

Algorithme faible complexité pour le calcul de la distribution de la distance minimale de codes polaires

Malek ELLOUZE¹ Romain TAJAN¹ Camille LEROUX¹ Christophe JÉGO¹ Charly POUILLIAT²

¹Université de Bordeaux, Bordeaux INP, Laboratoire IMS, UMR CNRS 5218, France

²Université de Toulouse, INPT-ENSEEIH, Laboratoire IRIT, UMR CNRS 5505, France

Résumé – Dans cet article, un algorithme de faible complexité permettant de calculer une borne précise de la distance minimale et de son nombre d’occurrences est introduit pour les codes polaires. Cette méthode peut être appliquée pour n’importe quelle stratégie de bits gelés et peut donc être étendue à n’importe quel type de codes polaires, dont ceux qui utilisent des bits gelés dynamiques ou les codes PAC. L’algorithme proposé permet une estimation précise de la probabilité d’erreur paquet d’un décodeur polaire par liste. Des résultats expérimentaux sont donnés pour les codes polaires du standard 5G pour une taille allant jusqu’à $N = 1024$.

Abstract – In this article, a low-complexity algorithm is provided to compute an accurate bound on both the minimum distance and its associated occurrence for polar codes. This method can be applied regardless of the frozen bits set. It can thus be extended to any type of polar code, including those using dynamic frozen bits or polarization-adjusted convolutional codes. The estimation of the minimum distance and the number of occurrences also enable to estimate the SCL decoder’s performances. Experimental results are given for 5G standard polar codes as large as $N = 1024$.

1 Introduction

Les codes polaires [2] suscitent un intérêt grandissant puisqu’il s’agit de la première famille de codes correcteurs d’erreurs à atteindre la capacité de Shannon pour un canal sans mémoire avec un décodage à annulations successives (SC). Cependant, le décodage SC est seulement efficace pour de grandes tailles de code. A taille finie, la propagation des erreurs dégrade fortement les performances de décodage. Celles-ci peuvent être améliorées en utilisant le décodage à annulations successives par liste (SCL) [12]. Ce constat a motivé la proposition de codes polaires avec de meilleures propriétés de distance [8, 9, 13]. Afin de prédire leurs performances, il est intéressant de pouvoir estimer le nombre de mots de codes de poids faibles pour n’importe quelle taille et/ou stratégie de bits gelés.

Les propriétés de distance des codes polaires ont été étudiées dans plusieurs travaux de recherche. Dans [6], les auteurs ont fourni une méthode pour calculer la distance minimale du code dans le cas où les bits d’information sont sélectionnés selon leur poids. [3] a permis d’estimer le nombre de mots de code de distance minimale pour des stratégies spécifiques de bits gelés. Dans [7], les auteurs ont pu déterminer le nombre de mots de code ayant une faible distance via des simulations de Monte Carlo avec de grandes tailles de listes. Dans [11], un algorithme est proposé pour calculer le nombre de mots de code ayant un poids minimal pour les PAC codes [8]. Une méthode pour énumérer les mots de codes de poids faibles pour des ensembles spécifiques a été proposée dans [10]. Cette contribution est à l’origine d’une méthode décrite dans [15] qui permet de calculer tout le spectre d’un code polaire. Elle s’appuie sur les groupes d’automorphisme des codes polaires afin de réduire sa complexité calculatoire. Cependant, sa complexité demeure élevée ce qui la rend inutilisable même pour des tailles modérées de code.

Dans cet article, un algorithme de faible complexité permettant

de calculer une borne très précise de la distance minimale et du nombre d’occurrences est présenté. Contrairement à [3, 6, 11, 15], cette méthode est universelle dans le sens où elle peut être appliquée à n’importe quelle stratégie de bits gelés. De plus, en considérant que tout code polaire peut être décrit comme l’union de cosets prédéfinis, une réduction drastique de complexité est obtenue en considérant seulement les cosets pertinents. Les résultats obtenus permettent d’estimer les taux d’erreur paquet pour un décodage par liste de codes polaires.

