Décodage des codes LDPC définis sur des groupes abéliens

Alban Goupil[1], Maxime Colas[1], Guillaume Gelle[1]

[1] CreSTIC — Université de Reims Champagne-Ardenne
Moulin de la Housse, BP 1039, 51687 Reims Cedex 2, France
prénom.nom@univ-reims.fr

Résumé – Une large classe de codes LDPC est définie et qui comprend les codes LDPC définis sur les corps finis, les anneaux et les groupes mais aussi des codes non-linéaires. Cela est rendu possible en étendant la notion de parité du code. Un algorithme rapide de type propagation de croyance est développé pour ce type de parité. Des exemples montrent des utilisations possibles de tels codes.

Abstract – We introduce a wide class of LDPC codes, large enough to include LDPC codes over finite fields, rings or groups as well as some non-linear codes. This class is defined by an extension of the parity-check equations involved in the code's definition. A belief propagation decoding procedure with the same complexity as for the decoding of LDPC codes over finite fields is presented for theses new parities. Examples are given that illustrate the interest of this new code family.

1 Introduction

Depuis la redécouverte par MacKay [1] des codes de Gallager [2], beaucoup de généralisations ont été proposées. Il est possible de les diviser en deux catégories. D'un côté l'irrégularité du graphe factoriel définissant le code a permis d'approcher asymptotiquement la capacité du canal [3, 4]. D'un autre côté, les codes LDPC non-binaires [5, 6] se sont montrés efficaces pour de faible taille de mot.

Cependant, l'algorithme de décodage est toujours issu de l'algorithme de la propagation de croyance (BP). Cet algorithme est basé sur la représentation graphique des codes. Il est suffisamment général pour pouvoir prendre en compte des parités un peu particulières comme celle définissant des codes non-binaires. Par contre sa complexité calculatoire dépend grandement de la spécialisation du calcul des messages envoyés par les noeuds de parité. En effet ces noeuds représentent les contraintes définissant le codes ; or celles-ci peuvent être particulièrement complexes. Suivant le type de code, il est toutefois possible d'obtenir un algorithme rapide. Par exemple, l'article [7] présente un algorithme BP rapide dans le cas des codes LDPC définis sur les corps finis GF(2^m).

La contribution de cet article est double. Dans un premier temps, nous montrons qu'il est possible de rendre les contraintes des noeuds de parité très générales sans toutefois sacrifier à la structure des codes LDPC. En effet, il n'est pas nécessaire de spécialiser le domaine des symboles du code outre mesure. La notion algébrique de groupe abélien suffit amplement. Par conséquent les codes LDPC sur des corps ou des anneaux deviennent des cas particuliers des codes LDPC définis sur des groupes. La seconde partie de cet article montre qu'un algorithme BP existe et qu'il est aussi rapide que l'algorithme BP spécifique aux codes LDPC définis sur GF(2^m). Grâce à cet algorithme, les codes LDPC définis sur des groupes ne sont pas qu'une simple généralisation purement théorique.

2 Codes LDPC sur les groupes

Les codes LDPC sont habituellement définis par une matrice de parité creuse. Chaque parité représente une contrainte que les mots de codes doivent satisfaire. Pour les codes définis sur un corps $\mathbb{K}$, ces parités sont linéaires :

$$\sum_i h_i \, v_i \equiv 0 \quad \text{dans } \mathbb{K} \tag{1}$$

Les symboles des mots de code v_i appartiennent à $\mathbb{K}$ et les multiplications par les éléments h_i de la matrice se font dans ce corps.

L'opération principale dans (1) est l'addition. La multiplication par h_i ne représente qu'un moyen de mixage des éléments à l'intérieur du corps. La généralisation que nous proposons consiste à ne garder que l'aspect additif. La structure algébrique la plus simple qui définit l'addition, la soustraction et la présence d'un zéro est le groupe. Pour des raisons de décodage, ces groupes sont pris abéliens (commutatifs). Nous supposerons maintenant que les symboles v_i des mots de code appartiennent à un groupe abélien G. Par ailleurs, les h_i des contraintes (1) deviennent de simples fonctions de G dans G. Les contraintes de parité s'écrivent alors :

$$\sum_i h_i(v_i) \equiv 0 \quad \text{dans } G \tag{2}$$

L'aspect creux du tableau contenant les fonctions h_i permettra d'avoir un décodage de type propagation de croyance.

