Recovering the ARP interleaver of an unknown turbo-code

Ilias Elfryakh[1], Sébastien Houcke[1], Stefan Weithoffer[2]

[1]IMT Atlantique - Département Mathematical and Electrical Engineering, Lab-STICC - SI3 - UMR 6285, Brest, France

[2]IMT Atlantique - Département Mathematical and Electrical Engineering, Lab-STICC - 2AI - UMR 6285, Brest, France

ilias.elfryakh@imt-atlantique.fr,
sebastien.houcke@imt-atlantique.fr,stefan.weithoffer@imt-atlantique.fr

Résumé − Dans cet article, nous proposons un algorithme efficace d'identification d'un entrelaceur ARP d'un turbo code à partir de quelques mots de code interceptés. Cet algorithme reprend les principes d'identification proposés dans [1] en les adaptant à la classe particulière des entrelaceurs ARP. L'intérêt principal est de nécessiter que quelques mots de code pour réaliser l'identification. Cela ouvre la porte à de nouvelles techniques de sécurité [2]

Abstract − In this article, we give an efficient algorithm for recovering the ARP permutation of a given turbo encoder when several noisy codewords are received. The algorithm presented here is based on a more general existing Turbo-code interleaver identification algorithm [1]. However our algorithm is able to work with less intercepted codewords which opens the door for security application [2].

Résumé

1 Introduction

Dans la plupart des systèmes de communication, l'utilisation de codes correcteurs est désormais d'usage. En effet, ces codes permettent de rendre les transmissions plus robustes aux aléas liés au média utilisé. Dans un contexte non-coopératif, il est courant que le récepteur (parfois intercepteur) n'ait pas de connaissances a priori sur la nature du code et doive identifier les différents paramètres du codeur de canal en aveugle. Ce problème possède une longue histoire et a été étudié pour différents type de codes : codes linéaires [3] [4] [5], codes LDPC [6] [7], codes convolutionels [14] [8] [9] [10] [11] [12] et turbo-codes [15] [16]. Nous nous focalisons dans ce papier au problème de reconstruction des turbo-codes et plus particulièrement de l'entrelaceur. Le processus de codage est présenté figure 1 et consiste à prendre un mot d'information u de longueur k et de le coder au moyen d'un premier code convolutif systématique de rendement ρ afin d'obtenir un mot dont les k premiers bits sont une copie du mot d'information, le reste étant les bits de redondance (notés v). L'entrée est alors permutée par un entrelaceur π et codée par un second code convolutif systématique. Seule la partie de redondance (notée cette fois w) est conservée. La sortie u, v, w est envoyée à travers un canal bruité et constitue un mot de code. Nous considérerons par la suite que l'intercepteur récupère une version erronée de ce mot notée x, y, z respectivement.

La procédure d'identification d'un turbo-code est de réussir à partir de M mots de codes bruités $c^1, \ldots, c^M$ d'identifier les deux codes convolutifs et l'entrelaceur. Les techniques mentionnées précédemment [12] pour reconstruire un code convolutif peuvent être utilisées pour identifier le premier codeur. Dans [13], les auteurs présentent une approche permettant d'identifier le deuxième codeur. Enfin dans [1] les auteurs proposent une méthode permettant d'identifier π sans aucun a priori sur l'entrelaceur utilisé. L'approche proposée est basée sur le maximum de vraisemblance i.e. trouver la permutation $\hat{\pi}$ qui maximise la probabilité $\mathbf{p}(\pi = \hat{\pi}|c^1, ..., c^M)$, où $c^1, ..., c^M$ sont les mots de code reçus. En ce basant sur leurs travaux, nous proposons une méthode d'identification spécifique lorsque des entrelaceurs ARP [17] sont utilisés. En effet, il est courant d'utiliser cette classe d'entrelaceur pour construire les turbo-codes. Ils possèdent des propriétés intéressantes aussi bien en terme de performance que de facilité d'implémentation [18],[19], [20].

La suite du papier est organisée ainsi : nous présentons tout d'abord les entrelaceurs ARP, puis nous présentons le principe d'identification de l'entrelaceur et enfin nous terminons par des simulations et une conclusion.

