4.18)$$

$$Ap^1 = LLR^1/Y^1/Q_2^1 \quad (4.19)$$

Ap^0et Ap^1 doivent être aussi normalisés comme suit:

$$Ap^0 = Ap^0/Ap^0 + Ap^1 \quad (4.20)$$

$$Ap^1 = Ap^1/Ap^0 + Ap^1 \quad (4.21)$$

Ainsi, l'information extrinsèque est alors le rapport entre Ap^0et Ap^1.

$$L_{2,s}^- = Ap^0/Ap^1 \quad (4.22)$$

$$L_2^- = [L_{2,p}^-, L_{2,s}^-] \quad (4.23)$$

Où, $L_{2,s}^-$ et $L_{2,p}^-$ signifient les parties systématique et de parité de L_2^-.

Cependant, $L_{2,p}^- = 1$ (car les bits de parité de DEC1 ne doivent pas changer).

La décision finale est obtenue directement à partir de la probabilité finale de DEC1 à chaque itération:

$$L_{j,1}^+ = Q_j^1 - Q_j^0 \qquad (\text{Pour } j = 1,2, \dots N) \quad (4.24)$$

$$\widehat{x_j} = \begin{cases} 1\ , si\ sign(L_{j,1}^+ = Q_j^1 - Q_j^0) > 0 \\ 0 \qquad\qquad si\ non \end{cases} \quad (4.25)$$

Le décodage itératif est arrêté si le flux décodé $\widehat{x_j}$ vérifie toutes les équations de contrôle de parité du code:

$$H.\widehat{x^T}_j = 0. \quad (4.26)$$

4.3 Conclusion

Etant donné que l'information extrinsèque des deux décodeurs a été obtenue en deux étapes:

• La première consiste à calculer une probabilité a postériori de la façon où seulement les informations utiles sont conservées. Ceci pour éviter les phénomènes de propagation d'erreurs lors des itérations comme dans PCGC et TCC avec quelques modifications.

• La seconde est de l'adapter pour l'autre décodeur (conversion entre probabilité et métrique).

La solution proposée est envisagée pour améliorer davantage le compromis entre la performance et la complexité par rapport aux autres schémas de codage existants, où les deux informations extrinsèques sont équilibrées de manière très efficace, car si une petite erreur survient dans une information extrinsèque elle causera de sérieuses erreurs à l'autre décodeur et le résultat divergera avec le processus d'itératif.

Chapitre 5

Ce chapitre étudie les codes correcteurs d'erreurs de courte longueur. Nous comparons donc les performances de ces codes dans des conditions identiques. Nous visons par cette comparaison de pouvoir juger le meilleur code en fonction de la performance et de la complexité et qui fournit un taux d'erreur proche de la limite de Shannon même avec une courte longueur de code.

A cet effet, nous avons mené un ensemble d'expériences sur le canal $AWGN$ sous Matlab. Dans ces simulations, la longueur de la séquence d'information est fixée à $K = 1024$ bits pour toutes les expériences. Deux cas sont étudiés dans notre approche en fonction de la moyenne des poids de la colonne (MCW) du code LDPC (Tableau 5.1). Les performances de l'approche proposée sont comparées à celles de $LDPC$ unique, $LDPC\text{-}BCH$, $PCGC$, TCC et $3D\text{-}TCC$ binaire.

$Approche\ proposées$	$DEC1$	$DEC2$
$Cas\ 1$	$MCW = 2.667$	$CL = 3$
$Cas\ 2$	$MCW = 2.5$	$CL = 3$

Tableau 5.1: Les deux expériences de l'approche proposée

La séquence d'information est générée de manière aléatoire. Les symboles codés sont modulés en BPSK et transmis sur un canal gaussien pour un SNR comprise entre 0 et 2,5 dB.

Pour vérifier l'efficacité de la méthode proposée par rapport aux autres méthodes, nous devons exécuter suffisamment de simulation (générer autant de formes de bruit et de séquences de bits aléatoirement) jusqu'à ce que nous ayons un nombre suffisant d'erreurs de bloc pour chaque valeur de SNR. Ce processus garantit que toutes les principales classes de mode de défaillance sont couvertes. Dans nos expériences, le nombre de simulations a atteint plus de 3×10^5 à partir de SNR = 2 dB.

5.1 Analyse du BER:

Pour faciliter la comparaison, nous avons représenté le paramètre de simulation Bit Error Rate (BER) des messages décodés pour les différents algorithmes de décodage sur les mêmes figures.

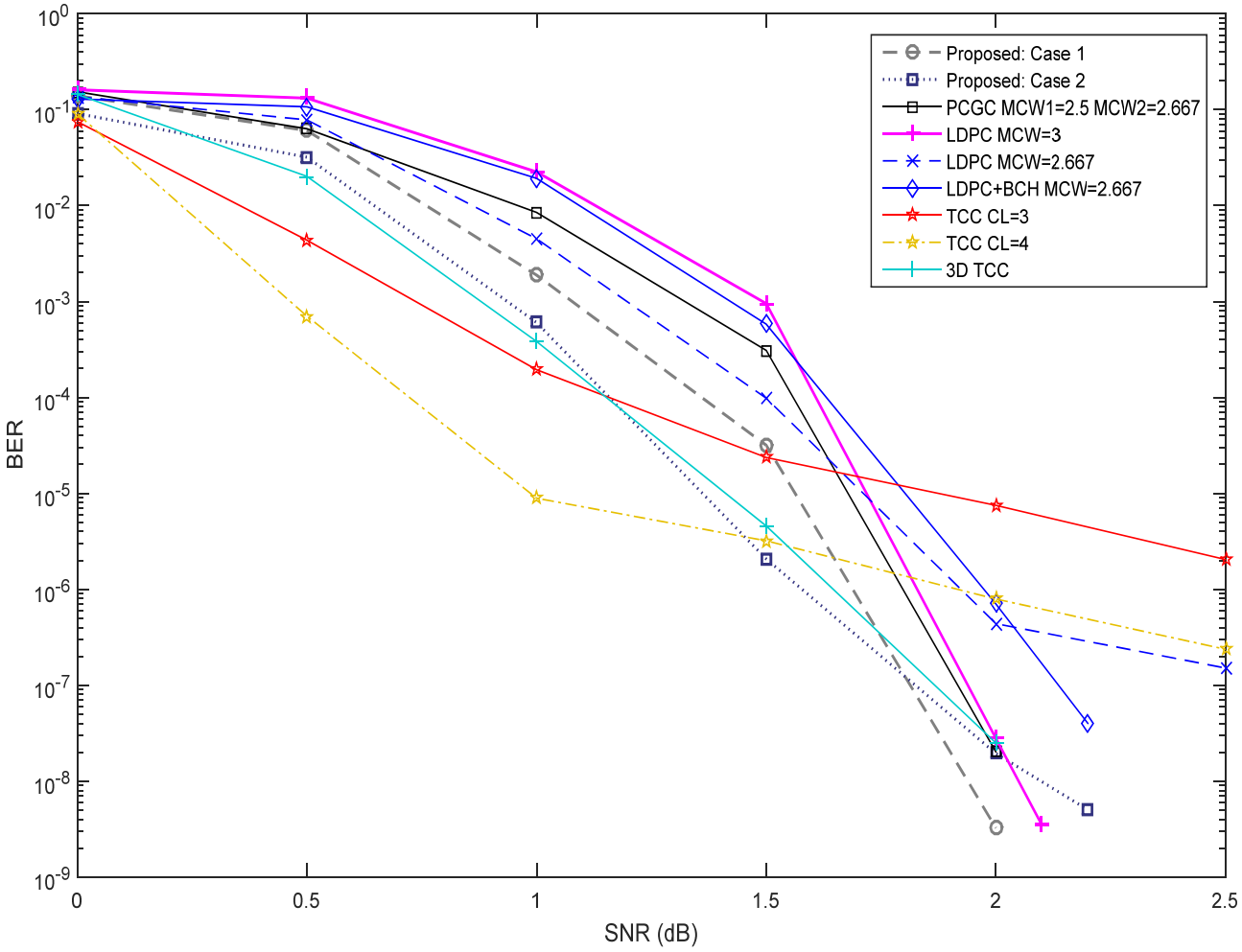


Figure 5.1: BER des différentes approches de décodage y compris notre proposition

En se référant à la Figure 5.1, l'analyse du BER pour les deux expériences (cas 1 et cas 2) considérées dans l'approche proposée, nous permet de constater : Dans le cas 1, aucun signe d'erreur plancher et aucune trace d'erreur détectée après 3×10^5 trames de 1024 bits et après le point 2 dB. La configuration avec MCW = 2.5 donne un BER inférieur à celui avec MCW = 2.667. Cependant, l'inverse est vrai au point 1,75 dB. Dans le cas 2, la courbe montre un plancher d'erreur très bas à partir du point 1,5 dB. Cette fiabilité des performances pour les deux cas est due au décodeur LDPC qui

reçoit des informations plus fiables et compréhensibles de l'autre décodeur convolutif et vice versa.

