N° d'ordre

THESE DE DOCTORAT

Présentée devant L'université Djillali Liabes de Sidi-Bel-Abbes
Faculté de Génie Electrique
Département de Télécommunications
Laboratoire : Télécommunications et de Traitement Numérique du Signal

Pour l'obtention du Diplôme de Doctorat LMD
En Electronique: Télécommunications

Mr. TEHAMI Mohammed Amine

Codes LDPC : Construction, analyse et performances

Soutenu le : 31/ 10 / 2018

Devant le jury composé de :

M. DJEBBAR Ahmed Bouzidi	Pr	Président	**UDL-SBA**
M. DJEBBARI Ali	Pr	Directeur de thèse	**UDL-SBA**
M. ELAHMAR Sid Ahmed	Pr	Examinateur	**UDL-SBA**
M. BOUZIANI Merahi	Pr	Examinateur	**UDL-SBA**
M. BENAISSA Mohammed	MCA	Examinateur	**CU-Ain Témouchent**

Année Universitaire : 2018-2019

Remerciements

En tout premier lieu, je remercie *Dieu*, tout puissant, de m'avoir donné la force pour survivre, ainsi que l'audace pour dépasser *toutes les* difficultés.

Cette thèse est le résultat des travaux de recherche menés au sein du laboratoire de télécommunications et de traitement numérique du signal (LTTNS).

Ce travail n'aurait pas pu voir le jour sans l'encadrement de mon directeur de thèse, professeur DJEBBARI Ali, qui a été mon mentor tout au long de ses dernières années. Je tiens à lui exprimer ma gratitude pour ses conseils, ses orientations et sa confiance dans les choix que j'ai pu faire tout au long de ma thèse, qui m'ont permis de travailler dans d'excellentes conditions.

J'exprime ma reconnaissance au professeur M. DJEBBAR Ahmed Bouzidi, pour avoir accepté de présider le jury de ma thèse. Je remercie également ainsi que les membres du Jury : le Professeur M. ELAHMAR Sid Ahmed de l'université de Sidi Bel Abbes, le Professeur M. BOUZIANI Merahi de l'université de Sidi Bel Abbes et le Docteur M. BENAISSA Mohammed de l'université de Ain Temouchent pour avoir rapporté mon manuscrit de thèse et m'avoir fait part de leurs remarques et leurs corrections.

L'ambiance de travail n'aurait pas été aussi agréable sans la présence de l'ingénieur de laboratoire Mme DASSI Leila. Je tiens à lui exprimer ma plus profonde gratitude pour son soutien indéfectible tout au long de ses années.

Je n'oublie pas l'ensemble des enseignants du département de télécommunications, en plus de ceux déjà cités plus haut, qui ont contribué à ma formation : Mr. Chouakri, Mr. Salah Belkhodja, Mme. Dekkich, Mr. Djebbari, Mr Fassi, Mr.Seddiki et Mme. Driz.

Je remercie aussi ma famille, et en particulier ma mère, mes frères et sœurs et ma futur femme, pour leur patience et la compréhension dont ils ont fait preuve tout au long de mes études.

Merci à tous et à toute

« À ma mère

à la mémoire de mon père

à mes frères et sœurs

à ma fiancée

à toute ma famille

à tous mes amis

à tous les doctorants du laboratoire TTNS des promotions 2011 et 2012 »

Résumé

La construction d'un code LDPC avec un faible plancher d'erreur, un encodage linéaire et décodage moins complexe est nécessaire pour répondre aux contraintes en termes de qualité de réception et d'implémentation. L'objectif de cette thèse est la construction des codes LDPC binaires à large girth, à décodage moins complexe et un encodage linéaire. Nous avons présenté deux contributions sur la construction des codes LDPC binaires. Notre première contribution, est une méthode de construction des codes LDPC à rendement fixe, basée sur une concaténation d'une matrice structurée, matrice de Hankel, et d'une matrice double diagonale. Les codes obtenus garantissent une très faible complexité de décodage tout en réduisant considérablement la mémoire de stockage de la matrice H et un codage a complexité linéaire (dû à la structure de la matrice double diagonale). Notre deuxième contribution, est une méthode de construction des codes LDPC à rendement variable. C'est une autre méthode de construction des codes LDPC basée sur une concaténation d'une matrice sous forme de plusieurs sous matrices de permutations, et une matrice double diagonale. Cette dernière méthode présente les mêmes avantages que la première méthode en termes de complexité d'encodage et de décodage. Par simulation, les codes conçus sont évalués en termes de BER et comparés à une transmission BPSK non codée et à d'autres codes conventionnels.

Abstract

Constructing an LDPC code with a low error floor, linear encoding and less complex decoding is required to reply the constraints in terms of quality of reception and implementation. The objective of this thesis is the construction of binary LDPC codes with large girth, less complex decoding and with a linear encoding. We have presented two contributions on the construction of the binary LDPC codes. The first contribution is a method of constructing, fixed rate LDPC codes based on a concatenation of a structured matrix, Hankel matrix, and dual diagonal matrix. The obtained codes guarantee a very low decoding complexity while reducing the storage memory of the matrix H, and linear encoding complexity (due to the structure of the double diagonal matrix). The second contribution is a method of constructing variable rate LDPC codes based on a concatenation of a matrix, in the form of several permutation sub-matrices, and a dual diagonal matrix.

This last method has the same advantages as the first method in terms of encoding and decoding complexity. By simulation, the designed codes are evaluated and compared to conventional codes.

ملخص

ان انشاء شفرة LDPC بأرضية خطأ منخفضة ,و ترميز خطي, و فك تشفير اقل تعقيدا , امر ضروري للإجابة على الشروط من حيث جودة الاستقبال و التنفيذ . الهدف من هذه الاطروحة هو انشاء رموز LDPC ثنائية ذات محيط كبير , فك تشفير اقل تعقيد و ترميز خطي . قدمنا مساهمتين بشان بناء رموز LDPC ثنائية . مساهمتان الاولى تتمثل في انشاء طريقة بناء شفرات LDPC ذات محيط واسع على اساس تسلسل مصفوفة منظمة , مصفوفة HANKEL , و مصفوفة مزدوجة القطرية . تضمن هذه الشفرات على فك ترميز جد منخفض في حين تقلل الى حد كبير ذاكرة تخزين المصفوفة H , كما تضمن ترميز خطي بسبب بنية المصفوفة مزدوجة القطرية . كما تتمثل مساهمتنا الثانية في انشاء طريقة بناء شفرات LDPC على اساس سلسلة من مصفوفة في شكل عدة مصفوفات شبه متعاقبة , و مصفوفة مزدوجة القطرية . و تتمتع الطريقة الاخيرة بنفس مزايا الطريقة الاولى من حيث تعقيد الشفرة و فك التشفير . من خلال التمثيلات البيانية, يتم تقييم الرموز المصممة من حيث معدل الخطأ (BER) و مقارنتها بشفرات تقليدية.

Table des matières

Liste des figures

Liste des tableaux

Liste des acronymes et des abréviations

Acronyme	Signification Anglaise	Traduction Française
AWGN	Additive White Gaussian Noise Channel	Canal à Bruit Additif Blanc Gaussien
BER	Bit Error Rate	Taux d'Erreur Binaire
BCH	Bose-Chaudhuri-Hocquenghem	Bose-Chaudhuri-Hocquenghem
BCJR	Bahl, Cocke, Jelinek and Raviv	Bahl, Cocke, Jelinek et Raviv
BMAP	Bit Maximum A Posteriori	Maximum A Posteriori par Bit
BML	Bit Maximum Likelihood	Maximum de Vraisemblance par Bit
BP	Belief Propagation	Propagation de Croyance
BPSK	Binary-Phase Shift Keying	Modulation par Déplacement de Phase Binaire
CRC	Cyclic-Redundancy Check	Contrôle par Redondance Cyclique
DSP	Spectral Density of Power	Densité Spectral de Puissance
DVB-S2	Digital Vidéo Broadcasting Satellite	Satellite de Radiodiffusion des Vidéos Numériques
FER	Frame Error Rate	Taux d'Erreur Trame
GF	Galois Field	Corps de Galois
LDPC	Low Density Parity Check	Matrice de Contrôle de Parité à Faible Densité
MATLAB	Mathwork Laboratory	Laboratoire de Mathématiques
PEG	Progressive-Edge-Growth	Croissance Progressive des Bords
QC	Quasi-Cyclic	Quasi-Cyclique
RA	Repeat Accumulate	Répétiton Accumulée
RS	Reed Solomon	Reed Solomon
SNR	Signal-to-Noise Ratio	Rapport Signal sur Bruit

SOVA	Soft Output Viterbi Algorithme	Algorithme de Viterbi à Sortie Douce
WMAP	Word Maximum A Posteriori	Maximum A Posteriori par Mot
WML	Word Maximum Likelihood	Maximum de Vraisemblance par Mot

Introduction générale

Le codage de canal consiste à transmettre de façon efficace la séquence d'information dénommée mot d'information en une séquence codée dénommée mot de code [1, 2] qui peut être sous forme binaire ou non-binaire. Dans cette thèse, on s'intéresse seulement aux codes binaires. Le principe du codage de l'information numérique est de bien récupérer les informations à la réception [1, 2]. La séquence codée reçue, doit être transformée par le récepteur en une séquence estimée qui doit être la même séquence transmise. En réalité, un bruit du canal de transmission entraîne des erreurs qui conduisent au problème de non fiabilité dans les systèmes de communication.

En 1948, Shannon [1] a montré qu'une transmission fiable des données peut être obtenue en ayant un code avec un débit de données arbitrairement proche de la capacité du canal, cela signifie que l'effet du bruit est à son minimum. A partir de cette date, différentes méthodes de construction des codes correcteurs d'erreurs ont été étudiées. L'objectif de la théorie des codes correcteurs d'erreurs est de réduire les erreurs lors de l'opération du décodage, en assurant une grande vitesse de transmission et avoir en même temps un faible complexité de codeur et de décodeur [2].

En 1963, Gallager [3, 4] a découvert les codes *LDPC* (Low-Density-Parity-Check) et il a présenté ces codes sous formes pseudo-aléatoires. Les bons codes LDPC (selon leur performances) sont générés par ordinateur (en particulier des codes avec de grande longueurs de blocs), mais le manque de structure de la matrice de contrôle de parité a causé un décodage très complexe. Ces codes ont été délaissés jusqu'à 1981, lorsque Tanner a développé une interprétation graphique [5]. Après l'invention des turbo-codes, ces codes ont été redécouverts au milieu des années 90 par Mackay et autres [6, 7, 8], Wilberg [9], Sipser et autres [10].

L'obtention des codes *LDPC* à faible plancher d'erreurs est souhaitable. Il est bien connu que le plancher d'erreurs des décodeurs *LDPC* est dû à la présence des topologies dans le graphe de Tanner. Ces topologies sont appelées "girth". Donc, pour la conception des codes *LDPC* binaires à un faible plancher d'erreurs, l'élimination des girth (cycles courts) est nécessaire. Cependant, la majorité des méthodes de construction proposées n'ont pas de structure particulière qui permet d'éliminer les petits cycles. Dans cette thèse, nous allons

concentrer notre travail sur la construction des codes *LDPC* exempt de cycles courts, avec un encodage linéaire et un décodage moins complexe.

Organisation de la thèse

Cette thèse est composée de quatre chapitres organisés comme suit :

Le premier chapitre est dédié à une brève introduction des concepts liés au codage de canal et ses techniques. Ensuite, les différents paramètres qui caractérisent les codes *LDPC* sont décrits. Après avoir introduit deux représentations de ces codes, les méthodes d'encodage sont présentées. Enfin, les opérations du décodage sont discutées.

Le deuxième chapitre est consacré à l'état de l'art sur les méthodes de construction des codes *LDPC*. Dans un premier temps, nous présentons les avantages et les inconvénients des deux constructions aléatoires et déterministes. Dans une deuxième partie, les deux constructions des codes *LDPC* sont décrites et illustrées par des exemples.

Dans le troisième chapitre, nous proposons deux méthodes déterministes de construction des codes *LDPC* irrégulier. La première méthode de construction, est basée sur une concaténation d'une matrice à structurée de Hankel et d'une matrice double diagonale. La deuxième méthode de construction, est basée sur une concaténation de plusieurs sous matrices de permutation et une matrice double diagonale. Dans une dernière partie, les avantages des deux méthodes de construction en termes notamment de complexité d'implémentation sont présentés.

Le quatrième chapitre est consacré à la simulation d'une chaîne de transmission complète basée sur l'algorithme de codage linéaire et l'algorithme du décodage propagation de croyance (*BP* : Belief Propgation). Les performances des deux codes *LDPC* proposés sont présentées et discutées. La dernière partie de ce chapitre, est dédiée à la comparaison des performances des codes *LDPC* proposés, avec une transmission *BPSK* non codée et les codes *LDPC* conventionnels.

Finalement, la conclusion générale de ce travail synthétise les différentes phases présentées dans ce document. Les perspectives de travaux futurs sont également décrites.

Chapitre 1

Introduction aux techniques du codage

1.1 Introduction

Shannon [1] a montré, en 1948, qu'il existe une limite au débit d'informations transmis en présence du bruit, appelée capacité du canal, et après plus de 40 ans de recherche Claude Berrou, Alain Glavieux et Punya Thitimajshima [11] ont montré comment réussir à s'approcher de cette fameuse limite avec une complexité raisonnable. Cette avancée a eu pour conséquence une redécouverte des codes de Gallager en 1962 [3, 4] dénommés codes *LDPC*. Ces codes sont basés sur des matrices de contrôle de parité pseudo-aléatoires de faible densité [4, 12]. Du fait de leur complexité d'encodage, de décodage et des moyens matériels de l'époque, ces codes n'ont pas suscités suffisamment d'intérêt au sein de la communauté de la théorie du codage. Cet oubli durera jusqu'à l'introduction des Turbo-codes et du principe itératif. Ainsi en 1995, Mackay a redécouvert les codes *LDPC* [6, 7, 8] par la suite de nombreux travaux se sont intéressés à ces deux familles de codes : Turbo-codes et codes *LDPC* et plus généralement à l'application du principe itératif dans un système de communication numérique.

Dans ce chapitre, nous allons présenter les notions de base du codage des codes *LDPC*. Ensuite, nous allons introduit les deux présentations de ces codes. Ainsi, les deux types du décodage sont décrits.

1.2 Théorème de Shannon

L'utilisation des codes correcteurs d'erreurs permet d'améliorer la communication à travers un canal bruité, en 1948 Shannon [1] a mis en évidence les règles de réalisation de cette idée :

Shannon a montré que si $Ha<Ca$ donc, il existe une méthode de codage qui garantit une transmission quasi-parfaite (Probabilité d'erreur faible).

Avec *Ha* : Entropie d'une source, *Ca* : Capacité du canal.

Ce résultat peut être jugé comme suit : Si des erreurs de transmission sont générées par un canal de transmission, l'opération de codage est effectuée en ajoutant de la redondance aux messages émis de sorte qu'on puisse retrouver les messages émis sans erreurs [1].

1.3 Modèle de communication numérique

La figure 1.1 représente un système de communication numérique, le vecteur d'information *d* de longueur *K* est constitué à partir d'une source dont les bits sont i.i.d. (indépendant et identiquement distribué) [13]. Le principe de base du codage de canal consiste à remplacer le vecteur d'information *d* à transmettre par un mot de code *c* qui contient de la redondance *p*, l'objectif de la redondance est de faire en sorte que les bits ajoutés ne comportent pas la compréhension globale du mot de code *c* [13, 14].

Le mot de code *c* de longueur *N* (la version codée de *d*) est produit par le codeur de canal, le rendement du code *R* est définit par $R = \frac{K}{N}$

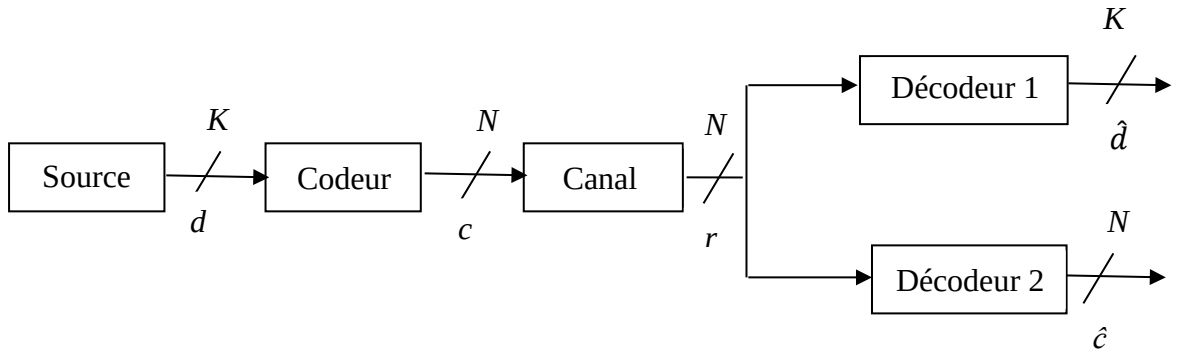


Figure 1. 1 – Modèle de communication numérique [14].

Dans la figure 1.1, il existe deux types de décodeurs : le décodeur '1' est conçu pour calculer une meilleure estimation $\hat{d}$ du mot d'information *d*. Le décodeur '2' est conçu pour calculer une meilleur estimation $\hat{c}$ du mot de code *c* et après on extraire $\hat{d}$ de $\hat{c}$ (dans le cas d'un décodage systématique).

On peut classer les décodeurs selon deux grandes familles [14, 15] : le décodeur à décision dures consiste à quantifier les échantillons du signal reçu en deux niveaux 0 et 1 qu'il fournit au décodeur. Dans le cas du décodeur à décision souple, le décodeur fonctionne à

partir des données fermes accompagnées d'une mesure de fiabilité qui est exploité par le décodeur.

1.4 Les codes correcteurs d'erreurs

Ce paragraphe est consacré à l'étude des caractéristiques des codes correcteurs d'erreurs, l'étude de la borne de Shannon et la capacité du canal [13], [16, 17, 18].

1.4.1 Caractéristiques des codes correcteurs d'erreurs

Les codes correcteurs d'erreur sont regroupés suivant leurs caractéristiques et leurs propriétés [19]. La première grande propriété [20] concerne la linéarité des codes et les deux autres propriétés concernent le caractère systématique et la distance minimale d'un code.

1.4.2 Définition

1.4.2.1 Propriété de linéarité

Un code c est dit linéaire [19], s'il existe une matrice génératrice $G \in GF_{(2)}$, (c'est-à-dire à coefficient N lignes et K colonnes à coefficients dans $\{0,1\}$), de rang K, telle que $\forall\, d \in \{0,1\}, c = d \times G$.

1.4.2.2 Le caractère systématique

Un code est dit systématique si le mot d'information d est contenu dans le mot encodé c [19]. Pour des codes correcteurs systématiques, le mot de code c est donc divisé en K bits systématiques et M bits de redondance avec (M=N-K). L'espace de calcul entre également en jeu dans la caractérisation des codes correcteurs. On définit un alphabet A qui représente l'ensemble des valeurs que peut prendre un élément de calcul de base pour un code donné. Les codes binaires utilisent l'alphabet binaire $F_2=\{0,1\}$. Certains codes utilisent les propriétés des corps de Galois. Dans ce cas, les éléments binaires sont regroupés en groupes de puissance de deux. Ces codes sont alors définis comme des codes non-binaires et leur alphabet correspond au corps de Galois noté F_{2q} ou $GF_{(2q)}$ [19].

1.4.2.3 La distance minimale

Le poids de Hamming [19], noté *wt*(***c***), d'un mot de code ***c*** est le nombre de ses éléments non nuls. Le poids minimal, noté w_{min}, d'un code est le plus petit poids de Hamming de tous les mots de code non nuls.

$$w_{min} = \min \text{ wt}(c) \tag{1.1}$$

Pour un code linéaire : sa distance minimale, notée *Dmin*, est égale à son poids minimal.

$$D_{min} = w_{min} \tag{1.2}$$

1.4.3 Borne de Shannon

La capacité d'un canal *AWGN* (Bruit Additif Blanc Gaussien) à bande limitée est donnée par [1] :

$$Ca = B \log_2(1 + SNR) \quad \text{[Bit/s]} \tag{1.3}$$

Dont *B* est la bande passante du canal et *SNR* est le rapport entre la puissance du signal transmis P_S et la puissance du bruit du canal P_N [1]. Notant que :

$$P_S = E_b R_I \tag{1.4}$$

Tel que E_b est l'énergie du bit d'information et R_I le débit de transmission [13].
Avec

$$P_N = N_0 B \tag{1.5}$$

Tel que N_0 est la densité spectrale de puissance unilatérale du bruit (Watt/Hz). (Voir figure 1.2).

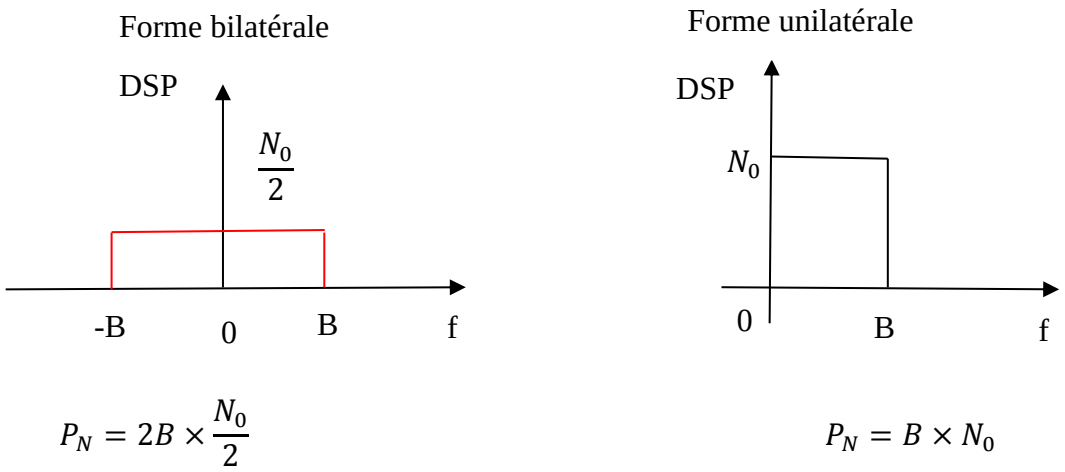


Figure 1. 2 – Densité spectrale du bruit d'un canal *AWGN* à bande limitée [1].

On remplace les équations (1.4) et (1.5) dans l'équation (1.3), on obtient :

$$Ca = B \log_2(1 + \frac{P_S}{P_N}) \quad (1.6)$$

$$Ca = B \log_2(1 + \frac{R_I E_b}{N_0 B}) \quad (1.7)$$

Le tableau 1.1 résume le théorème de Shannon :

Etat	Commentaire
$R_1 < Ca$	Communication fiable
$R_1 > Ca$	Communication non fiable

Tableau 1. 1 – Représentation de la fiabilité de communication [13].

Le théorème de Shannon peut être exprimé d'une autre manière [13, 14] :

$$\eta = \frac{\text{débit d'information}}{\text{bande passante}} = \frac{R_I}{B} \quad (1.8)$$

Avec η: représente l'efficacité spectrale.

Les deux bornes du rapport signal sur bruit E_b/N_0, pour une communication fiable est comme suit :

Le débit $R_I < Ca$, alors :

$$R_I < B\log_2(1+\frac{R_I E_b}{N_0 B})$$

Ce qui donne :

$$\frac{R_I}{B} < \log_2(1+\frac{R_I E_b}{N_0 B})$$

$$\eta < \log_2(1+\eta\frac{\mathrm{E_b}}{\mathrm{N_0}})$$

Ce qui donne :

$$(\frac{\mathrm{E}_b}{N_0}) > \frac{2^{\eta}-1}{\eta} \quad (1.9)$$

La fonction $(\frac{2^{\eta}-1}{\eta})$ est une fonction croissante, le minimum de E_b/N_0 est obtenu pour $\eta \longrightarrow 0$. On peut, à l'aide de la règle "d'Hôpital" [3, 15], calculer la limite de la fonction (2^{η}-*1/* η):

$$\lim_{n\to 0}(\frac{2^{\eta}-1}{\eta}) = \ln(2) = -1{,}59\ dB$$

Alors :

$$\left(\frac{E_b}{N_0}\right)_{min} = -1{,}59dB \quad (1.10)$$

Aussi, le débit d'information maximal (pour une communication fiable) est par définition égal à la capacité du canal [13, 14] :

$$R_{Imax} \triangleq Ca \quad (1.11)$$

Ainsi, l'efficacité spectrale est maximale pour un débit d'information maximal [13, 15] :

$$\eta_{max} = \frac{R_{Imax}}{B} \quad (1.12)$$

De même, on a :

$$R_{Imax} = B\log_2(1+\frac{R_{Imax}E_b}{N_0 B}) \quad (1.13)$$

$$\eta_{max} = \log_2(1+\frac{\eta_{max}E_b}{N_0}) \quad (1.14)$$

Et on obtient, le *SNR* pour une efficacité spectrale maximale :

$$\left(\frac{E_b}{N_0}\right)_{max} = \frac{2^{\eta\,max}-1}{\eta_{max}} \tag{1.15}$$

1.4.4 Capacité d'un canal AWGN

Dans cette partie, on s'intéresse juste à la largeur de bande du signal à transmettre car la bande passante du canal n'est pas usuelle dans l'étude des codes correcteurs d'erreurs, c'est pour cela on doit tenir en compte de tous les signaux présents dans la transmission [13].

Un signal qui a une largeur de spectre *B* peut être échantillonné avec une fréquence *2B* selon $F_e \geq 2B$ (le théorème de Nyquist Shannon) [1].

où F_e représente la fréquence d'échantillonnage.

Si chaque échantillon est codé sur R_{Id} bit, soit alors R_{Id} *[bit /ech]*.

Le débit de transmission R_I *[bit/sec]* est donné par :

$$R_I = R_{Id} \times 2B \tag{1.16}$$

$$R_{Id} = \frac{R_I}{2B} \qquad [bits/ech] \tag{1.17}$$

Le débit R_{Id} maximale est égal à une capacité, à partir de l'équation (1.11), on obtient :

$$C_d = Ca/2B$$

Ainsi, cette relation est donnée par [14], [18] :

$$C_d = \frac{B}{2B}\log_2\left(1 + \frac{R_I E_b}{N_0 B}\right) \tag{1.18}$$

$$C_d = \frac{1}{2}\log_2\left(1 + \frac{2R_{Id}E_b}{N_0}\right) \tag{1.19}$$

$$C_d = \frac{1}{2}\log_2\left(1 + \frac{2C_d E_b}{N_0}\right) \qquad [bit/ech] \tag{1.20}$$

Ce qui donne :

$$\left(\frac{E_b}{N_0}\right) = \frac{2^{2C_d}-1}{2C_d} \tag{1.21}$$

1.4.5 Performances d'un code et complexité d'implémentation

1.4.5.1 Performance d'un code

Un bon code dépend fortement des contraintes fixées suivant le type d'applications visées. Ces contraintes sont sa performance de correction d'erreurs et sa complexité d'implémentation [21]. En termes de performance de correction d'erreurs, deux critères sont nécessaires :

1. Une bonne performance dans la zone de convergence ("Good water-fall") (voir figure 1.3) caractérisée par un taux d'erreurs trame de 10^{-2} à 10^{-3} (acceptable pour les communications sans fils de type téléphonique 3G, 3G+, 4G), pour un rapport signal sur bruit faible.
2. Faible plancher d'erreur ("Low error-floor") (voir figure 1.3) caractérisé par un faible taux d'erreur de 10^{-7} à 10^{-9} (exigé par les communications satellitaires, fibre optique et les enregistrements magnétiques). Le *SNR* correspondant assure presque une transmission sans erreurs.

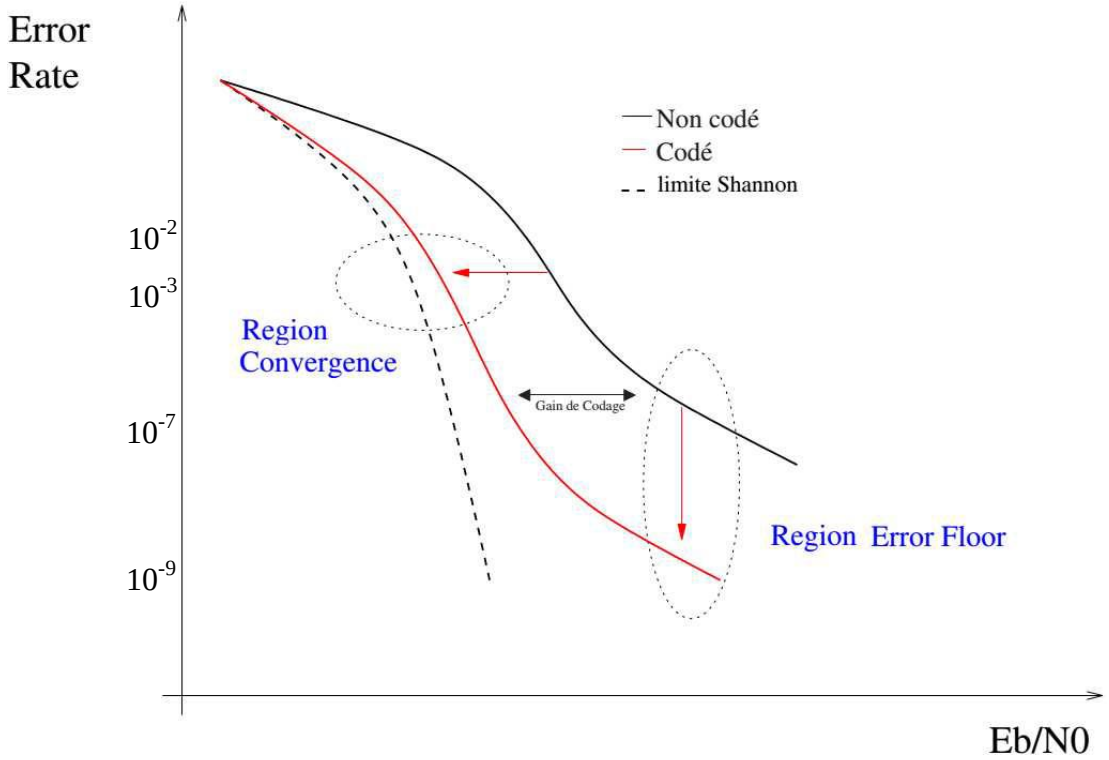


Figure 1. 3 – Les régions caractérisant les performances d'un système de codage [13, 21].

1.4.5.2 Complexité d'implémentation

Lors de l'implémentation d'un code (encodage et décodage), il y'a trois critères qui définissent la complexité de cette dernière [21].

1. Assurer un encodage linéaire (rapide), sachant que l'encodage des codes *LDPC* a une complexité quadratique avec la taille du code.
2. Pour réduire la complexité de décodage des codes *LDPC*, une architecture parallèle pour le décodeur est souhaitable.
3. Une faible latence : atteindre des performances voulus en un petit nombre d'itérations.

Le critère 1 (complexité d'implémentation) est satisfait, si on utilise des matrices doubles diagonales dans la matrice de contrôle de parité *H*. Quant à la conception quasi cyclique, elle permet de satisfaire le critère 1 (Performance). Notant que, la concaténation des matrices à structures quasi cyclique et à double diagonale est adaptée aux exigences des critères : critère 1 (performance) et critère 1 et 3 (complexité d'implémentation) [21].

1.4.6 Code en bloc

Le codage en bloc consiste à découper les informations à transmettre en blocs de K bits. Si on considère des codes binaires, on a 2^K mots différents. A chacun de ces 2^K mots non codés, est associé un mot de code unique de N bits [13]. Ainsi parmi les 2^N mots possibles de N bits, seuls 2^K sont des mots de code, tandis que le nombre de mots n'appartenant pas au code est plus important (égal à 2^N-2^K) [13]. La plupart des codes correcteurs d'erreurs utilisés sont des codes linéaires. Ils sont caractérisés par le fait que toute combinaison linéaire de mots de code donne un mot de code. Une caractéristique fondamentale des codes linéaires est que tout code linéaire admet une représentation systématique et est donc équivalent à un code systématique. Un code linéaire systématique est aussi appelé un code polynomial. En effet dans ce cas, les coefficients du polynôme générateur sont simples (ils ne dépendent pas des symboles précédemment codés), et la représentation polynomiale est la plus appropriée [13].

