Architecture matérielle pour un décodeur par programmation linéaire des codes LDPC

Imen DEBBABI[1], Bertrand LE GAL[2], Nadia KHOUJA[1], Fethi TLILI[1], Christophe JEGO[2]

[1]GRESCOM Laboratory, High School of Communications, Carthage University, Tunisia

[2]Bordeaux INP, Enseirb-Matmeca, CNRS IMS, UMR 5218, University of Bordeaux, Talence, France

imen.debbabi@supcom.tn, bertrand.legal@ims-bordeaux.fr nadia.khouja@isetcom.tn, fethi.tlili@supcom.rnu.tn, christophe.jego@ims-bordeaux.fr

Résumé − La méthode de direction alternative des multiplieurs (ADMM) est une approche récente utilisée pour le décodage des codes LDPC. Elle se base sur une approche de programmation linéaire (LP) qui est efficace en termes de taux d'erreur trame. Cependant, elle possède une forte complexité algorithmique et n'a pas encore été implémentée sur cible matérielle. Dans ce papier, nous proposons une première implantation de l'algorithme ADMM sur une cible FPGA. Son coût matériel est évalué et comparé avec les décodeurs LDPC de l'état de l'art qui utilisent l'approche de propagation de croyance.

Abstract − The Alternate Direction Method of Multipliers (ADMM) approach is a novel method for LDPC decoding using linear programming technique (LP). LP decoding of LDPC codes is efficient in terms of frame error rate performance, however it is complex in terms of computationnal complexity. It has not been yet implemented in hardware. In this paper, a first implementation of the ADMM LP decoding algorithm on an FPGA target is presented. Its hardware cost is evaluated and compared with the state of the art LDPC decoders using the belief propagation approach.

1 Introduction

Les codes correcteurs d'erreurs (ECC) sont employés dans la plupart des systèmes communicants afin de fiabiliser la transmission des données numériques. Les codes LDPC (Low Density Parity Check) ont été adoptés dans un nombre conséquent de standards de communication numérique. Traditionnellement, le décodage des codes LDPC est réalisé à l'aide d'algorithmes de propagation de croyance (BP) par passage de messages (MP).

Récemment, des techniques basées sur la programmation linéaire (LP) pour le décodage de codes LDPC ont été proposées [1]. Le décodage LP doit permettre une amélioration du taux d'erreur trame (FER) lors du décodage de codes LDPC par rapport aux approches BP [2]. Cependant, la forte complexité calculatoire des approches LP a restreint leur utilisation à des trames très courtes. Toutefois, des travaux récents dans le domaine du décodage LP [2] ont permis de réduire précieusement la complexité calculatoire du processus de décodage rendant l'implantation matérielle de décodeurs LP envisageable. En plus de réduire la complexité calculatoire du processus de décodage LP, ces travaux ont permis une reformulation du processus de décodage LP sous une forme algorithmique proche de celle du MP. L'algorithme résultant est communément appelé ADMM (Alternate Direction Method of Multipliers). Toutefois, les opérations mises en œuvre lors du processus de décodage sont très différentes.

Une première évaluation de la complexité calculatoire de l'algorithme ADMM et une comparaison avec celles des algorithmes BP sont présentées dans [3]. Cette dernière démontre que contrairement aux approches LP traditionnelles basées sur l'utilisation de solveurs, l'implantation de l'algorithme ADMM semble possible malgré l'utilisation d'opérations complexes dans le calcul des nœuds de parité. En effet, un travail a proposé des simplifications algorithmiques de la projection euclidienne, étape clef du calcul des nœuds de parité [4].

Dans cet article nous présentons la première implantation matérielle d'un décodeur LDPC basé sur l'algorithme de décodage ADMM. Dans un premier temps l'algorithme ADMM est détaillé. Ensuite, nous présentons l'architecture matérielle de décodage développée. Puis nous évaluons les performances de l'architecture pour une cible FPGA. Enfin, une comparaison de la complexité matérielle avec celles d'architectures BP est faite.

