Décodage de Turbo Codes Produit Ternaires

Loïc BARNAULT[1,2], David DECLERCQ[1], Jacques EUDES[2]

[1]ETIS ENSEA/UCP/CNRS UMR 8051,
6, avenue du Ponceau BP 44, F 95014 CERGY-PONTOISE CEDEX,
Tél. : +33 1 30 73 66 10 Fax. : +33 1 30 73 66 27

[2]Thalès, Colombes, France
Barnault@ensea.fr, Declercq@ensea.fr, Jacques.Eudes@fr.thalesgroup.com

Résumé – Dans cet article, nous présentons une généralisation de l'algorithme "list-based" ALD-FBBA appliqué aux codes blocs ternaires. Des itérations de cet algorithme permettent le décodage de turbo codes produits ternaires. Nous montrons que les performances de ces turbo codes produits ternaires sont comparables à celles des turbo codes produits binaires.

Abstract – In this paper, we present a generalization of the list-based ALD-FBBA algorithm applied to ternary bloc codes. Iterations of this algorithm permit to decode ternary turbo product codes defined by the interleaving of ternary bloc codes. We show that ternary turbo product codes performance are comparable to binary turbo product codes.

Introduction

Un code produit est défini par la concaténation en série de deux codes constituants de type bloc. Un code produit est ternaire lorsque ses codes constituants sont aussi de nature ternaire – les symboles prennent alors leurs valeurs dans l'ensemble $\{0; 1; 2\}$. Un turbo code produit ternaire est un code produit ternaire à décodage itératif, ce qui implique que ses codes constituants sont associés à un décodage SISO (Soft Input Soft Output). La première partie de cet article correspond à l'étude des codes constituants et de leur décodage par un nouvel algorithme généralisé de l'algorithme ALD-FBBA utilisé pour les codes blocs binaires [3]. La seconde partie définit les turbo codes produits ternaires et leur décodage itératif.

1 Décodage des codes constituants

1.1 Notation, Encodage

Soit $\mathcal{C}$, un code bloc ternaire de taille N, d'information K et de matrice génératrice $G_{(N,K)}$. Le codeur associé à $\mathcal{C}$ encode le vecteur d'information ternaire $I_{(K)}$ en un mot de code ternaire $C_{(N)}$ de la manière suivante,

$$C = GI \tag{1}$$

On définit d'autre part $L_{(N)}$, le vecteur de vraisemblances associé au mot de code C. Le $k^{ième}$ élément du vecteur L est composé de trois valeurs de probabilité et s'exprime ainsi,

$$L_k = [p(y_k|c_k=0) \;\; p(y_k|c_k=1) \;\; p(y_k|c_k=2)] = [l_0 \; l_1 \; l_2] \quad k \in [0\,;N\text{-}1] \tag{2}$$

1.2 Discussion sur le décodage

Pour être utilisé comme code constituant d'un turbo code produit, le code bloc $\mathcal{C}$ doit être associé à un décodage SISO (soft input, soft output). Les décodeurs SISO de codes binaires sont le plus souvent définis à partir de fiabilités construites sur une liste de mots de codes candidats [1]. On peut citer les deux algorithmes suivants,

L'algorithme *Chase* repose sur la construction d'un ensemble de mots de codes par un décodage algébrique de patterns dérivés de la décision dure en sortie du canal. Les différentes déclinaisons de l'algorithme Chase reposent sur la façon de définir ces patterns et satisfont à des degrés différents le compromis performance-complexité. Ainsi, Chase-I définit $\binom{N}{\lfloor d/2 \rfloor}$ patterns, Chase-II définit $2^{\lfloor d/2 \rfloor}$ patterns alors que Chase-III n'en définit que $\lfloor d/2 \rfloor + 1$ (d étant la distance de Hamming du code). La sortie souple de l'algorithme est construite en appliquant une marginalisation sur l'ensemble précédemment défini.