Pour passer à la représentation matricielle, on utilise directement le polynôme générateur qui permet de construire les lignes de la matrice génératrice [13].

Parmi les codes linéaires, un code est dit cyclique si toute permutation circulaire d'un mot de code donne un autre mot de code.

Parmi les codes cycliques les plus connus, on peut citer les codes de Hamming [22], les codes *CRC* (Cyclic Redundancy Check) [23], les codes *BCH* (Bose-Chaudhuri-Hocquenghem) [24, 25] et les codes *RS* (Reed Solomon) [26].
Les codes *LDPC* font partie des codes en blocs linéaires, mais ne sont a priori pas cycliques.

1.4.6.1 Définitions

Un code en bloc, définit sur le champ de Galois $GF_{(q)}$, peut être considéré comme un sous-espace vectoriel de dimension *K* [13, 14, 15] de l'espace vectoriel de dimension $GF_{(q)}{}^{N}$, le code peut être définit comme :

$$Ce = \left\{c^{(i)}, i \in \{0, \cdots, 2^{k-1}\}\right\} \tag{1.22}$$

Ce : représente les combinaisons des mots de codes.

La matrice génératrice *G* est une matrice de dimension $(K \times N)$, on peut obtenir un mot de code *c* comme suit [3], [13, 14, 15] :

$$c \in Ce \Rightarrow d \in GF_{(q)^K} /\ c = d \times G \tag{1.23}$$

La matrice de contrôle de parité *H* est une matrice de dimension $((N - K) \times N)$ dont ses éléments sont définit dans $GF_{(q)}$ et l'ensemble des mots de code est donné comme suit [3], [13, 14, 15] :

$$Ce = \left\{c^{(i)} / c^{(i)} \times H^t = 0\right\} \tag{1.24}$$

Cette matrice de contrôle de parité est constituée de $(M = N - K)$ lignes et chaque ligne représente une équation de contrôle de parité [3, 4], [13]. La condition de l'orthogonalité doit être satisfaite entre la matrice génératrice et la matrice de contrôle de parité.

$$G \times H^t = 0 \tag{1.25}$$

1.4.6.2 Graphe de Tanner

Tanner [5], [27] représente les codes en bloc en utilisant un graphe biparti dénommé graphe de Tanner, ce dernier offre une représentation complète des codes et explique l'algorithme du décodage. Pour une matrice *H* de dimension $(M \times N)$ (*N* colonnes et *M=N-K*

lignes), chaque nœud de variable V_n (avec $n \in \{1 \ldots N\}$) correspond à un bit c_n du mot de code (colonne) et chaque nœud de contrôle C_m (avec $m \in \{1 \ldots M\}$) correspond à une contrainte de contrôle de parité pc_m (ligne).

Un nœud de variable V_n est connecté à nœud de contrôle C_m si et seulement si $H\ (m,n)$ est non nul. Ainsi, le nombre des branches du graphe de Tanner égal au nombre des éléments non nuls de H [5].

Un code en bloc est définit par l'ensemble de ces mots de code '*Ce*', la matrice génératrice *G*, matrice de contrôle de parité *H* et son graphe de Tanner (voir figure 1.4) comme illustrés ci-dessous:

$$Ce = \begin{bmatrix} 0\,0\,0\,0\,0\,0\,0 & 0,0\,0\,1\,0\,1\,0\,1 & 1,1\,0\,1\,0\,1\,0\,1 & 0,1\,0\,1\,0\,1\,1 \\ 1\,0\,1\,0\,0\,0\,1 & 0,0\,0\,1\ 1\,1\,0\,1 & 1,1\,0\,1\,0\,0\,0\,0 & 0,0\,0\,0\,0\,1\,1 \\ 1\,0\,1\,1\,1\,1\,1 & 0,0\,0\,1\,1\,1\,1\,0 & 0,0\,0\,1\,1\,1\,1\,1 & 1,1\,1\,1\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0 & 1,1\,0\,1\,0\,0\,0\,1 & 0,1\,0\,0\,0\,1\,1\,1 & 1,1\,1\,1\,1\,1\,1 \end{bmatrix}$$

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

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

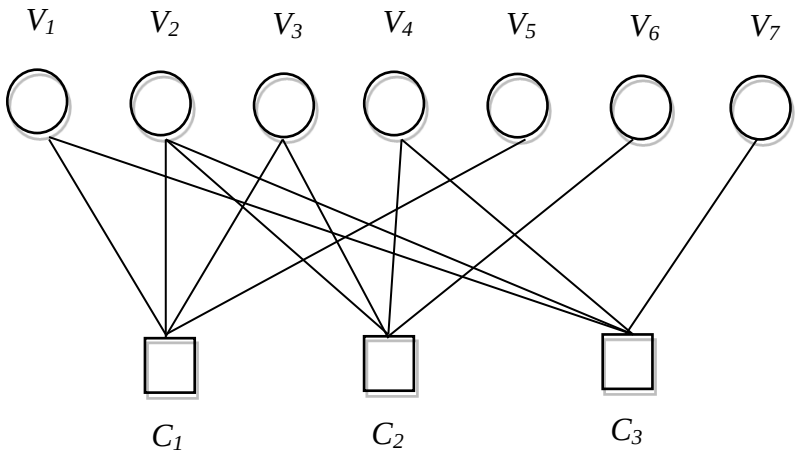


Figure 1. 4 – Graphe de Tanner d'un code en bloc (7,4).

Voici quelques notations nécessaires pour la représentation graphique [14].

Soit *N(m)* l'ensemble des symboles qui sont impliqués à la $m^{éme}$ contrainte de parité $N(m)=\{c_n/H(m,n)=1\}$.

Soit *N(m)/n* l'ensemble des symboles qui sont impliqués à la $m^{éme}$ contrainte de parité sauf le symbole c_n.

Soit *M(n)* l'ensemble des contraintes de parité aux quelles le symbole c_n est impliqué $M(n)=\{pc_m/H(m,n)=1\}$.

Soit *M(n)/m* l'ensemble des contraintes de contrôle de parité aux quelles le symbole c_n est impliqué sauf la contrainte pc_m.

Soit $C_{m,n}$ représente la parité des symboles impliqués aux contraintes de parité de l'ensemble *M(n)* sauf le symbole $c_{n.}$

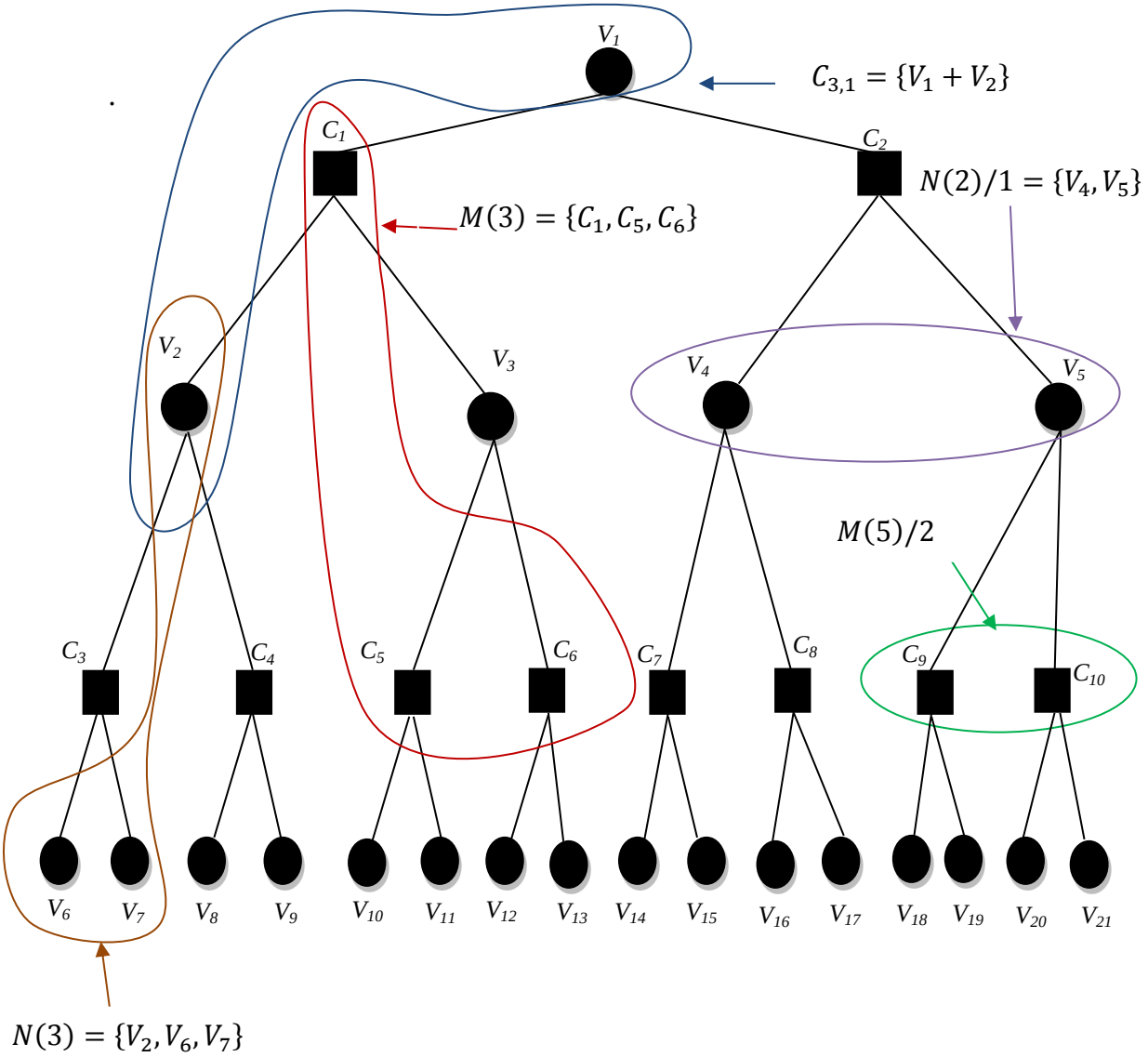


Figure 1. 5 – Illustration des ensembles à partir du graphe de Tanner [14].

1.4.6.3 Hypothèse d'un graphe sans cycle

Un graphe de Tanner est avec cycle (figure 1.6), (figure 1.7) s'il contient un chemin de départ et d'arrivée au même nœud. La longueur du plus petit cycle contenu dans le graphe de Tanner est appelée 'Girth' ou périmètre [5].

Figure 1. 6 – Graphe de Tanner (bipartie) d'un code *LDPC* avec cycle de longueur 4 (a) et sans cycle (b).

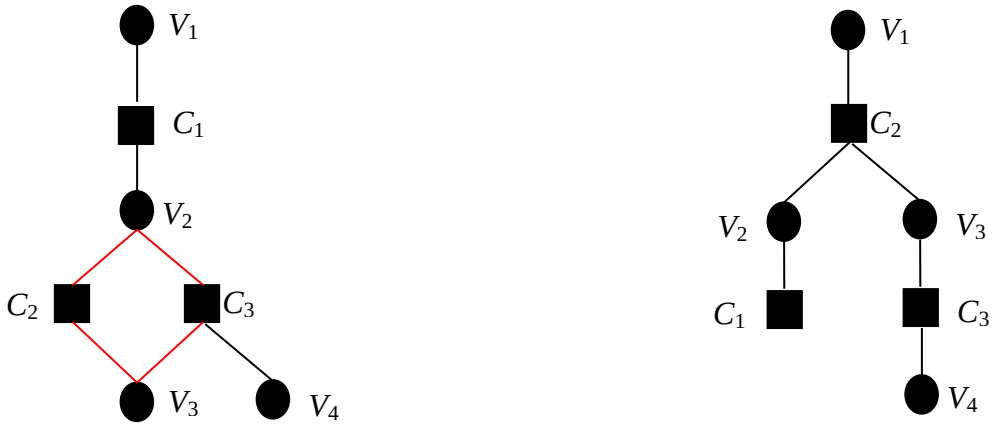


Figure 1. 7 – Graphe de Tanner (en arbre) d'un code *LDPC* avec un cycle de longueur 4 (a) et sans cycle (b).

1.5 Les codes LDPC

1.5.1 Bref historique

Les codes *LDPC* composent une classe des codes en bloc qui sont caractérisés par une matrice de contrôle de parité *H* creuse. Ils ont été décrits pour la première fois dans la thèse de

Gallager au début des années 60 [3, 4]. Ce travail a été délaissé pendant 30 années. Seulement quelques études rares se sont rapportées au cours de cette période. En particulier, Tanner a proposé une généralisation des codes de Gallager et représentations en graphe bipartite [5], [27].

En 1993, l'algorithme de décodage itératif des turbo-codes a été inventé par Berrou, Glavieux et Thitimajshima [11], [13], [28]. Les performances remarquables obtenues par les turbo-codes ont suscité plusieurs questions et beaucoup d'intérêt vers les techniques itératifs.

Au milieu des années 90, la redécouverte des codes de Gallager a été par Mackay et autres [6, 7, 8], Wilberg [9] et Sipser et autres [10]. Par conséquent, les codes *LDPC* sont à la confluence de deux grandes révolutions dans la communauté du codage de canal: la représentation du Tanner basée sur les graphes et les techniques de décodage itératif. La communauté scientifique a été motivée à suivre les travaux sur les codes *LDPC* grâce à la simplicité de l'analyse théorique de ces codes.

En 2004, et pour la première fois, un code *LDPC* a été normalisé par le satellite *DVB-S2* [29] et après ces codes ont été utilisés par les standards *IEEE* 802.16e [30] (Wi Max Mobile) et *IEEE* 802.11n [31](WIFI).

1.5.2 Définition

Les codes à faible densité sont des codes en blocs linéaires, le terme faible densité venant du fait que la matrice de contrôle *H* contient un faible nombre de valeurs non nulles veut dire que cette matrice est creuse [2]. Dans le cas particulier des codes *LDPC* binaires, la matrice de contrôle de parité contient un faible nombre de '1' devant le nombre de '0'. D'autre part, le graphe bipartite associé contient un faible nombre de connexions entre les nœuds de variable et les nœuds de contrôle. La faible densité de la matrice de contrôle de parité d'un code a pour but de réduire le nombre de cycles et augmenter la longueur des cycles [2], [8]. Aussi, lorsque la longueur *N* du mot de code augmente, le graphe de Tanner contient beaucoup moins de cycles [2, 5].

1.5.3 Représentation des codes LDPC

Un code *LDPC* peut se représenter selon deux formes :

1.5.3.1 Représentation matricielle

Pour un code *LDPC* de paramètres *(N, W_c,W_r)*, la relation de contrôle de parité entre un mot de code *c* et une matrice de parité *H* est définie par [3, 4] :

$$Hc^t = 0 \tag{1.26}$$

où *Wc* et *Wr*: représentent respectivement le poids de colonnes et de lignes

La matrice de contrôle de parité est de dimension ($M \times N$) avec N : nombre de colonnes ou la longueur du mot de code, M : nombre de lignes ou le nombre des équations de contrôle de parité, où K : représente les bits d'information ($K = N - M$). Cette matrice H est composée de (W_c) '1' par colonne et (W_r) '1' par ligne. Ainsi, chaque élément du mot de code participe à (W_c) équations de contrôle parité et chacune de ces équations est composée de (W_r) bits.

Exemple : On considère la matrice de contrôle de parité suivante d'un code produisant 4 bits de redondance.

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

C'est un code *LDPC* ou, le nombre de colonne (ou la longueur du mot de code) est N=8, le nombre de ligne (ou le nombre des équations de contrôle de parité) est M=4, le nombre de bits d'informations K=4.

La matrice H est composée d'un seul '1' par colonne et deux'1' par ligne et de zéros pour tous les autres éléments. Ainsi, chaque élément du mot de code contribue à une équation de contrôle parité et chacune de ces équations est utilisée par 2 bits.

1.5.3.2 Représentation graphique

Un graphe de Tanner [5] est une représentation graphique d'un code *LDPC*, ce graphe contient tous les éléments nécessaire du code et explique l'algorithme de décodage de ces codes.

La figure 1.8 représente un graphe de Tanner d'un code *LDPC* (8, 1, 2).

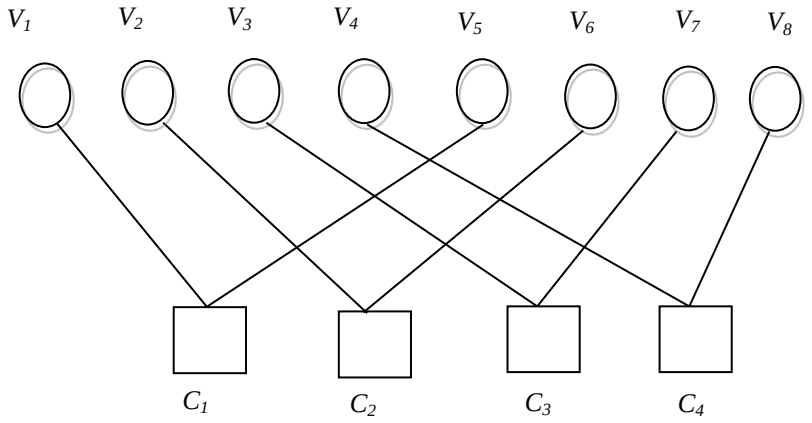


Figure 1. 8 – Graphe de Tanner d'un code *LDPC* (8, 1, 2).

Dans ce graphe, il y'a 8 nœuds de variable (longueur du mot de code ou nombre de colonnes dans la matrice *H*) et 4 nœuds de contrôle (contraintes de parité ou nombre de lignes dans la matrice *H*).

1.5.4 Classes des codes LDPC

La classe des codes *LDPC* engendre un très grand nombre de codes. Il est commode de les distinguer en deux classes:

-Les codes *LDPC* réguliers.

-Les codes *LDPC* irréguliers.

Un code *LDPC* est dit régulier [3, 4], lorsque la matrice de contrôle *H* contient un nombre constant de '1' dans chaque lignes et un nombre constant de '1'' dans chaque colonnes. Ainsi, chaque bit du mot de code contribue à un même nombre d'équations de parité et chacune des équations de parité utilise le même nombre de bits [14]. Ce code est noté code *LDPC (N, W_c, W_r)*.

Exemple:

Soit un code *LDPC* (8, 2, 2)

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

C'est un code *LDPC* régulier.

Un code LDPC est dit irrégulier si le nombre de '1' par ligne et le nombre de '1' par colonne varie et le principe d'irrégularité des variables d'un code *LDPC* irrégulier est défini par le polynôme suivant [3, 4], [32] :

$$\lambda(x) = \sum_{i\geq 1}^{w_{c\,max}} \tilde{\lambda}_i x^{i-1} \qquad (1.27)$$

Le coefficient λ_i est égal au rapport entre le nombre cumulé de '1' par colonne de poids *i* et le nombre total de '1' de la matrice *H*, par exemple :

$$\lambda(x) = 0.2x^4 + 0.8x^3$$

Cela est équivalent un code où 20% de '1' sont associés à des nœuds de variables de degré 5 et 80% de '1' sont associés à des nœuds variables de degré 4.

Autrement dit, le profil d'irrégularité peut être représenté par le polynôme suivant [3, 4], [32] :

$$\rho(x) = \sum_{j=2}^{w_{rmax}} \tilde{\rho}_j x^{j-1} \qquad (1.28)$$

Le coefficient ρ_j est égal au rapport entre le nombre cumulé de '1' des lignes de poids *j* et le nombre total de '1' de la matrice *H*.

1.5.5 Encodage des codes LDPC

Si la matrice *H* n'a pas de structure particulière, donc l'encodage d'un code *LDPC* peut être relativement complexe et pour cela un prétraitement complexe sur la matrice *H* est

nécessaire [13]. D'une autre manière, on peut construire directement la matrice *H* de façon à obtenir un code systématique très simple à encoder [13], [33]. Particulièrement, cette solution a été adoptée pour le code du standard *DVB-S2* de transmission numérique de télévision par satellite [29, 30, 31].

1.5.5.1 Encodage par matrice génératrice

Les codes *LDPC* étant des codes en bloc linéaires, l'encodage peut se faire par la matrice génératrice *G* de taille (*K*×*N*) du code (voir équation 1.30). Ces codes se définissent à partir de leur matrice de contrôle de parité *H* (voir équation 1.29), qui n'est pas systématique [13]. En utilisant l'algorithme, d'élimination, Gauss-Jordan, on peut transformer la matrice *H* en matrice systématique H_{sys}. L'inconvénient de cette technique est que la matrice génératrice G_{sys} du code systématique est dense donc, l'encodage devient plus complexe pour des codes de taille usuelle [13].

La matrice de contrôle de parité *H* est donnée par :

$$H_{sys} = [A \quad I_{N-K}] \tag{1.29}$$

$I_{N\text{-}K}$: matrice unité de taille (*N*-*K*×*N*-*K*)

La matrice génératrice *G* est donc :

$$G_{sys} = [I_K \quad A^T] \tag{1.30}$$

où *A* est matrice de dimension (*N* - *K*) × *K*.

1.5.5.2 Encodage à base de la forme triangulaire de H

Une solution permettant un encodage quasi-linéaire a été proposée par Richardson et autres. [33]. En utilisant des algorithmes qui permettent d'effectuer un prétraitement de la matrice de contrôle de parité *H*, on peut mettre cette dernière sous une forme presque triangulaire inférieure, comme illustré dans la figure (1.9) :

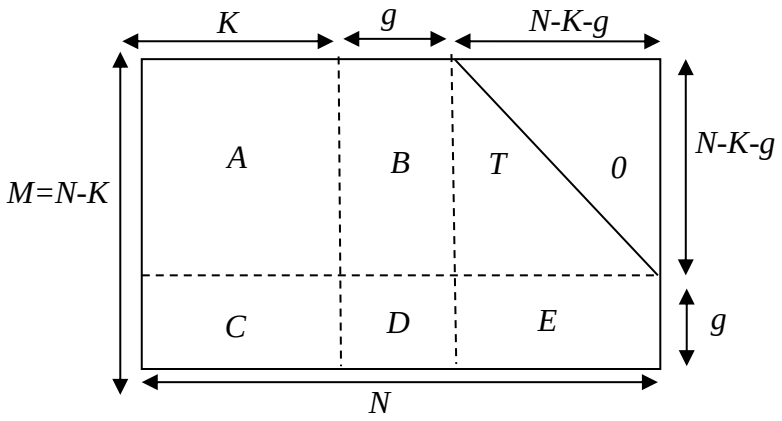


Figure 1. 9 – Représentation sous forme pseudo triangulaire inférieure de la matrice de contrôle de parité [33].

La figure I.9 montre la structure de la matrice *H'* obtenue après traitement de *H* (*une* décomposition de *H* en plusieurs sous matrices *A, B, T, C, D* et *E*) [13]. Lorsqu'on obtient cette forme, le processus (élimination de Gauss-Jordan) est appliqué (un processus pour mettre la sous matrice *E* nulle). Ce processus consiste à multiplier *H'* par H_{Gj} (équation 1.31) pour obtenir la matrice H_r (équation 1.32), d'où :

$$H_{Gj} = \begin{bmatrix} I_{M-g} & 0 \\ -ET^{-1} & I_g \end{bmatrix} \tag{1.31}$$

$$H_r = \begin{bmatrix} A & B & T \\ C' & D' & 0 \end{bmatrix} \tag{1.32}$$

Le mot de code recherché est décomposé en trois parties : $c = (d, P_1, P_2)$, où d représente le mot d'information, P_1 et P_2 représentent les bits de redondance cherchés, de tailles respectives g et N-g-K. Ainsi, le mot de code c doit vérifier l'équation (1.33) :

$$H_r . c^t = 0 \tag{1.33}$$

Ce qui donne :

$$[Ad] + [BP_1] + [TP_2] = 0 \tag{1.34}$$

$$[C'\mathrm{d}] + [D'P_1] = 0 \tag{1.35}$$

Si la matrice *D'*est non singulière, alors le premier vecteur de la redondance P_1 est comme suit :

$$P_1 = D'^{-1}[C'd] \tag{1.36}$$

En remplaçant la valeur de P_1 dans l'équation (1.34), on peut calculer P_2 comme suit :

$$P_2 = T'^{-1}([\mathrm{A}d] + [BP_1]) \tag{1.37}$$

1.5.5.3 Encodage à base de contrainte de construction

Cette méthode comporte une construction bien définie de la matrice *H* [30], [34], qui doit être divisée en deux sous matrices comme suit :

$$H = [H^d\ H^p] \tag{1.38}$$

Le mot de code *c* s'écrit comme $c = [d \quad p]$, avec d et p représentent respectivement le mot d'information et le mot de redondance. La relation de la parité s'écrit : $Hc^T = 0$, cela conduit à :

$$(H^d H^p)\begin{pmatrix} d \\ p \end{pmatrix} = 0 \tag{1.39}$$

$$H^d d = H^p p \tag{1.40}$$

$$p = dH^d(H^p)^{-1} \tag{1.41}$$

Dans [34], une autre méthode consiste à définir un vecteur de projection *v* :

$$v = H^d d \tag{1.42}$$

Le vecteur de redondance *p* s'obtient en replaçant l'équation (1.42) dans l'équation (1.40) :

$$p = v(H^p)^{-1} \tag{1.43}$$

Dans le cas où la matrice H^p est de type double diagonal (la norme *DVB-S2*), l'encodage se fait d'une manière très simple [34, 35]. Dans ce cas, le calcul des bits de redondance p_i peut se faire comme suit :

$$p_1 = \sum_j h_{1j}^d d_j \tag{1.44}$$

$$p_i = p_{i-1} + \sum_j h_{ij}^d d_j \quad mod\ (2) \tag{1.45}$$

Ou h_{ij}^d, p_i *et* d_j : représentent respectivement les éléments de H^d, du vecteur de redondance p et du mot d'information d.

1.6 Décodage optimale des codes LDPC

Il existe deux types de décodage optimal [14] :

1.6.1 Décodage par mot de code

Le but du décodeur est de trouver le mot décode $\hat{c}$ à partir du mot de code c, sachant le vecteur reçu r, via le canal, selon l'équation (1.46) :

$$\hat{c} = arg \max_{c\prime \in Ce} P_r(c = c'/r) \tag{1.46}$$

C'est le décodeur maximum a posteriori du mot de code (*W-MAP*) est donné comme suit: Word Maximum A Posteriori. En utilisant le théorème de Bayes la probabilité a posteriori $Pr(c = c'/r)$ est donnée comme suit

$$P_r(c/r) = \frac{P_r(r/c)P_r(c)}{P_r(r)} = \frac{P_r(r/c)P_r(c)}{\sum_{c\in Ce} P_r(r/c)P_r(c)} \tag{1.47}$$

Si les probabilités a priori *Pr(c)* sont égaux (équiprobables), l'équation (1.46) peut être exprimée comme suit [14] :

$$\hat{c} = arg \max_{c\prime \in Ce} P_r(r/c = c') \tag{1.48}$$

L'équation (1.48) est appelée décodage du mot de code par 'maximum de vraisemblance' (*W-ML*). $P_r(r/c)$ est appelée la fonction de vraisemblance lorsque r est fixé et il s'agit d'une fonction de densité de probabilité conditionnelle lorsque c est fixe. La seule manière d'obtenir un décodeur *W-MAP* optimal est de tester chaque mot de code, par exemple 2^K pour une source binaire. Si la source est équiprobable, les deux décodeurs *W-MAP* et *W-ML* sont équivalents et optimaux.

1.6.2 Décodage par symbole

Si le taux d'erreur binaire est concerné, le décodeur de bit (*BMAP*) et les décodeurs de bit (*B-ML*) donnent respectivement une estimation des symboles c_n du mot de code, comme illustrés ci- dessous :

$$\hat{c}_n = arg \max_{c\prime \in Ac} P_r(c_n = c_n\prime / r) \tag{1.49}$$

$$\hat{c}_n = arg \max_{c\prime \in Ac} P_r(r / c_n = c_n\prime) \tag{1.50}$$

où A_c représente l'alphabet des symboles du mot de code.

L'utilisation de l'algorithme *BCJR* (Bahl, Cocke, Jelinek and Raviv) [14, 36], dénommé l'algorithme vers l'avant et vers l'arrière, permet de calculer les probabilités a posteriori des symboles c_n, à condition que le code ait une représentation en treillis. L'algorithme de Viterbi à sortie douce (*SOVA*) [37] est un algorithme '*B-MAP*' sous-optimal qui ne nécessite qu'un seul traitement direct dans le treillis.

1.7 Décodage optimale des codes en bloc binaires

Le meilleur *FER* (Frame Error Rate) possible est atteint en utilisant l'algorithme de Viterbi : il nécessite une représentation du code en treillis [14]. Lorsque les codes en blocs ont une représentation en treillis [35, 36], la complexité de leur treillis augmente exponentiellement avec la taille du code. Puisque plusieurs représentations conduisent à des implémentations d'algorithmes irréalisables, de nombreux algorithmes de décodage sous-optimal ont été proposés pour réduire l'ensemble des mots de code *Ce*. Dans ce qui suit, on va s'intéresser aux taux d'erreur symbole optimale en utilisant l'algorithme *BCJR*.

c et *r* représentent respectivement le mot de code et le vecteur du signal reçu définies comme suit $r = \{r_1, r_2, \dots, r_N\}$ et $c = \{c_1, c_2, \dots, c_N\}$. Le vecteur *r* peut être divisé en deux sous-ensembles notés $\{r_n\}$ et $\{r_{i \neq n}\}$, avec :

$$r = \{r_n\} \cup \{r_{i \neq n}\}$$

La décision du décodage des codes en blocs se fait comme à partir de l'équation (1.49) [13, 14], [32] :

$$\begin{cases} \hat{c}_n = 0 & si \quad P_r(c_n = 0/r) > P_r(c_n = 1/r) \\ \hat{c}_n = 1 & si \quad P_r(c_n = 0/r) < P_r(c_n = 1/r) \end{cases} \tag{1.51}$$

La probabilité conditionnelle $P_r(c_n/r)$ est (voir annexe A.1) :

$$P_r(c_n/r) = \frac{P_r(r_n/c_n)P_r(c_n/r_{i\neq n})}{P_r(r_n/r_{i\neq n})} \tag{1.52}$$

De l'équation (1.52), on a :

$$\begin{cases} \hat{c}_n = 0 \quad si \quad \frac{P_r(c_n=0/r)}{P_r(c_n=1/r)} > 1 \Rightarrow \log\frac{P_r(c_n=0/r)}{P_r(c_n=1/r)} > 0 \\ \hat{c}_n = 1 \quad si \quad \frac{P_r(c_n=0/r)}{P_r(c_n=1/r)} < 1 \Rightarrow \log\frac{P_r(c_n=0/r)}{P_r(c_n=1/r)} < 0 \end{cases} \tag{1.53}$$

Selon l'équation (1.52), on a :

$$T_n = \log\frac{P_r(c_n=0/r)}{P_r(c_n=1/r)} = \log\frac{P_r(r_n/c_n=0)}{P_r(r_n/c_n=1)} + \log\frac{P_r(c_n=0/r_{i\neq n})}{P_r(c_n=1/r_{i\neq n})} \tag{1.54}$$

où T_n désigne l'ensemble d'information du bit *n*, c'est le logarithme du rapport entre les deux probabilités a posteriori du bit *n*. Le signe de T_n permet l'estimation du bit c_n et l'amplitude de T_n est la fiabilité de la décision.

On pose :

$I_n = \log\frac{P_r(r_n/c_n=0)}{P_r(r_n/c_n=1)}$, définie comme la quantité représentant l'information intrinsèque du bit *n*, elle est reliée à la valeur du signal reçu r_n et aux paramètres du canal de transmission (voir annexe A.2).

$E_n = \log\frac{P_r(c_n=0/r_{i\neq n})}{P_r(c_n=1/r_{i\neq n})}$, définie comme la quantité représentant l'information extrinsèque du bit *n*, c'est l'amélioration qu'on gagne si les symboles codés respectent les contraintes de contrôle de parité.