Pour les TCC à 4 et 8 états (Générateur polynomial $(7,5)_8$ et $(15,17)_8$ respectivement), nous pouvons voir que le taux d'erreur sur les bits (BER) diminue rapidement à mesure que le SNR augmente, mais il y a un point où la courbe ne chute pas aussi rapidement qu'avant. Cette dégradation des performances est due à un phénomène de plancher d'erreur. La courbe de la 3D-TCC binaire montre un bon compromis entre la cascade et les régions d'erreur par rapport à la TCC à 8 états.

Pour les deux configurations de LDPC, nous pouvons constater que les codes avec MCW = 2,667 surpassent les codes qui ont MCW = 3 pour un SNR faible. En revanche, pour un SNR élevé, le code avec MCW = 3 est le meilleur.

Cela signifie que les codes LDPC peuvent également afficher des planchers d'erreur s'ils ont des poids faibles dans la matrice *H*. Pour résoudre ce problème, un code de concaténation série LDPC-BCH a été proposé. La dégradation des performances dans la région de cascade pour les codes LDPC-BCH et 3D-TCC est principalement due à la perte de débit causée par la concaténation de code. Les auteurs de [56] et [57] ont présenté un algorithme de décodage itératif entre les codes LDPC et BCH pour réduire la dégradation des performances dans la région des cascades.

En comparant un ensemble de courbes BER des deux cas de notre méthode par rapport aux autres méthodes, nous pouvons constater que la courbe BER du TCC à 4 états surpasse les deux cas de notre approche jusqu'à 1,55 dB pour le cas 1 et jusqu'à 1,2 dB pour le cas 2, Au-delà de ces points, le schéma proposé fonctionne mieux que le TCC à 4 états. Il en va de même pour le TCC à 8 états, qui surpasse les deux cas de notre approche dans la région de convergence jusqu'à 1,65 dB pour le cas 1 et jusqu'à 1,45 dB pour le cas 2. PCGC est meilleur que le code LDPC avec MCW = 3 dans l'ensemble des régions SNR. La 3D-TCC peut réaliser un bon compromis entre la zone de convergence et la zone de plancher d'erreur au prix d'une complexité croissante [12]. Le cas 1 du schéma proposé est plus performant que le code LDPC, LDPC-BCH et PCGC dans toutes les régions SNR et ne présente pas de plancher d'erreur jusqu'à

un BER $=10^{-9}$, alors que le cas 2 se rapproche fortement de 3D-TCC et donne un bon gain de codage sur les codes LDPC, LDPC-BCH et PCGC jusqu'à 2 dB.

Sauf pour le TCC conventionnel, les courbes montrent clairement que 3D-TCC et le cas 2 sont les codes dominants dans la région SNR allant de 0 à environ 1,75 dB, tandis que TCC est encore plus robuste pour un SNR inférieur ; de 0 à 1,2 dB pour les décodeurs à 4 d'états et de 0 à 1,45 dB pour les décodeurs à 8 états.

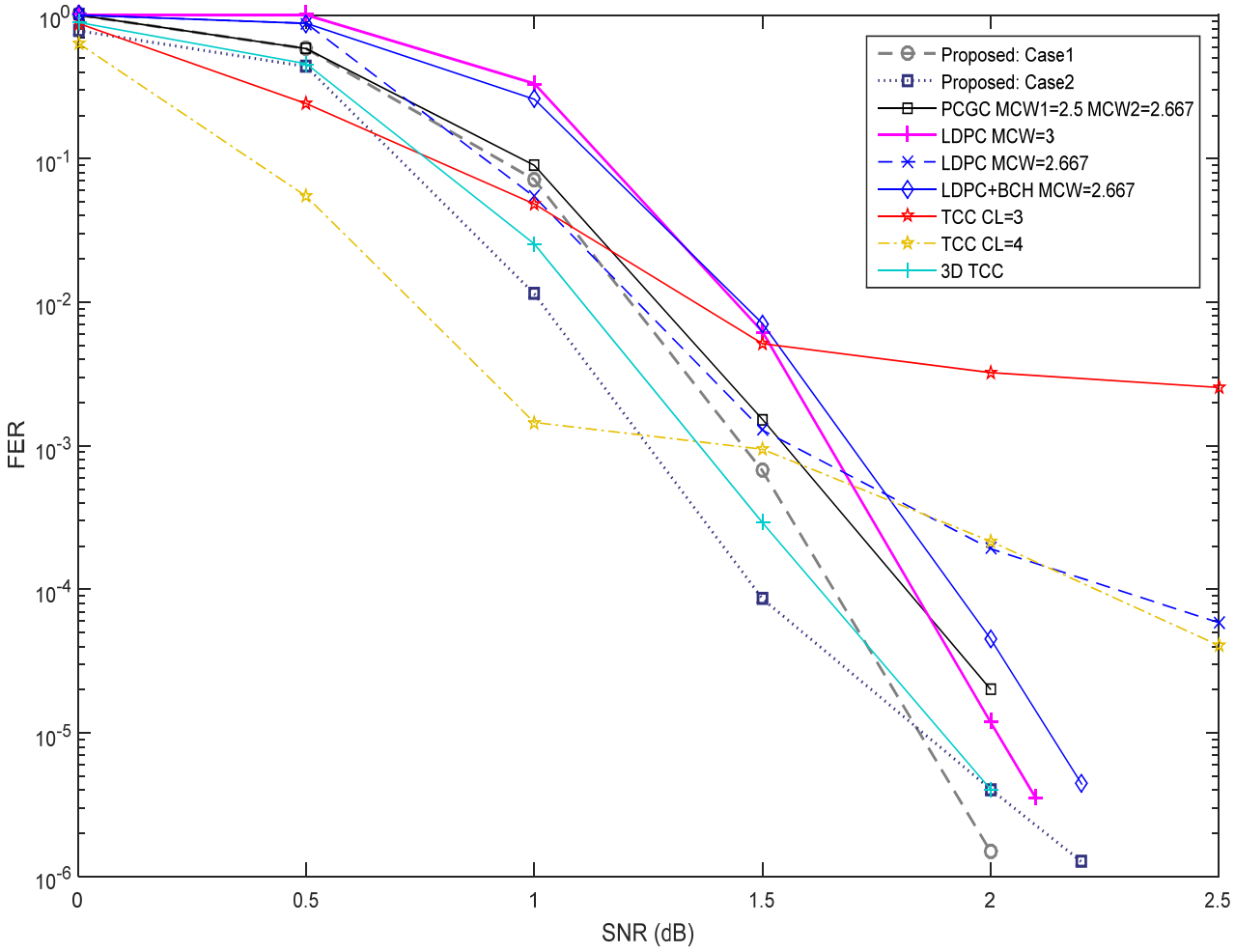


Figure 5.2 : FER des différentes approches de décodage y compris notre proposition

La Figure 5.2 permet de comparer la performance FER de certains codes par rapport à la méthode proposée. On peut voir que la FER des cas 1 et 2 de notre méthode est meilleure que celui de la TCC à 4 états et 8 états respectivement de 1,1 dB et 0,65 dB et de 1,46 dB et 1,21 dB. Comparé aux autres codes, un gain de

performance du cas 2 est observé jusqu'à 2 dB ; le point où la courbe croise la courbe de 3D-TCC.

