

Propagation de croyance et codes LDPC définis sur les groupes

Alban Goupil

8 décembre 2005



CRéSTIC

DéCom

Résumé

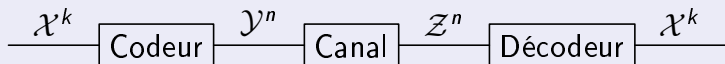
Nous introduirons une large classe de codes LDPC, qui est suffisamment grande pour prendre en compte les codes définis sur des corps finis, sur des anneaux et même plus généralement sur des groupes. Cela permet entre autre de pouvoir considérer des codes linéaires mais aussi non-linéaires. Un décodage par propagation de croyance sera présenté. Sa complexité reste quasiment identique à celui des codes LDPC définis sur un corps fini.

Plan de l'exposé

- 1 Introduction
- 2 Présentation des codes LDPC
- 3 Extensions des codes LDPC
- 4 Codes LDPC et les groupes
- 5 Exemples d'applications
- 6 Conclusion

Ce que Shannon nous promet

Paradigme de Shannon



Théorème de Shannon

- Les messages sont très libres : x^k
- Les mots de code aussi : y^n
- Le code est un simple dictionnaire
- Le canal est bien adapté au code : $y^n \rightarrow z^n$
- Décroissance de la probabilité d'erreur exponentielle en n

Mais en pratique...

Nécessité d'imposer des structures algébriques

- Le code impose une structure aux messages : espace vectoriel, module sur un anneau, polynôme
- Une structure est imposée au code (sous-espace vectoriel, ...)

Structures = Boîtes à outils

- Un encodage rapide — en utilisant une base, ...
- Un décodage souvent possible — par syndrome, ...
- Présence d'outils algébriques pour l'analyse — code AG, ...

Structures = Contraintes

- Codes courts si le décodage est algébrique
- Loin de la limite de Shannon
- Adéquation code/canal difficile (modulation, ...)
- \implies Construction des codes en fonction du type de canal

Turbo/LDPC codes

- Composés de plusieurs sous-codes
- Grande longueur
- Utilisent une permutation
- La limite de Shannon devient accessible

Décodage

- Propagation de croyance
- Algorithme simple qui utilise les décisions souples
- Mais sous-optimal

Définition

- Codes linéaires sur $\text{GF}(2)$
- Codes définis par une matrice de parité \mathbf{H} (parity check)

$$\mathbf{c} \in \mathcal{C} \iff \mathbf{H} \mathbf{c}^T = \mathbf{0}$$

- \mathbf{H} est une matrice creuse : majorité de 0 dans \mathbf{H} (low density)
- Attention : l'aspect creux est un concept asymptotique
- Régulier : nombre identique de 1 par lignes, idem pour les colonnes
- Problème d'encodage : il faut trouver une matrice génératrice \mathbf{G} , mais elles ne sont malheureusement pas nécessairement creuses.

Représentation graphique

Graphe de Tanner

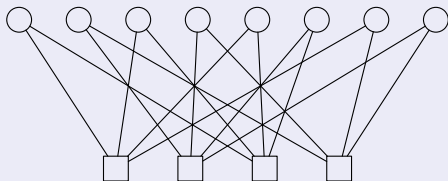
- Colonnes de \mathbf{H} \iff Variables (cercles)
- Lignes de \mathbf{H} \iff Parités (carrés)
- Chaque nœud représente une contrainte

Variable Contrainte d'égalité \implies code à répétition

Parité Contrainte de parité (binaire) \implies code de parité

Exemple

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

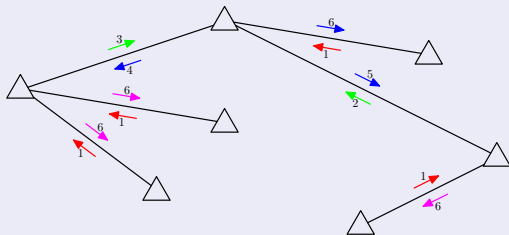


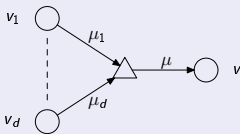
Propagation dans un graphe

Principe de la propagation

- Utilise le graphe comme support
- Chaque sommet calcule et envoie des messages en fonction
 - ▶ Des messages reçus
 - ▶ De la contrainte qu'il représente
- Principe d'information extrinsèque
- Problème : les cycles