Soit *M(n)* : l'ensemble des contraintes de parité aux quelles le bit c_n est impliqué (voir figure 1.5). $pc_{m,n}$ représente la parité des symboles impliqués aux contraintes de parité de l'ensemble *M(n)* sauf le symbole c_n (voir figure 1.5). $pc_{m,n}$ est définie comme suit :

$$pc_{m,n} = \sum_{j\in N(m)/n} c_j \tag{1.55}$$

Si $c_n = 1$ et on a : $pc_{m,n} + c_n = 0 \Rightarrow pc_{m,n}$ =1, pour toutes les contraintes *mϵM(n),* dans lesquelles le bit c_n participe. Si $c_n = 0$, alors $pc_{m,n}$=0, pour toutes les contraintes *mϵM(n),* dans lesquelles le bit c_n participe. Exprimons maintenant les probabilités contenues dans l'information extrinsèque en fonction des équations de contrôle de parité [14], [32], l'équation (1.54) devient :

$$T(c_n/r) = I_n + \log \frac{P_r(pc_{m,n}=0 \; pour \; toutes \; les \; m\epsilon M(n)/r_{i\neq n})}{P_r(pc_{m,n}=1 \; pour \; toutes \; les \; m\epsilon M(n)/r_{i\neq n})} \tag{1.56}$$

Puisque l'ensemble des bits associé à $pc_{m,n}$ est indépendant à l'ensemble des bits associés à $pc_{z,n}$ pour $z \neq m$ (voir figure 1.10).

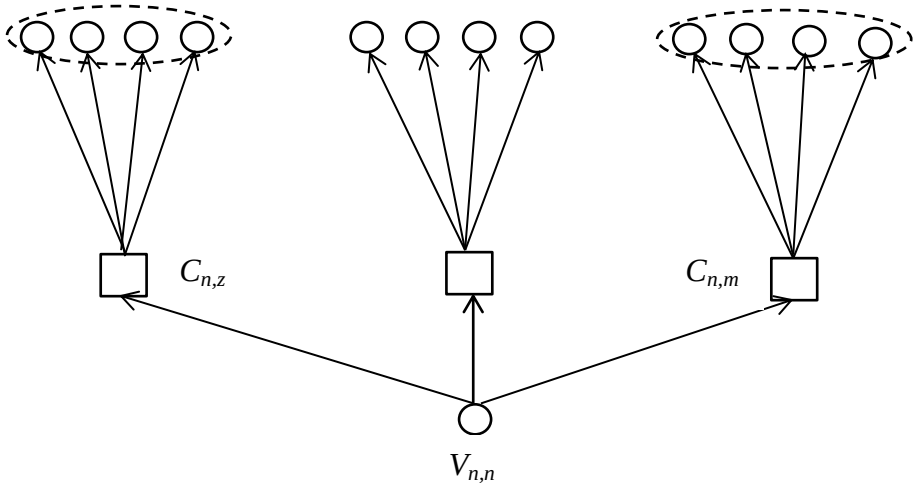


Figure 1. 10 – Indépendance conditionnelle des ensembles de bits sur le graphe de Tanner.

l'équation (1.56) devient [38]

$$T(c_n/r) = I_n + \log \frac{\prod_{m\epsilon M(n)} P_r(pc_{m,n}=0 \; /r_{i\neq n})}{\prod_{m\epsilon M(n)} P_r(pc_{m,n}=1 \; /r_{i\neq n})} \tag{1.57}$$

$$T(c_n/r) = I_n + \sum_{m\epsilon M(n)} \log \frac{P_r(pc_{m,n}=0 \; /r_{i\neq n})}{P_r(pc_{m,n}=1 \; /r_{i\neq n})} \tag{1.58}$$

par définition, le rapport de vraisemblance de ($pc_{m,n}/r_{i\neq n}$) est comme suit :

$$T(pc_{m,n}/r_{i\neq n}) = \log \frac{P_r(pc_{m,n}=0 \; /r_{i\neq n})}{P_r(pc_{m,n}=1 \; /r_{i\neq n})} \tag{1.59}$$

en remplaçant l'équation (1.59) dans l'équation (1.58), on obtient :

$$T(c_n/r) = I_n + \sum_{m\epsilon M(n)} T(pc_{m,n}/r_{i\neq n}) \tag{1.60}$$

en remplaçant l'équation (1.55) dans (1.60), on obtient :

$$T(c_n/r) = I_n + \sum_{m\epsilon M(n)} T\left(\sum_{j\in N(m)/n} c_j/r_{i\neq n}\right) \tag{1.61}$$

on invoque ''the tanh rule'' (voir annexe A.3), on obtient [14], [38] :

$$T(c_n/r) = I_n + 2\sum_{m\epsilon M(n)} tanh^{-1}\left(\prod_{j\in N(m)/n} \tanh\left(\frac{T(c_j/r_{i\neq n})}{2}\right)\right) \tag{1.62}$$

l'information $T(c_n/r)$ peut s'écrire comme suit :

$$T(c_n/r) = I_n + \sum_{m\epsilon M(n)} E_{n,m} \tag{1.63}$$

où

$$E_{n,m} = 2tanh^{-1}\left(\prod_{j\in N(m)/n} \tanh\left(\frac{T(c_j/r_{i\neq n})}{2}\right)\right) \tag{1.64}$$

par définition [38], on a :

$$E_n = \sum_{m\epsilon M(n)} E_{n,m} \tag{1.65}$$

ce qui donne :

$$E_n = \sum_{m\epsilon M(n)} 2tanh^{-1}\left(\prod_{j\in N(m)/n} \tanh\left(\frac{T(c_j/r_{i\neq n})}{2}\right)\right) \tag{1.66}$$

1.7.1 Les opérations de calcul

Comme illustrés dans les figures 1.11 et 1.12, l'information extrinsèque $E_{n,m}$ s'obtient à partir des informations partiels $T_{n,m,}$, représentant les messages transitant d'un nœud de variable vers un nœud de contrôle. Le calcul de l'information totale T_1, correspond à la profondeur de l'arbre entre le nœud de variable c_1 et le dernier nœud. Aussi, l'information totale T_3 peut être calculée en utilisant les résultats partiels obtenus lors du calcul de T_1 (voir figure 1.12). Les informations totales sont indépendantes donc, elles peuvent être traitées en

parallèle. Si un nœud du graphe est considéré et si les *N* variables du mot de code sont traitées au même temps donc, ce nœud est impliqué dans toutes les étapes de calcul [13].

Un processus d'analyse général pour les nœuds consiste à traiter tous les messages au même temps, c'est-à-dire que dans chaque arrivée d'un message à un nœud de variable c_n ou un nœud de contrôle pc_m, la valeur de ce nœud sera modifiée, aussi chaque nœud de variable ou nœud de contrôle traite toutes les messages de sortie et ainsi, on peut considérer ces nœuds comme des processeurs locaux de traitement indépendants [13], [38] . Ces processus de traitement sont dénommés 'mise à jour du nœud' (node update). Ces mises à jour sont répétées jusqu'au calcul de l'information totale de chaque bit. Chaque répétition est appelée une itération [13].

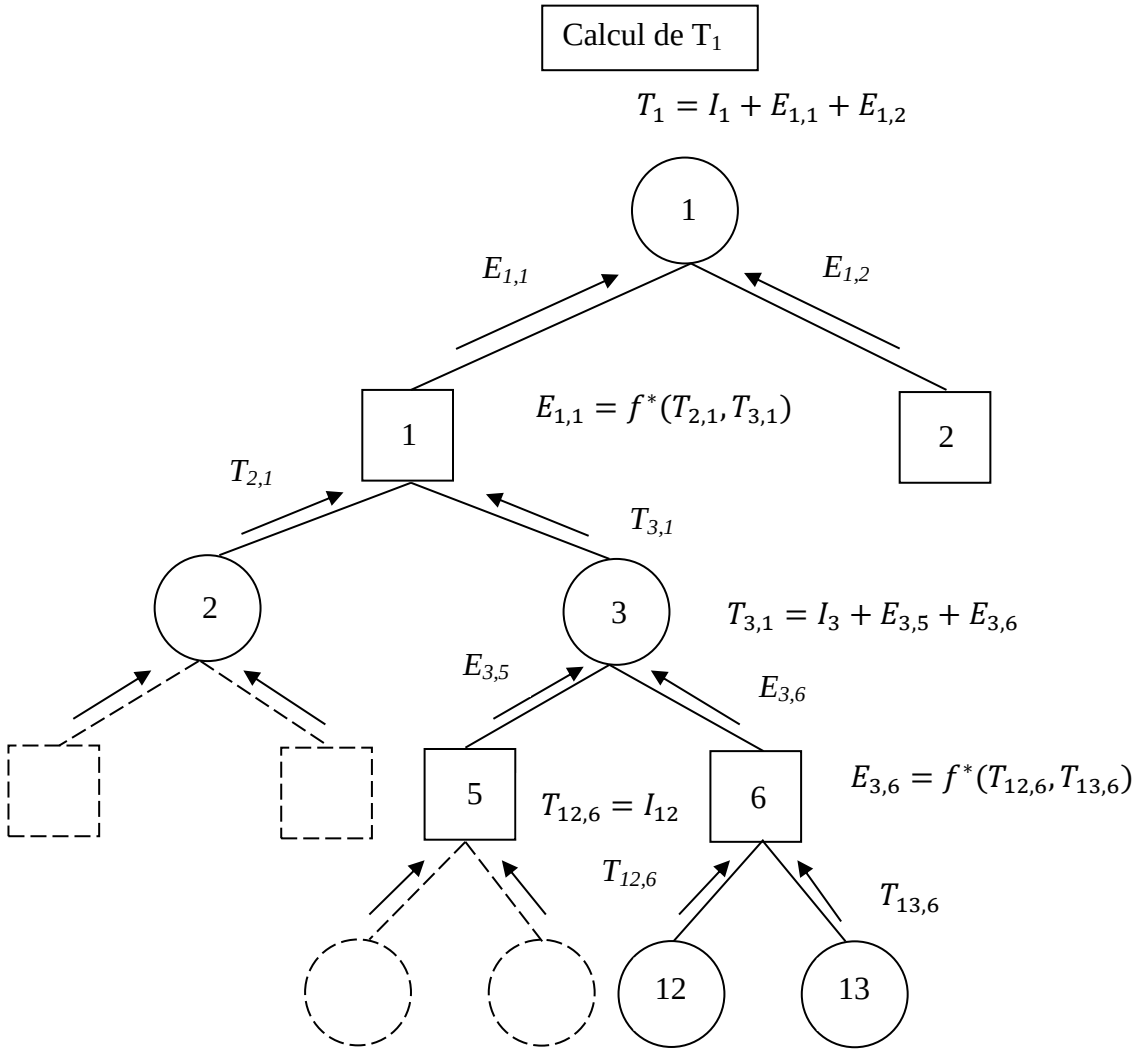


Figure 1. 11 – Les opérations de calcul de T_1 [14].

$$f^*(x) = -\log(\tanh(\frac{x}{2})) \tag{1.67}$$

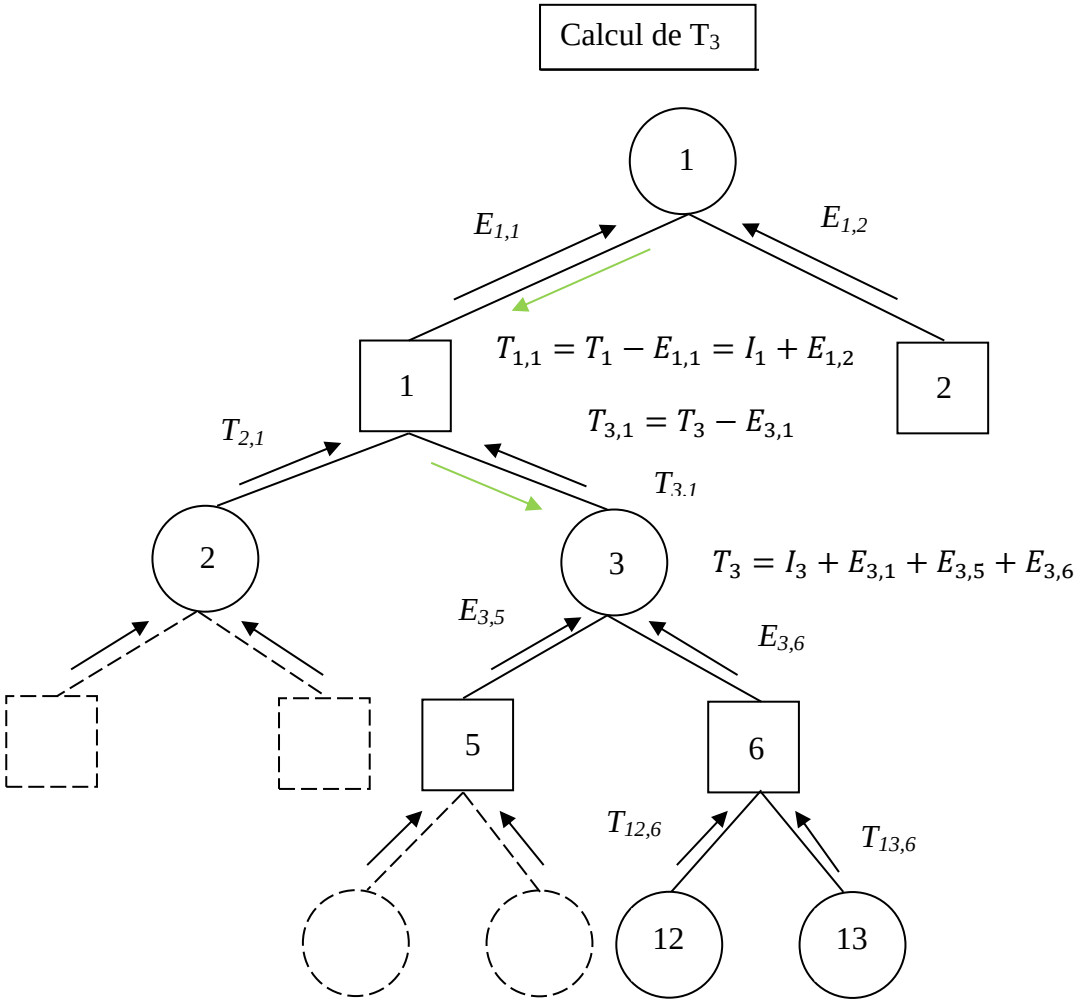


Figure 1. 12 – Les opérations de calcul de T_3 [14].

1.7.2 Les règle de mise à jour

L'information extrinsèque à l'itération *l* (voir équation 1.60) [12] [15], est exprimée comme suit :

$$E_{n,m}^{(l)} = 2\tanh^{-1}\sum_{j\epsilon N(m)\backslash n}\tanh(T^{(l-1)}{}_{c_j/r_{i\neq n}}/2) \tag{1.68}$$

Les règles de mise à jour de l'information extrinsèque sont illustrées sur la figure (1.13)

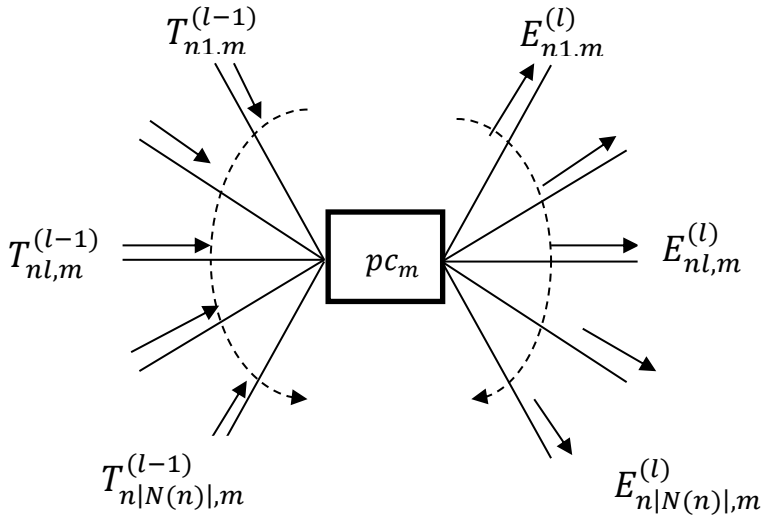


Figure 1. 13 – Mise à jour des messages : d'un nœud de variable à un nœud de contrôle [14, 15] [32].

L'information partielle [14, 15] [38] à l'itération *l*, est exprimée comme suit :

$$T_{n,m}^{(l)} = I_n + \sum_{m' \epsilon M(n) \backslash m} E_{n,m'}^{(l-1)} \tag{1.69}$$

où $m' \epsilon M(n) \backslash m$

Les règles de mise à jour de l'information partielle sont illustrées sur la figure (1.14)

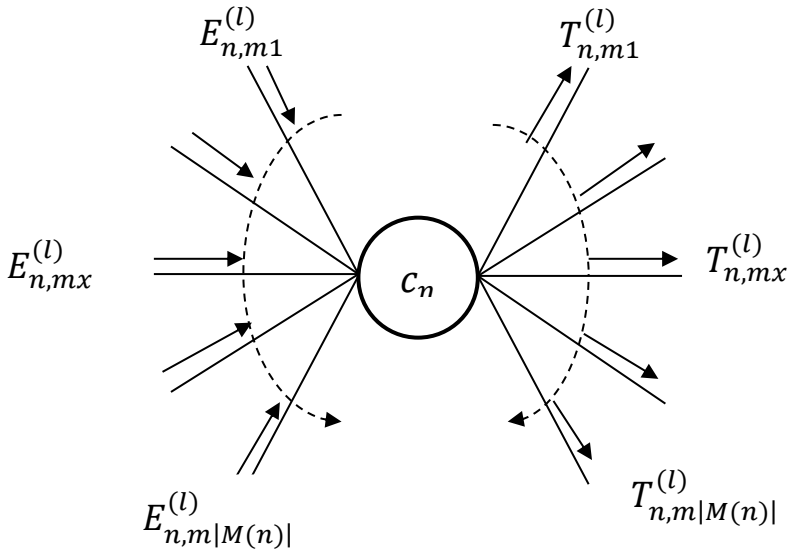


Figure 1. 14 – Mise à jour des messages : d'un nœud de contrôle à un nœud de variable [14, 15] [32].

en décomposant l'équation (1.68), on obtient [12] :

$$tanh\left(E_{n,m}^{(l)}/2\right) = \prod_{j\epsilon N(m)/n} tanh({T^{(l-1)}}_{c_j/r_{i\neq n}}/2) \tag{1.70}$$

ce qui donne :

$$sign(E_{n,m}^{(l)}) = \prod_{j\epsilon N(m)\backslash n} sign({T^{(l-1)}}_{c_j/r_{i\neq n}}) \tag{1.71}$$

$$tanh\left(\left|E_{n,m}^{(l)}\right|/2\right) = \prod_{j\epsilon N(m)\backslash n} tanh(\left|{T^{(l-1)}}_{c_j/r_{i\neq n}}\right|/2) \tag{1.72}$$

la fonction *f(x)* peut s'écrire comme suit :

$$f(x) = -\log(\tanh(\frac{x}{2})) = \log\frac{e^x+1}{e^x-1} \tag{1.73}$$

$$f\left(f(x)\right) = \log\frac{e^{\log\left(\frac{e^x+1}{e^x-1}\right)}+1}{e^{\log\left(\frac{e^x+1}{e^x-1}\right)}-1} = x \tag{1.74}$$

en prenant le logarithme de l'inverse l'équation (1.72), on obtient :

$$-\log\tanh\left(\left|E_{n,m}^{(l)}\right|/2\right) = -\log\prod_{j\epsilon N(m)\backslash n} tanh(\left|{T^{(l-1)}}_{c_j/r_{i\neq n}}\right|/2) \tag{1.75}$$

de l'équation (1.73), on obtient :

$$f\left(\left|E_{n,m}^{(l)}\right|\right) = -\sum_{j\epsilon N(m)\backslash n}\log\left(tanh(\left|{T^{(l-1)}}_{c_j/r_{i\neq n}}\right|/2)\right) \tag{1.76}$$

$$f\left(\left|E_{n,m}^{(l)}\right|\right) = \sum_{j\epsilon N(m)\backslash n} f\left(\left|{T^{(l-1)}}_{c_j/r_{i\neq n}}\right|\right) \tag{1.77}$$

comme on a : $f\left(f(x)\right) = x$ (voir équation (1.74)), on obtient :

$$\left|E_{n,m}^{(l)}\right| = f(\sum_{j\epsilon N(m)\backslash n} f\left(\left|{T^{(l-1)}}_{c_j/r_{i\neq n}}\right|\right)) \tag{1.78}$$

Les étapes de calcul de l'algorithme propagation de croyance sont résumées comme suit :

$$\begin{cases} Initialisation : \\ E_{n,m}^{(0)} = 0 \\ Mise\ à\ jour\ du\ noeud\ de\ contrôle : \\ E_{n,m}^{(l)} = \prod_{j \epsilon N(m) \backslash n} sign\left(T^{(l-1)}{}_{c_j / r_{i \neq n}}\right) \times f\left(\sum_{j \epsilon N(m) \backslash n} f\left(\left|T^{(l-1)}{}_{c_j / r_{i \neq n}}\right|\right)\right) \\ Mise\ à\ jour\ du\ noeud\ de\ variable : \\ T_{n,m}^{(l)} = I_n + \sum_{m' \epsilon M(n) \backslash m} E_{n,m'}^{(l)} \\ La\ derniere\ mise\ à\ jour\ du\ noeud\ de\ variable : \\ T_n^{(l)} = I_n + \sum_{m \epsilon M(n)} E_{n,m}^{(l)} \end{cases}$$

1.8 Conclusion

Dans ce chapitre, notions et définitions nécessaire à la compréhension des codes *LDPC* sont présentées. Aussi un bref historique et quelques propriétés sur les codes *LDPC* ont été donnés.

A propos de l'encodage des codes *LDPC,* trois approches basées respectivement sur la matrice génératrice, la forme de la matrice *H* et les contraintes de construction ont été décrites. Les deux types de décodage ont été présentés, ainsi l'algorithme propagation de croyance a été donné.

Les méthodes de construction aléatoires et déterministes feront l'objet du chapitre suivant.

Chapitre 2

Méthodes de construction des codes *LDPC*

2.1 Introduction

La construction d'un code *LDPC* doit être effectuée de manière à optimiser les performances de correction d'erreur et la complexité d'implémentation du décodeur [14].

Pour construire un code *LDPC* il faut : définir les positions de tous les éléments non nuls dans la matrice de contrôle de parité *H*. Le choix des paramètres, qui définissent ces codes, représente une difficulté principale dans la construction des codes *LDPC*.

Dans ce chapitre, certaines constructions aléatoires (de Gallager [3, 4] de Mackay [[6, 7, 8]) ainsi que certaines constructions déterministes (celle basée sur les protographes [[39, 40, 41], quasi-cyclique (*Q-C*) [42, 43, 44, 45], repeat accumulate [46, 47] et en treillis [48, 49, 50]) sont décrites en détails.

2.2 Etat de l'art sur les méthodes de construction des codes LDPC

2.2.1 Introduction

Construire un code *LDPC* revient à représenter sa matrice de contrôle de parité *H*, ce qui veut dire : définir les positions des '1' et des '0' dans cette matrice [3, 4]. Chaque construction doit prendre en considération le compromis suivant : Une matrice de contrôle de parité creuse (avoir des cycles de longueurs élevées), ce qui provoque une diminution de la distance minimale et en conséquence une diminution des performances du code. Pour avoir une distance minimale du code plus grande, la matrice de contrôle de parité doit être moins creuse, ce qui influe sur la convergence des algorithmes du décodage à cause de l'apparition des petits cycles [3, 4], [6, 7, 8].

La construction des codes *LDPC* est principalement classée en deux catégories [4], [51] : La construction aléatoire et la construction déterministe (structurée), ces deux types de constructions sont déterminés par les connexions entre les nœuds de variable et les nœuds de contrôle du graphe de Tanner. Chaque type de construction a ses avantages par rapport à l'autre. Dans la suite, plusieurs constructions sont exposées et illustrées par des exemples.

2.2.2 La construction aléatoire

2.2.2.1 Introduction

La construction aléatoire se réfère aux connexions non structurées entre les lignes et les colonnes dans la matrice de contrôle de parité [39], la conception de ces codes est réalisée par des algorithmes de recherche d'ordinateur. En fait, le processus de génération dans ce cas est un processus pseudo-aléatoire, mais il est dit aléatoire par souci de concision. La construction des codes *LDPC* aléatoire revient à construire la matrice *H* de dimension *N* par *M*, le rang de cette matrice est définit à partir des deux dimensions *N* et *K* comme suit [3, 4], [39] :

$$Rank\ (H) = N - K \tag{2.1}$$

Le rang des matrices aléatoire doit être contrôlé par la matrice *H* [8], dans le cas de la construction d'un code avec *R=1/3*, la matrice de dimension (3*x*, 2*x*), avec *x* : représente nombre entier aléatoire. La faible densité des codes *LDPC* diminue la probabilité d'avoir des lignes linéairement dépendant, surtout lorsque le poids de la colonne est impair [8]. Un autre point, sur les codes *LDPC* construits aléatoirement est que la génération de ces codes basés sur certaines règles de remplissage aléatoire de la matrice *H*, aboutit à des codes qui ont des profils d'irrégularité différents, mais ils ont à peu près les mêmes performances au niveau taux d'erreur binaire [39]. La construction des codes aléatoire se fait en ajoutant des connexions au graphe de Tanner entre les nœuds de contrôle et les nœuds de variable ou en ajoutant des '1' à la matrice de contrôle de parité *H*. La conception d'un code avec un rendement et girth souhaité, doit être faite en post-traitant les connexions de la matrice *H* pour supprimer les petits cycles (les cycles de longueur 4 dégradent les performances des codes) [3, 4], [6, 7, 8].

Les codes aléatoires ont des avantages et des inconvénients :

Les avantages [3, 4], [6, 7, 8], [52] :

-La construction aléatoire a de très bonnes performances de corrections d'erreurs.

-La simplicité de la construction.

-Les cycles n'ont pas d'importance dans cette construction.

-Les codes aléatoires (ceux de Mackay et *PEG* : Progressive Edge Growth) ont de meilleures performances par rapport aux codes structurées.

Les inconvénients [3, 4], [13, 14, 15], [39] :

-Une grande longueur du code a besoin de mémoire importante pour être stockée et utilisée dans le décodage et l'encodage, ce qui influe sur le temps de calcul, considéré comme plus important que celui du taux d'erreur.

-L'influence des cycles de petite longueur sur les performances du code.

-Les contraintes ajoutées pour construire des matrices de contrôle de parité rendre la construction et le codage plus complexe.

Parmi les codes aléatoires qui s'approchent de la limite de Shannon [1], on peut citer les codes issus de la construction de Gallager [3, 4] et ceux issus de la construction de Mackay [6, 7, 8].

2.2.2.2 La construction de Gallager

Dans ses premiers travaux [3, 4], Gallager a décrit le moyen de construire la matrice *H* des codes *LDPC* réguliers notés par (*N, Wc, Wr*).

où *N*, *Wr* et *Wc* représentent respectivement le nombre de colonne, le poids de lignes et celui de colonnes. La matrice est composée de deux parties : la première partie est structurée et l'autre partie est aléatoire. Le terme code de Gallager fait référence à un code qui correspond à l'équation (2.2) :

$$N \times W_r = M \times W_c \tag{2.2}$$

où *M*, représente le nombre de ligne. La structure de Gallager [3, 4] est donnée comme suit :

$$H = \begin{bmatrix} H_1 \\ \pi_1(H_1) \\ \vdots \\ \pi_i(H_1) \end{bmatrix} \tag{2.3}$$

où $\pi_i(H_1)$ représente la version permutée de la matrice H_1.

La construction de la matrice *H* se fait de la manière suivante :

-Construction de la sous matrice de permutation H_1 de dimensions $N \times (N/ Wr)$: telle que les *Wr* premières positions de la première ligne sont à '1' et les autres sont à '0'. Les lignes restantes sont obtenues par décalage (d_k) de la première ligne, comme suit :

$$D_k = k \times Wr, \qquad k = 1, \cdots, M-1 \tag{2.4}$$

-Les autres sous matrices $\pi_i(H_1)$ sont créés par des permutations aléatoires de la sous matrice H_1.

Ci-dessous une matrice *H* de Gallager de longueur (12, 3, 4) avec *Wr*=4 et *Wc*=3 :

$$H = \left[\begin{array}{cccccccccccc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ \hline 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ \hline 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}\right]$$

2.2.2.3 La construction de Mackay

Mackay a été le premier qui a introduit l'avantage de la construction des codes *LDPC* avec une matrice *H* creuse et il a montré que ces codes s'approchent beaucoup de la limite de Shannon [8], [52]. Mackay a introduit un ensemble de codes *LDPC* aléatoires (dénommé 'ensemble de matrices très creuses') dont, il a supposé que leur décodage diminuera leur probabilité moyenne d'erreur, mais sans qu'il donne une preuve à sa supposition. L'ensemble est définit comme suit [8] :

1. La matrice *H* est générée dont les bits '1' ne sont pas à la même position dans les colonnes.
2. La matrice *H* est créée en générant, d'une façon aléatoire, des colonnes de poids *Wc* et un poids de ligne uniforme *Wr*.
3. La matrice *H* est générée avec un poids de colonne *Wc* et un poids de ligne *Wr* uniforme tout en évitant le chevauchement de colonnes.
4. La matrice est construite selon des contraintes qui permettent de trouver un graphe de Tanner correspondant à un girth plus que 6.
5. La forme de la matrice est : $H = [C_1\ C_2]$ où C_2 est une matrice inversible.

2.2.3 La construction déterministe

2.2.3.1 Introduction

L'incertitude de garantir une performance optimale dans des constructions aléatoires, a conduit à l'utilisation de la construction des codes structurés. Les inconvénients des codes aléatoires ont conduit à la nécessité d'imposer une certaine régularité dans la construction de la matrice *H* [39]. Dans les constructions déterministes, les connexions lignes-colonnes sont

définies suivants des contraintes. De nombreuses constructions ont été développées, on peut citer les codes (basés sur les protographes [39, 40, 41], quasi-cyclique [42, 43, 44, 45], repeat accumulate [46, 47] et en treillis [48, 49, 50]).

2.2.3.2 La construction basée sur les protographes

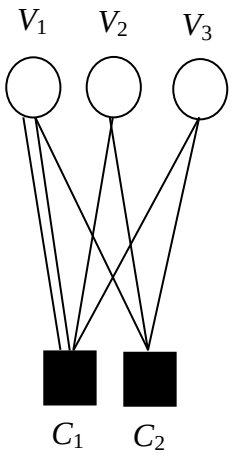
La construction d'un protographe d'un code *LDPC* [40, 41] débute à partir d'un petit graphe biparti, qui peut inclure des bords (connexions) parallèles. Le petit graphe biparti est utilisé pour obtenir un grand graphe de Tanner par une procédure ‘’copier et permuter’’. Le protographe est copié Q fois puis les bords sont copiés et connectés entre les copies du protographe sous certaines contraintes pour obtenir un graphe de Tanner large, correspond à une nouvelle matrice de contrôle de parité H. Si un nœud de variable V_{nn} du protographe est connecté à un nœud de contrôle C_{nm} du protographe, alors le nœud de variable V_{nn} ne peut être connecté qu'un seul nœud de contrôle C_{nm} [40, 41]. La construction élimine les bords parallèles du protographe principal en raison de permutation [40]. Ci-dessous un exemple de construction d'un code *LDPC* basé sur les protographes.