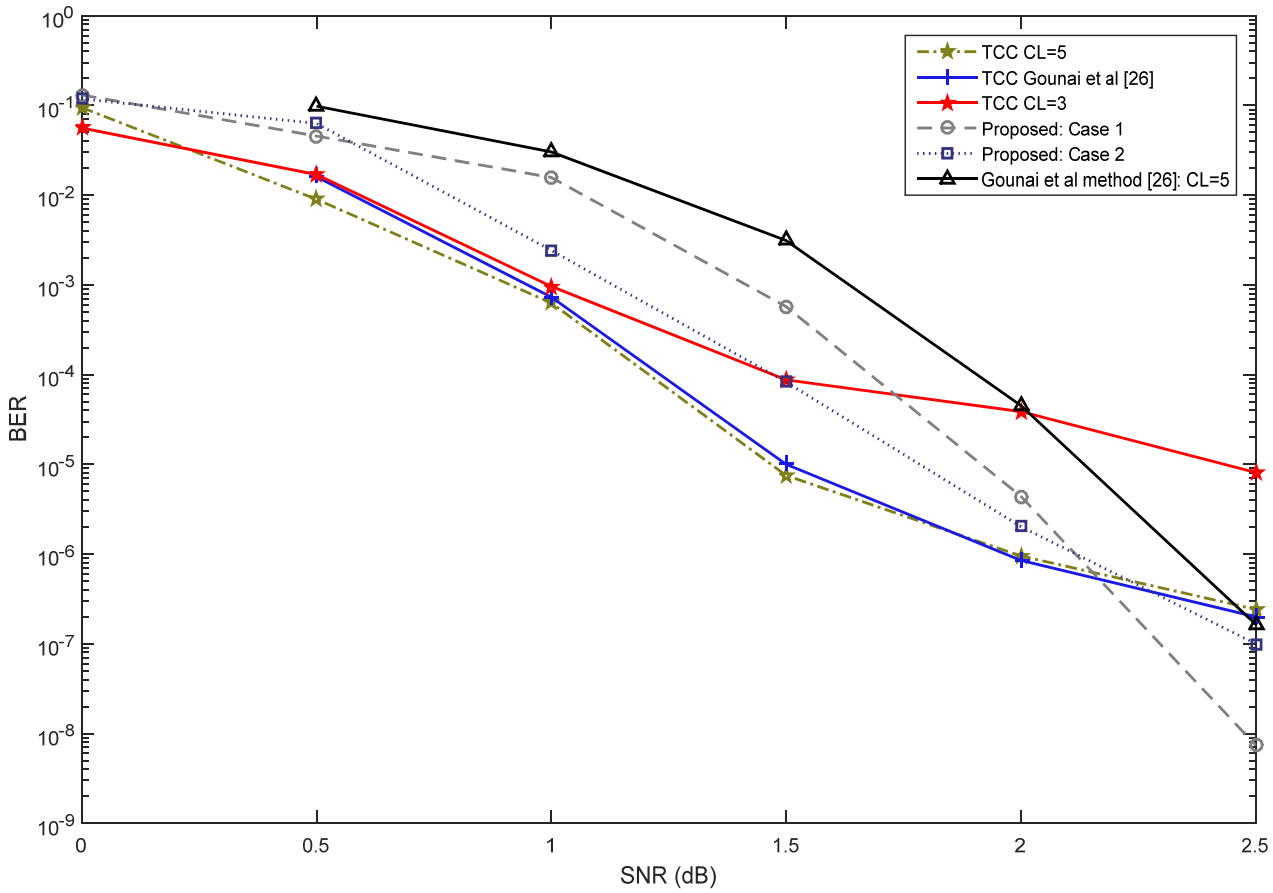


Figure 5.3. Comparaison du BER de notre approche avec celle de Gounai [28]

La Figure 5.3 montre la performance en termes de BER de notre méthode par rapport à celle de Gounai [28]. Notons que la longueur de la séquence d'information est K = 504 bits comme dans [28] pour tous les schémas de codage. Le TCC avec les mêmes paramètres que dans [28] (R = 1/3, CL = 5, polynôme générateur (31,27)) atteint des performances meilleures que TCC avec des paramètres (R = 1/3, CL = 3, générateur polynomial (7,5)). Cependant, cette amélioration est fournie au prix d'un quadruplement de la complexité du décodage. Par rapport à la méthode de Gounai, les gains de performance de la méthode proposée, pour les deux cas, sont correctement observés en utilisant seulement une longueur de contrainte CL = 3 pour le décodeur Log-MAP.

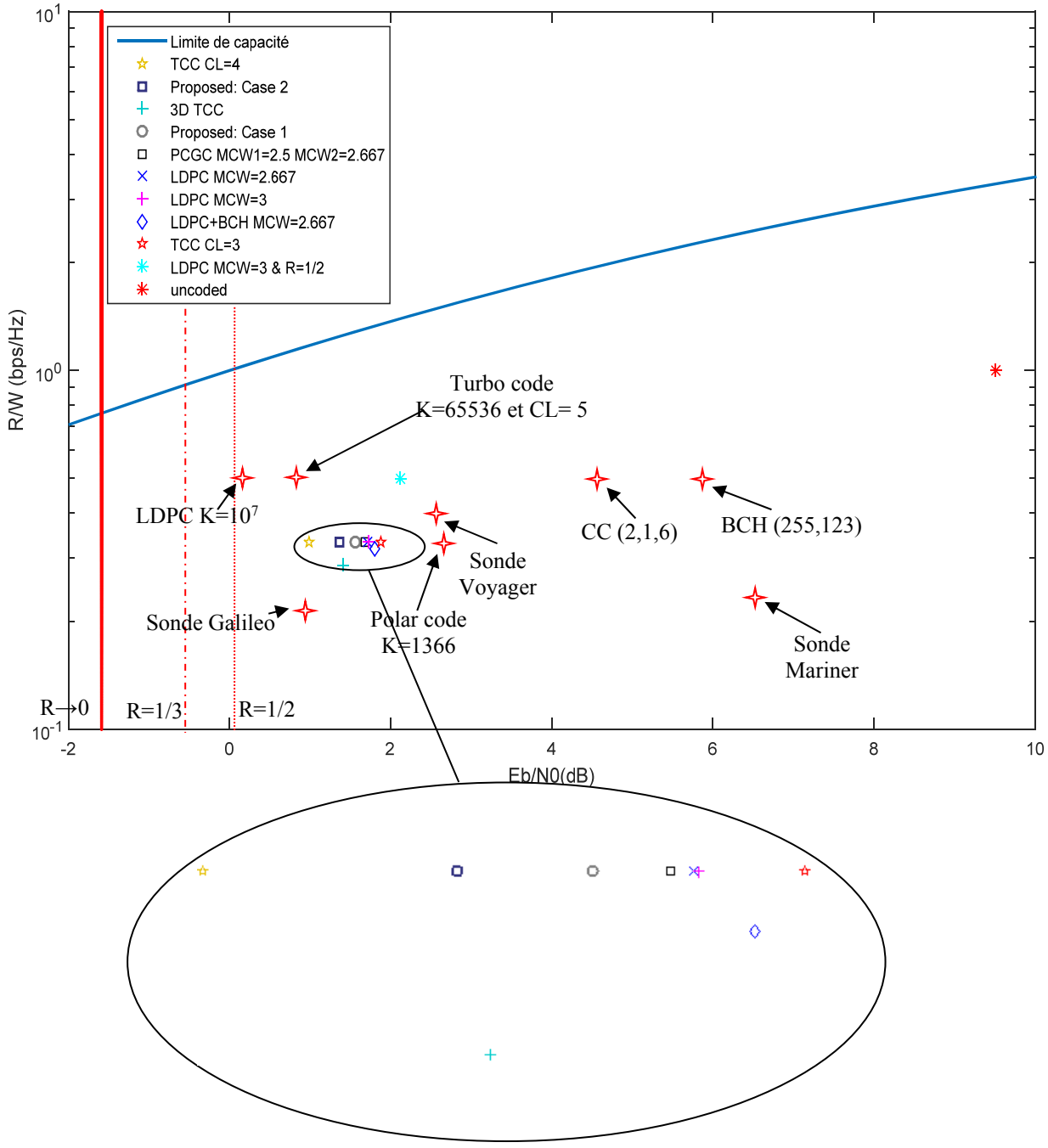


Figure 5.4 : Limite de Shannon et la performance de chaque décodeur à BER=10^{-5} ainsi que quelques performances de différents schémas de codage dans la littérature [58, 59, 60]