2 Décodage LP des codes LDPC

Un code LDPC $\mathcal{C}$ de taille N peut se décrire à l'aide d'une matrice de parité $\mathbf{H}$. Il se compose de M équations de parité avec $(M \leq N)$. Le rendement de codage de $\mathcal{C}$ est défini par $r = (N - M)/N$. La matrice H définissant $\mathcal{C}$ peut être représentée sous la forme d'un graphe

Algorithm 1 Algorithme de décodage ADMM

Initialiser $k = 0, T_i^{(0)} = 1.5 - \frac{\gamma_i}{\mu} - \frac{\alpha}{\mu}, i \in [1, N], \lambda_{ji}^{(0)} = 0, z_{ji}^{(0)} = 0.5, j \in [1, M]$
repeat
  for all j **do**
    STEP $-\phi_1-$
    for all $i \in \mathcal{N}(j)$: **do**
      Calcul du message du nœud de variable vers les nœuds de parité $T_{ij}^{(k)}$
      $T_{ij}^{(k)} = T_i^{(k)} - (z_{ji}^{(k)} + \lambda_{ji}^{(k)})$
      Clacul du vecteur d'entrée à la projection P_{ji}
      $P_{ji} = \rho\Pi_{[0,1]}(\frac{1}{d_{v_i} - 2\frac{\alpha}{\mu}} T_{ij}^{(k)}) + (1-\rho) z_{ji}^{(k)} - \lambda_{ji}^{(k)}$
    end for
    STEP $-\phi_2-$
    Projection Euclidienne du vecteur P de longueur d_c
    for all $i \in \mathcal{N}(j)$: **do**
      $w_{ji} = \Pi_{P_{d_{c_j}}}(P_{ji})$
    end for
    STEP $-\phi_3-$
    for all $i \in \mathcal{N}(j)$: **do**
      Calcul des messages du nœud de parité vers les nœuds de variable $z_{ji}^{(k+1)}$ et $\lambda_{ji}^{(k+1)}$
      $z_{ji}^{(k+1)} = w_{ji}$
      $\lambda_{ji}^{(k+1)} = w_{ji} - P_{ji}$
      Mise à jour du nœud de variable $T_i^{(k+1)}$
      $T_i^{(k+1)} = T_{ij}^{(k+1)} + (z_{ji}^{(k+1)} + \lambda_{ji}^{(k+1)})$
    end for
  end for
  $k = k + 1$
until $k \leq k_{max}$
Estimation des bits décodés par $\Pi_{[0,1]}(\frac{1}{d_{v_i} - 2\frac{\alpha}{\mu}} T_i)$

de Tanner où les nœuds variables (VN) sont indexés par $\mathcal{I} = \{1, ..., N\}$ et les nœuds de parité (CN) sont indexés par $\mathcal{J} = \{1, ..., M\}$. Chaque valeur '1' dans la matrice $\mathbf{H}$ à la position (i,j) correspond à un arc liant VN_i à CN_j dans le graphe de Tanner. Le degré des nœuds VN et CN sont respectivement notés d_{v_i} et d_{c_j}. Pour faciliter la formulation de l'algorithme de l'ADMM, $N_v(i)$ correspond aux indices des voisins du nœud $v_i \in$ `VN` et $N_c(j)$ à ceux des voisins du nœud $c_j \in$ `CN`.