Soit la matrice de contrôle de parité H suivante :

Un exemple de construction d'un code *LDPC* est donné à la figure (2.1) en utilisant un protographe de matrice B définit comme suit :

$$B = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

(a)



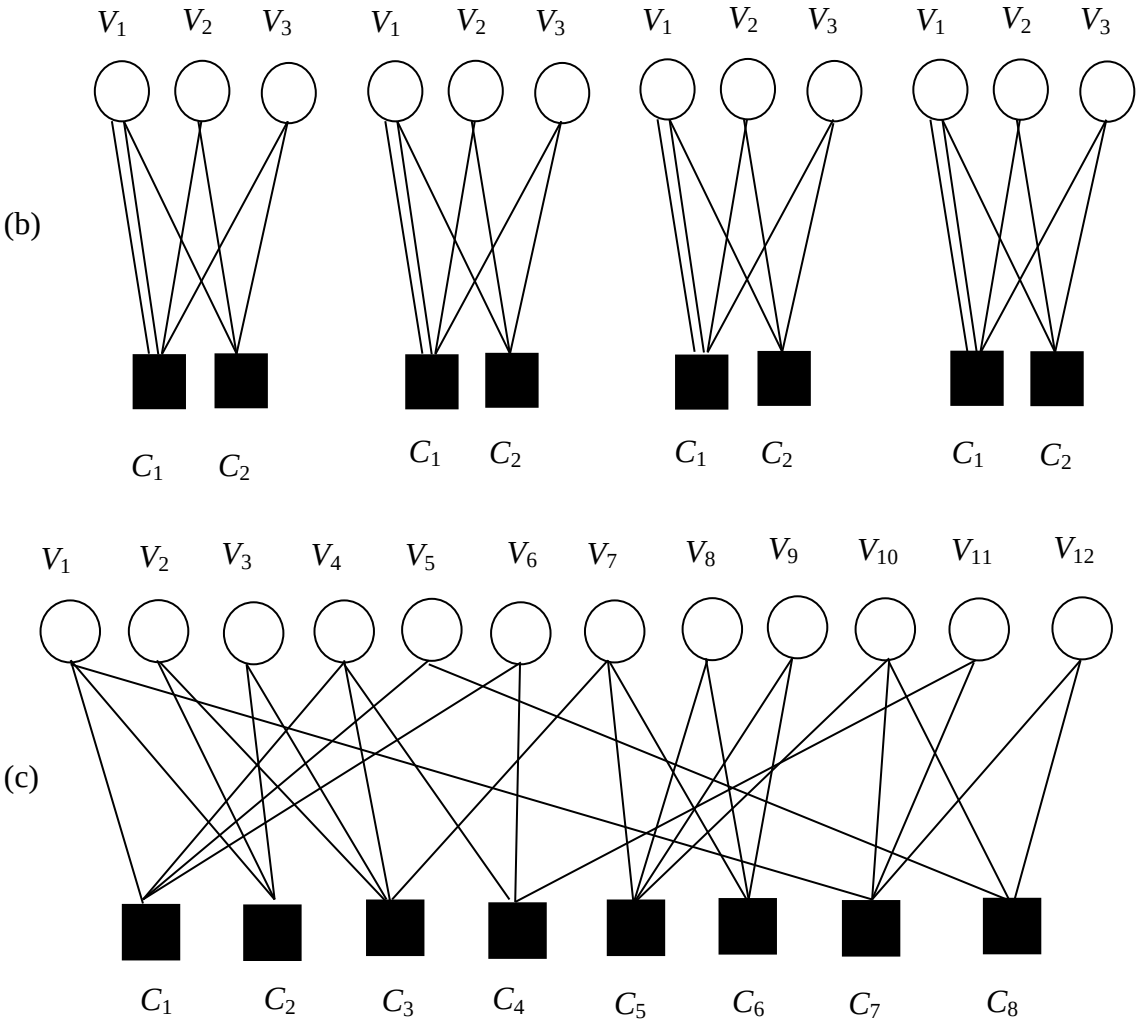


Figure 2. 1 – Illustration de la procédure copier et permuter avec Q=4 copies [40] : (a) le protographe de la matrice B ; (b) le protographe 'a' copié 4 fois ; (c) le protographe 'b' permuté.

La figure 2.1, montre un protographe copié 4 fois. Le graphe de Tanner obtenu est appelé graphe dérivé et correspond à un code *LDPC* protographique de longueur N=12, K=4 et R=1/4. Ces codes, basés sur les protographes, ont de nombreuses approches et méthodes pour optimiser leurs constructions [53, 54].

2.2.3.3 La construction quasi-cyclique

Par rapport aux codes *LDPC* construits de manière aléatoire, la matrice de parité d'un code *QC-LDPC* contient plusieurs sous-matrices de permutation ou sous-matrices nulles.

Ainsi, la mémoire de stockage peut être réduite considérablement (il suffit de mémoriser les premières lignes de chaque sous-matrice : les autres lignes sont formées par permutations circulaires). Cette structure est orientée sur la réalisation d'un décodeur plus efficace [55]. Il a été démontré, que le girth d'un code *QC-LDPC* a une limite inférieure définie par les positions des matrices de permutations circulaires [56], [21]. Ce qui permet de contrôler la longueur des cycles. La matrice de contrôle de parité, d'un code *LDPC* construit par des sous matrices de permutation [42, 43, 44, 45], peut être présentée comme suit :

$$H = \begin{bmatrix} Ip_{1,1} & Ip_{1,2} & \cdots & Ip_{1,L} \\ Ip_{2,1} & Ip_{2,2} & \cdots & Ip_{2,L} \\ \vdots & \vdots & \ddots & \vdots \\ Ip_{J,1} & Ip_{J,2} & \cdots & Ip_{J,L} \end{bmatrix} \tag{2.5}$$

où *Ix* représente une matrice identité de taille $z \times z$ circulairement permutée de x positions vers la droite. Le code ainsi définit est un code régulier de taille $N = L \times z$ et de rendement $R = 1- J/L$. Les coefficients de permutation $p_{j,l}$ de chaque sous matrice identité peuvent être construit selon des règles [57]. Un exemple de matrice *QC-LDPC* de longueur (12,9) avec des sous matrices de taille 3× 3 est donné comme suit :

$$H = \left[\begin{array}{ccc|ccc|ccc|ccc} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right]$$

2.2.3.4 Construction de type Repeat-Accumulate

La construction de cette famille de codes consiste à diviser la matrice de contrôle de parité *H* en deux parties, dont une partie est de type double diagonale. La matrice de contrôle de parité *H* peut s'écrire :

$$H = \begin{bmatrix} H_s & H_p \end{bmatrix} \tag{2.6}$$

Il existe deux types de codes repeat accumulate :

1. **Regular repeat accumulate (*RA*)** : c'est la famille dans laquelle, la sous matrice H_p est strictement double diagonale (voir équation 2.7) et la matrice *Hs* est régulière [47]. Des nombreux travaux ont portés sur l'étude de ces codes et des performances comparables à celles des codes *LDPC* irréguliers ont été obtenues [46].

$$H_p = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 1 & \ddots & 0 \\ 0 & 0 & 0 & \ddots & \ddots & 0 \\ \vdots & \vdots & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

2. **Irregular repeat accumulate (*IRA*) :** c'est le cas où la matrice H_p est strictement double diagonale et la matrice *Hs* est irrégulière [46]. La norme *DVB-S2* appartient à cette catégorie de codes *LDPC* irréguliers standardisés. Plusieurs méthodes de conception ont été proposées pour construire la matrice *Hs* [33], [57, 58]. Ces méthodes permettent une construction et un encodage très simple et un décodage moins complexe.

2.2.3.5 La construction en treillis

Dans cette section, la construction des codes *LDPC* avec de grande longueur de blocs (blocs représentent les colonnes) est abordée [48, 49, 50]. Les blocs sont définis comme des connexions entre les différentes pentes '*s*' du treillis. Suivant des indications dans [48, 49, 50], ces configurations sont simples et flexibles pour construire des codes à large girth. La matrice *H* est définie à partir d'une grille rectangulaire (un plan à 2 dimensions (*d*,*m*)). où m^2, *d* et $m \times d$ représentent respectivement le nombre de blocs, un nombre premier strictement inférieur à *m* et le nombre de lignes.

Une ligne dans la grille rectangulaire est une connexion de points, spécifiée par sa pente s $(0 \leq s \leq m-1)$ et le point de départ qui a les coordonnées$(0, a)$, contient tous les points qui ont les coordonnées suivantes :

$$\{(x, a + sx \bmod m) : 0 \leq x \leq d-1, 0 \leq a \leq m-1\} \tag{2.8}$$

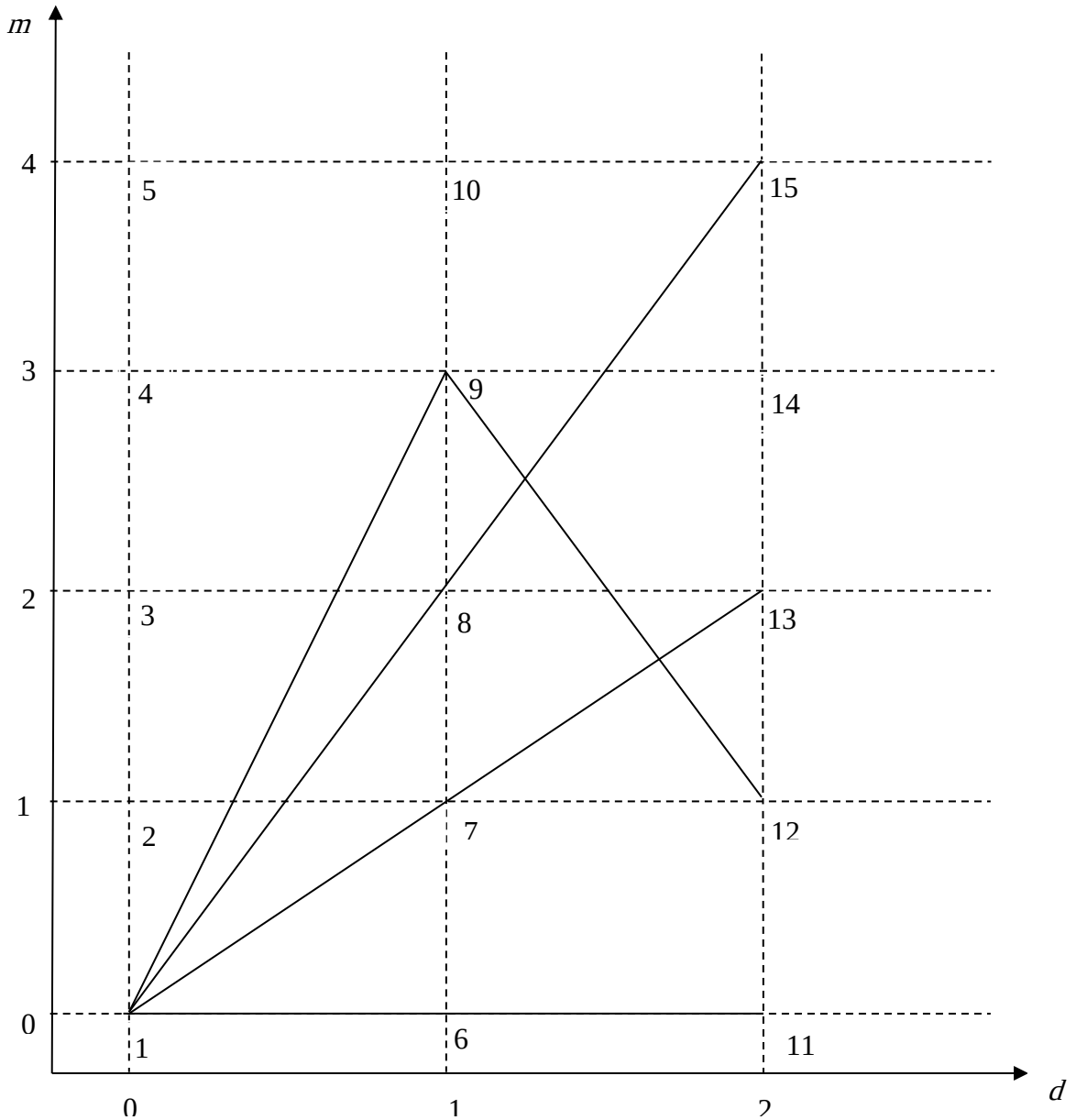


Figure 2. 2 – Exemple d'une grille rectangulaire : m=5, d=3 et s = 0, 1, 2, 3 et 4 [48, 49, 50].

Les positions des éléments non nuls sont comme suit :

pour s = 0 :

*{{*1, 6, 11*}*, *{*2, 7, 12*}*, *{*3, 8, 13*}*, *{*4, 9, 14*}*, *{*5, 10, 15*}}*

pour s = 1 :

*{{*1, 7, 13*}*,*{*2, 8, 14*}*, *{*3, 9, 15*}*, *{*4, 10, 11*}*, *{*5, 6, 12*}}*

pour $s = 2$:

*{{*1, 8, 15*}*,*{*2, 9, 11*}*, *{*3, 10, 12*}*, *{*4, 6, 13*}*, *{*5, 7, 14*}}*

pour $s = 3$:

*{{*1, 9, 12*}*,*{*2, 10, 13*}*, *{*3, 6, 14*}*, *{*4, 7, 15*}*, *{*5, 8, 11*}}*

et pour $s = 4$:

*{{*1, 10, 14*}*, *{*2, 6, 15*}*, *{*3, 7, 11*}*, *{*4, 8, 12*}*, *{*5, 9, 13*}}*

La matrice de contrôle de parité H correspondante aux positions ci-dessus est comme suit :

$$H = \left[\begin{array}{ccccc|ccccc|ccccc|ccccc|ccccc}
1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0\\
0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0\\
0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0\\
0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1\\
\hline
1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0\\
0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&1\\
0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0\\
\hline
1&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0&0&1&0&0\\
0&1&0&0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&1&0&0&1&0&0&0&0&0&0&0&1&0&0&1&0&0&0&0&0&0&0&1\\
0&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&1&0&0&1&0&0&0&0&0&0&0&1&0&0&1&0&0&0
\end{array}\right]$$

2.3 Conclusion

Dans ce chapitre, un état de l'art sur les méthodes de construction des codes *LDPC* a été présenté. Les avantages et les inconvénients des deux types de constructions aléatoire et déterministe ont été donnés.

Certaines constructions aléatoires (celle de Gallager et de Mackay) et déterministes (celle basée sur les protographes, quasi-cyclique, repeat accumulate et en treillis) ont été présentées et illustrées par des exemples.

Deux nouvelles méthodes de construction des codes *LDPC* binaires, à large girth, combinant un décodage moins complexe et un encodage linéaire feront l'objet du chapitre suivant.

Chapitre 3

Nouvelles méthodes de construction de codes *LDPC* irréguliers

3.1 Introduction

Dans ce chapitre, deux méthodes déterministes de construction des codes *LDPC* sont proposées. La première est basée sur l'utilisation d'une matrice de Hankel (matrice structurée) combinée avec une matrice double diagonale, quant à la deuxième méthode, elle utilise des sous matrices de permutation combinées avec une matrice double diagonale pour construire la matrice de contrôle de parité *H*.

Le girth est un élément clé qui influe sur les performances du code *LDPC* [4], [59]. Plusieurs recherches ont indiqué que les petits cycles augmentent la complexité de décodage [6, 7, 8], en conséquence les travaux sur la construction des codes *LDPC* avec un girth relativement large (plus de 4) sont toujours d'actualités [60, 61].

3.2 Les matrices structurées

Une matrice A de taille $n \times n$ est dite structurée si [62] :

1. Elle dépend d'un nombre moins de n^2 éléments.
2. Ses éléments sont distribués d'une manière simple.

Parmi les matrices structurées, on a :

3.2.1 Matrice de Toeplitz

T est une matrice de Toeplitz. Si les éléments situés à l'intersection des ligne i et colonnes j de la matrice T, notés $T_{i,j}$, alors on a :

$$T_{i,j} = t_{i-j} \tag{3.1}$$

La structure particulière de la matrice de taille $n \times n$ est obtenue à partir de l'équation (3.1) comme suit :

$$T = \begin{bmatrix} t_0 & t_{-1} & t_{-2} & \cdots & \cdots & t_{-n+1} \\ t_1 & t_0 & t_{-1} & \ddots & & \vdots \\ t_2 & t_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & t_1 & t_0 & t_{-1} \\ t_{n-1} & \cdots & \cdots & t_2 & t_1 & t_0 \end{bmatrix} \tag{3.2}$$

3.2.2 Matrice de Hankel

B est une matrice de Hankel dont ses éléments (entiers positif), notés $B_{i,j}$, sont définis selon l'équation (3.3)

$$B_{i,j} = b_{i+j} \tag{3.3}$$

La structure particulière de cette matrice *B* de taille $n \times n$ est obtenue à partir de l'équation (3.3) comme suit :

$$B = \begin{bmatrix} b_1 & b_2 & \cdots & b_n \\ b_2 & b_3 & \cdots & \vdots \\ \vdots & \vdots & \cdots & \vdots \\ b_n & b_{n+1} & \cdots & b_{2n-1} \end{bmatrix} \tag{3.4}$$

3.2.3 Matrice de Vandermonde

V est une matrice de Vandermonde si ses éléments, notés $V_{i,j}$, sont définis selon l'équation (3.5)

$$V_{i,j} = \left(\alpha_i^{j-1}\right) \tag{3.5}$$

La structure particulière de cette matrice *V* de taille *m*×*n* est obtenue à partir de l'équation (3.5) comme suit :

$$V = \begin{bmatrix} 1 & \alpha_1 & \cdots & \alpha_1^{n-1} \\ 1 & \alpha_2 & \cdots & \alpha_2^{n-1} \\ \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha_m & \cdots & \alpha_m^{n-1} \end{bmatrix} \tag{3.6}$$

3.2.4 Matrice de Cauchy

C est dite matrice de Cauchy si ses éléments, notés $C_{i,j}$, sont définis selon l'équation (3.7)

$$C = \left(\frac{1}{x_i - y_j}\right) \tag{3.7}$$

où les complexes $x_i - y_j \neq 0$

La structure particulière de cette matrice *C* de taille *n*×*n* est obtenue à partir de l'équation (3.7) comme suit :

$$C = \begin{bmatrix} \frac{1}{x_1 - y_1} & \cdots & \frac{1}{x_1 - y_n} \\ \vdots & & \vdots \\ \frac{1}{x_n - y_1} & \cdots & \frac{1}{x_n - y_n} \end{bmatrix} \tag{3.8}$$

3.3 Première méthode

3.3.1 Introduction

Cette méthode est proposée en se basant sur le constat de Mackay [8], concernant l'obtention d'une matrice *H* très creuse (voir construction de Mackay), qui stipule que : pour avoir une matrice *H* très creuse, il faut éviter le chevauchement de position des '1' dans les colonnes qui a pour conséquence l'élimination des cycles de longueur 4.

Soit *H,* une matrice de contrôle de parité de taille *M* par *N*, où *M* : le nombre de lignes et *N* le nombre de colonnes. La matrice *H* est représentée comme suit :

$$H = [H_1 H_2] \tag{3.9}$$

où H_2 est une matrice double diagonal, de taille *M* par *M*, la structure de cette dernière permet de calculer les bits de contrôle de parité avec une complexité linéaire [63, 64]. H_1 est une matrice carrée creuse de taille *M* par *M* obtenue après plusieurs traitement de la matrice de Hankel [65] afin d'éviter la redondance des colonnes dans la matrice *H*. La matrice H_1 est construite en deux étapes comme suit :

3.3.2 Etape 1

On utilise la matrice de Hankel [65] de taille $m \times m$ où m est considéré comme un nombre pair strictement supérieur à 3, où ses éléments (coefficients) sont définis comme suit :

$$a_{i,j} = i + j - 1 \tag{3.10}$$

où $1 \leq i \leq m, 1 \leq j \leq m$ représentent respectivement l'indice de la ligne et l'indice de la colonne.

La matrice de Hankel de taille $m \times m$ est représentée comme suit :

$$H_a = \begin{bmatrix} 1 & 2 & \dots & m \\ 2 & 3 & \dots & m+1 \\ \vdots & \vdots & \dots & \vdots \\ m & m+1 & \dots & 2m-1 \end{bmatrix} \tag{3.11}$$

On construit une matrice symétrique, notée H_b [66], dont les éléments $a_{i,j}$ sont définies selon l'équation (3.12) :

$$a_{i,j} = \begin{cases} a_{i,j} de\ H_a & \text{si } i + j \leq m + 1 \\ a_{(m+1)-i,(m+1)-j} & \text{sinon} \end{cases} \tag{3.12}$$

La matrice H_b obtenue est :

$$H_b = \begin{bmatrix} 1 & 2 & \dots & m \\ 2 & 3 & \dots & m-1 \\ \vdots & \vdots & \dots & \vdots \\ m & m-1 & \dots & 1 \end{bmatrix} \tag{3.13}$$

3.3.3 Etape 2

On considère chaque élément de la matrice H_b comme un indice. Pour éviter le chevauchement de positions des '1' dans les colonnes, il est nécessaire d'activer seulement deux indices *x* et *y* selon l'équation (3.14) et les autres indices sont à zéro [66] :

$$x + y = m \tag{3.14}$$

où $x \neq y \neq m$

A chaque indice activé (*x* et *y*) est assigné une sous matrice de permutation de taille (*m*-1) par (*m*-1). La matrice obtenue, notée H_1, est de taille $m\times (m\text{-}1)$ par $m\times (m\text{-}1)$ où $m\times(m\text{-}1)$ doit être égale à *M*. Finalement, la matrice *H* est construite par une simple concaténation de H_1 avec H_2 selon l'équation (3.9) [66].

Exemple

Etape 1 :

Soit la matrice de Hankel H_a de taille 4×4 est représentée comme suit :

$$H_a = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \end{bmatrix}$$

La matrice symétrique H_b est construite à partir de la matrice H_a selon l'équation (3.13) :

$$H_b = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 3 \\ 3 & 4 & 3 & 2 \\ 4 & 3 & 2 & 1 \end{bmatrix}$$

Etape 2 :

Les sous matrices de permutation, de taille 3×3, sont données comme suit :

La sous matrice associée à l'indice '1' est une sous matrice d'identité :

$$\text{'1'} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

La sous matrice associée à l'indice '2' est une sous matrice d'identité permuté une fois vers la droite :

$$‘2’ = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

La sous matrice associée à l’indice ‘3’ est une sous matrice d’identité permuté deux fois vers la droite :

$$‘3’ = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

La sous matrice associé à l’indice ‘4’ est une sous matrice zéro :

$$‘4’ = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Si on remplace tous les indices par leurs sous matrices correspondantes, la matrice H_1 contient plusieurs cycles de longueur 4. Cette dernière est obtenue comme suit :

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

En effet, la matrice H_1 possède plusieurs colonnes présentant des chevauchements de position des ‘1’ supérieur à 1 ce qui confirme l‘existence des cycles de longueur 4. Ces cycles sont mise en œuvre par le graphe de Tanner de la figure (3.1)

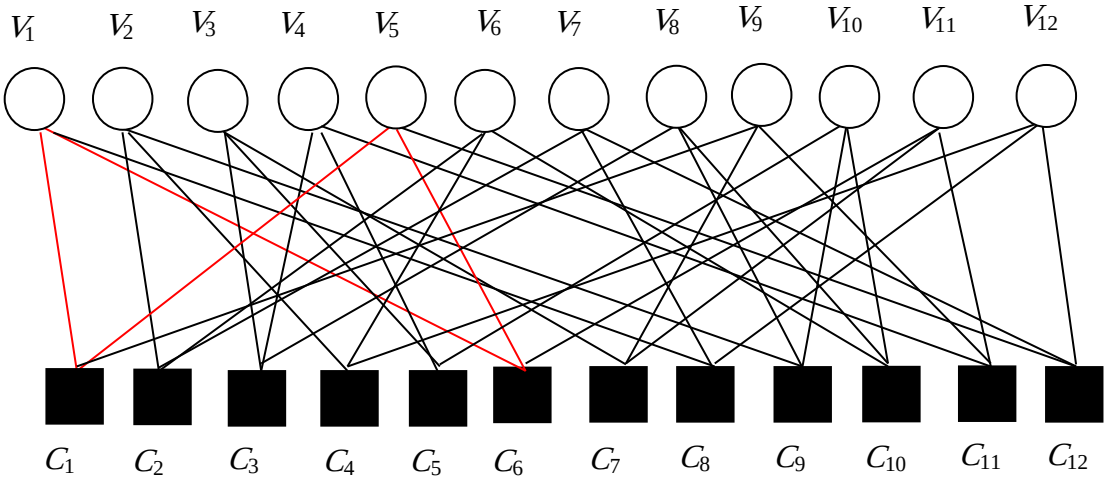


Figure 3. 1 – Graphe de Tanner de la matrice H_1 (12, 3, 3).

Afin d'éviter les cycles de longueur 4, on utilise le principe d'activation selon l'équation (3.14) (dans notre cas x=1 et y=3). La matrice H_1, dans le présent exemple, est obtenue à partir de la matrice H_b en activant seulement la sous matrice associé à l'indice '1' et la sous matrice associé à l'indice '3', une sous matrice zéro est associé aux indices non activés :

$$H_1 = \begin{bmatrix} \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 \\ \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & \mathbf{1} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{1} & 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{1} \end{bmatrix}$$

Ainsi la matrice H, concaténation de H_1 et H_2, est :

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

La figure (3.2) montre le graphe de Tanner de la matrice de contrôle de parité *H* exempté de cycles de longueur 4.

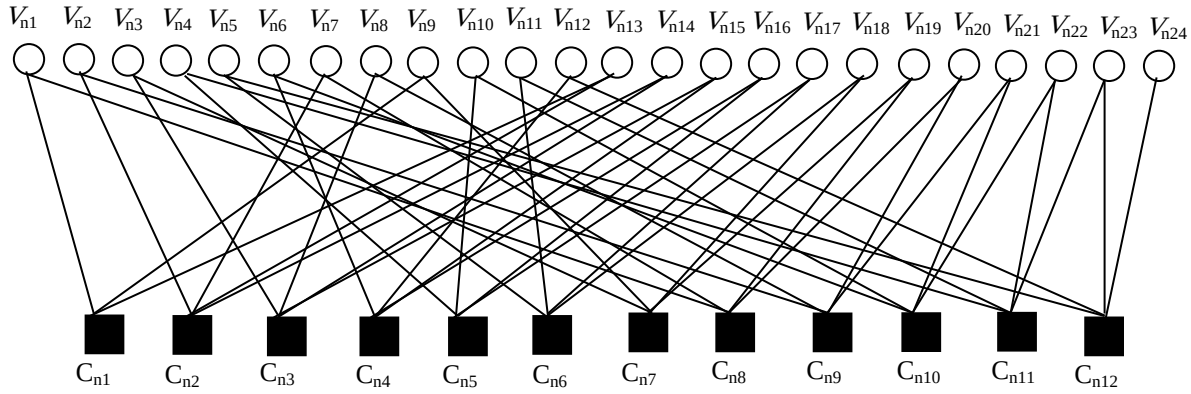


Figure 3. 2 – Graphe de Tanner de la matrice *H* issue de la première méthode.

3.4 Deuxième méthode

3.4.1 Introduction

L'objectif de cette deuxième méthode est la construction de codes *LDPC* exempte de cycles de longueur 4 et à rendement variable. La matrice de contrôle de parité *H* de ces codes s'écrit :

$$H = [H_1 \;\; H_2] \tag{3.20}$$

où H_1 est une matrice de taille $M\times(N-M)$ et H_2 est une matrice double diagonale de taille $M\times M$. La construction de H se fait en deux étapes [67]:

3.4.2 Etape 1

A partir d'une matrice d'identité I de taille $m\times m$ $(m>2)$, d'éléments notés $I_{i,j}$ on construit la matrice S, de taille $m\times m$, dont les éléments sont calculés selon l'équation (3.21) comme suit :

$$S_{i,j} = I_{(m+1-i),j} \tag{3.21}$$

3.4.3 Etape 2

On permute la matrice S (permutation de droite vers gauche) par n positions, où $n<m$. Les matrices obtenues après permutation sont notées $Sp_{(n)}$. La matrice H_1 est obtenue par concaténation des matrices d'identité I et des matrices $Sp_{(n)}$ selon l'équation (3.22) :

$$H_1 = \begin{bmatrix} I & I & \cdots & I \\ S_{p(1)} & S_{p(2)} & \cdots & S_{p(n)} \end{bmatrix} \tag{3.22}$$

La matrice H_1 est de taille $2m$ par $(n\times m)$. La matrice H est construite en concaténant les deux matrices H_1 et H_2 selon l'équation (3.20).

Exemple :

Etape 1

Soit la matrice d'identité I de taille 3×3, donnée comme suit :

$$I_{3x3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Par conséquent, la matrice S est obtenue selon l'équation (3.21) :

$$S_{3x3} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

Etape 2

Les deux versions '$Sp_{(1)}$'et '$Sp_{(2)}$' obtenues à partir de la matrice S sont comme suit :

$$S_{p(1)} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$S_{p(2)} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

La matrice H_1 de taille 6×6 est obtenue selon l'équation (3.22) comme suit :

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

Ainsi, la matrice H est de taille 6×12, concaténation de H_1 et H_2, est :

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

Le tracé du graphe de Tanner de la figure (3.3) montre l'inexistence des cycles de longueur 4.

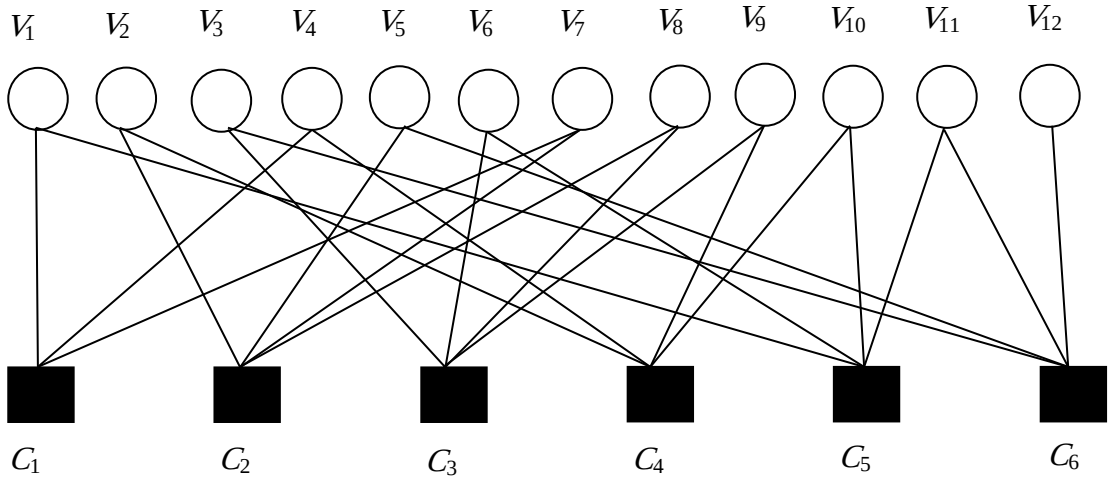


Figure 3. 3 – Graphe de Tanner de la matrice H issue de la deuxième méthode.

Il est à noter, qu'une procédure analytique de détection des cycles [68, 69, 70], a été appliquée sur les codes issus des deux méthodes de construction (voir Annexe D). Aucun cycle de longueur 4 n'a été détecté.

3.4.4 Codes LDPC irréguliers pour différents rendements

Le rendement d'un code R est définit comme $R = {}^{K}/_{N}$ où K et N représentent respectivement la longueur du mot d'information d et du mot de code c, le nombre des bits redondants est définit par $M = N - K$. Dans le cas de notre construction : $N = m(n+2)$, $M = 2 \times m$ et $K = n \times m$. On déduit que $R = \frac{n}{n+2}$. Afin d'obtenir différents rendements, la matrice H_1 doit être construite en utilisant les conditions suivantes :

1. n doit être strictement inférieur à m (pour éviter le chevauchement des positions des '1' dans les colonnes).
2. N doit diviser (n+2) (sachant que m est entier).

Le tableau 3.1 illustre les paramètres de ces codes *LDPC* proposés pour différents rendements.

n	$R=\frac{n}{n+2}$	m	$N=m(n+2)$	$M=2\times m$	$K=n\times m$
1	1/3	10…80…576…	30…240…1728…	20…160…1152…	10…80…576…
2	½	10…80…576…	40…320…2304…	20…160…1152…	20…160…1152…
4	2/3	10…80…576…	60…480…3456…	20…160…1152…	40…320…2304…
6	¾	10…80…576…	80…640…4608…	20…160…1152…	60…480…3456…
8	4/5	10…80…576…	100…800…5760…	20…160…1152…	80…640…4608…

Tableau 3. 1 – Codes *LDPC* irréguliers pour différentes rendements.

3.5 Les avantages des méthodes proposées

Les codes *LDPC* sont des codes en bloc linéaire, par conséquent ils sont exprimés comme l'espace nul de la matrice de contrôle de parité H [3], [71] c'est-à-dire que c est un mot de code si et seulement si

$$Hc^T = 0^T \tag{3.23}$$

Le mot de code binaire *c*, s'écrit: $c = [d \quad p]$ où *d* et *p* représentent respectivement les bits d'information et les bits de contrôle de parité. L'encodage des codes *LDPC* est exprimé comme suit :

$$(H_1 \quad H_2)\begin{pmatrix} d \\ p \end{pmatrix} = 0^T \tag{3.24}$$

où comme dans [72].

$$(H_2 \quad H_1)\begin{pmatrix} p \\ d \end{pmatrix} = 0 \tag{3.25}$$

où H_2 est une matrice double diagonale et H_1 est une matrice construite d'une façon déterministe. Basé sur la structure de H_2 et l'équation (3.25), les bits de contrôle de parité $p = \{p_i\}$ peuvent se calculer facilement à partir des bits d'information $d = \{d_j\}$ [72] (voir annexe B).

$$p_1 = \sum_j h_{1j}^1 \, d_j \tag{3.26}$$

et

$$p_i = p_{i-1} + \sum_j h_{ij}^1 \, d_j mod\ (2) \tag{3.27}$$

où h_{ij}^1 sont les éléments de H_1.