Plusieurs avantages découlent de cette définition des codes LDPC définis sur un groupe. Du points de vue du groupe G : il peut être fixée par une autre partie de la chaîne de communication comme, par exemple, la modulation utilisée. De plus il est possible de définir des codes performants où les symboles sont dans un domaine de taille 2^m mais qui n'ont pas de représentation binaire linéaire. Le code de Nordstrom-Robinson est non-linéaire en

binaire mais devient linéaire sur $\mathbb{Z}_4$ [8]. Par ailleurs, que les h_i soient n'importe quelles fonctions apporte un très grand choix pour permettre l'optimisation de ces codes. Ainsi, alors qu'il n'y a que q possibilités pour les h_i d'un code linéaire sur le corps $\mathrm{GF}(q)$ ce nombre monte à q^q pour un code sur un groupe de taille q. Même en se restreignant aux applications bijectives, le choix se porte sur $q!$ permutations possibles.

Cependant, le nombre de choix implique une optimisation sur les codes plus ardue.

Comme la définition des codes n'impose quasiment pas de contraintes ni sur le groupe ni sur les applications h_i, une procédure générale pour l'encodage de tels codes est malheureusement difficile à mettre en oeuvre.

3 Propagation de croyance

Les codes LDPC sur le corps $\mathrm{GF}(2^m)$ deviennent intéressants lorsque m devient grand. Toutefois l'algorithme brutal de décodage par propagation de croyance devient par la même occasion plus complexe. L'utilisation de la FFT permet cependant de revenir à une complexité raisonnable. L'algorithme que nous proposons ici pour décoder les codes LDPC sur les groupes utilise la même stratégie. Que les applications h_i soient générales n'implique pas de modifications substantielles. C'est pourquoi, l'algorithme ci-dessous redevient l'algorithme de décodage de [7] pour les codes linéaires sur $\mathrm{GF}(2^m)$.

La structure de groupe abélien est suffisamment riche pour pouvoir définir une transformée de Fourier [9]. Ainsi si un algorithme rapide existe alors le gain en complexité devient immédiat.

Tout d'abord, l'algorithme de propagation de croyance n'est pas modifié pour les messages issus des noeuds de variable. Seuls les noeuds de parité sont donc à considérer. Les notations sont celles de la figure 1. Le noeud de parité doit envoyer un message μ au noeud de variable v à partir des messages μ_i qui proviennent des noeuds de variable v_i. La contrainte représentée sur la figure 1 est $\sum_{i=1}^{n} h_i(v_i) + h(v) \equiv 0$ dans G.

Les messages μ_i sont des "vecteurs" qui à un élément g de G associe la "probabilité" que la variable vaille g.

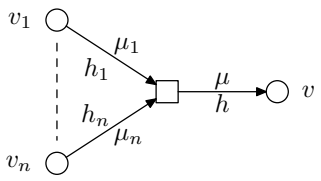

FIG. 1 – Le graphe au niveau d'un noeud de contrainte.

Selon [10, 11] le message μ est la marginalisation des messages entrants en considérant la contrainte du noeud :

$$\mu(x) = \sum_{\substack{x_1 \in G \\ \cdots \\ x_n \in G}} \Big[h(x) + \sum_{i=1}^{n} h_i(x_i) \equiv 0 \Big] \prod_{i=1}^{n} \mu_i(x_i) \qquad (3)$$

où $[P]$ vaut 1 si P est vérifiée, 0 sinon. Normalement $\sum_{x \in G} \mu(x) = 1$, mais comme seules les proportions relatives sont utiles, le coefficient de proportion est omis.