2 Entrelaceur ARP (Almost Regular Permutations) :

Deux familles d'entrelaceurs sont populaires pour la conception de turbo-code : les entrelaceurs Quadratic Permutation Polynomial (QPP) [21] et les entrelaceurs Almost Regular Permutations (ARPs) [17]. Les entrelaceurs ARPs

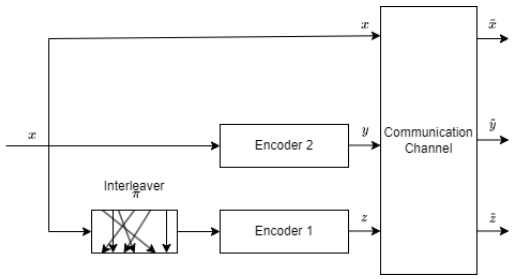

FIGURE 1 – Schéma de principe d'un Turbo-code

sont une extension des entrelaceurs RPs (Regular Permutations) qui peuvent être décrit ainsi :

$$\pi(i) = P.i \bmod K \text{ avec } i \in \{0, ..., K-1\} \tag{1}$$

où K est la taille de l'entrelaceur et P est la période du RP. P doit être premier avec K.

Les entrelaceurs ARP sont définis en partant d'un entrelaceur RP dans lequel on introduit un degré de désordre Q et un vecteur S de shift de longueur Q. Q doit être un diviseur de K :

$$\pi(i) = (P.i + S_{i \bmod Q}) \bmod K \tag{2}$$

Ils possèdent les propriétés suivantes :

Propriétés :

Soit $s_{max} = \frac{K}{Q} - 1$ et $Q_s = \{s.Q, \cdots, (s+1).Q - 1\}$.

— $\forall i, j \in \{0, \cdots, K-1\}$,

$$P.(i-j) \bmod K \neq (S_{j \bmod Q} - S_{i \bmod Q}) \bmod K$$

— $\forall s \in \{0, ..., s_{max}\}, \forall i \in Q_s$,

$$\pi(i) = (\pi(i - sQ) + s.PQ) \bmod K$$

La première propriété est une condition nécessaire et suffisante pour garantir la bijectivité de π. Elle permet de choisir un vecteur de shift. La seconde propriété exprime la structure périodique des entrelaceurs ARP : elle montre qu'un entrelaceur ARP peut être défini par ses Q premières valeurs. Cette propriété est importante car elle réduit la taille de la mémoire nécessaire pour enregistrer toutes les permutations candidates.

Lemma :

Soit π un entrelaceur ARP associé au triplet (K, Q, p) et r_1, r_2 deux entiers appartenant à l'intervalle $[0; Q-1]$. Nous définissons l'ensemble V_r par $V_r = \{\pi(r + sQ)/s \in \{0, \cdots, s_{max}\}\}$.

$$(V_{r_1} \cap V_{r_2} = \emptyset) \Leftrightarrow \pi(r_1) \notin V_{r_2}$$

Théorème :

Soient P un entier premier avec K et Q un diviseur de K (i.e. P est aussi premier avec Q). Alors le nombre d'entrelaceurs ARP associés au triplet (K, P, Q) est :

$$\left(\frac{K}{Q}\right)^Q \times Q!$$

3 Principe d'identification

3.1 Méthode initiale (Méthode de Tissier)

Dans [1], les auteurs proposent d'identifier π de manière incrémentale : la première étape consiste à estimer $\pi(0)$ à partir des M mots de code reçus. Pour chaque position candidate k (i.e. de 0 à $K-1$), les auteurs calculent la probabilité :

$$\mathbf{p}(\pi(0) = k | c^1, ..., c^M) \tag{3}$$

La position $\pi(0)$ est estimée par :

$$\pi(0) = \text{Argmax}_{k=0,..,K-1} \mathbf{p}(\pi(0) = k | c^1, ..., c^M)$$

Dans [13], les auteurs proposent une façon simple d'estimer la probabilité (3) et montrent que la "passe avant" de l'algorithme BCJR (Bahl-Cocke-Jelinek-Raviv Algorithm) permet de calculer :

$$\mathbf{p}(\pi(i) = j \,|\, \pi(0), ..., \pi(i-1), \tilde{x}^1, ..., \tilde{x}^M, \tilde{z}^1_{1..i}, ..., \tilde{z}^M_{1..i})$$

où $\tilde{z}^m_{1..i} = (\tilde{z}^m_1, ..., \tilde{z}^m_i)$ sont les i premiers bits désentrelacés. Une fois $\pi(0)$ estimé, il est possible d'itérer le processus pour identifier séquentiellement toutes les positions de l'entrelaceur. Les auteurs montrent que la complexité de l'algorithme est de l'ordre de $\mathcal{O}(K^2 M)$ [13].