L'algorithme *ALD-FBBA* détermine dans un premier temps une matrice d'encodage basée sur les symboles les plus fiables du mot de code reçu. Ensuite, il utilise cette matrice d'encodage pour construire un ensemble de mots de codes à partir de patterns dérivés de la décision dure en sortie du canal. Les patterns sont définis de manière à optimiser le compromis perfomance/complexité, en contrôlant la qualité et la taille de l'ensemble. La sortie souple de l'algorithme est également construite en appliquant une marginalisation sur l'ensemble précédemment défini.

Dans la section suivante, nous décrivons en détail la généralisation ternaire de l'algorithme ALD-FBBA.

1.3 Algorithme ALD-FBBA ternaire

Soit Y_D la sortie de la décision dure d'un mot de code bruité Y. On a $Y_D = C + B$ où B est un bruit discret de taille N à valeurs dans $\{0,1,2\}$ [1].

On veut construire un estimé $\tilde{C}$ du mot de code C en utilisant les symboles les moins bruités de Y_D. On sélectionne puis on ordonne les symboles les plus fiables (selon une fonction de fiabilité $\mathcal{F}$) qui permettent la reconstruction d'un mot de code. Cette sélection-permutation s'effectue par l'intermédiaire d'une matrice $Q_{\mathcal{F}}$ appliquée sur le mot de code bruité et décidé Y_D, le vecteur résultant est appelé U.

$$U = Q_{\mathcal{F}} Y_D = Q_{\mathcal{F}}(C + B) \tag{3}$$

avec la définition de l'encodage $C = GI$,

$$U = Q_{\mathcal{F}} GI + Q_{\mathcal{F}} B \tag{4}$$

La matrice $Q_{\mathcal{F}}$ séléctionne dans G les lignes correspondants aux symboles les plus fiables, sous la contrainte que l'on puisse reconstruire le mot de code. $Q_{\mathcal{F}}G$ est donc inversible par construction (pivot de Gauss). Ainsi,

$$(Q_{\mathcal{F}}G)^{-1}U = I + (Q_{\mathcal{F}}G)^{-1}Q_{\mathcal{F}}B \tag{5}$$

puis,

$$I = (Q_{\mathcal{F}}G)^{-1}U - (Q_{\mathcal{F}}G)^{-1}Q_{\mathcal{F}}B \tag{6}$$

de nouveau avec $C = GI$,

$$C = G(Q_{\mathcal{F}}G)^{-1}U - G(Q_{\mathcal{F}}G)^{-1}Q_{\mathcal{F}}B \tag{7}$$

on définit alors $G_{\mathcal{F}} = G(Q_{\mathcal{F}}G)^{-1}$ comme étant une matrice d'encodage secondaire,

$$C = G_{\mathcal{F}}U - G_{\mathcal{F}}Q_{\mathcal{F}}B \tag{8}$$

Une estimation du mot de code C est alors $\tilde{C} = G_{\mathcal{F}}U$ ce qui donne,

$$\tilde{C} = C + G_{\mathcal{F}}Q_{\mathcal{F}}B \tag{9}$$

Le choix de la fiabilité $\mathcal{F}$, fonction de la vraisemblance L_k d'un symbole, est effectué dans l'optique de minimiser l'erreur $G_{\mathcal{F}}Q_{\mathcal{F}}B$. La fonction de fiabilité que nous utilisons consiste en la différence entre les deux valeurs minimales de L_k.

$$\mathcal{F}(L_k) = l_{i_1} - l_{i_0} \tag{10}$$

avec i_0 l'indice du minimum, et i_1 l'indice du second minimum. Cette fonction de fiabilité est propre au décodeur ternaire.