Pour un groupe de taille q, selon cette équation, le calcul du message μ complet nécessite approximativement nq^{n+1} opérations. Ce calcul doit être repris autant de fois qu'il y a d'arêtes dans le graphe factoriel. Pour des tailles q de groupe modéré, le coût devient prohibitif.

Comme pour les codes sur les corps finis, l'usage de la transformée de Fourier simplifie le calcul. En effet (3) est équivalent à

$$\mu(x) \propto \mathcal{F}^{-1}\left(\prod_{i=1}^{n} \mathcal{F}^{*}\big(\mu_i^{h_i}\big) \right)\big(h(x)\big) \qquad (4)$$

où $\mathcal{F}$ est la transformée de Fourier et $\mathcal{F}^{-1}$ son inverse et où la fonction $\mu_i^{h_i}$ est définie par :

$$\mu_i^{h_i}(y) = \sum_{x \in h_i^{-1}(\{y\})} \mu_i(x) \qquad (5)$$

La transformée de Fourier transforme la convolution induite par la somme dans la contrainte de parité par une multiplication coefficient par coefficient. Il est donc plus rapide, si une transformée de Fourier existe, de passer dans le domaine de Fourier pour faire cette convolution.

La relation (4) permet la mise au point de l'algorithme calculant les messages issus des noeuds de parité :

initialisation $\mu'_1(g) = \cdots = \mu'_d(g) = \mu(g) = 0 \quad \forall g \in G$
pour $i = 1, \ldots, d$ **faire**
  pour $g \in G$ **faire** $\mu'_i(h_i(g)) \leftarrow \mu'_i(h_i(g)) + \mu_i(g)$ **fin**
  $\mu'_i \leftarrow \mathrm{FFT}^{*}(\mu'_i)$
fin
pour $g \in G$ **faire** $\mu'(g) \leftarrow \prod_{i=1}^{d} \mu'_i(g)$ **fin**
$\mu' \leftarrow \mathrm{IFFT}(\mu')$
pour $g \in G$ **faire** $\mu(g) \leftarrow \mu'(h(g))$ **fin**

Les procédures FFT et IFFT font le calcul de la transformée de Fourier et son inverse en utilisant le groupe abélien G. Les messages primés μ' sont des variables temporaires.

La complexité de cet algorithme dépend de celle des transformée de Fourier. Pour simplifier l'analyse, nous utiliserons la notation O. Sur un graphe ayant e arêtes, si on suppose que les procédures FFT et IFFT nécessitent $O(q \log q)$ opérations, alors une passe de l'algorithme de propagation de croyance nécessite $O(eq \log q)$ opérations. Si la transformée de Fourier n'est pas une version rapide, cette complexité croit comme $O(eq^2)$, qui est toutefois meilleure que le calcul direct (3).

Des algorithmes rapides de calcul de FFT existent pour la plupart des groupes intéressants. Par exemple la FFT sur un groupe du type $(\mathbb{Z}_2^p, +)$ correspond à une transformation de Hadamard rapide. De même les groupes cycliques du type $(\mathbb{Z}_{2^p}, +)$ utilisent une FFT classique telle que celle de Cooley et Tuckey. Une mise en oeuvre libre en langage C d'algorithmes FFT pour des tailles et des dimensions arbitraires se trouve à l'adresse `http://www.fftw.org` dans la librairie FFTW.

4 Exemples d'utilisation

Deux exemples d'utilisation sont présentés ci-dessous. Le premier utilise des codes déjà connus et améliore le

décodage par propagation de croyance, le second consiste à utiliser des codes p-adique et à les décoder sur des sous-groupes. Ces exemples tentent de montrer la faisabilité des codes définis sur les groupes et de leur décodage. Le potentiel des codes LDPC sur les groupes ne doit pas être limité à la simplicité de ces exemples.