Dans [72], une étude comparative entre les codes *LDPC* présentés par Ping et ceux présentés par Mackay [6, 72] a montré que si la matrice de contrôle de parité *H* est décomposée en $[H_2 \quad H_1]$, avec H_2 double diagonale, plusieurs avantages sont obtenus :

1. Une rapidité et une simplicité de l'encodage (par rapport aux méthodes d'encodage existantes).
2. H_2 est non singulière
3. Si H_1 est creuse, un faible espace de mémoire est utilisé pour stocker la matrice *H* dans l'encodeur.

En plus des avantages cités ci-dessus, les méthodes proposées garantissent d'obtenir les matrices H_1 de façon très creuse.

4. Vu leurs structures, l'espace de stockage est réduit considérablement (il suffit de mémoriser une ligne et les autres sont obtenues par permutation circulaires de la première ligne).
5. Absence des cycles de longueur 4, évitant ainsi l'apparition du phénomène dit de plancher d'erreur.

3.6 Complexité de décodage

La complexité du décodage des codes *LDPC* est directement liée au nombre de branches '*Br*' dans le graphe de Tanner ou aux nombres de '1' dans la matrice de contrôle de parité [13]. L'algorithme de décodage itératif 'propagation de croyance' comporte plusieurs étapes. À chaque étape, nous devons calculer l'information extrinsèque $E_{n,m}$ et totale $T_{n,m}$ qui sont associées au nœud correspondant. Notons *Br* le nombre de branches dans le graphe de Tanner. Par exemple, dans le cas d'un code régulier (N, W_c, W_r) le nombre de branches *Br* est donné par :

$$Br = W_c \times N = W_r \times M \tag{3.28}$$

Le tableau 3.2 compare le nombre de branches des codes *LDPC* proposés avec deux autres codes : les codes de Gallager et ceux de Mackay.

Longueur de blocs	Codes *LDPC* issus de la première méthode	Codes *LDPC* issus de la deuxième méthode	Codes de Gallager [3]	Codes de Mackay [6-8], [14]
N=480 et *M*=240	*Br* = 959	*Br* = 959	*Br* = 1440	*Br* = 1440
N=1000 et *M*=500	*Br* = 1999	*Br* = 1999	*Br* = 3000	*Br* = 3000

Tableau 3. 2 – Représentation du nombre de branches des codes *LDPC* proposés avec codes de Gallager et ceux de Mackay.

A partir du tableau 3.2, on peut constater que les codes *LDPC* proposés, issue des deux méthodes, ont un nombre réduit de branches par rapport aux codes de Gallager et ceux de Mackay, qui veut dire que les matrices *H* proposées sont vraiment creuse (nombre réduit de '1' par rapport à zéro), ce qui réduit la complexité de décodage.

3.7 Conclusion

Dans ce chapitre, nous faisons une brève description des matrices particulières dénommées matrices structurées. Deux contributions sur la construction des codes *LDPC* binaires ont été proposées et illustrées par des exemples. Notre première contribution est une méthode de construction des codes *LDPC* à large girth basée sur une concaténation d'une matrice structurée, matrice de Hankel, et d'une matrice double diagonale. Notre deuxième contribution est une autre méthode de construction des codes *LDPC* basée sur une concaténation d'une matrice, sous forme de plusieurs sous matrices de permutations, et une matrice double diagonale. Ainsi, la complexité des codes proposés a été calculée et comparée à d'autres codes. L'utilisation des codes *LDPC*, issue de la deuxième méthode, permet d'obtenir des rendements variables. L'évaluation des performances du décodeur *LDPC* avec l'algorithme propagation de croyance fera l'objet du chapitre suivant.

Chapitre 4

Evaluation des performances des codes *LDPC* proposés

4.1 Introduction

Dans ce chapitre, nous présentons une chaîne de transmission afin de tester les opérations d'encodage et de décodage. Nous allons illustrer chaque bloc de cette chaîne séparément. Après nous présentons les performances, obtenues par simulation, des codes *LDPC* issus des deux méthodes de construction proposées. Ces deux méthodes utilisent l'algorithme *BP* comme un algorithme de décodage. Ensuite, nous allons comparer les performances des codes *LDPC* proposés, en termes de *BER*, avec des données non codées et modulés en *BPSK* et avec d'autres codes.

4.2 Mise en œuvre de la chaîne de test

Dans le but de tester les opérations d'encodage et de décodage des codes *LDPC* conçus, une simulation d'une chaîne de transmission complète a été réalisée sur *MATLAB*. Cette chaîne est illustrée sur la figure 4.1. Elle est composée : d'une source binaire, de séquence aléatoire représentant les données *d*. Cette dernière est codée par un codeur *LDPC*. Le mot de code *c* (les symboles à transmettre) est d'abord modulé en *BPSK*, puis transmis via un canal *AWGN* (Additive White Gaussian Noise). Après démodulation et décodage, la mesure de la performance de transmission est effectuée en comparant la séquence des bits décodés $\widehat{\boldsymbol{d}}$ (bits estimés) à la séquence des bits initiaux $\boldsymbol{d}$ (bits transmis).

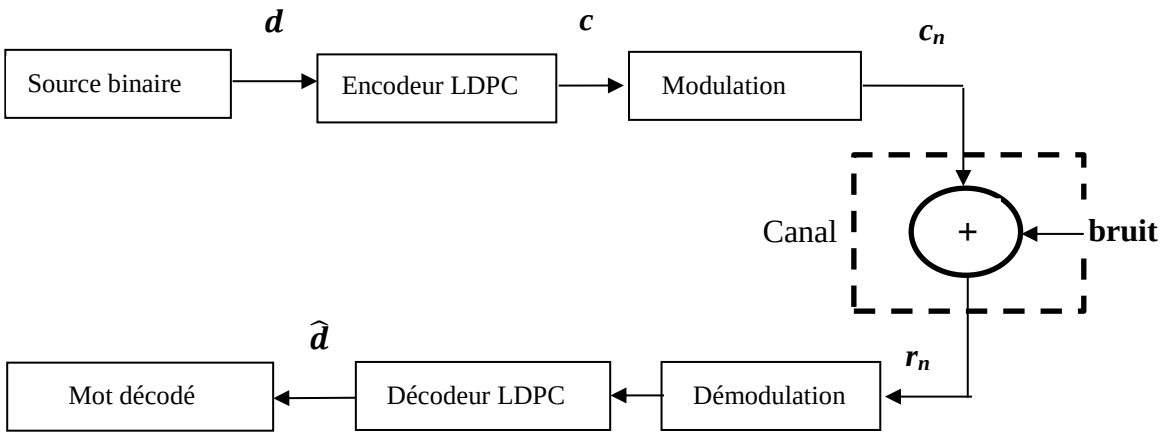


Figure 4. 1 – Chaîne de test de mise en œuvre sous *MATLAB*.

Cette chaîne est constituée, notamment, de :

Source binaire : constituée d'un générateur de données binaires aléatoires. Ce générateur crée un vecteur de message avec une probabilité a priori égale :$P_r[d_i = 1] = P_r[d_i = 0] = 0.5$

Encodeur LDPC : En termes de mise en œuvre de la simulation, l'encodage est effectué via une matrice de contrôle de parité *H* des codes *LDPC* proposés (voir chapitre 3).

Modulateur BPSK : Le signal *BPSK* est représenté par (le mapping) : $\{0,1\} = \{-\sqrt{E_b}, \sqrt{E_b}\}$. où E_b représente l'énergie moyenne par bit.

Canal : Le canal est modélisé par un canal *AWGN*. Le bruit résultant ajouté aux signaux reçus suit une distribution normale de moyenne nulle et variance $N_0 / 2$, où N_0 est la densité spectrale de puissance du bruit unilatérale.

Décodeur LDPC : le décodage *LDPC* est un processus qui calcule tous les messages passant par les nœuds du graphe de Tanner, dans notre cas le décodage se fait par l'algorithme *BP*.

4.3 Evaluation des performances des codes LDPC proposés

La qualité de la chaîne de transmission est évaluée par le *BER*. Pour évaluer le gain de codage apporté par l'algorithme *BP* présenté dans le chapitre 1, on a évalué les performances en termes de *BER* en utilisant la méthode Monté Carlo. Les performances d'une transmission des données codées et modulées en *BPSK* sont présentées dans le but de les comparer avec celles obtenues avec une transmission de données en *BPSK* non codée.

Les fonctions de densité de probabilité conditionnelle du signal reçu r_n sont données par :

Pour $c_n = 1$:

$$P_r(r_n/c_n = 1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(r_n - \sqrt{E_b}\right)^2}{2\sigma^2}\right) \quad (4.1)$$

Pour $c_n = 0$:

$$P_r(r_n/c_n = 0) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\left(r_n + \sqrt{E_b}\right)^2}{2\sigma^2}\right) \quad (4.2)$$

où $\sigma^2 = \frac{N_0}{2}$, représente la variance du bruit et $r_n = t_m + b$ est le signal reçu bruité. t_m est le mot de code, de longueur *N*, modulé en *BPSK*, *b* est le vecteur du bruit, de taille (1× *N*) donné par l'équation (4.3)

$$b = \sqrt{\frac{N_0}{2}} \times randn(1, N) \quad (4.3)$$

où la fonction : $randn(1, N)$génère une séquence binaire aléatoire de taille *N*.

Pour évaluer le *BER* par rapport au *SNR*, on procède comme suit :

Sachant que la densité spectrale unilatérale du bruit $N_{0\ uncoded}$, avant codage, est donnée :

$$N_{0\ uncoded} = \frac{E_b}{SNR} \quad (4.4)$$

où *SNR* représente la valeur du rapport signal sur bruit. La densité spectrale unilatérale du bruit N_0, après codage, s'obtient selon l'équation (4.5)

$$N_0 = \frac{N_{0\ uncoded}}{R} = \frac{E_b}{R \times SNR} \quad (4.5)$$

où *R* représente le rendement du code.

4.4 Résultats de simulation

4.4.1. Performances des codes LDPC issus de la première méthode

Les paramètres suivants sont utilisés :

- Code *LDPC* irrégulier issus de la première méthode.
- Longueur du mot de code : *N*=480 (Code C_1).
- Rendement *R*=1/2.
- Nombre d'itérations pour le décodeur *BP* est 80.
- Nombre d'itérations pour la simulation Monté Carlo est 10^6.

La figure 4.2 représente les performances du mot de code correspondant au code C_1, modulé en *BPSK* et transmis via un canal *AWGN*.

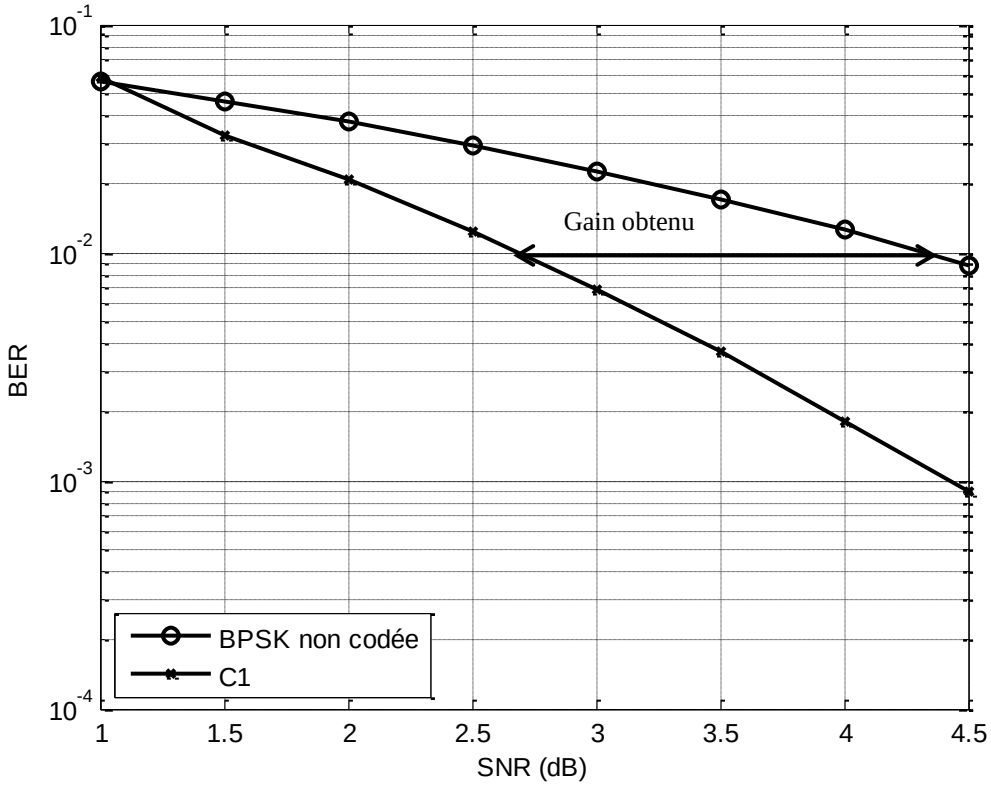


Figure 4. 2 – Performances du code C_1 versus *SNR*.

D'après la figure 4.2, on constate que le code C_1 offre de meilleures performances par rapport à une transmission en *BPSK* non codée. Pour un *BER* de 10^{-2}, un gain environ de 1.65*dB* est obtenu (4.35-2.7). Pour un *BER* de 10^{-3}, on obtient un gain de 2.35*dB* (6.75-4.4), la valeur de 6.75 correspond à un *BER* de 10^{-3} déterminée à partir de l'expression du $BER\ théorique$: $(\frac{1}{2}erfc(\sqrt{SNR}))$.

4.4.2 Performances des codes LDPC issus de la deuxième méthode

Les paramètres suivants sont utilisés :

- Code *LDPC* irrégulier issus de la deuxième méthode.
- Longueur du mot de code : N=480 (Code C_2).
- Rendement R=1/2.
- Nombre d'itérations pour le décodeur *BP* est 80.

- Nombre d'itérations pour la simulation Monté Carlo est 10^6.

La figure 4.3 représente les performances du mot de code correspondant au code C_2, modulé en *BPSK* et transmis via un canal *AWGN*.

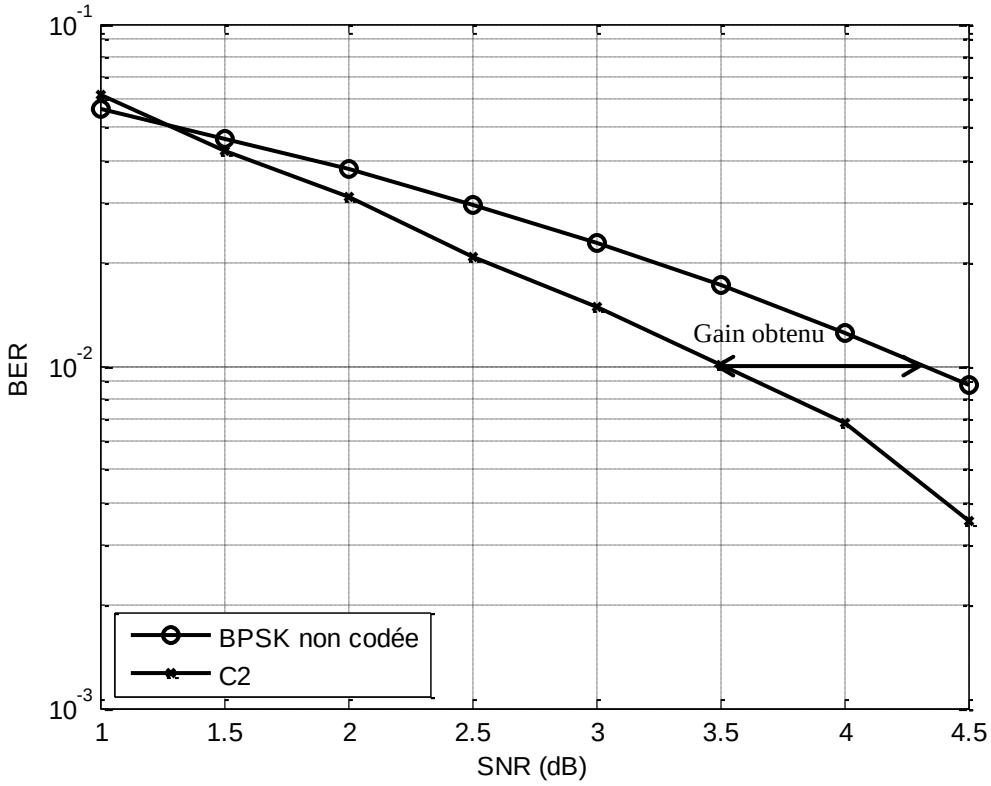


Figure 4. 3 – Performances du code C_2 versus *SNR*.

D'après la figure 4.3, on constate que le code C_2 offre de meilleures performances par rapport à une transmission en *BPSK* non codée. Pour un *BER* de 10^{-2}, on obtient un gain de $0.85dB$ (4.35-3.5), la valeur 4.35 correspond à un *BER* de 10^{-2} déterminée à partir de l'expression du $BER\ théorique$: $(\frac{1}{2}erfc(\sqrt{SNR}))$.

4.4.3 Comparaison entre les codes issus des deux méthodes de construction

Les paramètres suivants sont utilisés :

- Code *LDPC* irrégulier issus de la première méthode (Code C_1).
- Code *LDPC* irrégulier issus de la deuxième méthode (Code C_2).

- Longueur du mot de code : N=480.
- Rendement R=1/2.
- Nombre d'itérations pour le décodeur *BP* est 80.
- Nombre d'itérations pour la simulation Monté Carlo est 10^6.

La figure 4.4 représente les performances du mot de code correspondant aux codes C_1 et C_2, modulés en *BPSK* et transmis via un canal *AWGN*.

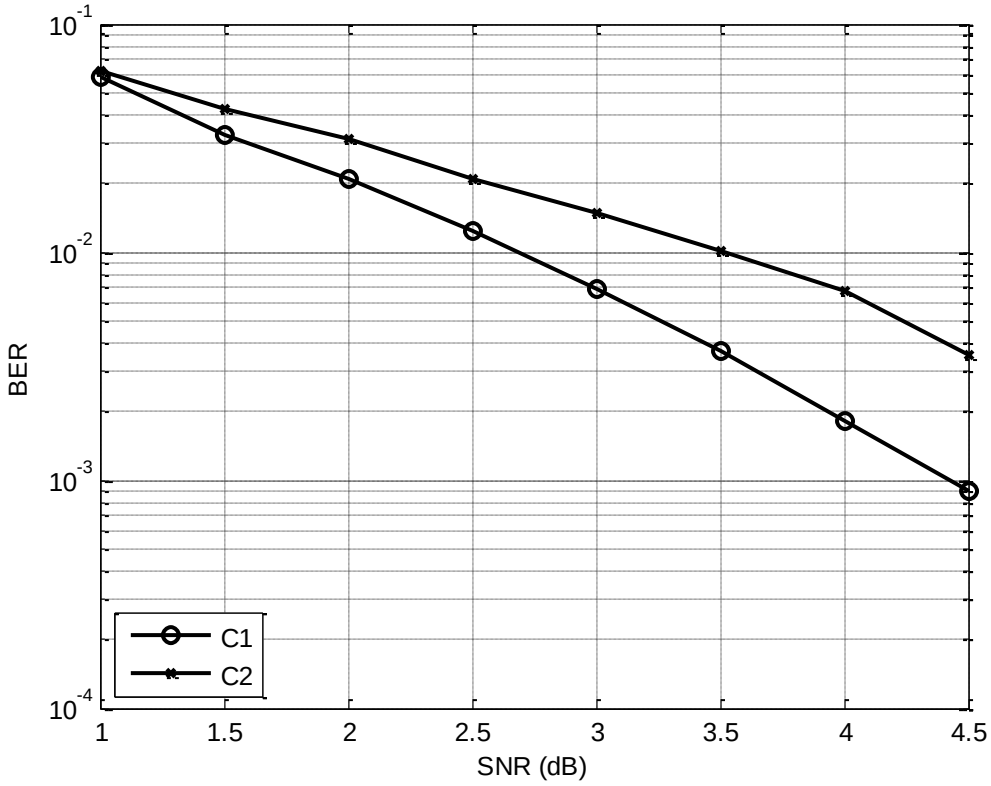


Figure 4. 4 – Performances des codes C_1 et C_2 versus *SNR*.

D'après la figure 4.4, on constate que le code C_1 offre de meilleures performances par rapport au code C_2. À titre d'exemple, pour un *BER* de 10^{-2}, un gain environ de 0.8*dB* est obtenu (3.5-2.7), cela peut être justifié du faite de la présence d'un grand nombre de cycles, dans le code C_2 par rapport au code C_1, ces derniers influent sur la convergence de l'algorithme de décodage *BP* (voir annexe *D*).

4.4.4 Comparaison des codes LDPC proposés avec d'autres codes

Les paramètres suivants sont utilisés :

- Code *LDPC* irrégulier issus de la première méthode (Code dénommé C_3).
- Longueur du mot de code : N=4324.
- Rendement R=1/2.
- Nombre d'itérations pour le décodeur *BP* est 80.
- Nombre d'itérations pour la simulation Monté Carlo est 10^4.

La figure 4.5 représente les performances du mot de code correspondant au code C_3, modulé en *BPSK* et transmis via un canal *AWGN*.

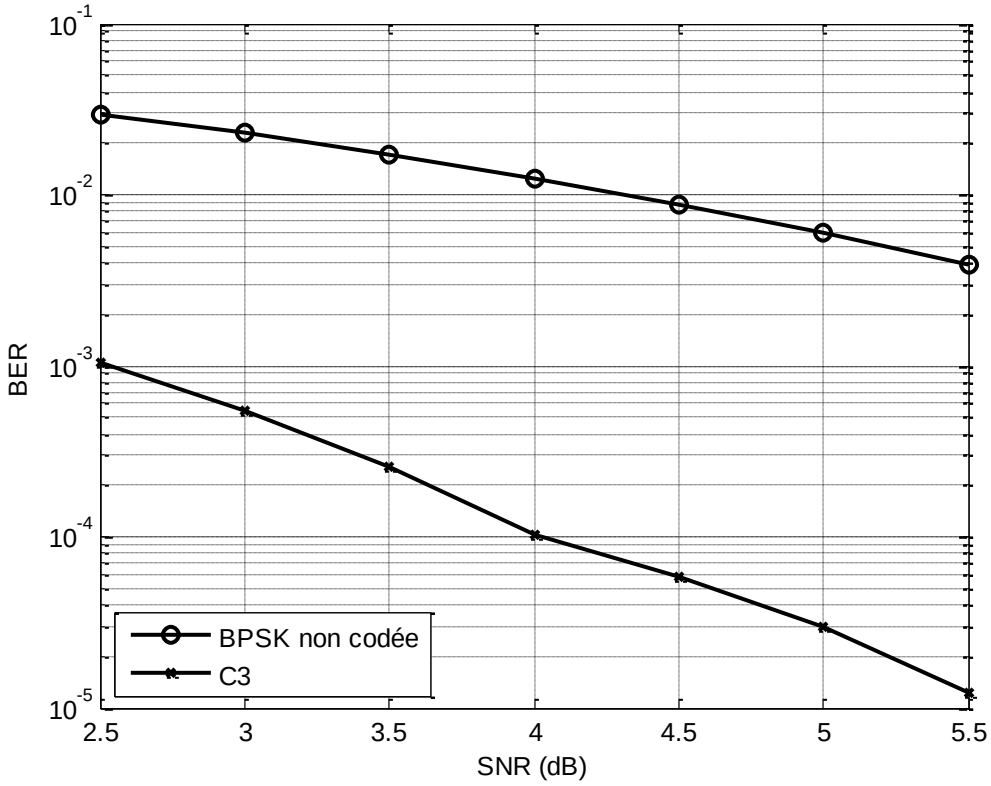


Figure 4. 5 – Performances du code C_3 versus *SNR*.

D'après la figure 4.5, on constate que le code C_3 offre de meilleures performances par rapport au code C_1 et à une transmission en *BPSK* non codée. Pour un *BER* de 10^{-3}, un gain environ de 1.9*dB* est obtenu (4.4-2.5) comparativement au code C_1 et un gain environ de

4.25*dB* est obtenu (6.75-2.5) comparativement à une transmission en *BPSK* non codée. Nous présentons sur le tableau 4.1 une comparaison en termes *BER* du code C_3, avec trois autres codes : code aléatoire (code de Gallager), le *GB-LDPC (*Geometry Based*)* et *TS-LDPC* code. (Turbo-Structured) pour de longue longueur (*N*=4324, pour C_3 ; *N*=4368, pour les autres codes).

BER	Code C_3	Code - Gallager [73]	Code *GB-LDPC* [73]	Code *TS-LDPC* [73]
	E_b/N_0			
10^{-3}	2.5	3	3	3
10^{-4}	4	4	4	3.8

Tableau 4. 1 – Comparaison du *SNR* pour plusieurs constructions de codes *LDPC* pour de longue longueur.

On note, pour un *BER* de 10^{-3}, le code C_3 offre un gain de performance environ 0.5 dB par rapport aux trois codes. Pour un *BER* de 10^{-4}, le code C_3 offre des performances similaires que celles du code de Gallager et du code *GB-LDPC* et présente une perte en gain d'environ 0,2 dB par rapport au code *TS-LDPC.* Nous pouvons dire, que le code C_3, pour de longue longueur peut atteindre la région de plancher d'erreur. Nous présentons dans le tableau 4.2 une comparaison en termes de *BER* des codes : C_1, Gallager, Mackay et *PEG* pour de faible longueur (*N*=480, pour C_1 ; *N*=504, pour les autres codes).

BER	Code C_1	Code - Gallager [74]	Code - Mackay [74]	Code *PEG-LDPC* [74]
	E_b/N_0			
10^{-2}	2.7	2	1.75	1.75
10^{-3}	4.4	2.7	2.3	2.3

Tableau 4. 2 – Comparaison du *SNR* pour plusieurs constructions de codes *LDPC* pour de faible longueur.

A partir du tableau 4.2 on constate, que pour un *BER* de 10^{-2}, le code C_1 présente une perte en gain environ 0,7 dB et 0,95 dB respectivement par rapport au code de Gallager et les codes de Mackay et *PEG-LDPC*. Pour un *BER* de 10^{-3}, le code C_1 est moins performant par rapport aux codes de Gallager, Mackay et *PEG-LDPC.* Il est à noter, que ces résultats

prévisibles sont dus à la structure qui caractérise les codes aléatoire (Gallager, Mackay, *PEG-LDPC*) (voir chapitre 2) [74].

4.5 Conclusion

Dans ce chapitre, nous avons élaboré, sous *MATLAB*, une simulation d'une chaîne de transmission complète, pour tester les opérations d'encodage et de décodage. Nous nous sommes plus concentrés sur l'évaluation des performances des codes proposés. Nous avons terminé par une comparaison des performances des codes construits par rapport, notamment, aux codes conventionnels (Gallager et Mackay). Les codes *LDPC*, issus de la première méthode, permettent d'avoir des résultats sensiblement identique, pour de longue longueur, aux codes conventionnels. Les codes issus de la deuxième méthode sont moins performants mais sont caractérisés par un encodage simple et une complexité de décodage réduite.

Conclusion générale et perspectives

Dans cette thèse, nous nous intéressons à la construction des codes *LDPC* binaires irrégulier à large girth, encodage linéaire et décodage moins complexe. Nous avons proposé deux contributions principales sur la construction des codes *LDPC* binaires.

Dans le premier chapitre, nous avons présenté les éléments de base relatifs aux techniques de codage canal et plus particulièrement aux codes de type *LDPC*. Ces codes forment une large famille des codes correcteurs d'erreurs, qui font parties des codes en bloc permettant de s'approcher de quelques fractions de dB de la limite de Shannon, ces codes sont définis à l'aide d'une matrice de contrôle de parité très creuse. Nous avons présenté les deux représentations des codes *LDPC* (représentation matricielle, représentation graphique). Concernant l'encodage, trois approches (en utilisant la matrice génératrice *G*, à base de la forme triangulaire de *H* et à base des contraintes de construction) ont été décrites. Deux types de décodage ont été présentés (décodage par mot de code, décodage par symbole). Le décodage est réalisé selon l'algorithme *BP*.

Le deuxième chapitre constitue un état de l'art sur les méthodes de construction des codes *LDPC,* ainsi les avantages et les inconvénients des deux types de constructions aléatoire et déterministe ont été donnés. Certaines constructions aléatoire (celle de Gallager et de Mackay) et déterministe (celle basée sur les protographes, quasi-cyclique, repeat accumulate et en treillis) ont été décrites et illustrées par des exemples.

Dans le troisième chapitre, une brève description des matrices particulières dénommées matrices structurées a été donnée. Deux nouvelles méthodes de construction des codes *LDPC* binaires irrégulier à large girth combinant un décodage moins complexe et un encodage linéaire ont été proposées et illustrées par des exemples. Les avantages de ces dernières ont été décrits. Ainsi la procédure de l'encodage linéaire, du a l'utilisation la matrice double diagonale, lors du codage des codes *LDPC* proposés a été illustrée. La complexité des codes proposés a été évaluée et comparée à d'autres codes.

Dans le quatrième chapitre, une chaîne de transmission a été mise en œuvre pour l'évaluation des performances des codes LDPC proposés. Concernant les performances, les codes *LDPC,* issus de la première méthode, permettent d'avoir des performances similaires, pour de longue longueur, à celles des codes conventionnels. Quant aux codes issus de la

deuxième méthode, leurs performances sont moins bonnes ‘dans la zone du plancher d’erreur’ mais ont l’avantage d’avoir un encodage simple et une complexité de décodage réduite.

Les travaux présentés dans cette thèse ont donné lieu aux publications suivantes :

M.A. Tehami, A. Djebbari, "Low-Density-Parity-Check Codes constructed from Hankel Matrices," *Journal of Telecommunications and Information Technology,* vol. no.3. 2018.

M.A. Tehami, A. Djebbari, "New Method to build Low-Density-Parity-Check Codes," *International Journal of Technology*, (en révision).

Perspectives :

Des investigations plus avancées sont proposées comme travaux futurs, concernant les codes *LDPC*. Parmi les perspectives envisageables :

-Il est intéressant : de combiner les méthodes de détection des cycles avec la méthode de conception de la matrice *H*, dans le but de construire un code *LDPC* à large girth.

-Il peut être aussi intéressant : de déterminer la distance minimale d’un code *LDPC*, dont la matrice *H* est de taille importante, qui reste un problème à résoudre.