Exemple : $\mu = 1 + \sum_i \mu_i$





- Les messages sont des tableaux contenant des probabilités :

$$\mu(x) = \Pr[v = x]$$

- Le calcul des messages est une marginalisation sous l'hypothèse d'indépendance des messages entrant :

$$\mu(x) = \sum_{x_1, \dots, x_d} \Pr[v = x | C, x_1, \dots, x_d] \prod \mu_i(x_i)$$

Modification du graphe sous-jacent

- Irrégularité du profile pour atteindre une plus grande capacité
 - ▶ Évolution de densité, EXIT charts, ...
 - ▶ Études asymptotiques
- Constructions plus algébriques
 - ▶ Faciliter l'encodage, la mémorisation du graphe, ...
 - ▶ Permutations particulières
 - ▶ Graphes particuliers (pour les stopping sets, graphes de Cayleys, ...)

Modification du type de contraintes (parités)

- Codes LDPC sur des corps finis
- Utilisation de treillis
- Mise en place de petits codes constituants à la place des parités simples.

Codes LDPC sur $GF(q)$

Corps finis $GF(q)$

- Ensemble fini de q éléments tel que
 - ▶ Addition, soustraction, 0, avec commutativité
 - ▶ Multiplication, division (sauf par zéro), 1, avec commutativité
 - ▶ Règles classiques de distributivité, ...
- Structures très riches (espace vectoriel, ...) mais très contraignantes, par exemple, q doit être de la forme p^m avec p premier

Codes LDPC sur $GF(q)$

- \mathbf{H} est maintenant une matrice creuse d'éléments de $GF(q)$
- Les parités sont de la forme :

$$\sum_i h_i c_i \equiv 0 \quad \text{dans } GF(q)$$

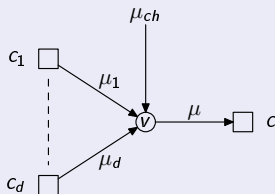
- Avantages : performances, adaptation aux modulations
- Inconvénients : complexité en $O(eq^2)$, sauf pour $GF(2^m)$ en $O(eq \log q)$

Décodage des codes LDPC sur $GF(q)$

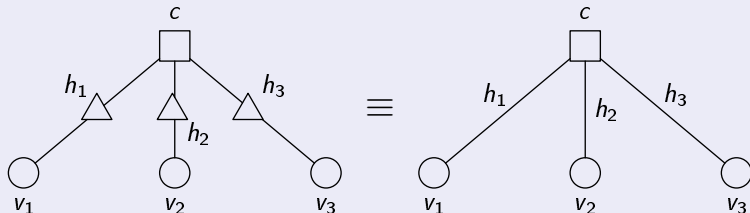
Nœuds de variables

- Pas de modification de l'algorithme de base
- Calcul d'un message :

$$\mu(\mathbf{x}) = \mu_{ch}(\mathbf{x}) \prod_{i=1}^d \mu_i(\mathbf{x}) \quad \forall \mathbf{x} \in GF(q)$$



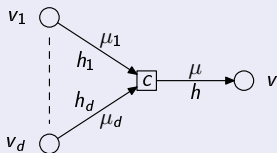
Nœuds de parités, étiquetage du graphe



Décodage des codes LDPC sur $GF(q)$

Nœuds de parités, passage en force

$$\mu(x) = \sum_{\substack{x_1 \in GF(q) \\ \vdots \\ x_d \in GF(q)}} \left[hx + \sum_{i=1}^d h_i x_i \equiv 0 \right] \prod_{i=1}^d \mu_i(x_i)$$



Nœuds de parités, domaine de Fourier

$$\mu(x) \propto \mathcal{F}^{-1} \left(\prod_{i=1}^d \mathcal{F}^* (\mu_i \circ h_i^{-1}) \right) (hx)$$

- $\mu \circ h$ est le message μ dont les éléments sont permutés en fonction de h
- \mathcal{F} est la transformée de Fourier. Dans le cas $GF(2^m)$, \mathcal{F} est équivalente à une transformée de Hadamard et ne nécessite donc que des additions et des soustractions de réels

Motivations

- Peut on faire plus général que les codes LDPC sur les corps tout en gardant la même complexité de décodage
- Meilleure adaptation entre le code et la modulation
- Plus de choix pour la matrice de parité **H**
- Représenter plus de contraintes de parités

Pourquoi se restreindre aux corps finis ?