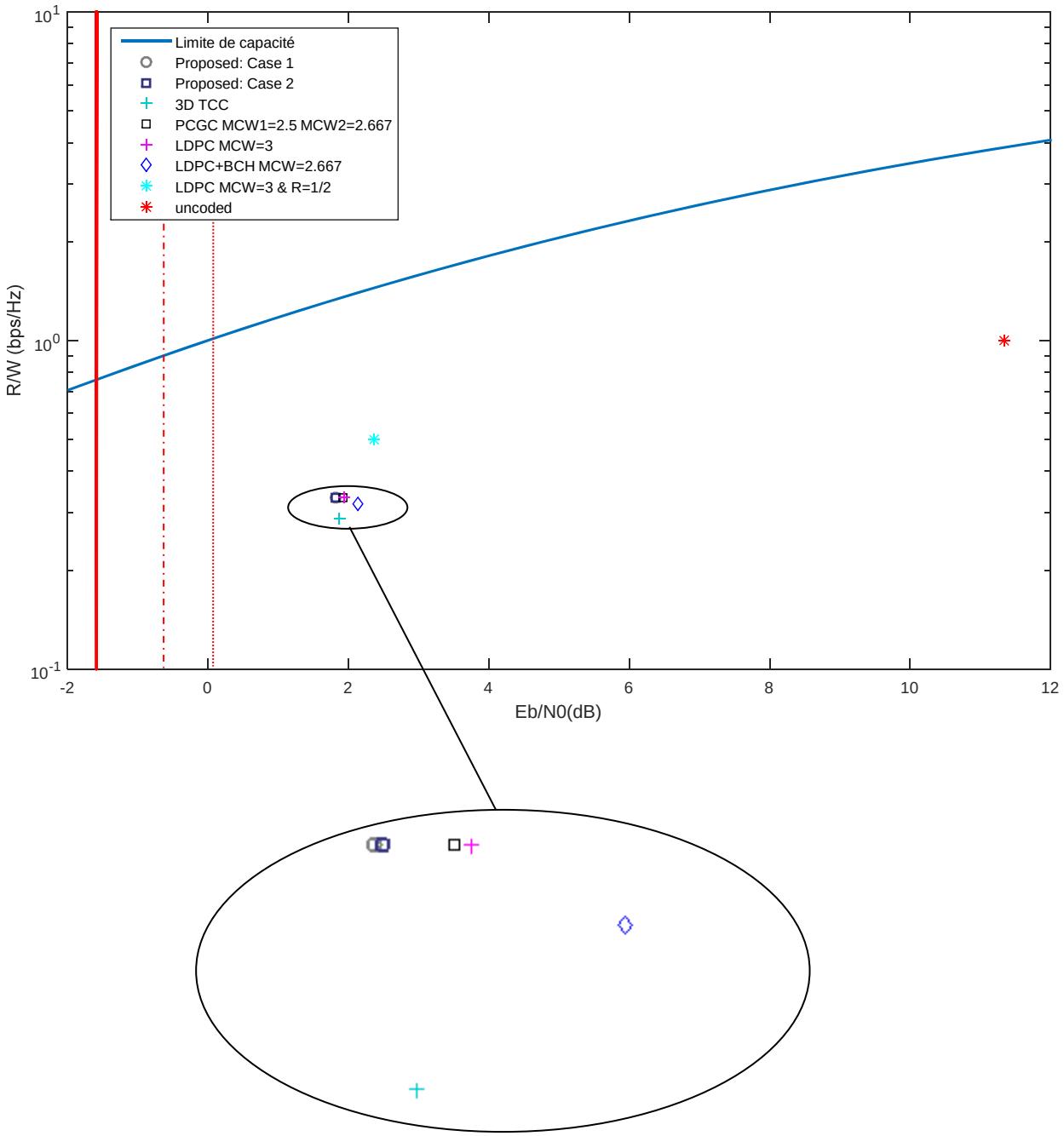


Figure 5.5 : Limite de Shannon et la performance de chaque décodeur à BER=10^{-7}

Les Figures 5.4 et 5.5 montrent la limite de capacité de Shannon dans le canal AWGN avec une probabilité d'erreur binaire de 10^{-5} et 10^{-7} respectivement. L'efficacité obtenue

de divers schémas de codage est également montrée. Parmi les schémas, les Turbo codes à 8 états atteignent la meilleure performance par rapport à la limite de capacité du canal AWGN pour une probabilité d'erreur de 10^{-5} à 0.98 dB suivi par le cas 2 de notre méthode à 1.36 dB. Pour une probabilité d'erreur de 10^{-7}, les deux cas de la méthode proposée atteignent les meilleures performances aux SNR 1.81 dB et 1.82 dB, suivi par 3D-TCC à 1.86 dB.

5.2 Analyse de complexité :

5.2.1 Complexité en termes de nombre d'opérations mathématiques :

Concernant le processus de décodage, le calcul de la complexité de l'implémentation logicielle des décodeurs Log-MAP et SPA par bit d'information et par itération basée sur les équations (3.9-22), dans le chapitre 3 section 2, et les équations (3.50-53, 3.57-61), dans le chapitre 3 section 3, est représenté dans les Tableaux 5.2-6 :

	$Additions$	$Soustractions$	$Multi\ par \pm 1$	exp	log	Max
α	S	S	0	S	S	1
β	S	S	0	S	S	1
γ	$7S$	$2S$	$2S$	$2S$	$2S$	0
L^e	0	2	0	0	0	0
L^+	$6S-2$	1	0	$2S$	2	0
$Total_1$	$15S-2$	$4S+3$	$2S$	$6S$	$4S+2$	2
$Total_2$	$K(30S-4)$	$K(8S+6)$	$4SK$	$12SK$	$K(8S+4)$	$4K$

Tableau 5.2: Calcul de complexité de Turbo décodeur

Où $S = 2^{CL-1}$ et CL est la longueur de contrainte du code convolutif.

Nous prenons en compte les multiplications par ± 1 dans l'algorithme Log-MAP (Out^{ant} et Out^{ult}), et la métrique de la branche de transition en direction aller dans le calcul de LLR. Par conséquent, $Total_1$ est le nombre total d'opérations requises par

un décodeur Log-MAP pour décoder un bit d'information par itération. Etant donné que deux décodeurs Log-MAP sont requis dans le turbo décodeur pour décoder K bits par itération, nous aurons donc :

$Total_2 = 2 \times K \times Total_1$.

Pour le cas de LDPC, nous prenons l'algorithme SPA tel qu'il est décrit dans [19].

	$Additions$	$Subtractions$	$Multiplications$	$divisions$	exp
f_j^1	1	0	1	2	1
f_j^0	0	1	0	0	0
δQ_{ij}	0	1	0	0	0
DR_{ij}	0	0	$d_j - 1$	0	0
R_{ij}^0	1	0	1	0	0
R_{ij}^1	0	1	1	0	0
Q_{ij}^0	1	0	$d_i - 1$	1	0
Q_{ij}^1	0	0	$d_i - 1$	1	0
Q_j^0	1	0	d_i	1	0
Q_j^1	0	0	d_i	1	0
L_j^+	0	1	0	0	0
$Total$	4	4	$d_j + 4d_i$	6	1

Tableau 5.3 : Nombre total d'opérations pour calculer chaque terme dans LDPC

Où d_i est le nombre de '1' par colonne, $d_j = 2d_i$ est le nombre de '1' par ligne de la matrice de contrôle de parité $H(M \times N)$.

Pour calculer la complexité pour tous les bits, le total des opérations dans le graphique de Tanner (l'ensemble des nœuds de contrôle et des nœuds de variable) dont nous avons besoin est :

	Nombre total des opérations nécessaires dans le graphique de Tanner pour tous les bits (LDPC)
f_j^1	N
f_j^0	N
δQ_{ij}	N
DR_{ij}	Md_j
Pdq	$-$
Pdq'	$-$
Pr_j^0	$-$
Pr_j^0	$-$
R_{ij}^0	Md_j
R_{ij}^1	Md_j
Q_{ij}^0	Nd_i
Q_{ij}^1	Nd_i
Q_j^0	N
Q_j^1	N
L_j^+	N

Tableau 5.4: Nombre total d'opérations pour LDPC

Alors, en se Basant sur des opérations mathématiques, la complexité de calcul dans la plate-forme logicielle des différents décodeurs (log-MAP, SPA [19] et BCH) est détaillée dans les Tableaux 5.5 et 5.6. Pour l'algorithme Log-MAP, nous prenons en compte les multiplications par ± 1:

$Methode$	$Add/Sous$	$Multi/div$	exp
$LDPC$	$I_2 \times [N(d_i + 5) + 2Md_j]$	$I_2 \times [M({d_j}^2 - d_j) + 2N(d_i^2 + d_i) + 5N]$	N
$LDPC\text{-}BCH$	$I_2 \times [N(d_i + 5) + 2Md_j] + (2n + 29)t + 1$	$I_2 \times [M({d_j}^2 - d_j) + 2N(d_i^2 + d_i) + 5N] + (4n + 24)t + 2n$	$N + 3t + 1$
$PCGC$	$I_1 \times \begin{pmatrix} I_2 \times [N(d_{i1} + 5) + 2Md_{j1}] + \\ I_3 \times [N(d_{i2} + 5) + 2Md_{j2}] + \\ 2(N - M) \end{pmatrix}$	$I_1 \times \begin{pmatrix} I_2 \times [M({d_{j1}}^2 - d_{j1}) + 2N(d_{i1}^2 + d_{i1}) + 5N] + \\ I_3 \times [M({d_{j2}}^2 - d_{j2}) + 2N(d_{i2}^2 + d_{i2}) + 5N] + \\ 12(N - M) \end{pmatrix}$	$I_1 \times 2N$