3.2 Méthode pour des entrelaceurs ARP - recherche exhaustive

Notre but est de réussir à proposer une méthode d'identification des entrelaceurs ARP fonctionnant avec très peu de mots de code interceptés tout en conservant les performances d'identification que l'algorithme initial. La réduction significative du nombre de mots de code interceptés pour l'identification est rendue possible du fait de la structure spéciale des entrelaceurs ARP bloc à bloc. Cette structure peut apporter une information supplémentaire sur les probabilités

$$\mathbf{p}(\pi(i) = j \,|\, \pi(0), ..., \pi(i-1), \tilde{x}^1, ..., \tilde{x}^M, \tilde{z}^1_{1..i}, ..., \tilde{z}^M_{1..i}).$$

En effet, lorsque l'on trouve la position $\pi(0)$, nous fixons aussi la position du premier bit de chacun des blocs. La difficulté est alors de trouver l'ordre des blocs : en effet, l'état des registres pour le premier bit sont connus (i.e. fixé à zéro tout comme pour l'algorithme initial) par contre ce n'est pas le cas pour les autres blocs de l'entrelaceur. Pour résoudre ce problème, nous proposons de parcourir l'ensemble $\mathcal{P}$ des entrelaceurs ARP de paramêtres (K, P, Q). Pour chaque $\pi_l \in \mathcal{P}$, on calcule :

$$p^l_i = \mathbb{P}(\pi(i) = \pi_l(i) / \hat{x}, \hat{z}, E^l_i)$$

où E^l_i représente l'événement que les i premières positions de la permutation correspondent aux positions de la $l^{\text{ième}}$ permutation de $\mathcal{P}$.

Les Q premières probabilités p_i^l pour $i = \{0, ..., Q-1\}$ sont calculées de la même façon que pour l'algorithme initial. On se sert de la structure par bloc de l'entrelaceur pour venir vérifier que l'on a trouvé la bonne permutation dans le bloc et identifier l'ordre des blocs.

Pour cela on évalue :

$$p^l = \prod_{i=0}^{K-1} p_i^l$$

On identifie l'entrelaceur comme celui qui maximise p^l (i.e. $\hat{\pi}$ correspond au $\hat{l}^{\text{ième}}$ entrelaceur de $\mathcal{P}$ tel que $\hat{l} = \text{Argmax}_l\ p^l$).

3.3 Méthode pour des entrelaceurs ARP - recherche en faisceau

La recherche exhaustive est coûteuse mais optimale, elle nous permet d'évaluer les performances de l'algorithme et d'avoir un point de comparaison avec d'autres approches sous-optimales. En pratique, nous proposons d'effectuer une recherche sur $\mathcal{P}$ en utilisant un algorithme de recherche en faisceau. C'est un algorithme de recherche heuristique qui explore un graphe en ne considérant qu'un ensemble limité de permutations candidates après chaque nouvelle position identifiée. L'algorithme calcule à chaque étape i, les probabilités d'au plus $K \times b_s$ permutations et sélectionne les b_s meilleures candidates (i.e. celle ayant les plus grandes valeurs de $p^l = \prod_{k=0}^{i} p_k^l$). La taille du faisceau b_s est un paramètre défini par l'utilisateur. La complexité de l'algorithme est alors en $O(Q \times K)$.

Après Q étapes, l'algorithme utilise la structure ARP pour recalculer les probabilités sur l'ensemble de la trame et d'identifier l'entrelaceur ayant la plus forte probabilité.

4 Simulation

Afin de mesurer l'intérêt de notre approche, nous évaluons les performances de celle-ci en terme de probabilité d'identification correcte de l'entrelaceur. Cette probabilité est estimée par simulation de Monte Carlo où à chaque réalisation, les mots de codes, le bruit et l'entrelaceur sont tirés aléatoirement. Pour chaque simulation, 500 tirages de Monte Carlo ont été réalisés. Nous considérons une modulation de phase à 2 états (BPSK) et un canal à bruit blanc additif Gaussien. Ceci nous permet d'avoir rapidement accès aux rapports de vraisemblance (LLRs) de chaque bit reçu mais n'a aucun caractère restrictif. La méthode pourra être mise en oeuvre avec tout type de modulation dès lors que l'on est en mesure de fournir les LLRs de chaque bit intercepté. La figure 2 compare les performances d'identification d'un entrelaceur ARP de taille $K = 256$ bits avec $Q = 8$ pour la recherche exhaustive, la recherche en faisceau pour différentes valeurs de bs et pour la méthode de Tissier [1]. Le rapport signal à bruit (RSB) considéré dans cette simulation est de 6dB.