$$h_1 c_1 + h_2 c_2 + \dots + h_d c_d \equiv 0 \quad \text{dans } GF(q)$$

- L'addition est primordiale puisque la parité en dépend
- La multiplication par h_i est par contre restrictive
 - ▶ On n'utilise pas la distributivité dans la parité
 - ▶ Elle ne représente qu'une permutation particulière des éléments de $GF(q)$

Codes LDPC sur un groupe

Ce que l'on garde pour définir des codes

- Un groupe G abélien fini \implies Transformée de Fourier
 - ▶ Un ensemble fini de q éléments
 - ▶ Addition, soustraction, 0
 - ▶ Commutativité
- Des fonctions de G dans G
 - ▶ Permutations
 - ▶ Mais aussi des applications non bijectives, non injectives, non surjectives

Le nouveaux type de parités

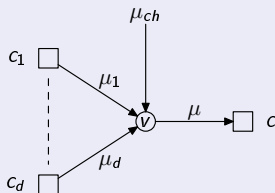
$$h_1(c_1) + h_2(c_2) + \cdots + h_d(c_d) \equiv 0 \quad \text{dans } G$$

- Peut représenter des codes LDPC
 - ▶ sur des corps, sur des anneaux, sur des groupes
 - ▶ non-linéaires
- La matrice de parités \mathbf{H} devient une matrice d'applications

Décodage des codes LDPC sur les groupes

Nœuds de variables

$$\mu(\mathbf{x}) = \mu_{ch}(\mathbf{x}) \prod_{i=1}^d \mu_i(\mathbf{x}) \quad \forall \mathbf{x} \in \mathbf{G}$$



Nœuds de parités, domaine de Fourier

$$\mu(\mathbf{x}) \propto \mathcal{F}^{-1} \left(\prod_{i=1}^d \mathcal{F}^* (\mu_i^{h_i}) \right) (h(\mathbf{x}))$$

- \mathcal{F} est la transformée de Fourier de G .
- μ^h est la fonction définie par $\mu^h(\mathbf{y}) = \sum_{\mathbf{x} \in h^{-1}(\{\mathbf{y}\})} \mu(\mathbf{x})$
- Même complexité que sur $\text{GF}(q)$ si G a une transformée de Fourier rapide

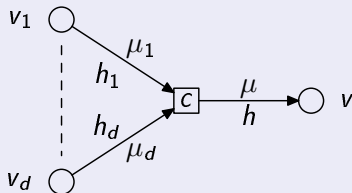
Décodage des codes LDPC sur les groupes

Algorithme pour les nœuds de parités

```
0 require  $\mu'_1(g) = \dots = \mu'_d(g) = \mu(g) = 0 \quad \forall g \in G$ 
1 for  $i = 1, \dots, d$  do
2   for  $g \in G$  do  $\mu'_i(h_i(g)) \leftarrow \mu'_i(h_i(g)) + \mu_i(g)$  done
3    $\mu'_i \leftarrow \text{FFT}^*(\mu'_i)$ 
4 done
5 for  $g \in G$  do  $\mu'(g) \leftarrow \prod_{i=1}^d \mu'_i(g)$  done
6  $\mu' \leftarrow \text{IFFT}(\mu')$ 
7 for  $g \in G$  do  $\mu(g) \leftarrow \mu'(h(g))$  done
```

Relativement à celui de $\text{GF}(q)$

```
2' for  $g \in G$  do  $\mu'_i(h_i g) \leftarrow \mu_i(g)$  done
:
:
7' for  $g \in G$  do  $\mu(g) \leftarrow \mu'(h g)$  done
```



Avantages

- Existence d'un algorithme BP de décodage rapide
- La forme du groupe est donnée par l'application, non plus par le code
- Adéquation code/canal
- Espace de recherche plus important : Exemple sur $GF(q)$

multiplications	permutations	applications
$q - 1$	$q!$	$q^q - 1$
- Extension immédiate pour un groupe G_i par variable v_i

Inconvénients

- Espace de recherche trop grand ?
- Un code tiré au hasard n'a pas autant de chance d'être bon que sur un corps fini
- Comment trouver un encodage ? Nous avons développé une modification de l'algorithme de décomposition LU pour matrice creuse, mais aucune certitude s'il trouve un encodage dès qu'il en existe un !