Tableau 5.5: Comparaison de la complexité de LDPC, LDPC-BCH et PCGC

$Methode$	$Add/Sous$	$Multi/div$	exp	log	max
TCC	$I_1 \times K(38S + 2)$	$I_1 \times K \times 4S$	$I_1 \times K \times 12S$	$I_1 \times K(8S + 4)$	$I_1 \times 4K$
$3DTCC$	$KI_1 \times [(38S_1 + 2) + 2\lambda(19S_2 + 1)]$	$(4S_1 + 4\lambda S_2) \times KI_1$	$KI_1(12S_1 + 12\lambda S_2)$	$KI_1 \times [(8S_1 + 4) + 2\lambda(4S_2 + 2)]$	$(4 + 4\lambda)KI_1$
$Proposée$	$I_2 \times [N(d_i + 5) + 2Md_j] + I_1 \times [K(19S + 8)]$	$I_2 \times [M({d_j}^2 - d_j) + 2N(d_i^2 + d_i) + 5N] + I_1 \times (2S + 13)K$	$I_2 \times N + I_1 \times [K(6S + 3)]$	$I_1 \times K(4S + 4)$	$I_1 \times 2K$

Tableau 5.6: Comparaison de la complexité du TCC, TCC 3D et la méthode proposée

Où $S = 2^{CL-1}$ avec CLest la longueur de contrainte du code convolutif, d_i est le nombre de 1's par colonne, $d_j = 2d_i$ est le nombre de 1's par ligne dans la matrice de contrôle de parité $H(M \times N)$, K est la longueur du mot de code, λ est la fraction des bits de parité à coder à partir des deux encodeurs en parallèle de 3D-TCC, S_1est le nombre d'états de ces encodeurs, S_2 est le nombre d'états du post-encodeur, n et t sont les paramètres du code BCH, I_1sont les super-itérations pour la structure turbo et I_2 et I_3 sont les itérations pour le premier et le deuxième décodeurs LDPC.

Le Tableau 5.7 résume les différents paramètres pour calculer la complexité des différentes méthodes.

Methode	*Paramètres*
Proposée $Cas\ 1$	$R = 1/3 \quad K = 1024$ $CL = 3 \quad S = 4$ $M = K \quad N = 2K \quad d_i = 2.667 \quad d_j = 5$
Proposée $Cas\ 2$	$R = 1/3 \quad K = 1024$ $CL = 3 \quad S = 4$ $M = K \quad N = 2K \quad d_i = 2.5 \quad d_j = 5$
TCC	$R = 1/3 \quad K = 1024$ $CL = 3\ or\ 4 \quad S = 4\ or\ S = 8$
$3DTCC$	$R = 1/3.5 \quad K = 1024$ $CL_1 = 4 \quad CL_2 = 3$ $S_1 = 8 \quad S_2 = 4 \quad \lambda = 1/4$
$LDPC$	$R = 1/3 \quad K = 1024$ $M = 2K \quad N = 3K \quad d_i = 3 \quad d_j = 6$
$LDPC\text{-}BCH$	$R = 1/3.1541 \quad n = 1023 \quad k = 973 \quad t = 5$ $M = 2n \quad N = 3n$ $d_i = 2.667 \quad d_j = 5$
$PCGC$	$R = 1/3 \quad K = 1024$ $M = K \quad N = 2K$ $d_{i1} = 2.667 \quad d_{i2} = 2.5 \quad d_j = 5$

Tableau 5.7: Paramètres pour l'application numérique

5.2.2 Complexité en termes de nombre d'itérations :

Le nombre maximal d'itérations de toutes les méthodes de décodage est résumé dans le Tableau 5.8.

Iterationlevel	TCC	$3D\text{-}TCC$	$LDPC$	$PCGC$	*Proposed*
Super-iterations	10	10	−	10	10
$Self\text{-}iterations$	−	−	100	2×50	50

Tableau 5.8: Nombre maximal d'itérations pour toutes les méthodes de décodage

Le nombre d'itérations pour TCC et 3D-TCC est fixé à 10, pour LDPC le nombre maximal des itérations est 100, pour le code concaténé en parallèle (PCGC) nous avons deux décodeurs LDPC avec un maximum de 50 itérations pour chacun et un nombre maximal d'itérations égal à 10 pour la structure turbo. Également pour la méthode proposée, nous avons un nombre maximal d'itérations de 50 et 10 pour le LDPC et la structure turbo respectivement.

Pour réduire au minimum le temps de retard de décodage, il est important d'utiliser un critère d'arrêt efficace pour arrêter le processus de décodage. Ainsi, dans la méthode proposée, nous utilisons LDPC dans DEC1 qui permet de tester si la solution de $H.\widehat{x^T} = 0$ est atteinte après chaque itération et, si c'est le cas, d'arrêter le décodage. Le nombre moyen correspondant de niveaux d'itérations requis par chaque décodeur pour K = 1024 bits est représenté sur la Figure 5.6. Nous constatons que plus le SNR est élevé plus le nombre total d'itérations requises est moindre.

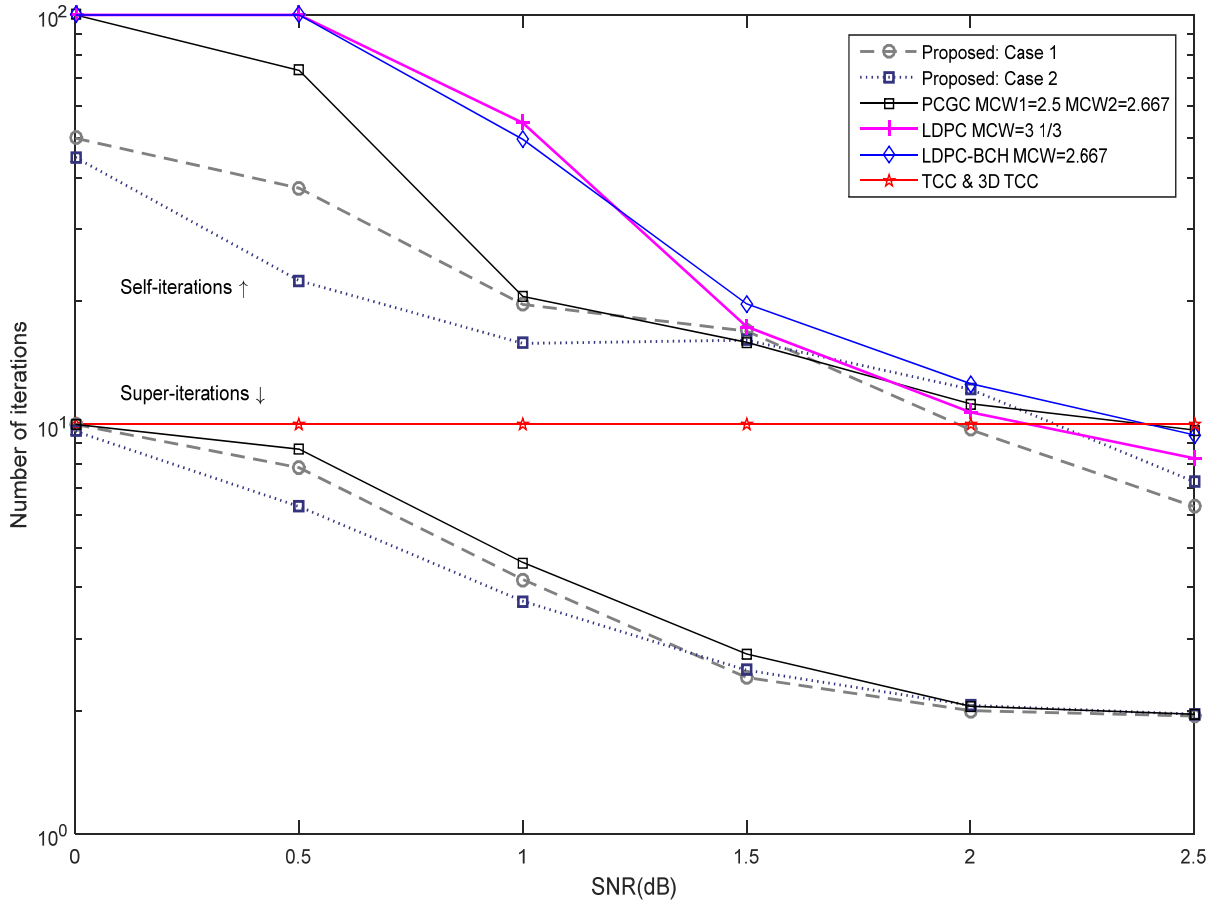


Figure 5.6: Nombre moyen d'itérations pour chaque décodeur pour $K = 1024$

Méthode	*Niveau d'itération*	1 dB	1,5 dB	2 dB
Proposée Cas 1	Super-itération I_1	4,18	2,41	2,00
	itération I_2	19,63	16,90	9,71
Proposée Cas 2	Super-itération I_1	3,70	2,51	2,06
	itération I_2	15,77	16,06	12,19
LDPC	itération I_2	54,50	17,33	10,71
LDPC-BCH	itération I_2	41,68	16,77	11,65
PCGC	Super-itération I_1	4,60	2,75	2,05
	itération I_2	11,73	10,71	6,31
	itération I_3	8,81	5,16	4,92

Tableau 5.9: Nombre moyen d'itérations nécessaires pour chaque décodeur

Pour chaque méthode de correction d'erreurs ayant un critère d'arrêt, le Tableau 5.9 résume le nombre d'itérations moyen nécessaire à chaque niveau d'itération en fonction de la puissance de bruit.