Annexes

A Algorithme du décodage des codes *LDPC*

Annexe A.1 :

Soit le vecteur reçu $r = \{r_1, r_2, \cdots, r_N\}$ à la sortie du canal et les ensembles $\{r_n\}$ et $\{r_{i\neq n}\}$ tel que :

$$r = \{r_n\} \cup \{r_{i\neq n}\}$$

la probabilité conditionnelle $P_r(c_n/r)$ s'écrit comme suit :

$$P_r(c_n/r) = P_r(c_n/r_n, r_{i\neq n}) \qquad (A.1.01)$$

puisque c_n, r_n sont indépendants de $r_{i\neq n}$, on a :

$$\begin{cases} P_r(c_n, r_n, r_{i\neq n}) = P_r(c_n/r_n, r_{i\neq n})P_r(r_n, r_{i\neq n}) & (A.1.02) \\ P_r(c_n, r_n, r_{i\neq n}) = P_r(r_n/c_n, r_{i\neq n})P_r(c_n, r_{i\neq n}) & (A.1.03) \end{cases}$$

$$\begin{cases} P_r(r_n, r_{i\neq n}) = P_r(r_n/r_{i\neq n})P_r(r_{i\neq n}) & (A.1.04) \\ P_r(c_n, r_{i\neq n}) = P_r(c_n/r_{i\neq n})P_r(r_{i\neq n}) & (A.1.05) \end{cases}$$

à partir des équations (A.1.01) et (A.1.02), on a :

$$P_r(c_n/r) = \frac{P_r(c_n, r_n, r_{i\neq n})}{P_r(r_n, r_{i\neq n})} \qquad (A.1.06)$$

en remplaçant (A.1.03) et (A1.04) dans (A.1.06) on obtient :

$$P_r(c_n/r) = \frac{P_r(r_n/c_n, r_{i\neq n})P_r(c_n, r_{i\neq n})}{P_r(r_n/r_{i\neq n})P_r(r_{i\neq n})} \qquad (A.1.07)$$

en remplaçant (A.1.05) dans (A.1.07) on obtient :

$$P_r(c_n/r) = \frac{P_r(r_n/c_n, r_{i\neq n})P_r(c_n/r_{i\neq n})P_r(r_{i\neq n})}{P_r(r_n/r_{i\neq n})P_r(r_{i\neq n})} \qquad (A.1.08)$$

sachant que :

$$P_r(r_n/c_n, r_{i\neq n}) = \frac{P_r(r_n, c_n, r_{i\neq n})}{P_r(c_n, r_{i\neq n})} \qquad (A.1.09)$$

$$\begin{cases} P_r(r_n, c_n, r_{i\neq n}) = P_r(r_n, c_n/r_{i\neq n})P_r(r_{i\neq n}) & (A.1.10) \\ P_r(c_n, r_{i\neq n}) = P_r(c_n/r_{i\neq n})P_r(r_{i\neq n}) & (A.1.11) \end{cases}$$

en remplaçant (A.1.10) et (A.1.11) dans (A.1.09), on obtient :

$$P_r(r_n/c_n, r_{i\neq n}) = \frac{P_r(r_n, c_n/r_{i\neq n})}{P_r(c_n/r_{i\neq n})} \qquad (A.1.12)$$

puisque c_n, r_n sont indépendants de $r_{i\neq n}$, on a :

$$P_r(r_n, c_n/r_{i\neq n}) = P_r(r_n, c_n)$$

$$P_r(c_n/r_{i\neq n}) = P_r(c_n)$$

d'où

$$P_r(r_n/c_n, r_{i\neq n}) = \frac{P_r(r_n, c_n/r_{i\neq n})}{P_r(c_n/r_{i\neq n})} = \frac{P_r(r_n, c_n)}{P_r(c_n)} = P_r(r_n/c_n) \qquad (A.1.13)$$

en remplaçant (A.1.13) dans (A.1.08), on obtient :

$$P_r(c_n/r) = \frac{P_r(r_n/c_n)P_r(c_n/r_{i\neq n})}{P_r(r_n/r_{i\neq n})} \qquad (A.1.14)$$

Annexe A.2. :

Cas de canal gaussien (*AWGN*) :

L'information intrinsèque I_n est :

$$I_n = \log\frac{P_r(r_n/c_n = 0)}{P_r(r_n/c_n = 1)}$$

Pour une modulation *BPSK* sur le mot de code *c*, on a :

$$\begin{cases} si \;\; c_n = 1 \;\; \rightarrow +A \\ si \;\; c_n = 0 \;\; \rightarrow -A \end{cases}$$

L'énergie par symbole $E_s = A^2$, les fonctions de densité de probabilité conditionnelle sont :

$$P_r(r_n/c_n = 1) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{\left(r_n-\sqrt{E_s}\right)^2}{2\sigma^2}\right) \qquad (A.2.01)$$

$$P_r(r_n/c_n = 0) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{\left(r_n+\sqrt{E_s}\right)^2}{2\sigma^2}\right) \qquad (A.2.02)$$

$$I_n = \log\left(\frac{\exp\left(-\frac{\left(r_n+\sqrt{E_s}\right)^2}{2\sigma^2}\right)}{\exp\left(-\frac{\left(r_n-\sqrt{E_s}\right)^2}{2\sigma^2}\right)}\right) \qquad (A.2.03)$$

$$I_n = \log\left(\exp\left(-\frac{\left(r_n+\sqrt{E_s}\right)^2}{2\sigma^2}\right) \times \exp\left(\frac{\left(r_n-\sqrt{E_s}\right)^2}{2\sigma^2}\right)\right) \qquad (A.2.04)$$

$$I_n = \log\left(\exp\left(\left(-\frac{\left(r_n+\sqrt{E_s}\right)^2}{2\sigma^2}\right)+\left(\frac{\left(r_n-\sqrt{E_s}\right)^2}{2\sigma^2}\right)\right)\right) \quad \text{(A.2.05)}$$

$$I_n = \log\left(exp\left(\frac{\left(-4r_n\sqrt{E_s}\right)}{2\sigma^2}\right)\right) \quad \text{(A.2.06)}$$

$$I_n = \frac{-2r_n\sqrt{E_s}}{\sigma^2} \quad \text{(A.2.07)}$$

Annexe A.3:

(The tanh rule) sur deux variables:

Soit $\theta = c_{1\oplus}c_2$, une équation de contrôle de parité dont c_1 et c_2 sont deux variables binaires dépendent de variables d_1 et d_2

Soit $p_r(\theta = 1/d_1, d_2) = p_\theta$ on a :

$$\frac{p_r(\theta=0/d_1,d_2)}{p_r(\theta=1/d_1,d_2)} = \frac{1-p_\theta}{p_\theta} \quad \text{(A.3.1)}$$

on a $T_n = \log\left(\frac{p_r(\theta=0/d_1,d_2)}{p_r(\theta=1/d_1,d_2)}\right)$

$$tanh(\,(T_n)\,) = tanh\left(\frac{1}{2}\log\left(\frac{p_r(\theta=0/d_1,d_2)}{p_r(\theta=1/d_1,d_2)}\right)\right)$$

$$tanh\left(\frac{1}{2}\log\left(\frac{p_r(\theta=0/d_1,d_2)}{p_r(\theta=1/d_1,d_2)}\right)\right) = tanh\left(\frac{1}{2}\log\left(\frac{1-p_\theta}{p_\theta}\right)\right) \quad \text{(A.3.2)}$$

Sachant que : $\tanh\left(x/2\right) = \frac{e^x-1}{e^x+1}$, de l'équation (A.3.2), on obtient :

$$tanh\left(\frac{1}{2}\log\left(\frac{p_r(\theta=0/d_1,d_2)}{p_r(\theta=1/d_1,d_2)}\right)\right) = \frac{e^{\log\left(\frac{1-p_\theta}{p_\theta}\right)}-1}{e^{\log\left(\frac{1-p_\theta}{p_\theta}\right)}+1} = 1-2p_\theta \quad \text{(A.3.3)}$$

Soit $p_1 = p_r(c_1 = 1/d_1, d_2)$ et $p_2 = p_r(c_2 = 1/d_1, d_2)$

si $\theta = 1$, on a $\theta = c_{1\oplus}c_2 = \bar{c_1}c_2 + \bar{c_2}c_1 = 1$, on obtient :

$p_r(\theta = 1/d_1, d_2) = p_r(c_1 = 0/d_1, d_2)p_r(c_2 = 1/d_1, d_2) + p_r(c_2 = 0/d_1, d_2)\, p_r(c_1 = 1/d_1, d_2)$

$$p_\theta = p_r(\theta = 1/d_1, d_2) = (1-p_1)p_2 + (1-p_2)p_1 \quad \text{(A.3.4)}$$

en remplaçant (A.3.4) dans (A.3.3) :

$tanh\left(\frac{1}{2}\log\left(\frac{p_r(\theta=0/d_1,d_2)}{p_r(\theta=1/d_1,d_2)}\right)\right) = 1-2[(1-p_1)p_2 + (1-p_2)p_1] = (1-2p_1)(1-2p_2)$

(A.3.5)

Sachant que $p_1 = p_r(c_1 = 1/d_1, d_2)$ et $p_2 = p_r(c_2 = 1/d_1, d_2)$

D'après l'équation (A.3.3), on a :

$$(1 - 2p_1) = \ tanh\left(\frac{1}{2}\log\left(\frac{p_r(c_1=0/d_1,d_2)}{p_r(c_1=1/d_1,d_2)}\right)\right) \quad \text{(A.3.6)}$$

$$(1 - 2p_2) = tanh\left(\frac{1}{2}\log\left(\frac{p_r(c_2=0/d_1,d_2)}{p_r(c_2=1/d_1,d_2)}\right)\right) \quad \text{(A.3.7)}$$

D'après l'équation (A.3.5), on a :

$$tanh\left(\frac{1}{2}\log\left(\frac{p_r(\theta=0/d_1,d_2)}{p_r(\theta=1/d_1,d_2)}\right)\right) = tanh\left(\frac{1}{2}\log\left(\frac{p_r(c_1=0/d_1,d_2)}{p_r(c_1=1/d_1,d_2)}\right)\right) tanh\left(\frac{1}{2}\log\left(\frac{p_r(c_2=0/d_1,d_2)}{p_r(c_2=1/d_1,d_2)}\right)\right) \quad \text{(A.3.8)}$$

$$tanh\left(\frac{T_{(\theta/d_1,d_2)}}{2}\right) = tanh\left(\frac{T_{(c_1/d_1,d_2)}}{2}\right) tanh\left(\frac{T_{(c_2/d_1,d_2)}}{2}\right) \quad \text{(A.3.9)}$$

(The tanh rule) sur plusieurs variables:

$$tanh\left(\frac{1}{2}\log\left(\frac{p_r(\theta=0/d_j)}{p_r(\theta=1/d_j)}\right)\right) = \prod_{i=1}^{n} tanh\left(\frac{1}{2}\log\left(\frac{p_r(c_j=0/d_j)}{p_r(c_j=1/d_j)}\right)\right) \quad \text{(A.3.10)}$$

$$\left(\log\left(\frac{p_r(\theta=0/d_j)}{p_r(\theta=1/d_j)}\right)\right) = 2tanh^{-1}\prod_{j=1}^{n} tanh\left(\frac{1}{2}\log\left(\frac{p_r(c_j=0/d_j)}{p_r(c_j=1/d_j)}\right)\right) \quad \text{(A.3.11)}$$

$$T_{(\theta/d_j)} = 2tanh^{-1}\prod_{j=1}^{n} tanh\left(\frac{1}{2}\log\left(T_{(c_j/d_j)}\right)\right) \quad \text{(A.3.12)}$$

B Encodage linéaire des codes *LDPC* proposés

Exemple 1

Soit une matrice de contrôle de parité H déterminée par la première méthode de dimension $M \times N = (12 \times 24)$ ci-dessous :

$$H = [H_1 H_2]$$

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

Soit le mot d'information d :

$$d = [1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0]$$

Le premier bit de contrôle de parité est calculé selon l'équation (1.44) :

$$p_1 = \sum_j h_{1j}^1 \, d_j$$

où h^1_{ij} représentent les éléments de la matrice H_1 ci-dessous :

La matrice H_1 est comme suit

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$p_1 =$

$h_{1\,1}^{1} d_1 + h_{1\,2}^{1}\, d_2 + h_{1\,3}^{1} d_3 + h_{1\,4}^{1} d_4 + h_{1\,5}^{1} d_5 + h_{1\,6}^{1} d_6 + h_{1\,7}^{1} d_7 + h_{1\,8}^{1} d_8 + h_{1\,9}^{1} d_9 + h_{1\,10}^{1} d_{10} + h_{1\,11}^{1} d_{11} +$

$h_{1\,12}^{1} d_{12}$

Selon H_1 et d, on obtient :

$$p_1 = 1 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0$$

$$p_1 = 1$$

Les autres bits de contrôle de parité $p = \{p_i\}$ peuvent se calculer selon l'équation (1.45) :

$$p_i = p_{i-1} + \sum_j h_{ij}^{1}\, d_j mod\ (2)$$

$p_2 = p_1 +$

$h_{2\,1}^{1} d_1 + h_{2\,2}^{1}\, d_2 + h_{2\,3}^{1} d_3 + h_{2\,4}^{1} d_4 + h_{2\,5}^{1} d_5 + h_{2\,6}^{1} d_6 + h_{2\,7}^{1} d_7 + h_{2\,8}^{1} d_8 + h_{2\,9}^{1} d_9 + h_{2\,10}^{1} d_{10} + h_{2\,11}^{1} d_{11} + h_{2\,12}^{1} d_{12}$

$$p_2 = 1 + 0 + 0 + 0 + 0 + 0 + 0 + 1 + 0 + 0 + 0 + 0 + 0$$

$$p_2 = 0$$

$p_3 =$

$p_2 +$

$h_{3\,1}^{1} d_1 + h_{3\,2}^{1}\, d_2 + h_{3\,3}^{1} d_3 + h_{3\,4}^{1} d_4 + h_{3\,5}^{1} d_5 + h_{3\,6}^{1} d_6 + h_{3\,7}^{1} d_7 + h_{3\,8}^{1} d_8 + h_{3\,9}^{1} d_9 + h_{3\,10}^{1} d_{10} + h_{2311}^{1} d_{11} +$

$h_{3\,12}^{1} d_{12}$

$$p_3 = 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 0$$

$$p_3 = 0$$

De façon similaire, on peut obtenir les autres bits :

$$p_4 = 0;\ p_5 = 0;\ p_6 = 0;\ p_7 = 0;\ p_8 = 0; p_9 = 0; p_{10} = 1;\ p_{11} = 1;\ p_{12} = 0$$

Ainsi, on obtient :

-Le vecteur de redondance : $p = [1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0]$

-Le mot de code $c = [d\ \ p]$ est :

$$c = [1\ 0\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ \ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 0]$$

Exemple 2

Soit une matrice de contrôle de parité H déterminée par la deuxième méthode de dimension $M \times N$ = (6× 12) ci-dessous :

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

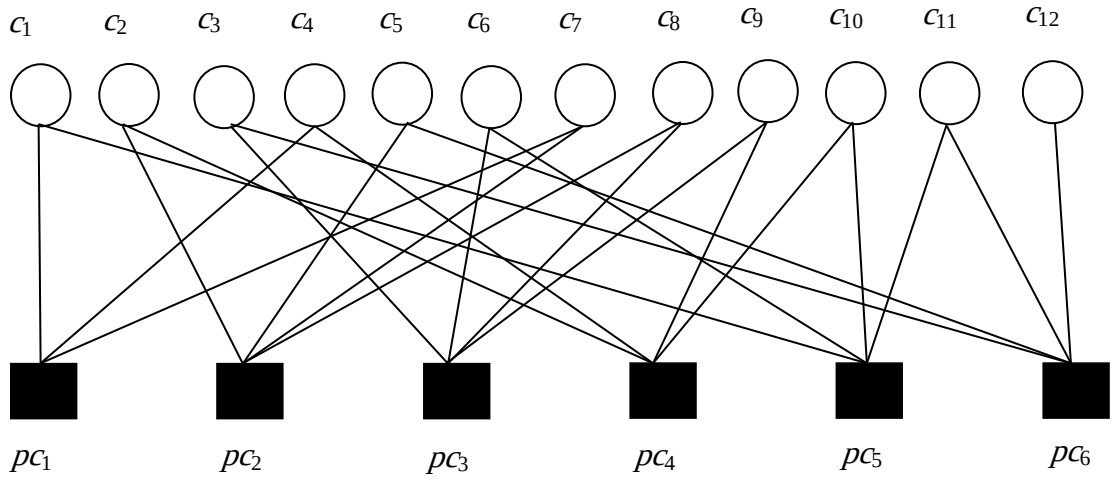
Pour un vecteur de données $d = [0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1]$, le mot de code c obtenu en utilisant les équations (1.44) et (1.45) est :

$$c = [0\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ 0\ 0\ 1\ 0\ 1\ 0\ ]$$

C Décodage des codes *LDPC* proposés

Exemple

Le graphe de Tanner de la matrice *H* de l'exemple 2 est comme suit :



où c_n représentent les nœuds de variables et les pc_m représentent les nœuds de contrôles.

Soit *N(m)* représente l'ensemble des bits qui interviennent dans la $m^{ême}$ équation de contrôle de parité. Donc, dans cet exemple on a :

$N(1) = \{c_1, c_4, c_7\}$; $N(2) = \{c_2, c_5, c_7, c_8\}$;$N(3) = \{c_3, c_6, c_8, c_9\}$;

$N(4) = \{c_2, c_4, c_9, c_{10}\}$; $N(5) = \{c_1, c_6, c_{10}, c_{11}\}$;$N(6) = \{c_3, c_5, c_{11}, c_{12}\}$.

D'une façon similaire :

Soit *M(n)* représente l'ensemble des équations de contrôle de parité qui contrôle le $n^{ème}$ bit du mot de code. Donc, dans cet exemple on a :

$M(1) = \{pc_1, pc_5\}$; $M(2) = \{pc_2, pc_4\}$; $M(3) = \{pc_3, pc_6\}$; $M(4) = \{pc_1, pc_4\}$;

$M(5) = \{pc_2, pc_6\}$; $M(6) = \{pc_3, pc_5\}$; $M(7) = \{pc_1, pc_2\}$; $M(8) = \{pc_2, pc_3\}$;

$M(9) = \{pc_3, pc_4\}$; $M(10) = \{pc_4, pc_5\}$; $M(11) = \{pc_5, pc_6\}$; $M(12) = \{pc_6\}$.

Soit *c* le mot de code (voir exemple 2) à émettre via un canal *AWGN* sous *BPSK* :

$$c = [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \; 0 \; 0 \; 1 \; 0 \; 1 \; 0]$$

En utilisant le mapping *BPSK* suivant :

Pour un bit 1 : $2(1) - 1 = 1$

Pour un bit 0 : $2(0) - 1 = -1$

Le mot de code modulé est

$$c_{mod} = [-1 \quad 1 \quad -1 \quad -1 \quad 1 \quad 1 \quad -1 \quad -1 \quad 1 \quad -1 \quad 1 \quad -1]$$

On pose :

E_b (énergie moyenne par bit)=1, *R* (rendement)=0.5.

Le mot transmis est :

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

On suppose $(SNR)_{dB}$=0, ce qui donne N_0 (densité spectrale unilatérale de bruit)=1, La variance du bruit ajoutée est $\sigma^2 = \frac{N_0}{2} = 1$, le vecteur du bruit *b* correspondant à $N_0/2$ est :

b= [-0.2803 0.8389 -1.6037 1.0133 -0.6790 0.5459 -0.8406 1.2036 0.4714 0.0795 0.1531 1.3582]

Le vecteur reçu :

$r_n = t_m + b$ = [-1.2803 1.8389 -2.6037 0.0133 0.3210 1.5459 -1.8406 0.2036 1.4714 -0.9205 1.1531 0.3582]

Le décodage se fait en utilisant l'algorithme propagation de croyance comme suit :

1) Initialisation :

Calcul de l'information intrinsèque I_n *:*

$$I_n = \frac{-2r_n}{\sigma^2}$$

Le vecteur des informations intrinsèques s'écrit :

$$I_n = [2.5606 \quad -3.6778 \quad 5.2074 \quad -0.0266 \quad -0.6420 \quad -3.0918 \quad 3.6812 \quad -0.4072 \quad -2.9428 \quad 1.8410 \quad -2.3062 \quad -0.7164]$$

-Initialisation de l'information extrinsèque (sous forme matricielle) :

$$E_n = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Calcul de l'information totale T_n :

On a :

$$T_n^{(0)} = I_n \qquad \text{(C.1)}$$

$$T_n^{(0)} = [2.5606 \quad -3.6778 \quad 5.2074 \quad -0.0266 \quad -0.6420 \quad -3.0918 \quad 3.6812 \quad -0.4072 \quad -2.9428 \quad 1.8410 \quad -2.3062 \quad -0.7164]$$

2) Mise à jour

Itération 1

-Mise à jour de l'information extrinsèque (des nœuds de contrôle)

$$E_{n,m}^{(l)} = \prod_{j \epsilon N(m) \backslash n} sign\left(T_{c_j}^{(l-1)}\right) \times f\left(\sum_{j \epsilon N(m) \backslash n} f\left(\left|T_{c_j}^{(l-1)}\right| / r_{i \neq n}\right)\right) \qquad \text{(C.2)}$$

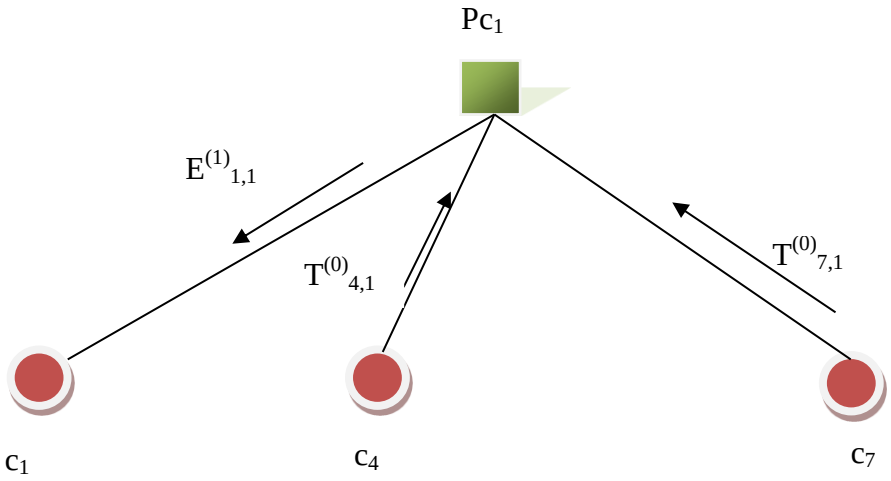
Sachant que :

$$sign\, T_n^{(0)} = [+1 \quad -1 \quad +1 \quad -1 \quad -1 \quad -1 \quad +1 \quad -1 \quad -1 \quad +1 \quad -1 \quad -1]$$

$|T_n^{(0)}|$ = [2.5606 -3.6778 5.2074 -0.0266 -0.6420 -3.0918 3.6812 -0.4072 -2.9428 1.8410 -2.3062 -0.7164]

- Calcul de $E_{1,1}^{(1)}$:

Considérant une partie du graphe de Tanner contenant le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



Selon le graphe de Tanner, on a : $N(1)/1 = \{c_4, c_7\}$

En remplaçant dans l'équation (C.2), on obtient:

$$E_{1,1}^{(1)} = \prod_{j\epsilon N(1)\backslash 1} sign\left(T_{c_j}^{(0)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 1} f\left(\left|T_{c_j}^{(0)}\right| / r_{i\neq 1}\right)\right)$$

Le signe de $E_{1,1}^{(1)}$ est :

$$sign\left(E_{1,1}^{(1)}\right) = sign\left(T_{c_4}^{(0)}\right) \times sign\left(T_{c_7}^{(0)}\right) = (-1)(+1) = (-1)$$

La valeur absolue de $E_{1,1}^{(1)}$est

$$\left|E_{1,1}^{(1)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 1} f\left(\left|T_{c_j}^{(0)}\right| / r_{i\neq 1}\right)\right) = f(f\left(\left|T_{c_4}^{(0)}\right|\right) + f\left(\left|T_{c_7}^{(0)}\right|\right))$$

Calculant maintenant $f\left(\left|T_{c_4}^{(0)}\right|\right)\ et\ f\left(\left|T_{c_7}^{(0)}\right|\right)$

On a

$$f(x) = \log\frac{e^x + 1}{e^x - 1}$$

$$f\left(\left|T_{c_4}^{(0)}\right|\right) = \log\frac{e^{\left|T_{c_4}^{(0)}\right|} + 1}{e^{\left|T_{c_4}^{(0)}\right|} - 1} = \log\frac{e^{0.0266} + 1}{e^{0.0266} - 1} = 4.3185$$

$$f\left(\left|T_{c_7}^{(0)}\right|\right) = \log\frac{e^{\left|T_{c_7}^{(0)}\right|} + 1}{e^{\left|T_{c_7}^{(0)}\right|} - 1} = \log\frac{e^{3.6812} + 1}{e^{3.6812} - 1} = 0.0504$$

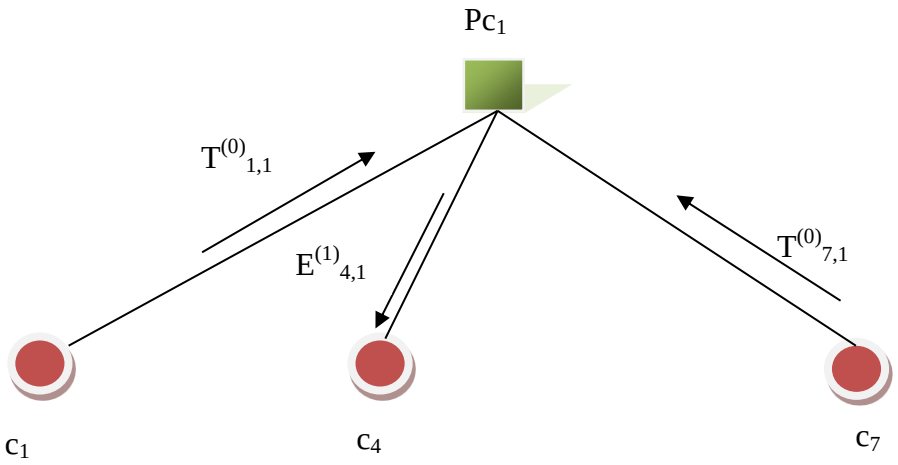
$$\left|E_{1,1}^{(1)}\right| = f(4.3185 + 0.0504) = f(4.3689) = \log\frac{e^{4.3689}+1}{e^{4.3689}-1}$$

$$\left|E_{1,1}^{(1)}\right| = 0.0253$$

$$E_{1,1}^{(1)} = -0.0253$$

- Calcul de $E_{4,1}{}^{(1)}$:

On prend la partie du graphe de Tanner contenant le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



Selon le graphe de Tanner, on a : $N(1)/4 = \{c_1, c_7\}$

En remplaçant dans l'équation (C.2), on obtient :

$$E_{4,1}^{(1)} = \prod_{j\epsilon N(1)\backslash 4} sign\left(T_{c_j}^{(0)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 4} f\left(\left|T_{c_j}^{(0)}\right| / r_{i\neq 4}\right)\right)$$

Le signe de $E_{4,1}^{(1)}$ est :

$$sign\left(E_{4,1}^{(1)}\right) = sign\left(T_{c_1}^{(0)}\right) \times sign\left(T_{c_7}^{(0)}\right) = (+1)(+1) = (+1)$$

La valeur absolue de $E_{4,1}^{(1)}$ est :

$$\left|E_{4,1}^{(1)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 4} f\left(\left|T_{c_j}^{(0)}\right| / r_{i\neq 2}\right)\right) = f(f\left(\left|T_{c_1}^{(0)}\right|\right) + f\left(\left|T_{c_7}^{(0)}\right|\right))$$

Calculant maintenant $f\left(\left|T_{c_1}^{(0)}\right|\right)$

$$f\left(\left|T_{c_1}^{(0)}\right|\right) = \log\frac{e^{\left|T_{c_1}^{(0)}\right|}+1}{e^{\left|T_{c_1}^{(0)}\right|}-1} = \log\frac{e^{2.5606}+1}{e^{2.5606}-1} = 0.1548$$

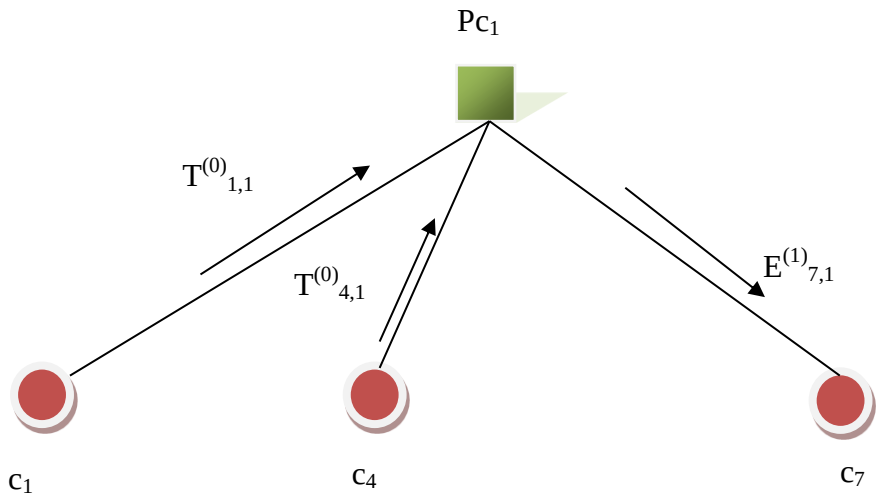
$$\left|E_{4,1}^{(1)}\right| = f(0.1548+0.0504) = f(0.2052) = \log\frac{e^{0.2052}+1}{e^{0.2052}-1}$$

$$\left|E_{4,1}^{(1)}\right| = 2.2803$$

$$E_{4,1}^{(1)} = +2.2803$$

- Calcul de $E_{7,1}^{(1)}$:

Le nœud de contrôle de parité Pc_1 est lié aux nœuds de variable comme suit :



Selon le graphe de Tanner, on a : $N(1)/7 = \{c_1, c_4\}$

En remplaçant dans l'équation (C.2), on obtient :

$$E_{7,1}^{(1)} = \prod_{j\epsilon N(1)\backslash 7} sign\left(T_{c_j}^{(0)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 7} f\left(\left|T_{c_j}^{(0)}\right|/r_{i\neq 7}\right)\right)$$

Le signe de $E_{7,1}^{(1)}$ est :

$$sign\left(E_{7,1}^{(1)}\right) = sign\left(T_{c_1}^{(0)}\right) \times sign\left(T_{c_4}^{(0)}\right) = (+1)(-1) = (-1)$$

La valeur absolue de $E_{7,1}^{(1)}$ est :

$$\left|E_{7,1}^{(1)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 7} f\left(\left|T_{c_j}^{(0)}\right|/r_{i\neq 7}\right)\right) = f(f\left(\left|T_{c_1}^{(0)}\right|\right) + f\left(\left|T_{c_4}^{(0)}\right|\right))$$

On a

$$\left|E_{7,1}^{(1)}\right| = f(0.1548 + 4.3185) = f(4.4733) = \log\frac{e^{4.4733}+1}{e^{4.4733}-1}$$

$$\left|E_{7,1}^{(1)}\right| = 0.0228$$

$$E_{7,1}^{(1)} = -0.0228$$

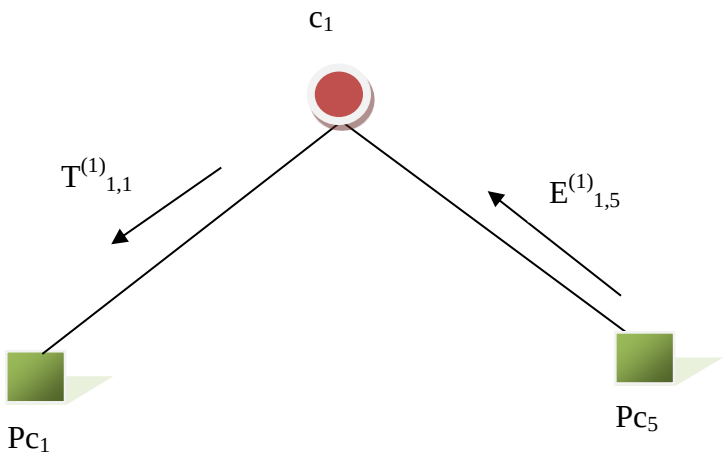
Ainsi de la même façon, on peut calculer toutes les informations extrinsèques. Ci-dessous toutes les informations extrinsèques calculées suite à l'exécution du programme de calcul itératif.

$$E_n^{(1)} = \begin{bmatrix} -0.0253 & 0 & 0 & 2.2803 & 0 & 0 & -0.0228 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.1187 & 0 & 0 & 0.3672 & 0 & -0.1187 & 0.5767 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.3331 & 0 & 0 & 0.3614 & 0 & 2.2699 & 0.3668 & 0 & 0 & 0 \\ 0 & 0.0174 & 0 & 1.4539 & 0 & 0 & 0 & 0 & 0.0184 & -0.0228 & 0 & 0 \\ 1.2165 & 0 & 0 & 0 & 0 & -1.1235 & 0 & 0 & 0 & 1.5177 & -1.2889 & 0 \\ 0 & 0 & -0.1751 & 0 & 0.5717 & 0 & 0 & 0 & 0 & 0 & 0.2118 & -0.5138 \end{bmatrix}$$

-Mise à jour de l'information partielle (des nœuds de variable) :

- Calcul de $T_{1,1}^{(1)}$:

La partie du graphe de Tanner contenant le nœud de variable c_1 et les nœuds de contrôle qui lui sont connectés est comme suit :



Selon le graphe de Tanner, on a : $\mathrm{M}(1)/1 = \{\mathrm{Pc}_5\}$

De l'équation (1.69), on a :

$$T_{n,m}^{(l)} = I_n + \sum_{m' \in M(n)/m} E_{n,m'}^{(l)} \qquad \text{(C.3)}$$

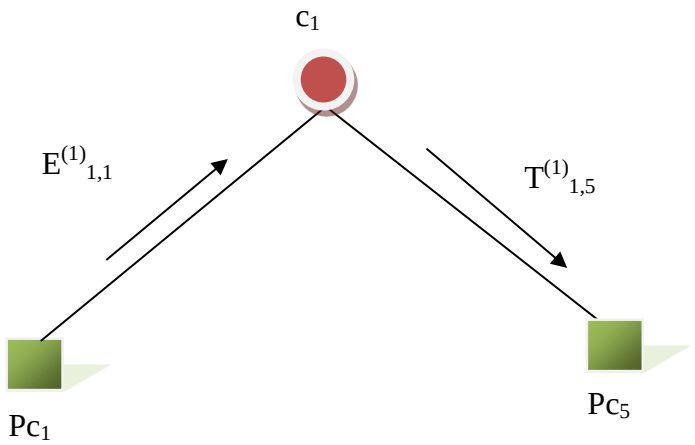
Pour : *n*=1 ; *m*=1, l'équation (C.3) devient :

$$T_{1,1}^{(1)} = I_1 + \sum_{m' \in M(1)/1} E_{1,m'}^{(1)}$$

$$T_{1,1}^{(1)} = I_1 + E_{1,5}^{(1)} = (2.5606) + (1.1265) = 3.7771$$

- Calcul de $T_{1,5}^{(1)}$:

On prend une partie du graphe de Tanner contenant le nœud de variable c_1 et les nœuds de contrôle qui lui sont connectés.