4.1 Regroupement de bits

Les codes algébriques classiques peuvent rarement être décodés par un algorithme de propagation de croyance. En effet le graphe factoriel correspondant contient un grand nombre de petits cycles. L'idée du regroupement de bit suppose de découper les mots de code en plusieurs blocs de p bits. Ces blocs contiennent donc des symboles de $\mathbb{Z}_2^p$. Les mots de code considérés sur $\mathbb{Z}_2^p$ vérifient des équations de parité issues de la matrice de parité du code binaire. Le graphe factoriel se transforme ainsi en un graphe plus petit.

Par exemple, la matrice de parité du code de Hamming étendu est :

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \tag{6}$$

En regroupant les bits deux par deux, les mots de code sur $G = \mathbb{Z}_2^2$ sont définis par la matrice de parité issue de (6) :

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 0 & a & b \\ 0 & 1 & b & a \end{bmatrix} \tag{7}$$

où 1 est l'application identité, où a est l'application linéaire $\left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$, et b l'application linéaire $\left[\begin{smallmatrix} 1 & 1 \\ 1 & 1 \end{smallmatrix}\right]$. Que l'application b ne soit pas une bijection de G dans G, ne pose pas de difficulté de décodage par propagation de croyance grâce à la généralité de la définition (2).

Le graphe factoriel issu de (6) comporte beaucoup plus de 4-cycles que celui issu de la matrice de parité (7). Le regroupement peut se faire naturellement sur plus de bits, donc sur des groupes $\mathbb{Z}_2^p$ de plus en plus grand. Par exemple, un regroupement par blocs de quatre bits amène un décodage selon le maximum de vraisemblance car le graphe factoriel, dont la matrice de parité est de la forme $[1\ c]$, devient un arbre.

Tout les codes linéaires binaires se prêtent facilement au regroupement de bits. Cela permet en plus d'avoir la facilité d'encodage d'un code linéaire binaire tout en ayant un décodage par BP efficace. Avec cette technique, le compromis entre les performances et la complexité de décodage devient adaptable.

Nous avons considéré le code binaire résidu quadratique étendu QR $[48, 24]$. La matrice de parité utilisée est systématique. Les mots de codes sont découpés par blocs de deux, de huit, de six, jusqu'à douze bits. Le découpage par bloc de 24 bits correspond au décodage selon le maximum de vraisemblance.

Le décodage par propagation de croyance de ce code a été simulé sur un canal gaussien en utilisant une BPSK. Le nombre d'itérations à été fixé à 10 avec la condition d'arrêt prématuré classique.

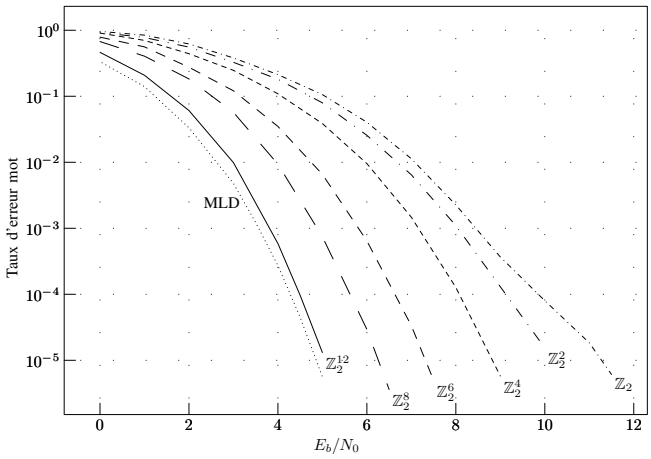

FIG. 2 – Décodage du code QR $[48, 24]$ sur un canal gaussien en fonction du groupe.

Les résultats de la simulation, affichés sur la figure 2, montrent que le regroupement des bits améliore le décodage. Naturellement, la complexité du décodage augmente par la même occasion. Cependant cette augmentation n'est pas prohibitive, et la complexité totale reste raisonnable.

Comme les groupes utilisés sont du type $(\mathbb{Z}_2^p, +)$, la transformée de Fourier sur le groupe utilisé est équivalente à une transformation de Hadamard rapide. Par conséquent, seules des additions et des soustractions sont effectuées.