Méthode	*Ccomplexité total par itération*	*Complexité globale* à $SNR = 1{,}5$
3DTCC	$570{,}5 \times K$	$5705 \times K$
TCC $CL = 4$	$506 \times K$	$5060 \times K$
TCC $CL = 3$	$258 \times K$	$2580 \times K$
Cas 1 *proposé*	$250{,}45 \times K$	$1989{,}92 \times K$
Cas 2 *proposé*	$246 \times K$	$1851{,}98 \times K$
PCGC	$202{,}45 \times K$	$4108{,}49 \times K$
LDPC	$198 \times K$	$3382{,}35 \times K$
LDPC-BCH	$191{,}92 \times K$	$2662{,}77 \times K$

Tableau 5.10: Comparaison de la complexité de différents décodeurs

Le Tableau 5.10 fournit une application numérique pour décrire la complexité de chaque méthode de correction d'erreurs par bit/par une seule itération, ainsi par un nombre moyen des itérations correspondant pour le SNR=1.5 dB.

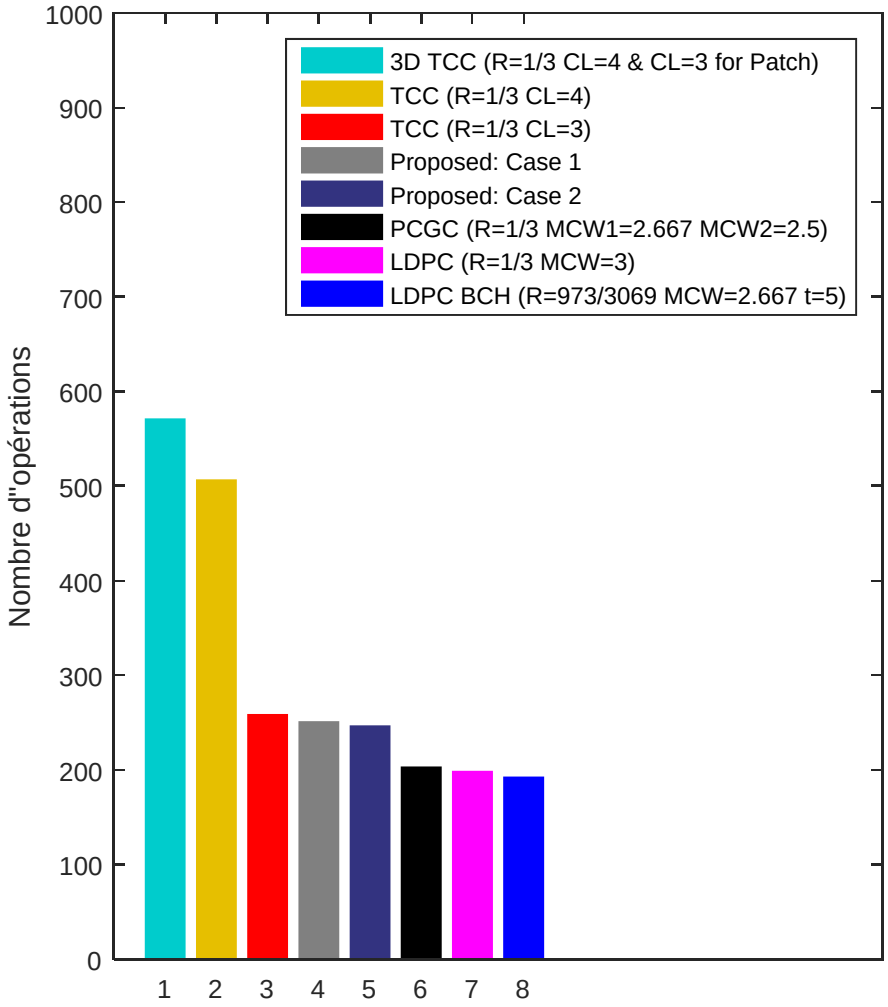


Figure 5.7: Complexité de calcul par bit, par itération

La Figure 5.7 montre que la complexité, par itération, dans LDPC est plus faible par rapport aux calculs globaux de TCC à 8 états et de 3D-TCC, sans inclure les opérations d'entrelacement/désentrelacement. Dans notre approche, la complexité est légèrement supérieure à celle de LDPC mais reste acceptable du fait qu'elle n'a pas besoin de processus d'entrelacement.

Notons enfin que la complexité de calcul dépend du nombre d'itérations. En fait, l'augmentation du nombre d'itérations augmente les calculs et provoque un long retard de décodage. D'autre part, à l'exception de TCC et 3D-TCC, toutes les méthodes de décodage étudiées dans ce chapitre ont un caractère itératif qui est modérément sensible au SNR. TCC et 3D-TCC permettent un décodage continu jusqu'à un nombre fixe d'itérations sans possibilité de détecter les erreurs.

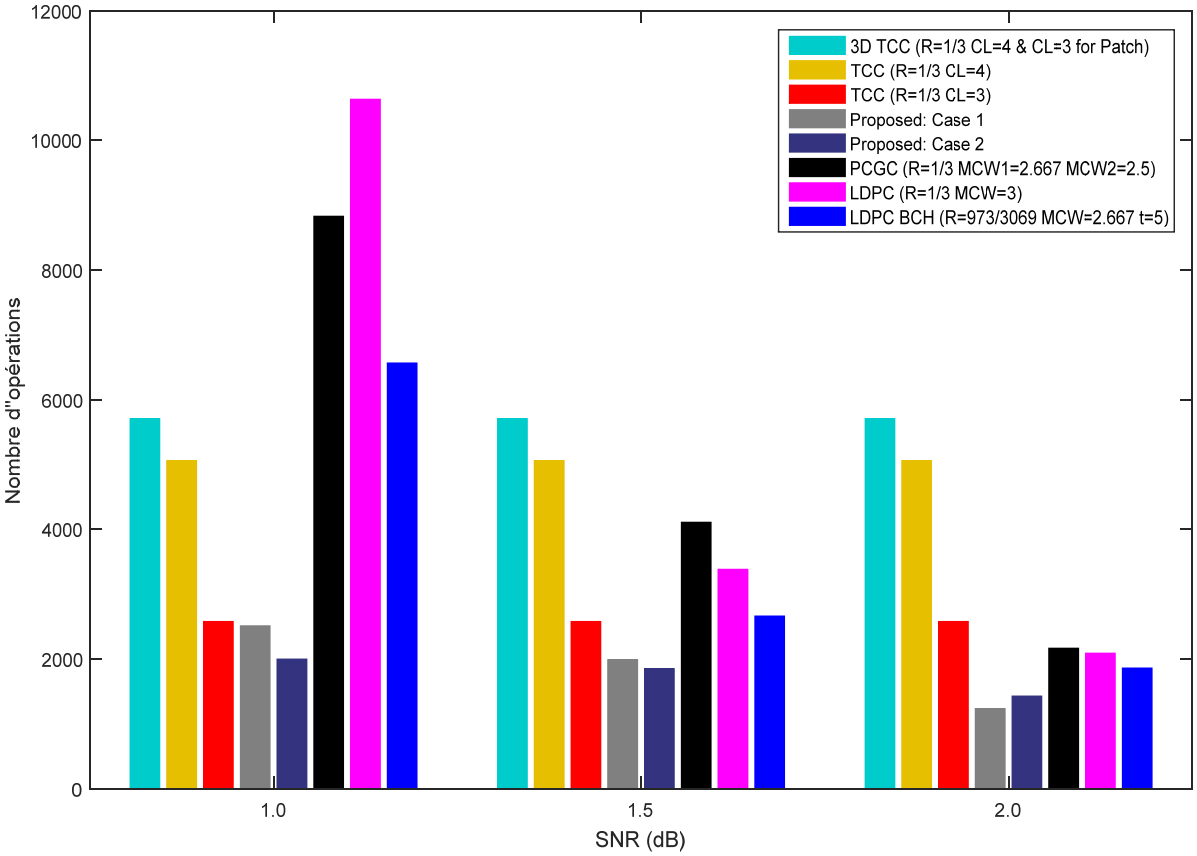


Figure 5.8: Complexité globale du calcul en fonction du nombre d'itérations

Comme nous pouvons le voir dans la Figure 5.8, les deux cas de notre approche atteignent la plus faible complexité de calcul parmi toutes les méthodes de codage dans toute la gamme de SNR. En outre, le cas 2 peut obtenir les mêmes performances de 3D-TCC avec une faible consommation d'énergie et une complexité de calcul réduite.