2 Préliminaires

2.1 Codes linéaires en blocs

Considérons $\mathbf{a}_i^j \triangleq (a_i, a_{i+1}, \dots, a_j)$ et $\mathbb{F}_2 = \{0, 1\}$ le groupe fini d’ordre 2. Soit \mathbb{F}_2^N l’espace de tous les N -uplets \mathbf{x}_0^{N-1} définis dans \mathbb{F}_2 , doté de son opération usuelle d’addition notée \oplus . Soit $\mathcal{C}(N, K)$ un code linéaire en bloc de longueur N avec K bits d’informations et de rendement $R = K/N$. Ces 2^K mots de code forment un sous-espace de dimension K dans \mathbb{F}_2^N . Le poids de Hamming d’un mot de code $\mathbf{c} \in \mathbb{F}_2^N$, défini comme le nombre d’éléments non nuls est noté $w(\mathbf{c})$.

La distance de Hamming $d(\mathbf{c}, \check{\mathbf{c}}) = w(\mathbf{c} \oplus \check{\mathbf{c}})$ entre deux mots de code \mathbf{c} and $\check{\mathbf{c}}$ correspond au nombre d’éléments pour lequel ils diffèrent. La distance minimale du code \mathcal{C} est donnée par :

$$d^* = \min_{\mathbf{c} \in \mathcal{C}, \mathbf{c} \neq \mathbf{0}} w(\mathbf{c}) \quad (1)$$

2.2 Codes polaires

Un code polaire $\mathcal{C}(N, K)$ est caractérisé par sa longueur $N = 2^n$. La matrice génératrice d’un code polaire s’obtient à

partir de la sous-matrice de K lignes de la matrice de Kronecker polaire $G = G_2^{\otimes n}$ où $G_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ est la matrice du noyau et $(\cdot)^{\otimes n}$ définit le produit n -ième de Kronecker. Les mots de code d'un code polaire sont obtenus comme $\mathbf{c} = \mathbf{u}G$, où $\mathbf{u} \in \mathbb{F}_2^N$. \mathbf{u} est un vecteur d'information binaire dans lequel K positions sont allouées aux bits d'information, tandis que le reste des positions est *gelé/fixé* à des valeurs connues. Nous notons \mathcal{F} l'ensemble des indices des composantes de \mathbf{u} correspondant aux bits gelés. Dans notre étude, sans perte de généralité, nous supposons que les bits gelés sont fixés à 0. Cette méthode peut être appliquée à d'autres types de codes polaires comme indiqué dans la section 4.3.

Comme expliqué dans [15], étant donné $\mathbf{u}_0^{i-1} \in \mathbb{F}_2^{i-1}$ et $u_i \in \mathbb{F}_2$, un coset polaire $\mathcal{C}_N^{(i)}$ peut être exprimé comme :

$$\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i) = \{[\mathbf{u}_0^{i-1}, u_i, \mathbf{u}_{i+1}^{N-1}]G | \mathbf{u}_{i+1}^{N-1} \in \mathbb{F}_2^{N-i-1}\} \quad (2)$$

Un coset polaire $\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i)$ décrit donc l'espace des mots de code générés par le préfixe \mathbf{u}_0^i sans tenir compte des bits gelés contenus dans \mathbf{u}_{i+1}^{N-1} .

Pour un coset polaire donné $\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i)$, $w^*(\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i))$ correspond au poids minimal et $A^*(\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i))$ est le nombre d'occurrences associé.

3 Enumération optimisée du poids minimal

Dans cette section, nous utilisons une méthode qui sera détaillée dans de prochains travaux permettant de calculer la distance minimale et le nombre d'occurrences associé pour calculer une borne précise de la distance minimale et son nombre d'occurrence pour un code polaire. Cette méthode ne sera pas détaillée dans cet article par manque d'espace. Nous montrons également que dans tous les cas étudiés, nous sommes capables de retrouver la valeur exacte de la distance minimale et du nombre d'occurrences pour tout coset ($\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i)$).

Nous avons montré dans ces travaux qu'il est possible de calculer le poids minimal ainsi que son nombre d'occurrences associé pour chaque coset polaire à partir des graphes facteurs utilisés pour le décodage SC. Ces graphes étant des arbres, des règles régissant la propagation du poids minimal ont été déduites et permettent de calculer le poids minimal et le nombre d'occurrences associé pour tout coset.