Le polytope de parité correspond à l'enveloppe convexe de tous les vecteurs binaires de taille d_{c_j} possédant un nombre pair de 1. Supposons qu'un mot de code $x \in \mathcal{C}$ soit transmis au travers d'un canal bruité et que le vecteur d'information reçu soit r, alors Barman *et al.* [5] ont démontré que le problème de décodage ML décrit dans [1] peut être reformulé comme suit :

$$\begin{aligned} &\text{minimiser } \gamma^T x \\ &\text{Sujet à } T_j x = z_j; z_j \in P_{d_{c_j}} \forall j \in \mathcal{J}, \end{aligned} \tag{1}$$

avec γ le vecteur d'information provenant du canal (log-likelihood). Son i^{th} élément est défini par $\gamma_i = \log \frac{Pr(r_i|x_i=0)}{Pr(r_i|x_i=1)}$.

La description du problème d'optimisation à double variable, et sa résolution sont décrites dans [2, 5]. Une amélioration des performances de décodage via des facteurs de pénalisation a été ensuite proposée dans [6, 7]. La fonction objectif de la formulation LP devient $\gamma^T x - \alpha\|x - 0.5\|_2^2$. α est un coefficient de pénalisation dont la valeur dépend des paramètres du code LDPC [8].

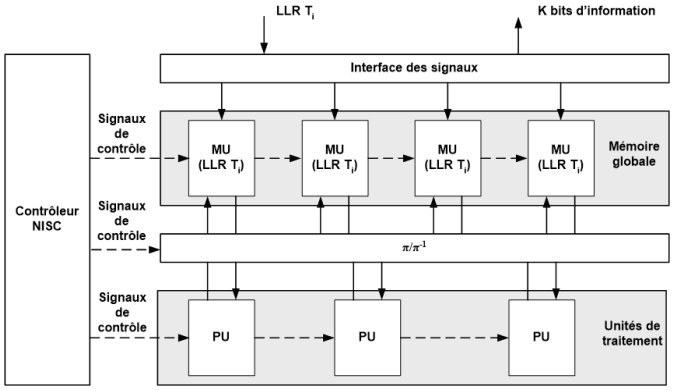

FIGURE 1 – Décodeur LDPC basée sur une architecture NISC.

Une formulation de l'ADMM avec les calculs de parité ordonnancés par couche [9, 10] est présentée dans l'algorithme 1. L'ordonnancement par couches a été sélectionné car il permet de réduire d'un facteur 2 la complexité calculatoire du décodage pour une correction équivalente [9,10]. De plus la complexité mémoire est elle aussi réduite.

Cette formulation algorithmique de l'ADMM est proche de celles des algorithmes BP usuels. En effet, le processus de décodage se compose principalement de 3 étapes : le calcul des messages entre les VNs et le CN, le calcul de la valeur propre de CN et la génération des messages de CN aux VNs. Toutefois, il s'agit des seules similitudes entre les deux approches de décodage. Ainsi, le calcul interne des nœuds de parité qui est basé sur une projection Euclidienne est plus complexe. Ces différences sont exposées plus précisément dans [9, 10]. Différents algorithmes permettent de réaliser la projection Euclidienne [5]. Dans le cadre de l'implantation matérielle présentée, la version proposée dans [4] a été sélectionnée. En effet cette dernière réduit les dépendances de contrôle ainsi que le recours aux fonctions de tri durant la projection.

3 Modèle d'architecture de décodage

De nombreux travaux ont porté sur l'implantation de décodeurs LDPC sur des cibles matérielles. Afin de minimiser le temps de conception, des approches automatisées basées sur des modèles d'architectures génériques de décodeur semi-parallèle ont été développées [11]. Afin de minimiser l'effort de développement et en raison des similitudes de l'algorithme ADMM avec les algorithmes BP, le modèle d'architecture et le flot de conception associé présentés dans [11] ont été adaptés afin de supporter l'algorithme ADMM. La formulation algorithmique retenue de l'ADMM est organisée autour de l'exécution des nœuds CN. En conséquence, l'architecture matérielle d'implantation se décompose en 4 parties principales (Figure 1) :