L'algorithme ALD-FBBA consiste à construire un sous ensemble $\mathcal{S}$ de mots de codes, concentrés au voisinage du mot de code bruité reçu (Le voisinage étant défini comme les mots de codes les plus fiables au sens de $\mathcal{F}$). Pour cela, on définit des variations au voisinage du mot de code candidat,

$$C^{(0)} = \tilde{C} = G_{\mathcal{F}}U \tag{11}$$

Le voisinage de $C^{(0)}$ correspond aux mots de codes $C^{(i)}$ dont la fiabilité des symboles qui diffèrent de $C^{(0)}$ est faible.

Divers types d'ensembles $\mathcal{S}$ peuvent être utilisés. Le plus efficace serait l'univers des mots de codes, ce qui correspondrait à un décodage au sens du maximum de vraisemblance. Il n'est cependant pas utilisable en raison de sa complexité. Il s'agit donc de trouver un bon compromis entre qualité du décodage et complexité. L'ensemble $\mathcal{S}_1$ décrit ci-dessous est intéressant puisque pour une faible complexité il ne fait varier que les l symboles supposés les moins fiables. Le cardinal de cet ensemble est de 3^l mots de codes, l est un paramètre qui permet d'explorer un voisinage plus ou moins étendu autour du mot de code candidat. Quant à l'ensemble $\mathcal{S}_2$, il est utilisé pour pouvoir effectuer par la suite l'étape de construction de la sortie souple. L'ensemble que nous utilisons est $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$.

$$\mathcal{S}_1 = \left\{ C^{(i)} = G_{\mathcal{F}} \left(U + [\overbrace{0 \cdots 0}^{K} \; i_0 \cdots i_{l\text{-}1} \; 0 \cdots 0]^T \right), \forall (i_0 \cdots i_{l\text{-}1}) \in \{0;1;2\}^l \right\} \tag{12}$$

$$\mathcal{S}_2 = \left\{ C^{(i)} = G_{\mathcal{F}} \left(U + [\overbrace{0 \cdots 0}^{t} \; i_r 0 \cdots 0]^T \right), \forall \, i_r \in \{1;2\}, \; \forall t \in [0\,;K\text{-}1] \right\} \tag{13}$$

L'ensemble $\mathcal{S}$ contient donc les mots de code candidats dont les fiabilités vont produire la sortie souple du décodeur. A chaque élément d'un mot de code $C^{(i)}$, on associe sa probabilité contenu dans l'entrée souple L. Pour chaque mot de code, on effectue le produit $W^{(i)}$ des probabilités associées aux symboles de $C^{(i)}$.

$$W^{(i)} = \prod_k L_k[C_k^{(i)}] \tag{14}$$

$W^{(i)}$ représente donc la fiabilité du mot de code $C^{(i)}$. Pour calculer la sortie souple S, on cherche à estimer la probabilité $S_k[j]$ qu'a l'élément C_k d'avoir la valeur j. On définit alors $S_k[j]$ comme étant le maximum des $W^{(i)}$ associés aux mots de codes satisfaisant $C_k^{(i)} = j$.

$$S_k[j] = Max_{C_k^{(i)} = j}(W^{(i)}) \tag{15}$$

1. Pour le calcul $(C + B)$, les opérations se font modulo 3.

2 Décodage itératif d'un turbo code produit ternaire

2.1 Notation

Maintenant que nous disposons d'un décodage SISO de codes blocs ternaires, nous allons l'appliquer au décodage itératif d'un code produit ternaire. Soient $\mathcal{C}_1$ le code ligne et $\mathcal{C}_2$ le code colonne du code produit, N_1 (resp. N_2) et R_1 (resp. R_2) sont les tailles et rendements de $\mathcal{C}_1$ (resp. $\mathcal{C}_2$). Le code produit a donc une taille $N = N_1 N_2$, et un rendement $R = R_1 R_2$. La figure 1 représente la structure de ce code produit. Le code est envoyé sur un canal additif Gaussien dont l'entrée est une constellation $3PSK$.