5.3 Conclusion

Dans ce chapitre nous avons discuté les résultats de notre méthode par rapport à divers codes correcteurs d'erreurs, qui sont les codes conventionnels (LDPC et TCC) ainsi que certaines méthodes utilisant différents types de concaténation (Concaténation en série, en parallèle et hybride). Nous avons comparé donc les performances de ces codes dans des conditions identiques en analysant pareillement leurs complexités algorithmiques.

Pour résumer, l'analyse et les résultats montrent que notre approche est meilleure en termes de performance (Cascade/Plancher d'erreur) et de performance/complexité et fournit également des gains de codage significatifs par rapport à LDPC, LDPC-BCH et PCGC.

Conclusion

Conclusion générale :

Dans cette thèse, nous avons discuté un schéma de décodage plus efficace avec moins de complexité pour le codage Turbo. Ceci est basé sur la concaténation parallèle des codes LDPC et convolutionnels. La complexité de calcul est analysée et la décision de fiabilité de la nouvelle approche est simulée en comparaison avec LDPC unique, LDPC-BCH, PCGC, TCC et 3D-TCC.

Malgré les très bonnes performances de TCC dans les SNR’s faibles, il ne peut pas être utilisé pour la détection d'erreurs. Pour cette déficience de capacité de détection d'erreurs et le phénomène de plancher d'erreur, le TCC n'est pas approprié pour plusieurs applications de communication lorsqu'elles nécessitent un gain de codage élevé, un plancher d'erreur faible et une consommation d'énergie réduite.

La 3D-TCC présente un bon compromis entre la zone de convergence et la zone du plancher d’erreur mais, malheureusement, au prix d'une complexité croissante. Les codes LDPC peuvent battre TCC et 3D-TCC en termes de capacité de détection d'erreurs, de performance, d’absence de plancher d'erreur et de complexité de décodage réduite. Cependant, ils ne fonctionnent pas aussi bien que le TCC et 3D-TCC en SNR faible. De plus, PCGC a une bonne performance par rapport au LDPC, mais avec un retard de décodage excessif dû à ses trois niveaux d'itérations.

Les avantages déduits des codes TCC et LDPC dans cette analyse sont : plancher d'erreur réduit, bonne performance, moins de complexité, capacité de détection d'erreurs et délai de décodage réduit. Notre méthode vise à profiter de tous ces avantages pour fournir un code de correction d'erreur qui convient à toute la gamme des valeurs de SNR. L'originalité de notre contribution est que l'information extrinsèque échangée entre les deux décodeurs est calculée et utilisée plus efficacement que dans [28], ce qui permet à chaque décodeur d'être plus intelligible pour l'autre.

À partir des résultats de notre analyse, nous arrivons à la conclusion que le système proposé peut réaliser les meilleurs compromis entre la performance dans les régions de convergence et plancher d'erreur, la complexité et la latence par rapport à toutes les approches de décodage étudiées dans cette thèse. Ce qui en fait un schéma de codage prometteur pour les communications dans l'espace lointain et les communications mobiles des prochaines générations.

Comme perspectives, des recherches similaires à [61] et [62] peuvent être utilisées pour améliorer les performances et / ou réduire la complexité des algorithmes SPA et Log-MAP. En outre, nous pourrons considérer les travaux tels que [63] et [64] pour étendre notre approche pour le cas non-binaire.

Références

Références:

[1] C. E. Shannon. “A Mathematical Theory of Communication”, Bell System Tech. J., vol. 27, Introduction, Part 1&2 pp. 379-423, Part 3&4 pp. 623-656, Juillet et Octobre 1948.

[2] Thibaud Tonnellier. “Contribution to the improvement of the decoding performance of turbo codes : algorithms and architecture”, Thèse de doctorat, Université de Bordeaux, 2017.

[3] Madiagne Diouf. “Conception Avancée des Codes LDPC Binaires pour des Applications Pratiques”, Thèse de doctorat, Université de Cergy-Pontoise,2015.

[4] T-H. Lin, WJ. Kaiser, GJ. Pottie. “Integrated low-power communication system design for wireless sensor networks”, IEEE Commun. Mag. 42(12), 142–150 (2004).

[5] Y Zhu, W Wu, J Pan, Y Tang. “An energy-efficient data gathering algorithm to prolong lifetime of wireless sensor networks”, Comput. Commun. 33(5), 639–647 (2010).

[6] SL. Howard, C. Schlegel, K. Iniewski. “Error control coding in low-power wireless sensor networks: when is ECC energy-efficient?”, EURASIP. J. Wirel. Commun. Netw. 2, 1–14 (2006).

[7] F. Kienle, N. Wehn, H. Meyr. “On complexity, energy- and implementation-efficiency of channel decoders”, IEEE Trans. Commun. 59(12), 3301–3310 (2011).

[8] C. Berrou, A. Glavieux, and P. Thitimajshima. “Near Shannon limit error correcting coding and decoding : turbo-codes”, In Proc. of ICC ’93, Genève, vol. 2, pp. 1064-1070, Mai 1993.

[9] C. Berrou and A. Glavieux. “Near Optimum Error Correcting Coding and Decoding: Turbo Codes”, IEEE Transactions on Communications, vol. COM-44, No 10, pp. 1261-1271, October 1996.

[10] L. C. Perez, J. Seghers, and D. J. Costello, Jr. “A Distance Spectrum Interpretation of Turbo Codes”, IEEE Trans. Inform. Theory, vol 42, No 6, pp 1698-1709, Nov 1996.

[11] C. Berrou, A. Graelli Amat, Y. Ould-Cheikh-Mouhamedou, and Y. Saouter. “Improving the distance properties of turbo codes using a third component code : 3D turbo codes”, IEEE Trans. Commun, vol. 57, No. 9, pp. 2505–2509, Sep. 2009.

[12] E. Rosnes, A. Graelli Amat. “Performance Analysis of 3-D Turbo Codes”, IEEE Transactions on Information Theory, vol. 57,No 6, pp. 3707-3720, 2011.

[13] Y. Ould-Cheikh-Mouhamedou. “A Simple and Efficient Method for Lowering the Error Floors of Turbo Codes that Use Structured Interleavers”, IEEE Communications Letter, Vol. 16, No 3, pp 392 – 395, March 2012.

[14] Th. Tonnellier, C. Leroux, B. Le Gal. “Lowering the Error Floor of Turbo Codes with CRC Verification”, IEEE Wireless Communications Letters, Vol 5, No 4, pp 404 – 407, Aug 2016.

[15] Defeng Ren , Jianhua Ge, Jing Li. “Modified Collision-Free Interleavers for High-Speed Turbo Decoding”, Wireless Personal Communications, Vol 68, No 3, pp 939-948, February 2013.

[16] Rahul Shrestha, Roy Paily. “Design and Implementation of a Linear Feedback Shift Register Interleaver for Turbo Decoding”, Chapter, Progress in VLSI Design and Test, Vol 7373, pp 30-39, 2012.

[17] Hyeji Kim, Youngjoo Lee and Ji-Hoon Kim. “Low-complexity CRC-aided early stopping unit for parallel turbo decoder”, Electronics Letters, Vol. 51 No. 21 pp. 1660–1662, 8th October 2015.

[18] Zheng Ma, Pingzhi Fan, Wai Ho Mow and Qingchun Chen. “A joint early detection-early stopping scheme for short-frame turbo decoding”, AEU - International Journal of Electronics and Communications, Vol.65, No 1, pp. 37-43, January 2011.

[19] D. J. C. Mackay and R. M. Neal. “Near Shannon Limit Performance of Low Density Parity Check Codes”, Electron. Lett, vol 33, No 6, pp. 457-458, Mars 1997.

[20] D. J. C. MacKay and R. M. Neal. “Good codes based on very sparse matrices”, in Proc. Cryptography Coding. 5th IMA Conf., C. Boyd, Ed., Berlin, Germany, 1995, pp. 100-111, Springer.

[21] T. J. Richardson, and R. L. Urbanke, "Efficient Encoding of Low-Density Parity-Check Codes", IEEE Trans. Information Theory, vol. 47, no. 2, pp. 638-656, Feb. 2001.