Selon le graphe de Tanner, on a : $\mathrm{M}(1)/5 = \{\mathrm{Pc}_1\}$

Pour : *n*=1 ; *m*=5, l'équation (C.3) devient :

$$T_{1,5}^{(1)} = I_1 + \sum_{m' \in M(1)/5} E_{1,m'}^{(1)}$$

$$T_{1,5}^{(1)} = I_1 + E_{1,1}^{(1)} = (2.5606) + (-0.0253) = 2.5353$$

Ainsi de la même façon, on déduit les valeurs de $T^{(1)}_{n,m}$ (par application d'un programme de calcul) :

$$T_{n,m}{}^{(1)} = \begin{bmatrix} 3.7771 & 0 & 0 & 1.4273 & 0 & 0 & 3.5625 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3.6604 & 0 & 0 & -0.0703 & 0 & 3.6584 & 1.8627 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5.0323 & 0 & 0 & -4.2153 & 0 & 1.1695 & -2.9244 & 0 & 0 & 0 \\ 0 & -3.5591 & 0 & 2.2537 & 0 & 0 & 0 & 0 & -2.5760 & 3.3587 & 0 & 0 \\ 2.5353 & 0 & 0 & 0 & 0 & -2.7304 & 0 & 0 & 0 & 1.8182 & -2.0944 & 0 \\ 0 & 0 & 4.8743 & 0 & -0.2748 & 0 & 0 & 0 & 0 & 0 & 3.5951 & -0.7164 \end{bmatrix}$$

Mise à jour de l'information totale $T^{(1)}_n$:

$$T_n^{(l)} = I_n + \sum_{m \in M(n)} E_{n,m}^{(l)} \qquad \text{(C.4)}$$

Pour *n*=1, l'équation (C.4) devient :

$$T_1^{(1)} = I_1 + E_{1,1}^{(1)} + E_{1,5}^{(1)} = 2.5606 + (-0.0253) + (1.2165) = 3.7518$$

Ainsi de la même façon, on peut calculer toutes les informations totales. Le vecteur de l'information totale $T_n^{(1)}$ est donné suite au programme de calcul comme suit :

$T_n^{(1)}$ =[3.7518 -3.5417 4.6992 3.7076 0.2968 -3.8539 3.5397 2.4394 -2.5576 3.3359 -3.3833 -0.2026]

Selon l'équation (1.54) on a :

$$T_n = \log \frac{P_r(c_n=0/r)}{P_r(c_n=1/r)} \qquad \text{(C.5)}$$

En utilisant l'équation (C.5), le mot décodé est

$$\hat{c}^{(1)} = [0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1]$$

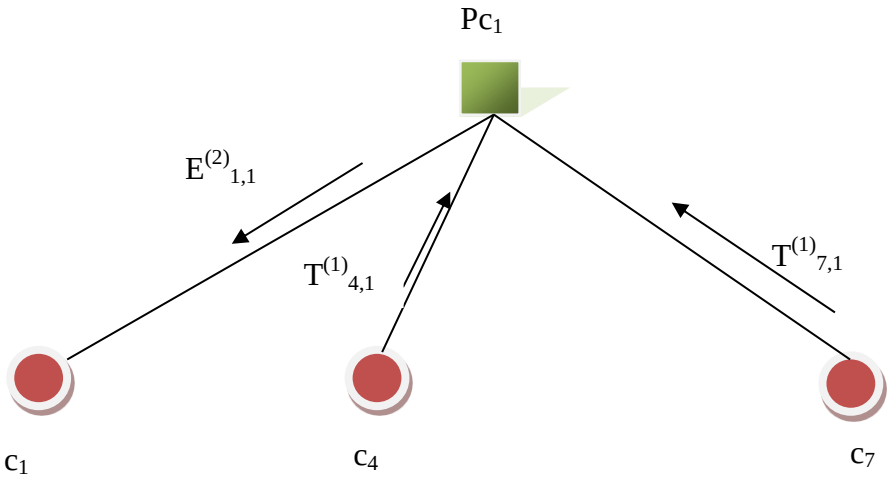
$\hat{c}^{(1)}$ est différent de *c* donc, on continue les itérations.

Itération 2

Mise à jour de l'information extrinsèque (des nœuds de contrôle)

- Calcul de $E_{1,1}^{(2)}$:

On prend une partie du graphe de Tanner contient le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



En remplaçant dans l'équation (C.2), on obtient :

$$E_{1,1}^{(2)} = \prod_{j\epsilon N(1)\backslash 1} sign\left(T_{c_j}^{(1)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 1} f\left(\left|T_{c_j}^{(1)}\right| / r_{i\neq 1}\right)\right)$$

Le signe de $E_{1,1}^{(2)}$ est :

$$sign\left(E_{1,1}^{(2)}\right) = sign\left(T_{c_4}^{(1)}\right) \times sign\left(T_{c_7}^{(1)}\right) = (+1)(+1) = (+1)$$

La valeur absolue de $E_{1,1}^{(2)}$est

$$\left|E_{1,1}^{(2)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 1} f\left(\left|T_{c_j}^{(1)}\right| / r_{i\neq 1}\right)\right) = f(f\left(\left|T_{c_4}^{(1)}\right|\right) + f\left(\left|T_{c_7}^{(1)}\right|\right))$$

Calculant maintenant $f\left(\left|T_{c_4}^{(1)}\right|\right)\ et\ f\left(\left|T_{c_7}^{(1)}\right|\right)$

$$f\left(\left|T_{c_4}^{(1)}\right|\right) = \log\frac{e^{\left|T_{c_4}^{(1)}\right|}+1}{e^{\left|T_{c_4}^{(1)}\right|}-1} = \log\frac{e^{1.4273}+1}{e^{1.4273}-1} = 0.4895$$

$$f\left(\left|T_{c_7}^{(1)}\right|\right) = \log\frac{e^{\left|T_{c_7}^{(1)}\right|}+1}{e^{\left|T_{c_7}^{(1)}\right|}-1} = \log\frac{e^{3.5625}+1}{e^{3.5625}-1} = 0.0568$$

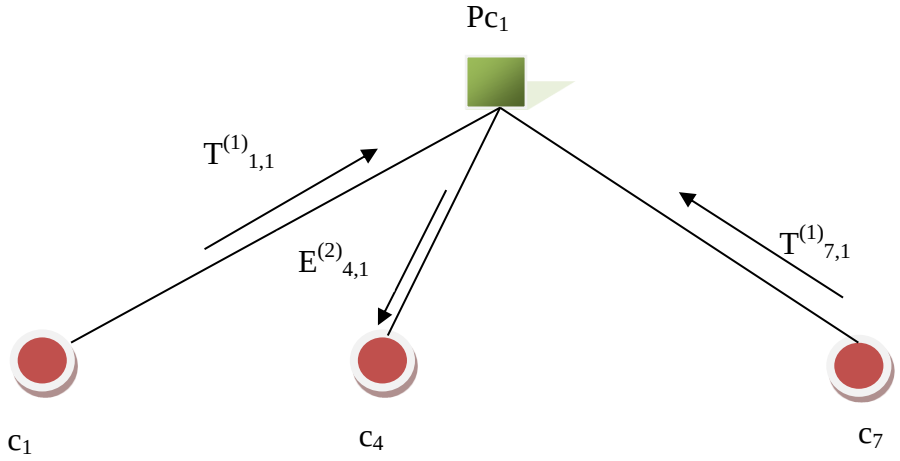
$$\left|E_{1,1}^{(2)}\right| = f(0.4895 + 0.0568) = f(0.5463) = \log\frac{e^{0.5463}+1}{e^{0.5463}-1}$$

$$\left|E_{1,1}^{(2)}\right| = 1.3221$$

$$E_{1,1}^{(2)} = +1.3221$$

- Calcul de $E_{4,1}^{(2)}$:

En prenant la partie du graphe de Tanner contenant le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



En remplaçant dans l'équation (C.2), on obtient :

$$E_{4,1}^{(2)} = \prod_{j\epsilon N(1)\backslash 4} sign\left(T_{c_j}^{(1)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 2} f\left(\left|T_{c_j}^{(1)}\right| / r_{i\neq 4}\right)\right)$$

Le signe de $E_{4,1}^{(2)}$ est :

$$sign\left(E_{4,1}^{(2)}\right) = sign\left(T_{c_1}^{(1)}\right) \times sign\left(T_{c_7}^{(1)}\right) = (+1)(+1) = (+1)$$

La valeur absolue de $E_{4,1}^{(2)}$ est :

$$\left|E_{4,1}^{(2)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 4} f\left(\left|T_{c_j}^{(1)}\right| / r_{i\neq 2}\right)\right) = f\left(f\left(\left|T_{c_1}^{(1)}\right|\right) + f\left(\left|T_{c_7}^{(1)}\right|\right)\right)$$

Calculant maintenant $f\left(\left|T_{c_1}^{(1)}\right|\right)$

$$f\left(\left|T_{c_1}^{(1)}\right|\right) = \log\frac{e^{\left|T_{c_1}^{(1)}\right|}+1}{e^{\left|T_{c_1}^{(1)}\right|}-1} = \log\frac{e^{3.7771}+1}{e^{3.7771}-1} = 0.0458$$

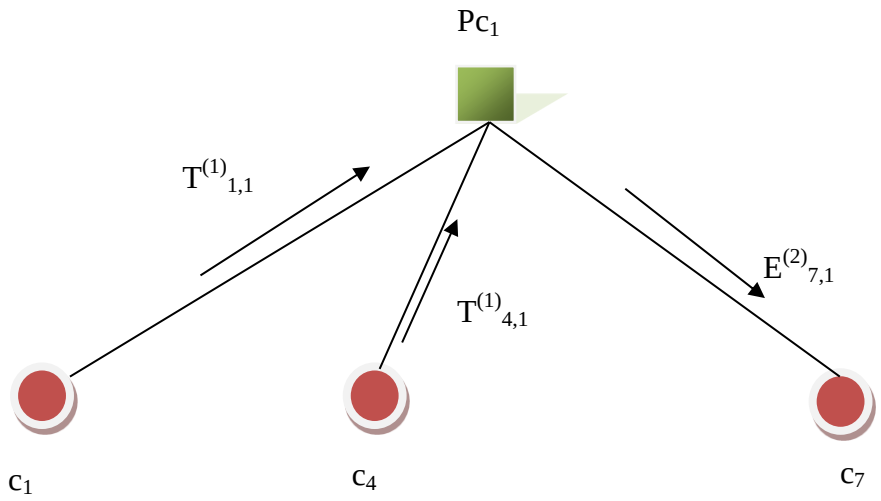
$$\left|E_{4,1}^{(2)}\right| = f(0.0458 + 0.0568) = f(0.1026) = \log\frac{e^{0.1026}+1}{e^{0.1026}-1}$$

$$\left|E_{4,1}^{(2)}\right| = 2.9714$$

$$E_{4,1}^{(2)} = +2.9714$$

-Calcul de $E_{7,1}^{(2)}$:

En prenant la partie du graphe de Tanner contenant le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



En remplaçant dans l'équation (C.2), on obtient :

$$E_{7,1}^{(2)} = \prod_{j\epsilon N(1)\backslash 7} sign\left(T_{c_j}^{(1)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 7} f\left(\left|T_{c_j}^{(1)}\right| / r_{i\neq 7}\right)\right)$$

Le signe de $E_{7,1}^{(2)}$ est :

$$sign\left(E_{7,1}^{(2)}\right) = sign\left(T_{c_1}^{(1)}\right) \times sign\left(T_{c_4}^{(1)}\right) = (+1)(-1) = (+1)$$

La valeur absolue de $E_{7,1}^{(2)}$ est :

$$\left|E_{7,1}^{(2)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 7} f\left(\left|T_{c_j}^{(1)}\right| / r_{i\neq 7}\right)\right) = f\left(f\left(\left|T_{c_1}^{(1)}\right|\right) + f\left(\left|T_{c_4}^{(1)}\right|\right)\right)$$

$$\left|E_{7,1}^{(2)}\right| = f(0.0458 + 0.4895) = f(0.5353) = \log\frac{e^{0.5353}+1}{e^{0.5353}-1}$$

$$\left|E_{7,1}^{(2)}\right| = 1.3415$$

$$E_{7,1}^{(2)} = +1.3415$$

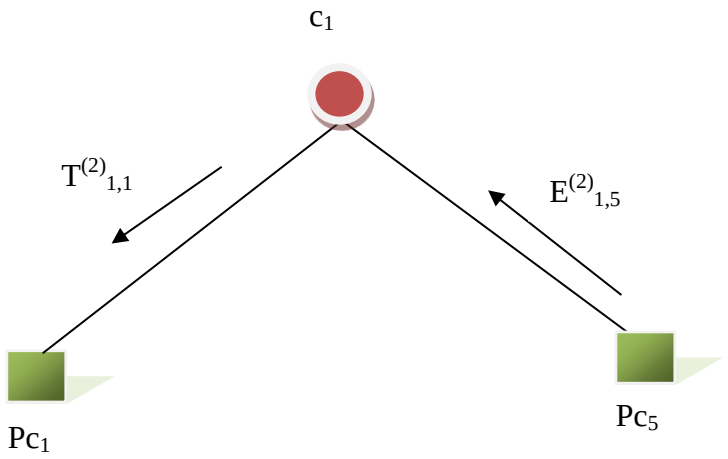
Ainsi de la même façon, on peut calculer toutes les informations extrinsèques (obtenues par programme de calcul) :

$$E_n^{(2)} = \begin{bmatrix} 1.3223 & 0 & 0 & 2.9716 & 0 & 0 & 1.3416 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.0488 & 0 & 0 & -1.5844 & 0 & 0.0488 & 0.0635 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.1477 & 0 & 0 & -0.1501 & 0 & 2.5916 & -0.1624 & 0 & 0 & 0 \\ 0 & -1.5460 & 0 & 1.9762 & 0 & 0 & 0 & 0 & -1.7893 & 1.5748 & 0 & 0 \\ 1.0821 & 0 & 0 & 0 & 0 & -1.0461 & 0 & 0 & 0 & 1.3387 & -1.2072 & 0 \\ 0 & 0 & -0.0889 & 0 & 0.6640 & 0 & 0 & 0 & 0 & 0 & 0.0925 & 0.2560 \end{bmatrix}$$

Mise à jour de l'information partielle (des nœuds de variable) :

- Calcul de $T_{1,1}^{(2)}$:

En prenant la partie du graphe de Tanner contenant le nœud de variable c_1 et les nœuds de contrôle qui lui sont connectés.



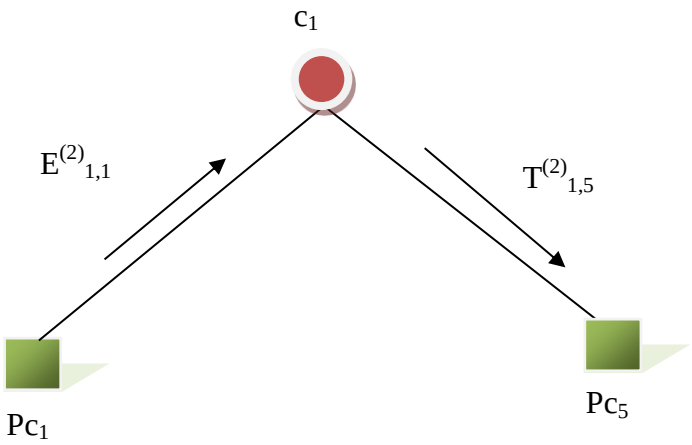
Pour : *n*=1 ; *m*=1, l'équation (C.3) devient :

$$T_{1,1}^{(2)} = I_1 + \sum_{m' \in M(1)/1} E_{1,m'}^{(2)}$$

$$T_{1,1}^{(2)} = I_1 + E_{1,5}^{(2)} = (1.9244) + (0.2360) = 2.1604$$

- Calcul de $T_{1,5}^{(2)}$:

En prenant la partie du graphe de Tanner contenant le nœud de variable c_1 et les nœuds de contrôle qui lui sont connectés.



Pour : *n*=1 ; *m*=5, l'équation (C.3) devient :

$$T_{1,5}^{(2)} = I_1 + \sum_{m' \in M(1)/5} E_{1,m'}^{(2)}$$

$$T_{1,5}^{(2)} = I_1 + E_{1,1}^{(2)} = (2.5606) + (1.3223) = 3.8829$$

Ainsi de la même façon, on déduit les valeurs de $T^{(2)}_{n,m}$ (par programme de calcul) :

$$T_{n,m}^{(2)} = \begin{bmatrix} 3.6427 & 0 & 0 & 1.9496 & 0 & 0 & 3.7300 & 0 & 0 & 0 & 0 & 0 \\ 0 & -5.2238 & 0 & 0 & 0.0220 & 0 & 5.0228 & 2.1844 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5.1185 & 0 & 0 & -4.1379 & 0 & -0.3437 & -4.7321 & 0 & 0 & 0 \\ 0 & -3.7266 & 0 & 2.9450 & 0 & 0 & 0 & 0 & -3.1052 & 3.1797 & 0 & 0 \\ 3.8829 & 0 & 0 & 0 & 0 & -3.2419 & 0 & 0 & 0 & 3.4158 & -2.2137 & 0 \\ 0 & 0 & 5.3551 & 0 & -2.2264 & 0 & 0 & 0 & 0 & 0 & -3.5134 & -0.7164 \end{bmatrix}$$

Mise à jour de l'information totale T_n :

Pour *n*=1, l'équation (C.4) devient :

$$T_1^{(2)} = I_1 + E_{1,1}^{(2)} + E_{1,5}^{(2)} = 2.5606 + (1.3223) + (1.0821) = 4.9650$$

Ainsi de la même façon, on peut calculer toutes les informations totales. Le vecteur de l'information totale $T_n^{(2)}$ (obtenu par un programme de calcul) est :

$T_n^{(2)}$ =[4.9651 -5.2726 5.2662 4.9211 -1.5623 -4.2880 5.0717 2.2479 -4.8944 4.7544 -3.4209 -0.4604]

Selon l'équation (C.5), le mot décodé est :

$$\hat{c}^{(2)} = [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1]$$

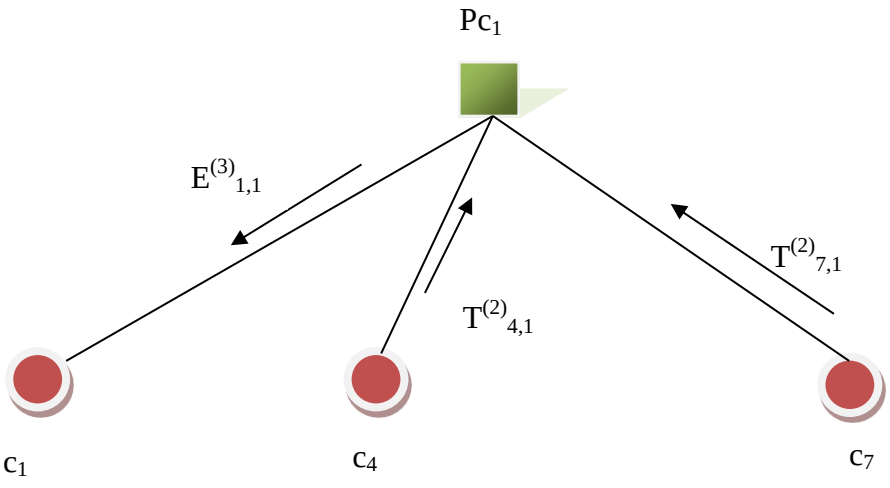
$\hat{c}^{(2)}$ est différent de *c* donc, on continue les itérations.

Itération 3

Mise à jour de l'information extrinsèque (des nœuds de contrôle)

- Calcul de $E_{1,1}^{(3)}$:

En prenant la partie du graphe de Tanner contenant le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



En remplaçant dans l'équation (C.2), on obtient :

$$E_{1,1}^{(3)} = \prod_{j\epsilon N(1)\backslash 1} sign\left(T_{c_j}^{(2)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 1} f\left(\left|T_{c_j}^{(2)}\right| / r_{i\neq 1}\right)\right)$$

Le signe de $E_{1,1}^{(3)}$ est :

$$sign\left(E_{1,1}^{(3)}\right) = sign\left(T_{c_4}^{(2)}\right) \times sign\left(T_{c_7}^{(2)}\right) = (+1)(+1) = (+1)$$

La valeur absolue de $E_{1,1}^{(3)}$ est :

$$\left|E_{1,1}^{(3)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 1} f\left(\left|T_{c_j}^{(2)}\right| / r_{i\neq 1}\right)\right) = f(f\left(\left|T_{c_4}^{(2)}\right|\right) + f\left(\left|T_{c_7}^{(2)}\right|\right))$$

Calculant maintenant $f\left(\left|T_{c_4}^{(2)}\right|\right)$ et $f\left(\left|T_{c_7}^{(2)}\right|\right)$

$$f\left(\left|T_{c_4}^{(2)}\right|\right) = \log\frac{e^{\left|T_{c_4}^{(2)}\right|}+1}{e^{\left|T_{c_4}^{(2)}\right|}-1} = \log\frac{e^{1.9496}+1}{e^{1.9496}-1} = 0.2866$$

$$f\left(\left|T_{c_7}^{(2)}\right|\right) = \log\frac{e^{\left|T_{c_7}^{(2)}\right|}+1}{e^{\left|T_{c_7}^{(2)}\right|}-1} = \log\frac{e^{3.7300}+1}{e^{3.7300}-1} = 0.0480$$

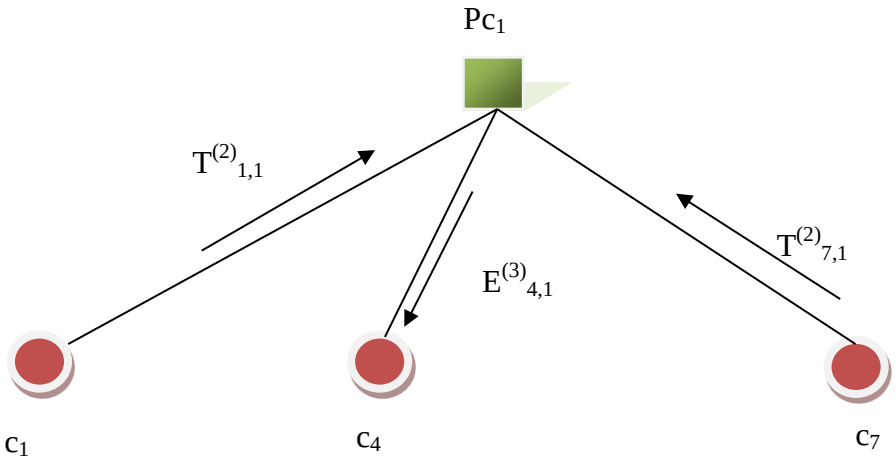
$$\left|E_{1,1}^{(3)}\right| = f(0.2866 + 0.0480) = f(0.3346) = \log\frac{e^{0.3346}+1}{e^{0.3346}-1}$$

$$\left|E_{1,1}^{(3)}\right| = 1.7972$$

$$E_{1,1}^{(3)} = +1.7972$$

- Calcul de $E_{4,1}^{(3)}$:

En prenant la partie du graphe de Tanner contenant le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



En remplaçant dans l'équation (C.2), on obtient :

$$E_{4,1}^{(3)} = \prod_{j\epsilon N(1)\backslash 4} sign\left(T_{c_j}^{(2)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 4} f\left(\left|T_{c_j}^{(2)}\right| / r_{i\neq 4}\right)\right)$$

Le signe de $E_{4,1}^{(3)}$ est :

$$sign\left(E_{4,1}^{(3)}\right) = sign\left(T_{c_1}^{2}\right) \times sign\left(T_{c_7}^{(2)}\right) = (+1)(+1) = (+1)$$

La valeur absolue de $E_{4,1}^{(3)}$ est :

$$\left|E_{4,1}^{(3)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 4} f\left(\left|T_{c_j}^{(2)}\right| / r_{i\neq 2}\right)\right) = f(f\left(\left|T_{c_1}^{(2)}\right|\right) + f\left(\left|T_{c_7}^{(2)}\right|\right))$$

Calculant maintenant $f\left(\left|T_{c_1}^{(2)}\right|\right)$

$$f\left(\left|T_{c_1}^{(2)}\right|\right) = \log\frac{e^{\left|T_{c_1}^{(2)}\right|}+1}{e^{\left|T_{c_1}^{(2)}\right|}-1} = \log\frac{e^{3.6427}+1}{e^{3.6427}-1} = 0.0524$$

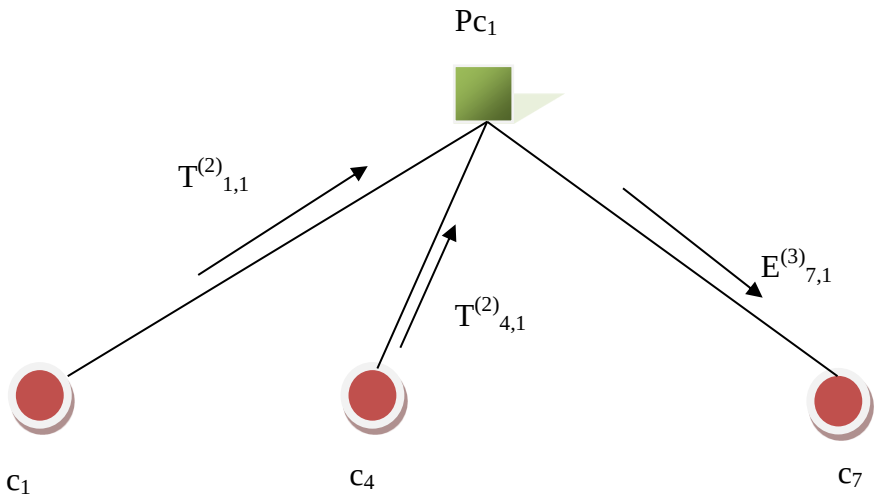
$$\left|E_{4,1}^{(3)}\right| = f(0.0524 + 0.0480) = f(0.1004) = \log\frac{e^{0.1004}+1}{e^{0.1004}-1}$$

$$\left|E_{4,1}^{(3)}\right| = 2.9927$$

$$E_{4,1}^{(3)} = +2.9927$$

- Calcul de $E_{7,1}^{(3)}$:

En prenant la partie du graphe de Tanner contenant le nœud de contrôle Pc_1 et les nœuds de variable qui lui sont connectés.



En remplaçant dans l'équation (C.2), on obtient :

$$E_{7,1}^{(3)} = \prod_{j\epsilon N(1)\backslash 7} sign\left(T_{c_j}^{(2)}\right) \times f\left(\sum_{j\epsilon N(1)\backslash 7} f\left(\left|T_{c_j}^{(2)}\right| / r_{i\neq 7}\right)\right)$$

Le signe de $E_{7,1}^{(3)}$ est :

$$sign\left(E_{7,1}^{(3)}\right) = sign\left(T_{c_1}^{(2)}\right) \times sign\left(T_{c_4}^{(2)}\right) = (+1)(+1) = (+1)$$

La valeur absolue de $E_{7,1}^{(3)}$ est :

$$\left|E_{7,1}^{(3)}\right| = f\left(\sum_{j\epsilon N(1)\backslash 7} f\left(\left|T_{c_j}^{(2)}\right| / r_{i\neq 7}\right)\right) = f(f\left(\left|T_{c_1}^{(2)}\right|\right) + f\left(\left|T_{c_4}^{(2)}\right|\right))$$

$$\left|E_{7,1}^{(3)}\right| = f(0.0524 + 0.2866) = f(0.3390) = \log\frac{e^{0.3390}+1}{e^{0.3390}-1}$$

$$\left|E_{7,1}^{(3)}\right| = 1.7845$$

$$E_{7,1}^{(3)} = +1.7845$$

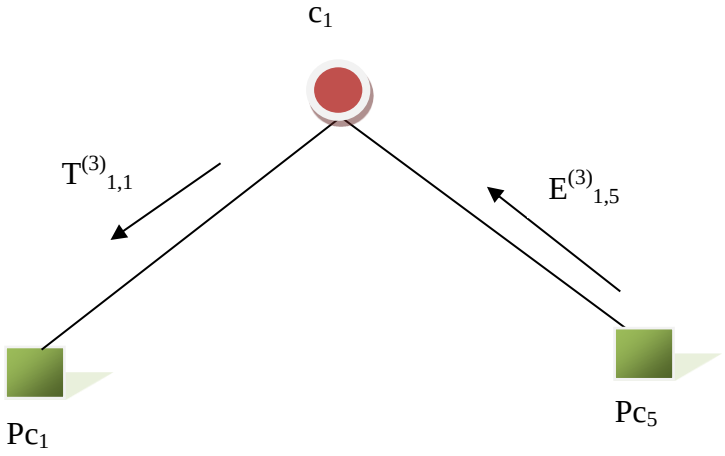
Ainsi de la même façon, on peut calculer toutes les informations extrinsèques (par programme de calcul)

$$E_n^{(3)} = \begin{bmatrix} 1.7972 & 0 & 0 & 2.9929 & 0 & 0 & 1.7845 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.0173 & 0 & 0 & -2.0847 & 0 & -0.0174 & -0.0215 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.3268 & 0 & 0 & 0.3335 & 0 & 3.4821 & 0.3287 & 0 & 0 & 0 \\ 0 & -1.9788 & 0 & 2.2064 & 0 & 0 & 0 & 0 & -2.1387 & 2.1121 & 0 & 0 \\ 1.7163 & 0 & 0 & 0 & 0 & -1.8217 & 0 & 0 & 0 & 1.7848 & -2.3827 & 0 \\ 0 & 0 & -0.5337 & 0 & 0.6649 & 0 & 0 & 0 & 0 & 0 & 0.5625 & 1.9526 \end{bmatrix}$$

Mise à jour de l'information partielle (des nœuds de variable) :

- Calcul de $T_{1,1}^{(3)}$:

En prenant la partie du graphe de Tanner contenant le nœud de variable c_1 et les nœuds de contrôle qui lui sont connectés.