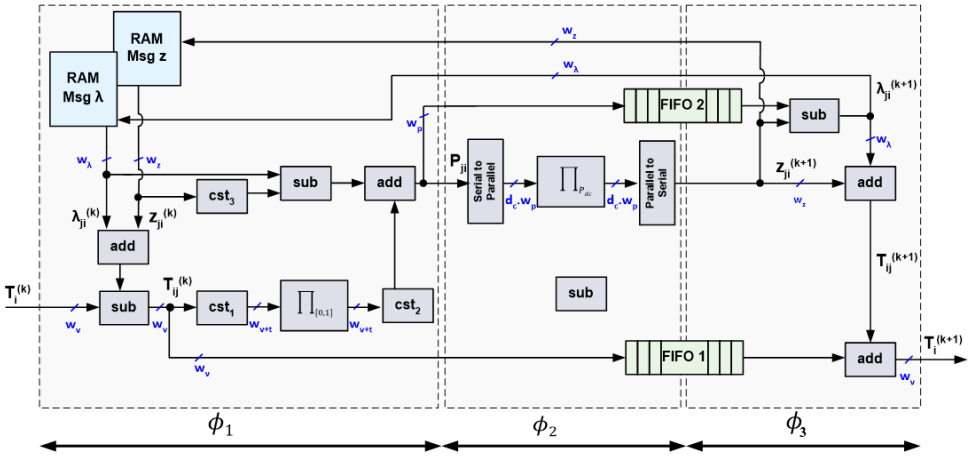

FIGURE 2 – Architecture de l'unité de traitement des CNs pour le décodeur ADMM.

— les unités de mémorisation des VNs (MU),
— les unités de traitement des CNs (PU) avec la mémorisation des messages,
— un réseau d'entrelacement des données,
— un séquenceur qui contrôle les calculs.

Toutefois afin de permettre l'implantation de l'algorithme de décodage ADMM, ce modèle architectural a dû être modifié. Les modifications ont porté sur l'architecture du PU. Tout d'abord, les nœuds CNs émettent 2 messages distincts par VN voisin auquel ils sont connectés. En conséquence, le nombre d'unités de mémorisation au sein des PUs a été doublé. Ensuite, l'unité de calcul traditionnelle décrite dans [11,12] a été remplacée par celle détaillée dans la Figure 2. Il est à noter que le module Π_P correspond à l'architecture matérielle de la projection Euclidienne décrite dans [4]. La latence de calcul est augmentée de d_c cycles d'horloge par rapport à celle décrite dans [11, 12].

L'algorithme ADMM a de bonnes performances en termes de correction d'erreurs. Cependant ce dernier nécessite une précision importante au niveau des données. Les décodeurs LDPC basés sur l'algorithme min-sum utilisent en général 6 bits de quantification pour les messages et 8 ou moins pour les données des nœuds variables [11, 12]. Pour ne pas dégrader les performances de décodage de l'ADMM, il est nécessaire d'utiliser un nombre de bits supérieur. Pour maintenir les performances du format flottant, 12 bits sont nécessaires pour les variables T_i, 10 bits pour les messages z et 11 bits pour les messages λ. Les coefficients de pénalisation ont été quant à eux quantifiés sur 8 bits (ρ, $1-\rho$, $\frac{\alpha}{\mu}$, $\frac{1}{d_v - 2\frac{\alpha}{\mu}}$). Il est à noter que le format de quantification de la projection Euclidienne utilisé est celui proposé dans [4], à savoir 10 bits au total avec 6 bits pour la partie fractionnaire.

4 Résultats d'implantation

Les résultats de synthèse logique de 5 décodeurs LDPC développés sont récapitulés dans le tableau 1. Ces résultats ont été obtenus à l'aide de l'outil Xilinx ISE 14.7. Le FPGA cible est un Virtex-6 (xc6vlx240t-3ff1156).