Les groupes $(\mathbb{Z}_2^p, +)$ qui apparaissent avec le regroupement de bits des codes linéaires binaires sont identiques aux groupes additifs des corps finis $\mathrm{GF}(2^p)$. Cependant la matrice de parité modifiée par le regroupement n'est quasiment jamais une matrice de parité valide pour un code linaire sur $\mathrm{GF}(2^p)$, car les applications de cette matrice ne sont pas obligatoirement des multiplications par un élément du corps. Le regroupement de bits est donc rendu possible grâce aux groupes $(\mathbb{Z}_2^p, +)$ mais surtout grâce à la généralité des applications intervenant dans (2).

4.2 Code 2-adique

Le second exemple d'utilisation des codes définis sur les groupes concerne les groupes du type $(\mathbb{Z}_{2^p}, +)$. La transformée de Fourier dans ce cas correspond à la transformée classique. Un algorithme rapide est donc disponible.

Les auteurs de [12] généralisent le code de Hamming binaire étendu au corps 2-adique. La matrice de parité est dans ce cas :

$$\mathcal{H} = \begin{bmatrix} 1 & \lambda & \lambda-1 & -1 & & & & 1 \\ & 1 & \lambda & \lambda-1 & -1 & & & 1 \\ & & 1 & \lambda & \lambda-1 & -1 & & 1 \\ & & & 1 & \lambda & \lambda-1 & -1 & 1 \end{bmatrix} \tag{8}$$

avec λ un entier algébrique 2-adique défini par l'équation $\lambda^2 - \lambda + 2 = 0$. Le développement 2-adique de λ est $0 + 2 + 4 + 32 + 128 + 256 + \cdots$.

À partir de cette matrice de parité, il est possible de construire une famille de codes sur les groupes $\mathbb{Z}_{2^p}$ en ne gardant des coefficients de $\mathcal{H}$ que leur reste modulo 2^p.

Les codes obtenus ne sont pas linéaires sur $\mathrm{GF}(2^p)$, mais sont linéaires sur $\mathbb{Z}_{2^p}$. Il devient possible d'obtenir ainsi des codes intéressants. Par exemple la projection de $\mathcal{H}$ sur $\mathbb{Z}_4$ est équivalent au code de Nordstrom-Robinson en utilisant le codage de Gray entre les éléments de $\mathbb{Z}_4$ et leur représentation binaire : $0 \to 00, 1 \to 01, 2 \to 11, 3 \to 10$.

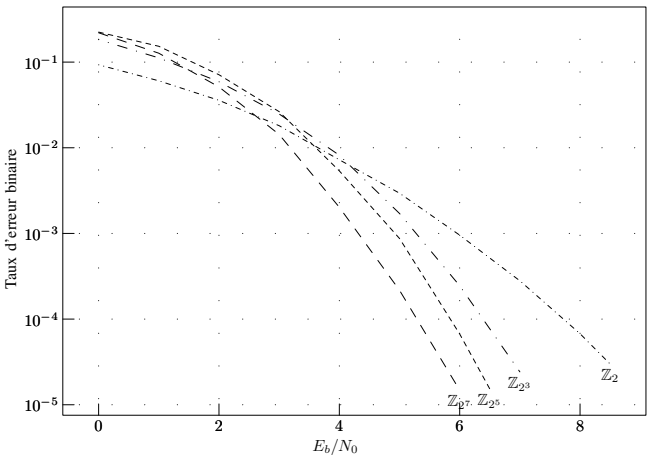

FIG. 3 – Performances sur canal gaussien des codes du Hamming étendu sur les anneaux $\mathbb{Z}_{2^n}$.