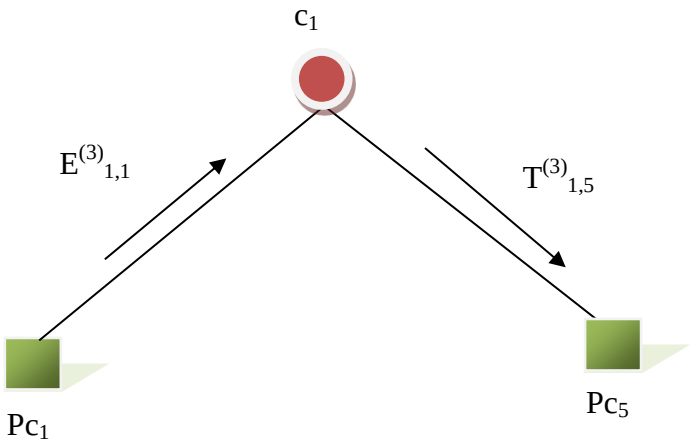
Pour : *n*=1 ; *m*=1, l'équation (C.3), devient :

$$T_{1,1}^{(3)} = I_1 + \sum_{m' \in M(1)/1} E_{1,m'}^{(3)}$$

$$T_{1,1}^{(3)} = I_1 + E_{1,5}^{(3)} = (2.5606) + (1.7163) = 4.2769$$

- Calcul de $T_{1,5}^{(3)}$:

En prenant la partie du graphe de Tanner contenant le nœud de variable c_1 et les nœuds de contrôle qui lui sont connectés.



Pour : *n*=1 ; *m*=5, l'équation (C.3), devient :

$$T_{1,5}^{(3)} = I_1 + \sum_{m' \in M(1)/5} E_{1,m'}^{(3)}$$

$$T_{1,5}^{(3)} = I_1 + E_{1,1}^{(3)} = (2.5606) + (1.7972) = 4.3578$$

Ainsi de la même façon, on déduit les valeurs de $T^{(3)}_{n,m}$ (par application d'un programme de calcul) :

$$T_{n,m}^{(3)} = \begin{bmatrix} 4.2769 & 0 & 0 & 2.1798 & 0 & 0 & 3.6638 & 0 & 0 & 0 & 0 & 0 \\ 0 & -5.6566 & 0 & 0 & 0.0229 & 0 & 5.4657 & 3.0749 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4.6737 & 0 & 0 & -4.9135 & 0 & -0.4287 & -5.0815 & 0 & 0 & 0 \\ 0 & -3.6605 & 0 & 2.9663 & 0 & 0 & 0 & 0 & -2.6141 & 3.6258 & 0 & 0 \\ 4.3578 & 0 & 0 & 0 & 0 & -2.7583 & 0 & 0 & 0 & 3.9531 & -1.7437 & 0 \\ 0 & 0 & 4.8806 & 0 & -2.7267 & 0 & 0 & 0 & 0 & 0 & -4.6889 & -0.7164 \end{bmatrix}$$

Mise à jour de l'information totale T_n :

En remplaçant les valeurs de $E_{1,1}^{(3)}$ *et* $E_{1,5}^{(3)}$ dans l'équation (C.4), la valeur de $T_1^{(3)}$ est :

$$T_1^{(3)} = I_1 + E_{1,1}^{(3)} + E_{1,5}^{(3)} = 2.5606 + (1.7972) + (1.7163) = 6.0741$$

Ainsi de la même façon, on peut calculer toutes les informations totales $T^{(3)}_n$. Le vecteur de l'information totale $T_n^{(3)}$ est :

$T_n^{(3)}$ =[6.0741 -5.6393 4.3469 5.1727 -2.0618 -4.5799 5.4483 3.0534 -4.7528 5.7379 -4.1264 1.2362]

Selon l'équation (C.5), le mot décodé est

$$\hat{c}^{(3)} = [0 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0]$$

Puisque $\hat{c}^{(3)}$ est identique au *c* donc, on arrête les itérations.

D Analyse des cycles

D.1 Procédure de détection des cycles

L'algorithme de détection des cycles contenus dans le graphe de Tanner, proposé par Li et al, est présenté comme suit :

D.1.1 La construction de la matrice relative

La matrice de contrôle de parité *H* est représentée comme :

$$H = [h_{kl}]_{(N-K)\times N}$$

où $h_{kl} = 0$ ou 1 et *k*=1,…, *M* et *l*=1,…..*N*.

où *N* et *M* représente respectivement le nombre de colonnes et de lignes de la matrice *H*.

Les éléments ha_{kl} de la nouvelle matrice de contrôle de parité H_a sont obtenus, à partir de la matrice de contrôle de parité *H*, selon l'équation (D.1) :

$$\begin{cases} Si\ h_{kl} = 0 & \text{alors } ha_{kl} = 0 \\ sinon & ha_{kl} = L_x \end{cases} \quad \text{(D.1)}$$

où L_x et *x* représentent respectivement les connexions et leurs numéros entre le nœud *k* et le nœud *l*.

La matrice relative *Mr* est construite, à partir de la matrice identité *I* et la nouvelle matrice H_a, selon l'équation (D.2) :

$$Mr = \begin{bmatrix} I_{KXK} & Ha^T \\ Ha & I_{(N)X(N)} \end{bmatrix} \quad \text{(D.2)}$$

Les éléments de la matrice relative mr_{ij} sont définis comme suit :

1. Si $i=j$ donc, $mr_{ij}=1$
2. Si $i\neq j$ donc, $mr_{ij} = 0$ ou $mr_{ij}=L_x$

où $i = 1, \dots, N+K$ et $j = 1, \dots, N+K$

D.1.2 Calcul des matrices relatives intégrées

Afin d'obtenir tous les cycles passant par un nœud, l'algorithme doit calculer tous les chemins qui émergent de ce nœud vers les autres nœuds. La matrice relative intégrée, notée $Mr_i^{(n-1)}$, permet de donner tous les chemins possibles entre le premier nœud ,dénommé nœud de source, et les autres nœuds (nœuds de variable et de contrôle).

La matrice relative intégrée $Mr_{ij}^{(n-1)}$ est représentée selon l'équation (D.3) :

$$Mr_i^{(n-1)} = \left[mr_{ij}^{(n-1)}\right] \tag{D.3}$$

où *n, n-1, i* et j représentent respectivement le nombre total des nœuds (nœuds de variable + nœuds de contrôle), le temps d'intégration, l'indice du nœud de source et l'indice des autres nœuds.

D.1.3 Le calcul de tous les cycles passant par un nœud de source

Les éléments de la matrice relative $mr_{ij}^{(n-1)}$ sont définis à partir de l'équation (D.4) comme suit :

$$mr_{ij}^{(n-1)} = L_a L_b \dots L_c + L_d L_e \dots L_f + \cdots + L_g \tag{D.4}$$

où *a,b,c,d,e,f,g* prenant leurs valeurs de 1,2,…, *n.*

L_g : le point de connexion directe correspondant.

On peut modifier l'équation (D.4) pour obtenir les cycles passant du nœud *i* vers le nœud *j* comme suit :

$$C_{ij} = (mr_{ij}^{(n-1)} - L_g).L_g = L_a L_b \dots L_c L_g + L_d L_e \dots L_f L_g \tag{D.5}$$

En utilisant tous les nœuds (de variable et de contrôle) comme des nœuds de source, on peut obtenir tous les cycles contenus dans le graphe de Tanner.

D.2 Application de la méthode détection des cycles

Exemple

Soit *H* la matrice de contrôle d'un code *LDPC* de l'exemple 2, avec (*M*=6, *N*=12) représentée comme suit :

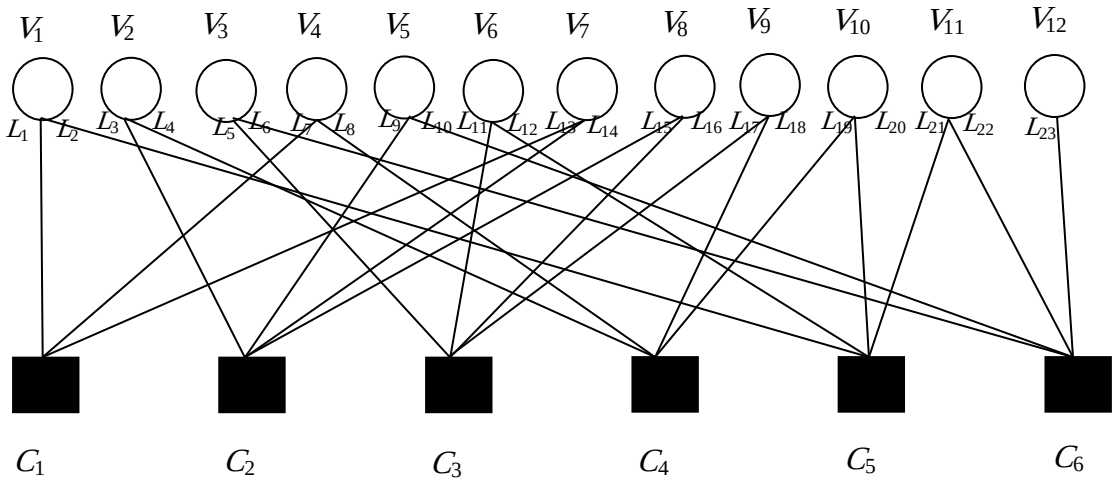
$$H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0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

Etape 1 : Construction de la matrice *Ha*

La matrice *Ha* est donnée selon l'équation (D.1) comme suit :

$$\boldsymbol{Ha} = \begin{pmatrix} L_1 & 0 & 0 & L_7 & 0 & 0 & L_{13} & 0 & 0 & 0 & 0 & 0 \\ 0 & L_3 & 0 & 0 & L_9 & 0 & L_{14} & L_{15} & 0 & 0 & 0 & 0 \\ 0 & 0 & L_5 & 0 & 0 & L_{11} & 0 & L_{16} & L_{17} & 0 & 0 & 0 \\ 0 & L_4 & 0 & L_8 & 0 & 0 & 0 & 0 & L_{18} & L_{19} & 0 & 0 \\ L_2 & 0 & 0 & 0 & 0 & L_{12} & 0 & 0 & 0 & L_{20} & L_{21} & 0 \\ 0 & 0 & L_6 & 0 & L_{10} & 0 & 0 & 0 & 0 & 0 & L_{22} & L_{23} \end{pmatrix}$$

Le graphe de Tanner de la matrice *Ha* est illustré comme suit :



Etape 2 : Construction de la matrice relative *Mr*

La matrice relative *Mr* est donnée comme suit (voir équation D.2) :

$$Mr=\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_1 & 0 & 0 & 0 & L_2 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_3 & 0 & L_4 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_5 & 0 & 0 & L_6 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_7 & 0 & 0 & L_8 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_9 & 0 & 0 & 0 & L_{10} \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & L_{11} & 0 & L_{12} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & L_{13} & L_{14} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & L_{15} & L_{16} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & L_{17} & L_{18} & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & L_{19} & L_{20} & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & L_{21} & L_{22} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & L_{23} \\
L_1 & 0 & 0 & L_7 & 0 & 0 & L_{13} & L_{14} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & L_3 & 0 & 0 & L_9 & 0 & 0 & L_{15} & L_{16} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & L_5 & 0 & 0 & L_{11} & 0 & 0 & L_{17} & L_{18} & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & L_4 & 0 & L_8 & 0 & 0 & 0 & 0 & 0 & L_{19} & L_{20} & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
L_2 & 0 & 0 & 0 & 0 & L_{12} & 0 & 0 & 0 & 0 & L_{21} & L_{22} & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & L_6 & 0 & L_{10} & 0 & 0 & 0 & 0 & 0 & 0 & L_{23} & 0 & 0 & 0 & 0 & 0 & 1
\end{pmatrix}$$

Etape 3 : Obtention des cycles.

En prenant le nœud ‘1’ comme un nœud de source, l’équation (D.3) devient :

$$Mr_1^{(17)} = [mr_{1\,1}^{(17)}\ mr_{1\,2}^{(17)}\ mr_{1\,3}^{(17)}\ mr_{1\,4}^{(17)}\ mr_{1\,5}^{(17)}\ mr_{1\,6}^{(17)}\ mr_{1\,7}^{(17)}\ mr_{1\,8}^{(17)}\ mr_{1\,9}^{(17)}\ mr_{1\,10}^{(17)}\ mr_{1\,11}^{(17)}\ mr_{1\,12}^{(17)}\ mr_{1\,13}^{(17)}\ mr_{1\,14}^{(17)}\ mr_{1\,15}^{(17)}\ mr_{1\,16}^{(17)}\ mr_{1\,17}^{(17)}\ mr_{1\,18}^{(17)}]$$

Dans la matrice $Mr_1^{(17)}$, les éléments qui possédent une connexion directe sont : $m_{1\,13}^{(17)}$ et $m_{1\,17}^{(17)}$, ces éléments sont donnés selon l’équation (D.4) comme suit :

$$m_{1\,13}^{(17)} = L_2L_{12}L_{11}L_5L_6L_{10}L_9L_3L_4L_8L_7 + L_2L_{12}L_{11}L_{16}L_{15}L_{14}L_{13} + L_2L_{12}L_{11}L_{17}L_{18}L_8L_7 + L_2L_{12}L_{11}L_{16}L_{15}L_3L_4L_8L_7 + L_2L_{20}L_{19}L_{18}L_{17}L_{16}L_{15}L_{14}L_{13} + L_2L_{20}L_{19}L_8L_7 + L_2L_{20}L_{19}L_4L_3L_{14}L_{13} + L_2L_{21}L_{22}L_{10}L_9L_3L_4L_8L_7 + L_2L_{21}L_{22}L_{10}L_9L_{14}L_{13} + L_2L_{21}L_{22}L_6L_5L_{16}L_{15}L_{14}L_{13} + L_2L_{21}L_{22}L_6L_5L_{17}L_{18}L_8L_7 + L_1$$

$$m_{1\,17}^{(17)} = L_1L_7L_8L_4L_3L_9L_{10}L_6L_5L_{11}L_{12} + L_1L_{13}L_{14}L_{15}L_{16}L_{11}L_{12} + L_1L_7L_8L_{18}L_{17}L_{11}L_{12} + L_1L_7L_8L_4L_3L_{15}L_{16}L_{11}L_{12} + L_1L_{13}L_{14}L_{15}L_{16}L_{17}L_{18}L_{19}L_{20} + L_1L_7L_8L_{19}L_{20} + L_1L_{13}L_{14}L_3L_4L_{19}L_{20} + L_1L_7L_8L_4L_3L_9L_{10}L_{22}L_{21} + L_1L_{13}L_{14}L_9L_{10}L_{22}L_{21} + L_1L_{13}L_{14}L_{15}L_{16}L_5L_6L_{22}L_{21} + L_1L_7L_8L_{18}L_{17}L_5L_6L_{22}L_{21} + L_2$$

Les cycles associés au nœud '1' (dans notre travail on s'interesse aux cycles de longueur 4) sont donnés, à partir des éléments $m_{1\,13}^{(17)}\ et\ \ m_{1\,17}^{(17)}$, selon l'équation (D.5)

$$C_{1\,13} = L_2L_{12}L_{11}L_5L_6L_{10}L_9L_3L_4L_8L_7L_1 + L_2L_{12}L_{11}L_{16}L_{15}L_{14}L_{13}L_1 + L_2L_{12}L_{11}L_{17}L_{18}L_8L_7L_1 + L_2L_{12}L_{11}L_{16}L_{15}L_3L_4L_8L_7L_1 + L_2L_{20}L_{19}L_{18}L_{17}L_{16}L_{15}L_{14}L_{13}L_1 + L_2L_{20}L_{19}L_8L_7L_1 + L_2L_{20}L_{19}L_4L_3L_{14}L_{13}L_1 + L_2L_{21}L_{22}L_{10}L_9L_3L_4L_8L_7L_1 + L_2L_{21}L_{22}L_{10}L_9L_{14}L_{13}L_1 + L_2L_{21}L_{22}L_6L_5L_{16}L_{15}L_{14}L_{13}L_1 + L_2L_{21}L_{22}L_6L_5L_{17}L_{18}L_8L_7L_1 \quad .$$

$$C_{1\,17} = L_1L_7L_8L_4L_3L_9L_{10}L_6L_5L_{11}L_{12}L_2 + L_1L_{13}L_{14}L_{15}L_{16}L_{11}L_{12}L_2 + L_1L_7L_8L_{18}L_{17}L_{11}L_{12}L_2 + L_1L_7L_8L_4L_3L_{15}L_{16}L_{11}L_{12}L_2 + L_1L_{13}L_{14}L_{15}L_{16}L_{17}L_{18}L_{19}L_{20}L_2 + L_1L_7L_8L_{19}L_{20}L_2 + L_1L_{13}L_{14}L_3L_4L_{19}L_{20}L_2 + L_1L_7L_8L_4L_3L_9L_{10}L_{22}L_{21}L_2 + L_1L_{13}L_{14}L_9L_{10}L_{22}L_{21}L_2 + L_1L_{13}L_{14}L_{15}L_{16}L_5L_6L_{22}L_{21}L_2 + L_1L_7L_8L_{18}L_{17}L_5L_6L_{22}L_{21}L_2 \quad .$$

Il n'existe aucun cycle de longueurs 4, le plus petit cycle associé au nœud '1' est de longueur 6.

En prenant tous les nœuds comme des nœuds de source, on constate l'inexistence des cycles de longueur 4.

Exemple 2

De (l'exemple 1, annexe B), la matrice de contrôle de parité *H* de dimension $M \times N = (12 \times 24)$ ci-dessous :

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

En prenant le nœud ‘1’ comme un nœud de source et en utilisant les équations (D.1), (D.2), (D.3), (D.4) et (D.5), les cycles associés (dans notre travail, on s’intéresse aux cycles de longueur 4) au nœud ‘1’ sont :

$$C_{1\,25} = L_2L_{14}L_{13}L_3L_4L_{16}L_{15}L_5L_6L_{18}L_{17}L_1 + L_2L_{14}L_{13}L_{26}L_{25}L_1$$
$$+ L_2L_{14}L_{13}L_{27}L_{28}L_3L_4L_{16}L_{15}L_5L_6L_{18}L_{17}L_1$$
$$+ L_2L_{38}L_{37}L_{36}L_{35}L_{34}L_{33}L_{32}L_{31}L_{30}L_{29}L_{28}L_{27}L_{26}L_{25}L_1 + L_2L_{38}L_{37}L_{18}L_{17}L_1$$
$$+ L_2L_{39}L_{40}L_{16}L_{15}L_5L_6L_{18}L_{17}L_1 + L_2L_{38}L_{37}L_6L_5L_{28}L_{25}L_1$$
$$+ L_2L_{14}L_{13}L_3L_4L_{40}L_{39}L_{38}L_{37}L_{36}L_{35}L_{34}L_{33}L_{32}L_{31}L_{30}L_{29}L_{28}L_{27}L_{26}L_{25}L_1$$

$$C_{1\,32} = L_1L_{17}L_{18}L_6L_5L_{15}L_{16}L_4L_3L_{13}L_{14}L_2 + L_1L_{25}L_{26}L_{13}L_{14}L_2$$
$$+ L_1L_{17}L_{18}L_6L_5L_{15}L_{16}L_4L_3L_{28}L_{27}L_{13}L_{14}L_2 + L_1L_{25}L_{26}L_{27}L_{28}L_5L_6L_{37}L_{38}L_2$$
$$+ L_1L_{25}L_{26}L_{27}L_{28}L_{29}L_{30}L_{31}L_{32}L_{33}L_{34}L_{35}L_{36}L_{37}L_{38}L_2 + L_1L_{17}L_{18}L_{37}L_{38}L_2$$
$$+ L_1L_{25}L_{26}L_{27}L_{28}L_{29}L_{30}L_{31}L_{32}L_{33}L_{34}L_{35}L_{36}L_{37}L_{38}L_{39}L_{40}L_4L_3L_{13}L_{14}L_2$$

Il existe des cycles, de longueur 6, 8, 10, 12, 14, 16, 22 associé au nœud ‘1’ et on constate l’inexistence des cycles de longueur 4.

En prenant tous les nœuds comme des nœuds de source, on montre que les cycles de longueurs 4 sont absents.

Bibliographie

[1] C. E. Shannon, "A Mathematical Theory of Communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, 1948.

[2] I. Adjudeanu, "Codes correcteurs d'erreurs LDPC structurés," Mémoire de master: Faculte de Sciences et de Genie Universite Laval, Québec, 2010.

[3] R. Gallager, "Low-Density Parity-Check Codes," Report: MA, MIT Press, Cambridge, 1963.

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

[5] R. Tanner, "A Recursive Approach to Low Complexity Codes," *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533-547, 1981.

[6] D. Mackay and R. M. Neal, "Near Shannon Limit Performance of Low Density," *Electronics Letters*, vol. 32, no. 18, p. 1645, 1996.

[7] D. Mackay, "Relationships between sparse graph codes," Information-based Induction Sciences Edition, 2000.

[8] D. Mackay and R. Neal, "Good codes based on very sparse matrices," *IEEE Transactions On Information Theory*, vol. 45, no. 2, pp. 399-431, 1999.

[9] N. Wiberg, Codes and Decoding, PhD. Dissertation, Linköping, University Linköping, Sweden, 1996.

[10] M. Sipser and D. A. Spielman, "Expander codes," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1710-1722, 1996.

[11] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *IEEE International Conference on Communications, Technical Program, Conference Record*, Geneva, 1993.

[12] J. R. Barry, "Low-density parity-check codes," Report, Georgia Institute of Technology, Georgia, 2001.

[13] C. Berrou, Code and Turbo-code, Bretagne: ENST Bretagne Edition, 2010.

[14] F. Guilloud, "Architecture générique de décodeur de codes LDPC," PhD. Dissertation, Télécom ParisTech, Paris, 2004.

[15] M. Senouci, "Etude des codes LDPC (Low Density Parity Check Codes)," Mémoire de fin d'étude, University Djillali Liabes, Sidi Bel Abbes, 2010.

[16] G. Forney, "The viterbi algorithm," *Proceedings of the IEEE,* vol. 61, no. 3, pp. 268-278, 1973.

[17] D. Mackay, Information theory, inference and learning algorithms, Cambridge: Cambridge University Press, 2003.

[18] L. Wehenkel, "Théorie de l'information et du codage," Report, Univérsité de liège, Faculté des sciences appliquées, Liège, 2003.

[19] S. Mahmoud, "Contribution au décodage à décision douce des codes correcteurs en bloc," PhD. Dissertation, Télécom Bretagne; Université de Bretagne Occidentale, Brest, 2016.

[20] J. Dion, "Etude et implémentation d'une architecture de décodage générique et flexible pour codes correcteurs d'erreurs avancés," PhD. Dissertation, Télécom Bretagne, Université de Bretagne Occidentale, Brest, 2013.

[21] M. Diouf, "Conception Avancée des codes LDPC binaires pour des applications pratiques," PhD. Dissertation, Université Cergy Pontoise; Université Cheikh Anta DIOP, Paris, 2015.

[22] R. W. Hamming, "Error detecting and error correcting codes," *Bell Labs Technical Journal,* vol. 29, no. 2, pp. 147-160, 1950.

[23] C. Modlin, "Error Control Coding in DSL Systems," in *Fundamentals of DSL Technology*, London, Taylor and Francis Group, 2005, p. 233.

[24] J. Massey, "Step-by-step decoding of the Bose-Chaudhuri-Hocquenghem codes," *IEEE Transactions on Information Theory,* vol. 11, no. 4, pp. 580-585, 1965.

[25] K. Serir, "Application des codes correcteurs d'erreurs Reed Muller,"Mémoire de Master, Université Abou Bakr Belkaid, Tlemcen, 2011.

[26] B. Vucetic, V. Ponapalam and J. Vu, "Low complexity soft decision decoding algorithms for reed-solomon codes," *IEICE transactions on communications,* vol. 84, no. 3, pp. 392-399, 2001.

[27] A. Yahya, O. Sidek and M. Salleh, "A new quasi-cyclic low density parity check codes," in *IEEE Symposium on Industrial Electronics & Applications, ISIEA*, Kuala Lumpur, 2009.

[28] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE transactions on Information Theory*, vol. 13, no. 2, pp. 260-269, 1967.

[29] ETSI, E. N. 302 307 v1. 1.1, "Digital Video Broadcasting (DVB): Second generation framing structure, channel coding and modulation system for Broadcasting. Interactive Services, News Gathering and other broadband satellite applications," Report, European Telecommunications Standards Institute, Paris, 2004.

[30] T. Brack, M. Alles and F. Kienle, "A synthesizable IP core for WIMAX 802.16 e LDPC code decoding," in *IEEE 17th International Symposium on Personal, Indoor and Mobile Radio Communications*, Helsinki, 2006.

[31] IEEE 802 LAN/MAN STANDARDS COMMITTEE, "IEEE Standard for Information technology-Telecommunication and information exchange between systems-Local and metropolitan area networks-Specific requirements Part11," Report, IEEE, London, 2009.

[32] J. Dore, "Optimisation conjointe de codes LDPC et de leurs architectures de d'ecodage et mise en oeuvre sur FPGA," PhD. Dissertation, INSA, Rennes, 2007.

[33] T. J. Richardson and R. L. Urbanke, "Efficient encoding of low-density parity-check codes," *IEEE transactions on information theory*, vol. 47, no. 2, pp. 638-656, 2001.

[34] J. Dore, M. Hamon and P. Penard, "A structured LDPC code construction for efficient encoder design," in *IEEE International Conference on Communications, ICC'06*, Paris, 2006.

[35] J. Wolf, "Efficient maximum likelihood decoding of linear block codes using a trellis," *IEEE Transactions on Information Theory*, vol. 24, no. 1, pp. 76-80, 1978.

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

[37] J. Hagenauer and P. Hoeher, "A Viterbi Algorithm with soft-decision outputs and its applications," in *Proceedings of the IEEE Globecom'89*, Dallas, 1989.

[38] T. Moon, Error correction coding. Mathematical Methods and Algorithms, Logan: Jhon Wiley and Son, Edition, 2005.

[39] T. Richardson and R. Urbanke, "Multi-edge type LDPC codes," in *Workshop honoring Prof. Bob McEliece on his 60th birthday, California Institute of Technology*, Pasadena, 2002.

[40] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," *IPN progress report*, vol. 42, no. 154, pp. 142-154, 2003.

[41] K. Wang, Y. Xiao and K. Kim, "Construction of protograph LDPC codes with circular generator matrices," *Journal of Systems Engineering and Electronics*, vol. 22, no. 5, pp. 840-847, 2011.

[42] D. Mitchell, R. Smarandache and M. Lentmaier, "Quasi-cyclic asymptotically regular LDPC codes," in *IEEE Information Theory Workshop (ITW)*, Dublin, 2010.

[43] Z. Li, C. L and L. Zeng, "Efficient encoding of quasi-cyclic low-density parity-check codes," *IEEE Transactions on Communications*, vol. 54, no. 1, pp. 71-81, 2006.

[44] Y. Wang, J. Yedidia and S. Draper, "Construction of high-girth QC-LDPC codes," in *IEEE 5th International Symposium onTurbo Codes and Related Topics*, Lausanne, 2008.

[45] L. Chen, J. Xu and I. Djurdjevic, "Near-Shannon-limit quasi-cyclic low-density parity-check codes," *IEEE Transactions on Communications,* vol. 52, no. 7, pp. 1038-1042, 2004.

[46] S. Johnson and S. R. Weller, "Practical Interleavers for Repeat--Accumulate Codes," *IEEE Transactions on Communications*, vol. 57, no. 5, 2009.

[47] R. Tanner, "On quasi-cyclic repeat-accumulate codes," in *Proceedings of the Annual Alerton Conference on Communication Control and Compting*, Illinois, 1999.

[48] B. Vasic and O. Milenkovic, "Combinatorial constructions of low-density parity-check codes for iterative decoding," *IEEE Transactions on information theory,*

vol. 50, no. 6, pp. 1156-1176, 2004.

[49] B. Vasic, A. Kuznetsov and E. Kurtas, "Lattice low-density parity check codes and their application in partial response systems," in *IEEE International Symposium on Information Theory, Proceedings*, Laussane, 2002.

[50] E. M. Kurtas and B. Vasic, Advanced error control techniques for data storage systems, Boca Raton: CRC Press, 2005.

[51] A. Y. Lulu, "Construction Of Ldpc Codes Using Randomly Permutated Copies Of Parity Check Matrix," Master Thesis, Islamic University Of Gaza, Gaza, 2012.

[52] S. Chung, G. D. Forney and T. J. Richardson, "On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit," *IEEE Communications letters,* vol. 5, no. 2, pp. 58-60, 2001.

[53] H. Zhong and T. Zhang, "Block-LDPC: A practical LDPC coding system design approach," *IEEE Transactions on Circuits and Systems I: Regular Papers,* vol. 52, no. 4, pp. 766-775, 2005.

[54] S. Lin and D. J. Costello, Error control coding, India: Pearson Education India Edition, 2004.

[55] M. G. Luby, M. Mitzenmacher and M. Shokrollahi, "Improved low-density parity-check codes using irregular graphs," *IEEE Transactions on Information Theory,* vol. 47, no. 2, pp. 585-598, 2001.

[56] S. Myung, K. Yang and J. Kim, "Quasi-cyclic LDPC codes for fast encoding," *IEEE Transactions on Information Theory,* vol. 51, no. 8, pp. 2894-2901, 2005.

[57] S. Johnson, "Low-density parity-check codes from combinatorial designs," PhD, Dissertation, University of Newcastle, Newcastle, 2003.

[58] R. Echard, "On the construction of some deterministic low-density parity-check codes," PhD, Disertation, George Mason University, Washington, 2003.

[59] K. Liu, Z. Fei and J. Kuang, "A novel algorithm for removing cycles in quasi-cyclic LDPC codes," in *IEEE 20th International Symposium on Personal, Indoor and Mobile Radio Communications*, Tokyo, 2009.

[60] J. Li, K. Liu and S. Lin, "Algebraic quasi-cyclic LDPC codes: Construction, low error-floor, large girth and a reduced-complexity decoding scheme," *IEEE*

Transactions on communications, vol. 62, no. 8, pp. 2626-2637, 2014.

[61] D. Nguyen, S. K. Chilappagari and M. W. Marcellin, "On the construction of structured LDPC codes free of small trapping sets," *IEEE Transactions on Information Theory,* vol. 58, no. 4, pp. 2280-2302, 2012.

[62] H. Khalil, "Matrices structurées et matrices de Toeplitz par blocs de Toeplitz en calcul numérique et formel," PhD, Dissertation, Université Claude Bernard-Lyon I, Lyon, 2008.

[63] Q. Guo-lei and Z. Dong, "Design of structured LDPC codes with quasi-cyclic and rotation architecture," in *3rd International Conference on Advanced Computer Theory and Engineering (ICACTE)*, Lianyungang, 2010.

[64] M. Yang, W. Ryan and Y. Li, "Design of efficiently encodable moderate-length high-rate irregular LDPC codes," *IEEE Transactions on Communications,* vol. 52, no. 4, pp. 564-571, 2004.

[65] A. Ahmed and Y. Hu, "3M relationship pattern for detection and estimation of unknown frequencies for unknown number of sinusoids based on Eigenspace analysis of Hankel matrix," PhD, Dissertation, School of Engineering, Design and Technology, London, 2013.

[66] M. A. Tehami and A. Djebbari, "Low-Density-Parity-Check Codes constructed from Hankel Matrices," *Journal of Telecommunications and Information Technology,* no. 3, 2018.

[67] M. Tehami and Djebbari A, "New Method to build Low-Density-Parity-Check Codes," *International Journal of Technology*, p. submitted and révised, 2018.

[68] R. Chen, H. Hunag and G. Xiao, "Relation between parity-check matrixes and cycles of associated Tanner graphs," *IEEE communications letters,* vol. 11, no. 8, pp. 674-676, 2007.

[69] B. Li, G. Wang and H. Yang, "A new method of detecting cycles in Tanner graph of LDPC codes," in *International Conference on Wireless Communications & Signal Processing, WCSP*, Nanjing, 2009.

[70] D. Fu-sheng, "An Algorithm to Calculate Entire Routes between Communication Network Nodes based on Logic Algebra," *Mathematics in Practice and Theory,* vol. 2, p. 30, 2006.

[71] W. Sulek, "Seed graph expansion for construction of structured LDPC codes," in *6th International Symposium on Wireless Communication Systems, ISWCS*, Tuscany, 2009.

[72] L. Ping, W. K. Leung and N. Phamdo, "Low density parity check codes with semi-random parity check matrix," *Electronics Letters*, vol. 35, no. 1, pp. 38-39, 1999.

[73] J. M. Moura, J. Lu and H. Zhang, "Structured Low-Density-Parity-Check Codes," *IEEE Signal Processing Magazine*, vol. 21, no. 1, pp. 42-55, 2004.

[74] X. Hu, E. Eleftheriou and D. M. Arnold, "Regular and Irregular Progressive Edge-Growth," *IEEE Transaction On Information Theory*, vol. 51, no. 1, pp. 386-398, 2005.