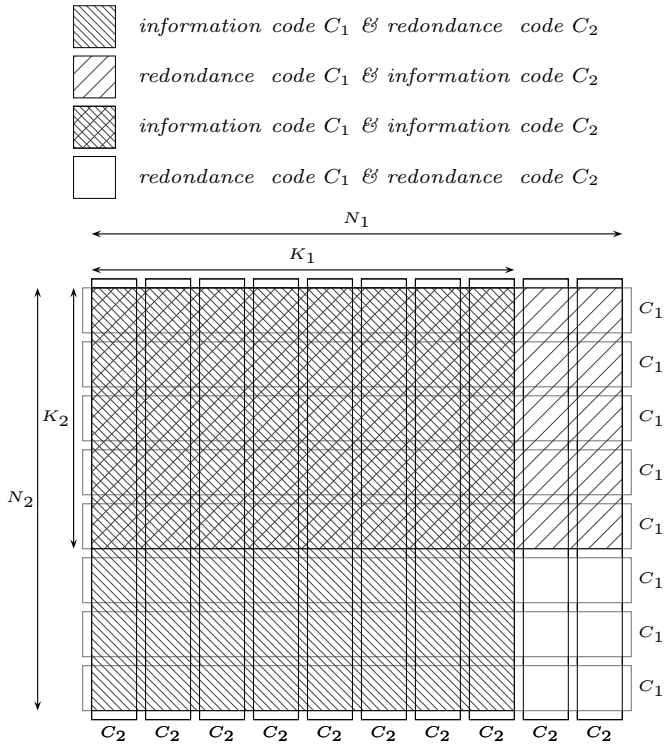

FIG. 1 – *Structure d'un code produit, constitué d'un entrelacement de codes blocs lignes et de codes blocs colonnes*

La figure 2 compare à E_b constante, les distances libres de la BPSK (***/***), de la QPSK (■), et de la 3PSK (▲). Les distances libres $dl^{(BPSK)}$ et $dl^{(QPSK)}$ sont identiques, alors que la distance libre $dl^{(3PSK)}$ est légèrement superieure.

$$\begin{aligned} dl^{(BPSK)} &= 2\sqrt{E_b} \\ dl^{(3PSK)} &= \sqrt{3\,log_2(3)}\sqrt{E_b} \\ &\approx 2.1806\sqrt{E_b} \\ dl^{(QPSK)} &= 2\sqrt{E_b} \end{aligned} \tag{16}$$

Pour définir les turbos codes produits les plus performants à priori, nous avons choisis des codes constituants possédant des distances minimales les plus grandes possibles [4].

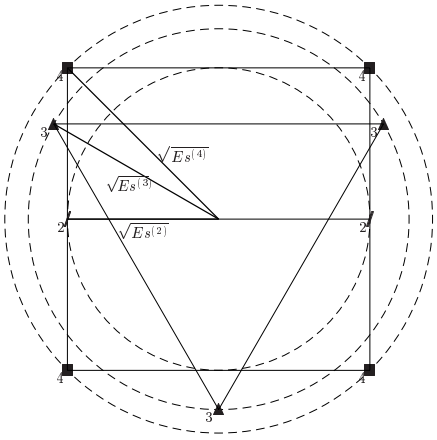

FIG. 2 – *Représentation des distances libres $dl^{(BPSK)}$, $dl^{(3PSK)}$ et $dl^{(QPSK)}$ pour une energie par bit E_b identique.*

2.2 Décodage itératif

Une itération du décodage de ces turbo codes produits se décompose en deux étapes. La première correspond au décodage de chaque code bloc associé aux lignes, et la seconde au décodage des codes blocs associés aux colonnes. Chacune de ces étapes a pour entrée les vraisemblances ainsi que l'information extrinsèque issue de l'étape précédente pondérée par un coefficient α pour les lignes, et β pour les colonnes [3][5]. Le schéma suivant décrit le fonctionnement d'une itération de décodage, L étant les vraisemblances, $E_n^{(L)}$ l'information extrinsèque en entrée de l'étape ligne, et $E_n^{(C)}$ l'information extrinsèque en entrée de l'étape colonne.