Les résultats de la simulation du décodage des codes issus de la projection de (8) sur $\mathbb{Z}_2$, $\mathbb{Z}_8$, $\mathbb{Z}_{32}$ et $\mathbb{Z}_{128}$ sont présentés sur la figure 3. Le canal utilisé est un canal gaussien à entrée binaire. La correspondance entre les éléments de $\mathbb{Z}_{2^p}$ et leur représentation binaire est donnée par le codage de Gray. Cela permet d'augmenter la distance minimale de Lee, comme l'indique [12]. Le nombre d'itérations du décodage par propagation de croyance est au maximum 20. Comme pour l'exemple précédent, l'algorithme stoppe prématurément dès qu'un mot de code est trouvé.

Il faut remarquer que la structure de (8) implique un grand nombre (25) de cycles de longueur quatre dans le graphe factoriel. Le décodage s'en ressent et une représentation plus adéquate serait bénéfique. Cependant, il devient possible de faire un regroupement par bloc de deux symboles ou plus, comme dans l'exemple précédent. Dans ce cas, un algorithme de FFT est directement disponible.

Le rendement des codes reste inchangé selon la taille du groupe par contre la taille binaire en dépend directement. Le graphe factoriel du code de Hamming 2-adique ne se modifie pas avec la taille du groupe utilisé. Il devient donc possible de pouvoir décoder un ensemble de codes sans changer la structure du décodeur.

5 Conclusion

Nous avons introduit ici une généralisation des codes LDPC. Cette généralisation est rendue possible grâce à l'utilisation de la structure de groupe dans la définition des contraintes de parité. Ces codes peuvent être aussi bien linéaires que non. Un algorithme de décodage par propagation de croyance existe et de complexité raisonnable dès qu'une transformée de Fourier rapide est disponible pour le groupe de définition.

Un premier exemple d'utilisation est donnée par le regroupement par bloc des bits des mots d'un code binaire. Cet exemple montre que l'approche est viable et que le compromis entre les performances et la complexité de décodage est ajustable. Le second exemple montre qu'il est possible de concevoir des codes performants qui ne soient ni binaires ni linéaires.

Remerciement

Les auteurs tiennent à remercier David Declercq pour les discutions fructueuses qui ont permis l'aboutissement de ce travail et de l'intérêt qu'il y porte.

Références

[1] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. on Inf. Theory*, vol. 45, no. 2, pp. 399–431, Mar. 1999.

[2] R. G. Gallager, *Low density parity check codes.* M.I.T press, 1963.

[3] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, "Analysis of low density codes and improved designs using irregular graphs," in *Proceedings of 30th ACM STOC*, May 1998.

[4] ——, "Improved low-density parity-check codes using irregular graphs," *IEEE Trans. on Inf. Theory*, vol. 47, no. 2, pp. 585–598, Feb. 2001.

[5] M. C. Davey and D. J. C. MacKay, "Low density parity check codes over $\mathrm{GF}(q)$," *IEEE Commun. Lett.*, vol. 2, no. 6, pp. 165–167, Jun. 1998.

[6] D. Sridhara and T. E. Fuja, "LDPC codes over rings for PSK modulations," *IEEE Trans. on Inf. Theory*, vol. 51, no. 9, pp. 3209–3220, Sep. 2005.

[7] D. Declercq and M. Fossorier, "Decoding algorithms for nonbinary LDPC codes over $\mathrm{GF}(q)$," *IEEE Trans. on Commun.*, vol. 55, no. 4, pp. 633–643, Apr. 2007.

[8] A. R. Hammons, Jr., P. V. Kumar, A. R. Calderbank, N. J. A. Sloane, and P. Solé, "The $\mathbb{Z}_4$-linearity of Kerdock, Preparata, Goethals, and related codes," *IEEE Trans. on Inf. Theory*, vol. 40, no. 2, pp. 301–319, Mar. 1994.

[9] S. Lang, *Algebra.* Springer, 2002.

[10] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, "Factor graphs and the sum-product algorithm," *IEEE Trans. on Inf. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[11] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. on Inf. Theory*, vol. 46, no. 2, pp. 325–343, Mar. 2000.

[12] A. R. Calderbank and N. J. A. Sloane, "Modular and p-adic cyclic codes," *Designs, Codes and Cryptography*, vol. 6, no. 1, pp. 21–35, July 1995.