Comme détaillé dans [15], tout code polaire \mathcal{C} peut être exprimé comme l'union disjointe des cosets :

$$\mathcal{C} = \bigcup_{\substack{u_i \in \{0, 1\}, \forall i \notin \mathcal{F} \\ u_i = 0, \forall i \in \mathcal{F}}} \mathcal{C}_N^{(s)}(\mathbf{u}_0^{s-1}, u_s = 0) \quad (3)$$

où s décrit l'index du dernier bit gelé, i.e $s = \max(\mathcal{F})$.

Cette expression résulte du fait qu'à partir du dernier bits gelé, les cosets décrivent tous les mots de code d'un code polaire. La preuve de (3) est assez triviale étant donnée qu'à partir du dernier bit gelé, les cosets décrivent tout l'ensemble des mots de codes d'un code polaire. La distance minimale d^* peut donc

être exprimée comme suit :

$$d^* \triangleq w^*(\mathcal{C} \setminus \mathcal{C}_N^{(N-1)}(\mathbf{0}_0^{N-2}, u_{N-1} = 0)) = \min(d_1, d_2) \quad (4)$$

où

$$d_1 = \min_{\mathbf{u}_0^{s-1} \neq \mathbf{0}_0^{s-1}} w^*(\mathcal{C}_N^{(s)}(\mathbf{u}_0^{s-1}, u_s = 0)), \quad (5)$$

$$d_2 = w^*(\mathcal{C}_N^{(i)}(\mathbf{0}_0^{i-1}, u_i = 1 | i \in [s+1, N-1])).$$

Pour obtenir les valeurs exactes de d^* et A^* , le nombre total de cosets à explorer peut être exprimé comme :

$$n_c = 2^\gamma + (N - s) \quad (6)$$

où γ , appelé mixing factor dans [15], décrit le nombre total de bits d'information avant le dernier bit gelé. Il en découle que même pour des tailles de code modérées, la complexité calculatoire est très élevée.

Afin de palier cette limitation calculatoire, nous proposons un algorithme à complexité réduite qui fournit une estimation précise de la distance minimale et du nombre d'occurrences associé. En effet, il est possible de démontrer que pour tout préfixe \mathbf{u}_0^i dont le coset associé a un poids minimal plus grand que celui d'un mot de code appartenant au code peut être ignoré pendant l'exploration des différents chemins.

De plus, afin de réduire davantage la taille de l'espace exploré, d_2 est calculé en premier lieu avec le nombre d'occurrences associés.

L'algorithme 1 détaille le calcul de la distance minimale et du nombre d'occurrences associé. Cela est réalisé de la manière suivante :

- Pour un code polaire et une stratégie de bits gelés, $w^*(\mathcal{C}_N^{(i)}(\mathbf{0}_0^{i-1}, u_i = 1 | i \in [s+1, N-1]))$ est calculé en premier lieu avec le nombre d'occurrences associé.
- Tous les chemins possibles \mathbf{u}_0^i à un étage de décodage i tel que $i \leq s$ sont énumérés.
- Pour chacun des chemins mentionnés, $d_2 = \mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i)$ est calculé.
- Les chemins tel que $w^*(\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i)) > d_2$ sont supprimés.
- Si le nombre de chemins considérés dépasse une valeur maximale L , alors seuls L chemins ayant le plus faible $w^*(\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i))$ sont gardés.

Cette énumération de chemins peut être vue comme un décodeur liste introduit dans [12] où les métriques sont basées sur le poids minimal au lieu des rapports de vraisemblance (LLRs).

Soit l_i la taille de la liste à chaque étage i . Il est important de souligner que si, durant l'exécution, $l_i < L \quad \forall i \in [1, s]$ alors l'algorithme proposé calcule la distance minimale et son nombre d'occurrences associés exacts et non pas juste une borne.