[22] Myung, S., Yang, K., & Kim, J. Quasi-Cyclic LDPC Codes for Fast Encoding. IEEE Transactions on Information Theory, vol 51, no 8, pp. 2894–2901, 2005.

[23] European Telecommunications Standards Institute. “Digital video broadcasting (DVB) second generation framing structure, channel coding and modulation systems for broadcasting, interactive services, news gathering and other broadband satellite applications”. DRAFT EN 302 307 DVBS2-74r15, 2003.

[24] Carlo Condo. “Concatenated Turbo/LDPC codes for deep space communications: performance and implementation”**,** SPACOMM 2013, The Fifth International Conference on Advances in Satellite and Space Communications, 21 – 26 April 2013, Venice, Italy.

[25] Damian A. Morero and Mario R. Hueda. “Novel serial code concatenation strategies for error floor mitigation of low-density parity-check and turbo product codes”, Canadian Journal of Electrical and Computer Engineering, Vol.36, No 2, pp 52 – 59, 2013.

[26] H. Behairy and S. C. Chang. “Parallel Concatenated Gallager codes”, Electronics Letters, vol 36, No 24, pp 2025-2026, nov 2000.

[27] H. Behairy and M. Benaissa. “Multiple Parallel Concatenated Gallager Codes: Design and Decoding Techniques”, IETE Journal of Research, Vol 59, No 6, Nov-Dec 2013.

[28] S. Gounai, T. Ohtsuki and T. Kaneko. “Performance of Concatenated Code with LDPC Code and RSC Code”, 2006 IEEE International Conference on Communications, pp. 1195-1199, Istanbul, 2006.

[29] E. Arikan, “Channel polarization: A method for constructing capacity-achieving codes”, EEE International Symposium on Information Theory, 2008.

[30] B. Oudjani, H. Tebbikh, N. Doghmane. “Modification of extrinsic information for parallel concatenated Gallager/Convolutional code to improve

performance/complexity trade-offs". AEU - International Journal of Electronics and Communications. Volume 83, January 2018, Pages 484-491.

[31] Research and Development for Space Data System Standards, "Low density parity check codes for use in near-earth and deep space applictions", experimental specification, ccsds 131.1-O-2, Orange Book, Issue 2, September 2007.

[32] John G. Proakis, Masoud Salehi, "Digital Communications", Fifth Edition, McGraw-Hill, New York, 2008.

[33] Md. Taslim Arefin, Fazlay Rabby Reza, "AWGN & Rayleigh Fading Channel: A Throughput Analysis for Reliable Data Transmission". VDM Verlag Dr. Müller, 5 August 2011.

[34] M.R.Soleymani, Yingzi Gao, U. Vilaipornsawai, "Turbo coding for satellite and wireless communications", Kluwer Academic Publishers, 2002.

[35] Bernd Friedrichs, "Error-control coding. Berlin" Springer-Verlag, Erwartet demnächst, 2010.

[36] M. Valenti and R. Seshadri. Topic: "Turbo and LDPC Codes: Implementation, Simulation, and Standardization." West Virginia University, Morgantown, Juin 2006.

[37] Robert J. McEliece; Laif Swanson. "Reed–Solomon Codes and the Exploration of the Solar System". JPL, 20 August 1993.

[38] R. Ludwig, J. Taylor, "Voyager Telecommunications Manual", JPL DESCANSO (Design and Performance Summary Series), March 2002.

[39] Huffman, William Cary; Pless, Vera S. "Fundamentals of Error-Correcting Codes". Cambridge University Press, 2003.

[40] Thierry Lestable, Moshe Ran "Error Control Coding for B3G/4G Wireless Systems: Paving the Way to IMT-Advenced Standards", John Wiley & Sons, Ltd, Mar 2011.

[41] J. S. Ramsey "Realization of Optimum Interleavers", IEEE Trans. on Inform. Th., Vol. IT-16, No. 3, pp. 338-345, May 1970.

[42] Dan Zhang ; Pingyi Fan ; Zhigang Cao. “Interference cancellation for OFDM systems in presence of overlapped narrow band transmission system”, IEEE Transactions on Consumer Electronics, Vol 50 , No 1, pp 108 – 114, 2004.

[43] O.F. Acikel ; W.E. Ryan. “Punctured turbo-codes for BPSK/QPSK channels”, IEEE Transactions on Communications, Vol 47 , No 9, pp 1315 – 1323, 1999.

[44] Giovanni E. Corazza , “Digital Satellite Communications”, Springer, Edition 1, 2007.

[45] R. M. Tanner, “A recursive approach to low complexity codes”, IEEE Trans. Inform. Theory, vol. IT-27, no. 9, pp. 533-547, Sep. 1981.

[46] http://www.kozintsev.net/soft/ldpc_distr.zip

[47] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, “Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate”, IEEE Transactions on Information Theory, vol. IT-20, pp. 284-287, 1974.

[48] Wu, Yufei. “Turbo Code Simulator”, MPRG lab, Virginia Tech, http://www.ee.vt.edu/»yufei , Nov 1998.

[49] P. Robertson, P. Hoeher and E. Villebrun, “Optimal and Sub-Optimalmaximum a Posteriori Algorithms Suitable for Turbo Decoding”, European Transactions on Tele- communications, Vol. 8, No. 2, 1997, pp. 119-125.

[50] A. J. Viterbi, “An Intuitive Justification and a Simplified Implementation of the MAP Decoder for Convolutional Codes”, IEEE Journal on Selected Areas in Communications, Vol. 16, No. 2, 1998, pp. 260-264.

[51] Mandal; Arijit Saha; Nilotpal Manna. “Information Theory, Coding and Cryptography”. Pearson India, 2013.

[52] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman. “Improved low-density parity check codes using irregular graphs”, IEEE Transactions on Information Theory, Vol. 47, pp. 585-598, Feb. 2001.

[53] Claude Berrou, Alexandre Graell i Amat, Youssouf Ould-Cheikh-Mouhamedou, and Yannick Saouter “Improving the Distance Properties of Turbo Codes Using a Third Component Code: 3D Turbo Codes”, IEEE Transactions on Communications, VOL. 57, NO. 9, September 2009.

[54] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Serial concatenation of interleaved codes: performance analysis, design and iterative decoding," IEEE Trans. Inform. Theory, vol. 44, pp. 909-926, May 2005.

[55] Kbaier Ben Ismail D, Douillard C, Kerouédan S. “Reducing the convergence loss of 3-dimensional turbo codes”, 6th Int. Symp. Turbo Codes and Related Topics, Brest 2010, 146-150.

[56] Pin-Han Chen, Jian-Jia Weng and Chung-Hsuan Wang. “BCH Code Selection and Iterative Decoding for BCH and LDPC Concatenated Coding System”, IEEE Communications Letters, Vol 17, No5, pp 980-983, May 2013.

[57] Shin-Lin Shieh. “Concatenated BCH and LDPC Coding Scheme With Iterative Decoding Algorithm for Flash Memory”, IEEE Communications Letters, Vol 19, No 3, pp 327-330, March 2015.

[58] Christian B. Schlegel, Lance C. Perez. “Trellis and Turbo Coding”, Wiley-IEEE Press, 2004.

[59] Sae-Young Chung G. David Forney Jr, Tom Richardson, Rüdiger L. Urbanke. “On the Design of Low-density Parity-check Codes within 0.0045 dB of the Shannon Limit”, IEEE Communications Letters, Volume: 5, Issue 2, pp58-60, Feb 2001.

[60] G. D. Forney, Jr. and G. Ungerboeck , “Modulation and coding for linear Gaussian channels”, IEEE Transactions on Information Theory, vol.44, no.6, pp. 2384-2415, Oct 1998.

[61] Abdelilah Kadi, Said Najahb, Mostafa Mrabti. “An exponential factor appearance probability belief propagation algorithm for regular and irregular LDPC codes”, AEU - International Journal of Electronics and Communications, Vol 69, No 6, pp 933-936, June 2015.

[62] M Martina, Stylianos Papaharalabos, P. Takis Mathiopoulos, Guido Masera. “Simplified Log-MAP Algorithm for Very Low-Complexity Turbo Decoder Hardware Architectures”, IEEE Transactions on Instrumentation and Measurement, Vol 63, No 3, pp 531-537, March 2014.

[63] Yuanyi Zhao, Martin Johnston, Charalampos Tsimenidis, Li Chen. “Non-binary turbo-coded physical-layer network coding on impulsive noise channels”, Electronics Letters, Vol 52, No 24, pp 1984 – 1986, Nov 2016.

[64] B. Shams, D. Declercq, V. Heinrich. “Diversity of non-binary cluster-LDPC codes using the EMS algorithm”, International Journal of Electronics and Communications (AEÜ), Vol 69, No 2, pp 492-499, February 2015.