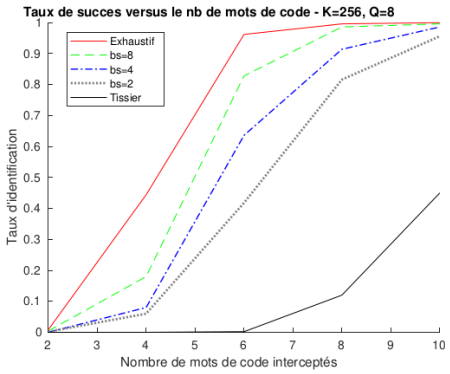

FIGURE 2 – Influence du nombre de faisceau sur les performances

On constate qu'il est possible d'identifier l'entrelaceur avec beaucoup moins de mots de code que l'approche originale. La figure 3 montre l'influence du type d'entrelaceur sur les performances. Nous avons testé 4 entrelaceurs différents : deux de taille $K = 128$ avec $Q = 4$ et $Q = 8$ et deux de taille $K = 256$ avec $Q = 4$ et $Q = 8$. Les courbes sont obtenues pour la recherche exhaustive et comparées aux performances de la méthode de Tissier [1]. Le rapport signal à bruit est toujours fixé à 6dB.

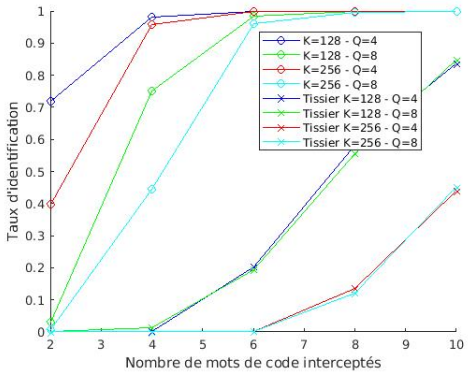

FIGURE 3 – Performance d'identification en fonction du nombre de mots interceptés

Enfin la figure 4 montre l'influence du rapport signal sur bruit (RSB) sur les performances d'identification. Le nombre de mots de code interceptés est fixé à 10 pour cette simulation. La méthode proposée est beaucoup plus robuste au bruit que celle de Tissier et al.

5 Conclusion

Nous avons proposé un algorithme permettant d'identifier un entrelaceur ARP d'un Turbo-Code. L'approche proposée permet d'identifer l'entrelaceur de manière fiable avec moins de 10 mots de code interceptés. Lorsque que les paramètres de l'entrelaceur utilisé sont connus (i.e. P et Q), cela est possible avec une complexité raison-

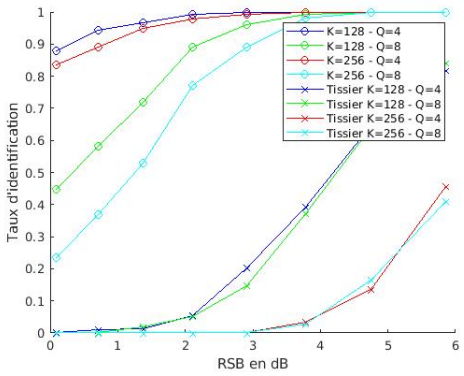

FIGURE 4 – Performance d'identification en fonction du RSB

nable. L'émetteur peut alors changer régulièrement d'entrelaceur, le récepteur légitime sera alors capable de suivre ces changements [2]. L'intercepteur sera lui incapable d'identifier le code utilisé.

Si les paramètres sont inconnus, il est alors nécessaire de faire une recherche exhaustive sur les paramètres possibles. Notons néanmoins que P et Q ne peuvent pas prendre n'importe quelles valeurs (cf. equation 2).

Références

[1] J. -P. Tillich, A. Tixier and N. Sendrier, "Recovering the interleaver of an unknown turbo-code," 2014 IEEE International Symposium on Information Theory, Honolulu, HI, USA, 2014, pp. 2784-2788.

[2] Stefan Weithoffer , Rami Klaimi , Charbel Abdel Nour, "Mitigating Blind Detection Through Protograph Based Interleaving for Turbo Codes", Military Communications Conference 2021, Nov 2021, San Diego, United States.

[3] A. Valembois, "Detection and recognition of a binary linear code," Discrete Applied Mathematics, vol. 111, pp. 199–218, Jul. 2001.