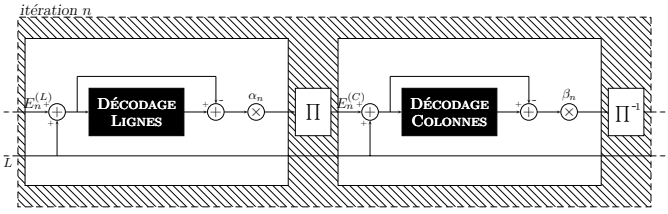

FIG. 3 – *Les deux étapes d'une itération de décodage*

Le graphique 4 compare les performances d'un turbo code produit ternaire associé à une $3PSK$ et celles d'un turbo code produit binaire associé à une $QPSK$ pour une efficacité spectrale identique $\eta = 0.72$ et pour une information binaire de l'ordre de 424 bits (taille ATM).

$$\eta = R_{ter} log(3)/log(2) = 2R_{bin} \tag{17}$$

Le code ternaire possède les caractéristiques suivantes $\mathcal{C}_1(N = 26-1,\ K = 18-1,\ d_{min} = 6)$ et $\mathcal{C}_2(N = 26-2,\ K = 18-2,\ d_{min} = 6)$, le rendement est de $R = 0.45$, l'information binaire equivalente est $K_{bin} = 431$, et le nombre de symboles $3PSK$ en sortie est 600. Le code binaire est défini par $\mathcal{C}_1(N = 35-0.74,\ K = 22-1.41,\ d_{min} = 6)$ et $\mathcal{C}_2(N = 35-0.74,\ K = 22-1.41,\ d_{min} = 6)$, ainsi $R = 0.36$, $K = 424$ et le nombre de symboles $QPSK$ en sortie est 587 (Les tailles des codes constituants ne sont pas entières en raison de poinçonnages non uniformes). Ces résultats montrent des performances comparables pour le turbo code produit binaire d'une part, et le turbo code produit ternaire d'autre part.