Regroupement de bits

Construction

- Groupe $G = \mathbb{Z}_2^p$ est l'ensemble des vecteurs binaires de longueur p
- On regroupe une matrice de parité de taille $mp \times np$ pour obtenir une matrice de parité de taille $m \times n$:

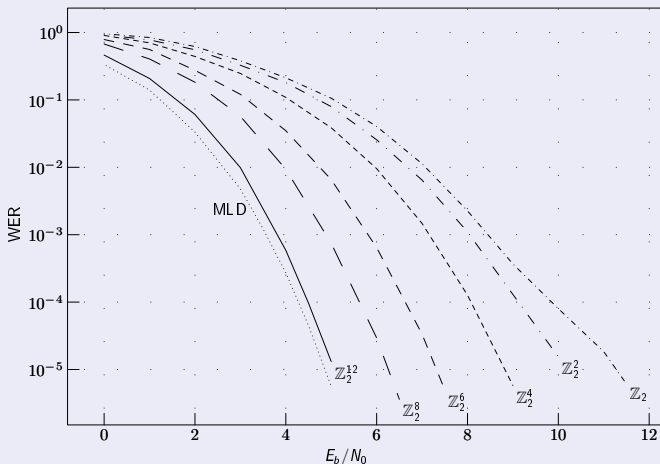
$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \end{bmatrix} \quad \text{devient} \quad \begin{matrix} \mathbf{H}_2 = \begin{bmatrix} 1 & 0 & a & b \\ 0 & 1 & b & a \end{bmatrix} \\ \text{ou} \\ \mathbf{H}_4 = [1 \quad c] \end{matrix}$$

Motivations

- Éliminer des cycles dans le graphe sous-jacent
- Encodage par le code binaire
- Décodage par les groupes \implies meilleures performances
- Ajustement du compromis performances/complexité

Application au QR[48,24,12]

- Regroupement par blocs de 1 à 12 bits : \mathbb{Z}_2^p pour $p = 1, \dots, 12$
- Canal AWGN avec modulation BPSK et 10 itérations maximum



Code 2-adic

- Le code de Hamming est un code cyclique défini sur \mathbb{Z}_2
- Il est possible de remonter ce code dans \mathbb{Z}_4 et même \mathbb{Z}_8, \dots
- La matrice de parité est, dans le corps 2-adic \mathbb{Q}_2 ,

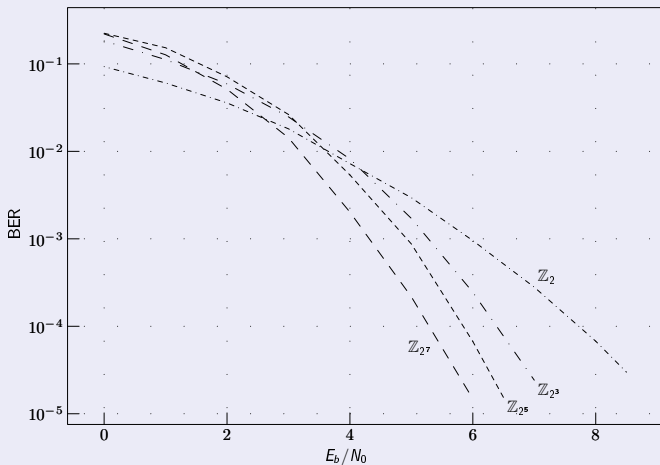
$$\mathbf{H}_\infty = \begin{bmatrix} 1 & \lambda & \lambda - 1 & -1 & & & & 1 \\ & 1 & \lambda & \lambda - 1 & -1 & & & 1 \\ & & 1 & \lambda & \lambda - 1 & -1 & & 1 \\ & & & 1 & \lambda & \lambda - 1 & -1 & 1 \end{bmatrix}$$

avec λ tel que $\lambda^2 - \lambda + 2 = 0$ ($\lambda = 0 + 2 + 4 + 32 + 128 + 256 + \dots$)

- Pour obtenir le code correspondant dans \mathbb{Z}_{2^p} il suffit de prendre la matrice dont les coefficients sont les restes de la division par 2^p de ceux de \mathbf{H}_∞
- Tous les codes cycliques peuvent être « remontés 2-adiquement »

Application au Hamming 2-adic

- Codage de Gray
- Canal AWGN avec modulation BPSK et 20 itérations maximum



Résumé

- Suppression d'une grande partie des contraintes algébriques
- Grand espace de recherche de code
- Bit clustering \implies compromis complexité / performances
- Codes uniquement creux (pour l'algorithme de BP)

Études futures

- Généraliser l'évolution de densité pour ces nouveaux codes
- Recherche des bons coefficients pour la matrice de parités
- Étudier l'adaptation groupe / modulation
- LDPC p -adic ?
- UEP, ...