[4] J. Barbier, G. Sicot, and S. Houcke, "Algebraic Approach of the Reconstruction of Linear and Convolutional Error Correcting Codes," in World Academy of Science, Engineering and Technology, vol. 16, Nov. 2006, pp. 66–71.

[5] A Bonvard, S Houcke, R Gautier, M Marazin, "Classification based on Euclidean distance distribution for blind identification of error correcting codes in noncooperative contexts", IEEE Transactions on Signal Processing 66 (10), 2018, 2572-2583

[6] M. Cluzeau and J. Tillich, "On the Code Reverse Engineering Problem," in Proc. of the IEEE Int. Symp. Information Theory. Toronto, Canada : IEEE, 2008, pp. 634–638.

[7] M. Cluzeau and M. Finiasz, "Recovering a Code's Length and Synchronization from a Noisy Intercepted Bitstream," in Proc. of the IEEE Int. Symp. Information Theory. Seoul, Korea : IEEE, 2009, pp. 2737–2741.

[8] E. Filiol, "Reconstruction of Convolutional Encoders over GF (q)," in Cryptography and Coding : 6th IMA Int. Conf., 1997, pp. 101–109.

[9] P. Lu, L. Shen, X. Luo, and Y. Zou, "Blind Recognition of Punctured Convolutional Codes," in Proc. of the IEEE Int. Symp. Information Theory. Chicago, USA : IEEE, Jul. 2004, p. 457.

[10] J. Dingel and J. Hagenauer, "Parameter Estimation of a Convolutional Encoder from Noisy Observation," in Proc. of the IEEE Int. Symp. Information Theory. Nice, France : IEEE, Jun. 2007, pp. 1776–1780.

[11] F. Wang, Z. Huang, and Y. Zhou, "A Method for Blind Recognition of Convolution Code Based on Euclidean Algorithm," in International Conference on Wireless Communications and Mobile Computing. Shanghai : IEEE, Sep. 2007, pp. 1414–1417.

[12] M. Côte and N. Sendrier, "Reconstruction of convolutional codes from noisy observation," in Proc. of the IEEE Int. Symp. Information Theory. Seoul, Korea : IEEE, 2009, pp. 546–550.

[13] M. Cluzeau, M. Finiasz, and J. Tillich, "Methods for the Reconstruction of Parallel Turbo Codes," in Proc. of the IEEE Int. Symp. Information Theory. Austin, Texas, USA : IEEE, Jun. 2010, pp. 2008–2012.

[14] M. Marazin, R. Gautier, and G. Burel, "Algebraic method for blind recovery of punctured convolutional encoders from an erroneous bit- stream," IET Signal Processing, vol. 6, no. 2, pp. 122–131, Apr. 2012.

[15] J. Barbier, "Reconstruction of turbo-code encoders," in Proceedings of SPIE, vol. 5819, 2005, pp. 463–473.

[16] R. Gautier, M. Marazin, and G. Burel, "Blind Recovery of the Second Convolutional Encoder of a Turbo-Code when its Systematic Outputs are Punctured," in 7-th IEEE-Communications 2008, Bucharest, Romania, 2008, pp. 345–348.

[17] C. Berrou, Y. Saouter, C. Douillard, S. Kerouedan and M. Jezequel, "Designing good permutations for turbo codes : towards a single model," 2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577), Paris, France, 2004, pp. 341-345.

[18] Ronald Garzón Bohórquez ;Charbel Abdel Nour ;Catherine Douillard, "On the Equivalence of Interleavers for Turbo Codes", IEEE Wireless Communications Letters 2015 Vol 4, Issue 1.

[19] Weithoffer, S., Griebel, O., Klaimi, R., Nour, C. A., Wehn, N., "Advanced hardware architectures for turbo code decoding beyond 100 Gb/s.", In 2020 IEEE Wireless Communications and Networking Conference (WCNC) (pp. 1-6).

[20] Weithoffer, S., Nour, C. A., Wehn, N., Douillard, C., Berrou, C., "25 years of turbo codes : From Mb/s to beyond 100 Gb/s". In 2018 IEEE 10th International Symposium on Turbo Codes and Iterative Information Processing (ISTC) (pp. 1-6).

[21] J. Sun and O. Y. Takeshita. "Interleavers for turbo codes using permutation polynomials over integer rings", IEEE Trans. on Inf. Theory, 51(1) :101–119, January 2005.