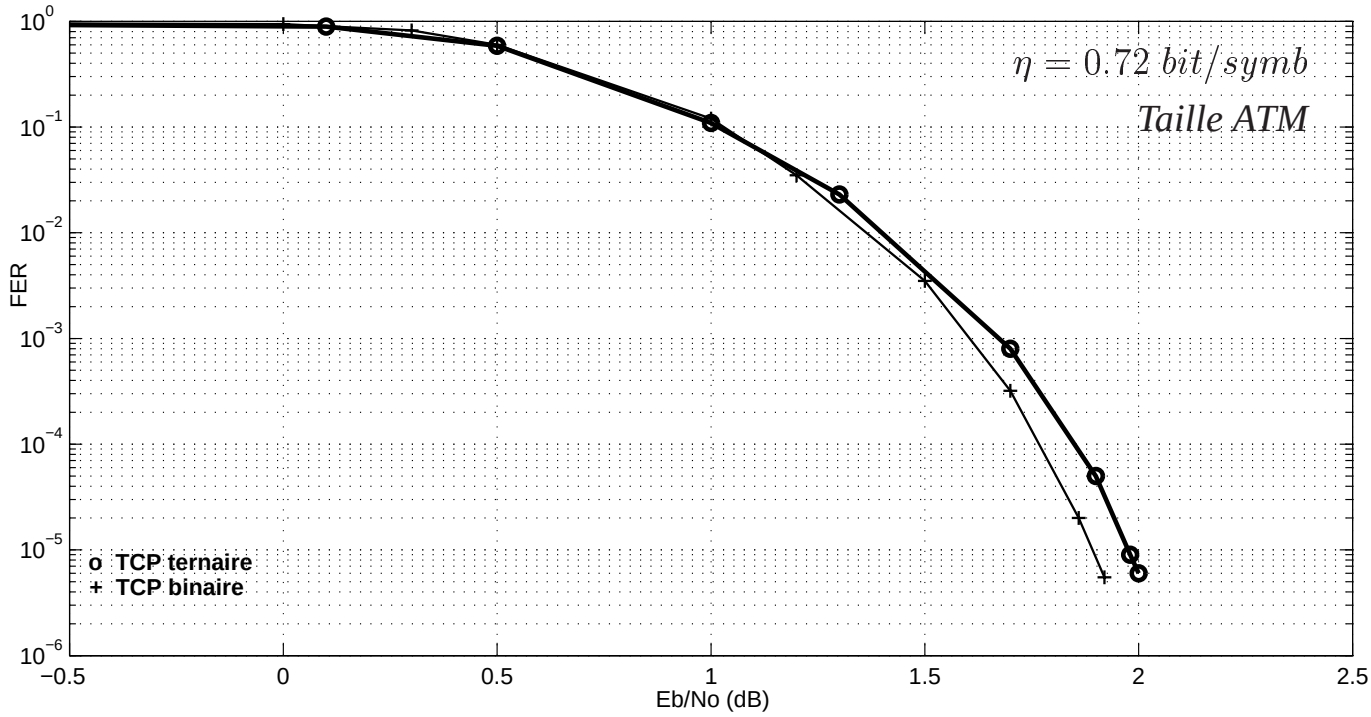

FIG. 4 – *Performances comparées d'un turbo code produit binaire et d'un turbo code produit ternaire de taille ATM, pour une efficacité spectrale* $\eta = 0.72$

Considérons une chaine de communication complète avec code et modem. Si l'on s'interesse à la transmission de paquets de petites tailles (ATM par exemple) on s'apperçoit qu'il existe une plage d'efficacité spectrale pour laquelle la $QPSK$ ne peut etre démodulée correctement et la $BPSK$ n'offre pas une capacité satisfaisante. Pour une efficacité spectrale $\eta \in [0.5\ ,\ 0.7]$, l'utilisation d'une constellation $3PSK$ facilite la démodulation tout en permettant une capacité proche de la $QPSK$. Pour que cette chaine de communication ternaire puisse concurrencer ce qui existe en binaire, elle doit comporter un code ternaire aux performances comparables à celles d'un code binaire.

3 Conclusion

Nous proposons dans cet article une extention de l'algorithme ALD-FBBA associé au décodage itératif des turbo codes produit ternaires.

Les résultats montrent que les performances des turbo codes binaires et ternaires sont comparables. Ceci nous permet d'utiliser une chaine de communication entièrement ternaire en conservant de bonnes performances au niveau du code tout en facilitant la démodulation. En effet la constellation $3PSK$ est plus robuste aux erreurs de synchronisation car c'est la constellation complexe ayant la plus grande distance libre à E_n/N_0 fixé. Enfin, il pourait être intéressant de compléter cette étude en comparant les performances de ce turbo code produit ternaire avec celles d'un code LDPC ternaire.

Références

[1] Philippa. A. Martin, Desmond. P.Taylor, Marc. P. C. Fossorier, "Soft-Input Soft-Output List-Based Decoding Algorithm", ISIT 2002.

[2] C. Barrios Vicente & J.H. Weber, "Dynamic Chase decoding algorithm", Proceedings IEEE Information Theory Workshop, Paris, France, pp. 312-315, March 31 - April 4, 2003.

[3] Juing. Fang, Fabien. Buda & Emmanuel Lemois, "Turbo Product Code : A Well Suitable Solution To Wireless Packet Transmission For Very Low Error Rates", 2^{nd}, International Symposium on Turbo Codes & Related Topics, Brest, France, 2000.

[4] F.R.Kschischang & S. Pasupathy, "Some Ternary and Quaternary Codes and Associated Sphere Packings", IEEE Transactions on information theory, VOL. 38, NO.2, March 1992.

[5] R. Pyndiah, "Near optimum decoding of product codes : block turbo codes", IEEE Trans. on Communications, Vol. 46, n° 8, August 1998.