Nous pouvons constater que la complexité mémoire des décodeurs ADMM varie en fonction de deux paramètres : la taille du code LDPC et le nombre d'unités PU. Le premier paramètre influe sur la taille des mémoires contenant les nœuds VN et les messages tandis que le second impacte uniquement les mémoires dédiées aux messages. La complexité en termes d'éléments logiques et de bascules est quant à elle principalement liée au nombre d'unités de traitement. Les fréquences maximales de fonctionnement varient de 84 à 102 MHz selon les configurations.

Afin de positionner les décodeurs ADMM par rapport aux travaux implémentant des versions sous optimales de l'algorithme BP, des décodeurs LDPC traditionnels ont été implantés à partir du même modèle architectural [11]. Ces décodeurs implantent l'algorithme NMS avec $\alpha = 0.75$. Le format de quantification interne de ces décodeurs est de 6 bits pour les messages et de 8 bits pour les nœuds de variable. Les résultats de synthèse pour ces décodeurs sont fournis dans le tableau 1.

Les fréquences maximales de fonctionnement des décodeurs NMS avoisinent les 150 MHz pour l'ensemble des configurations. Ces bonnes fréquences de fonctionnement permettent d'atteindre des débits de $\times 1.5$ à $\times 2$ supérieurs à ceux des décodeurs ADMM alors que le nombre de cycles par itération est similaire. En ce qui concerne la complexité matérielle, les décodeurs ADMM sont plus onéreux, ils consomment entre $6\times$ et $7.5\times$ plus de LUTs et $5.5\times$ plus de registres que les décodeurs NMS. La complexité mémoire des décodeurs ADMM est quant à elle entre $\times 1.2$ et $\times 1.7$ supérieure.

L'augmentation de la complexité calculatoire provient principalement de la différence de coût de l'unité de traitement. En effet, le coût d'implantation du PU pour les décodeurs NMS et de 174 LUTs et 132 registres. Dans le cas d'un décodeur ADMM, le coût est de 2213 LUTs et 1299 registres. Plusieurs facteurs expliquent ce résultat.

TABLE 1 – Caractéristiques des décodeurs LDPC (100 itérations de décodage)

Code LDPC	96×48		2640×1320		4000×2000		8000×4000		9216×4608	
Parallélisme $P\|M_u$	4|4		22|22		22|22		21|21		24|32	
Nombre de trames	1		2		1		1		1	
Algorithme	ADMM	NMS [11]	ADMM	NMS [11]	ADMM	NMS [11]	ADMM	NMS [11]	ADMM	NMS [11]
LUTs	8634 (5%)	1152 (1%)	48795 (32%)	7947 (5%)	48451 (32%)	7957 (5%)	45223 (30%)	7760 (5%)	67888 (45%)	11650 (12%)
Registres	6344 (2%)	1119 (1%)	29119 (9%)	5635 (1%)	29149 (9%)	5637 (1%)	27897 (9%)	5457 (1%)	43310 (14%)	7960 (2%)
Slices	3757 (9%)	594 (1%)	17571 (46%)	3044 (8%)	16216 (43%)	3151 (8%)	14305 (37%)	2918 (7%)	22288 (59%)	4587 (12%)
$18k$ Blocs RAMs	9 (1%)	9 (1%)	100 (12%)	78 (9.3%)	100 (12%)	78 (9.3%)	125 (15%)	73 (10%)	180 (21%)	116 (14%)
DSP48	12 (1%)	– –	66 (8%)	– –	66 (8%)	– –	63 (8%)	– –	96 (12%)	– –
Fréquence max. (MHz)	102.09	153.77	92.38	157.52	81.52	157.52	84.67	148.06	84.67	151.00
Débit (Mbps)	0.68	1.07	11	19	4.94	9.6	6.08	9.1	7.72	12.54

Tout d'abord, le format de quantification interne au décodeur ADMM est supérieur à celui nécessaire pour les décodeurs NMS. Cela accroît fortement l'utilisation d'éléments logiques pour implanter les opérations arithmétiques et logiques. De plus, la complexité calculatoire de l'unité de calcul est bien supérieure à celle de l'unité de calcul d'un décodeur NMS. En effet, le calcul des messages et la mise à jour des nœuds VNs sont plus complexes et cela sans compter l'étape de projection Euclidienne. Cette dernière utilise à elle seule 40% des LUTs et 10% des bascules.

En ce qui concerne le surcoût au niveau mémoire, comme indiqué précédemment, ceci c'est dûe au fait que l'algorithme ADMM a besoin de mémoriser 2 messages (z et λ) distincts entre les nœuds VN et CN. Les mémoires dédiées au stockage des messages sont donc de facto doublées. De plus le format de quantifications moins favorable pour l'ADMM accentue le surcoût.

Pour résumer les premiers résultats d'implantation des décodeurs ADMM démontrent la faisabilité de sa mise en œuvre. Cependant ces derniers sont onéreux en termes de mémoire et de ressources calculatoires. Toutefois, comme le démontrent les courbes de performances [2, 5, 8] leurs performances de décodage sont plus intéressantes. Par ailleurs, il est important de noter que le décodage LP des codes LDPC est un axe de recherche nouveau. Il est probable que cette première implantation puisse être grandement améliorée.

5 Conclusion

Le décodage LP des codes LDPC offre des performances intéressantes au niveau du pouvoir de correction. De plus, l'ADMM permet de rendre le décodage LP de codes LDPC possible. Dans cet article, nous avons présenté la première implantation matérielle de cet algorithme pour une formulation par couche sur un circuit FPGA. Cette architecture démontre la faisabilité de son implantation. Elle ouvre la voie à des travaux complémentaires visant à réduire le surcoût matériel important vis-à-vis des décodeurs LDPC traditionnels dont la capacité de correction est moindre.

Références

[1] J. Feldman., "Decoding error-correcting codes via linear programming," Ph.D. dissertation, Massachussets Institute of Technology, 2003.

[2] X. Zhang, Ph.D. dissertation, University of California San Diego, 2012.

[3] I. Debbabi, N. Khouja, F. Tlili, B. L. Gal, and C. Jego, "Evaluation of the hardware complexity of the ADMM approach for LDPC decoding," in *Proc. of WCNC*, 2016.

[4] M. Wasson and S. C. Draper, "Hardware based projection onto the parity polytope and probability simplex," in *In Proc. of the Asilomar Conference*, 2015.

[5] S. Barman, X. Liu, S. C. Draper, and B. Recht, "Decomposition methods for large scale LP decoding," *IEEE Trans. on Information Theory*, vol. 59, 2013.

[6] X. Liu, S. Draper, and B. Recht, "Suppressing pseudocodewords by penalizing the objective of LP decoding," in *Proc. of ITW*, 2012.

[7] H. Wei, X. Jiao, and J. Mu, "Reduced-complexity linear programming decoding based on ADMM for LDPC codes," *IEEE Com. letters*, vol. 19, no. 6, 2015.

[8] X. Jiao, H. Wei, J. Mu, and C. Chen, "Improved ADMM penalized decoder for irregular low-density parity-check codes," *IEEE Com. letters*, 2015.

[9] I. Debbabi, B. L. Gal, N. Khouja, F. Tlili, and C. Jego, "Fast Converging ADMM Penalized Algorithm for LDPC Decoding," *IEEE Com. letters*, vol. 20, no. 4, 2016.

[10] ——, "Comparison of different schedulings for the ADMM based LDPC decoding," in *Proc. of ISTC*, 2016.

[11] B. Le Gal, C. Jego, and C. Leroux, "A flexible NISC-based LDPC decoder," *IEEE Trans. on Signal Processing*, vol. 62, no. 10, 2014.

[12] C. Marchand, L. Conde-Canencia, and E. Boutillon, "Architecture and finite precision optimization for layered LDPC decoders," *Springer, JSPS*, 2011.