TABLE 1 : Résultats pour des codes polaires ayant des longueurs entre 128 et 1024

N	128			256			512			1024	
K	96	64	42	192	128	85	384	256	170	768	512
R	$\frac{3}{4}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{3}{4}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{3}{4}$	$\frac{1}{2}$	$\frac{1}{3}$	$\frac{3}{4}$	$\frac{1}{2}$
s	73	98	113	193	201	225	385	417	453	771	897
γ	41	34	27	129	73	54	257	161	111	515	385
d^*	4	8	16	8	8	16	8	8	16	8	16
A^*	96	304	1304	61536	96	816	53440	64	352	24960	36032
n_c	$2.1e^{12}$	$1.7e^{10}$	$1.3e^8$	$6.8e^{38}$	$9.4e^{21}$	$1.8e^{16}$	$2.3e^{77}$	$2.9e^{48}$	$2.5e^{33}$	$1.07e^{155}$	$7.8e^{115}$
n_r	114	167	1826	406055	210	788	317031	450	654	358938	178870
L_m	1	10	230	15122	1	42	6690	1	17	2113	4514

Algorithme 1 : Calcul de d^* and A^* pour un code polaire

Entrées : Code polaire $\mathcal{C}(N, K, \mathcal{F}, L)$
 $s \leftarrow$ index du dernier bit gelé
pour $i \in [s + 1, N - 1]$ **faire**
 | Calculer w^* and A^* de $\mathcal{C}_N^{(i)}(\mathbf{0}_0^{i-1}, u_i = 1)$
fin
 $d_{start} \leftarrow \min w^*(\mathcal{C}_N^{(i)}(\mathbf{0}_0^{i-1}, u_i = 1))$
 $A_{start} \leftarrow$ Occurrences de d_{start}
pour $i \in [1, s]$ **faire**
 | Calculer w^* and A^* de $\mathcal{C}_N^{(i)}(\mathbf{u}_0^{i-1}, u_i)$ pour
 | chaque chemin $\in \mathcal{L}$
 | Supprimer les chemins pour lesquels $w^* > w_{start}$
 | **si** $L > \mathcal{L}$ **alors**
 | | $\mathcal{L} \leftarrow$ Garder les L chemins ayant le plus faible
 | | poids minimal;
 | **sinon**
fin
 $d^* \leftarrow \min_{\mathbf{u}_0^{s-1} \neq \mathbf{0}_0^{s-1}} w^*(\mathcal{C}_N^{(s)}(\mathbf{u}_0^{s-1}, u_s = 0))$
si $d^* \neq d_{start}$ **alors**
 | $d^* \leftarrow \min(d^*, d_{start})$
 | $A^* \leftarrow$ Occurrences de d^*
sinon
fin
 Retourner (d^*, A^*)

4 Résultats expérimentaux

Dans cette section, nous présentons des résultats de calcul de la distance minimale et du nombre d'occurrences associé pour des codes polaires dont la taille va jusqu'à $N = 1024$. Nous estimons également les performances d'un décodeur SCL à partir des valeurs sur d^* et A^*

4.1 Calcul de d^* et A^*

L'algorithme 1 a été appliqué sur une large sélection de codes polaires, pour différents rendements et des longueurs de codes allant jusqu'à $N = 1024$ *. Lors des expérimentations, les

Il est possible de calculer d^ and A^* pour des tailles de code encore plus grande et n'importe quel rendement

structures de bits gelés utilisées sont celles spécifiées dans le standard 5G [1].

La taille maximale de la liste est $L = 10^6$. Le tableau 1 donne la distance minimale d^* et le nombre d'occurrences associé A_{w^*} pour chaque code étudié. La taille maximale de liste L_m et le nombre réel de cosets n_r explorés durant l'exécution de l'algorithme sont également donnés.

Le code (128,64) est également étudié dans [15] et on retrouve bien les mêmes valeurs de d^* et A^* .

La simplification introduite dans l'algorithme 1 diminue le nombre de cosets explorés de manière drastique. Ceci peut être observé en comparant les valeurs de n_d and n_r . Cet effondrement de la complexité explique la capacité de l'algorithme proposé à être appliqué à des codes polaire de grande taille. Parmi toutes les configurations explorées dans les expérimentations, la taille maximale de liste est de $L_m = 15122$. Cela signifie que pour ces configurations, nous sommes capables de calculer la vraie distance minimale et le nombre exact d'occurrences associé.

4.2 Approximation des performances du SCL

Le calcul de d^* et A^* pour $N = 256$ et K variant entre 1 et N et pour la configuration de bits gelés du standard 5G est présenté dans la figure 1. Cela montre d'une part que nous sommes en mesure de calculer la distance minimale et le nombre d'occurrences associé pour tous les rendements d'un code polaire avec une stratégie de bits gelés donnée. De plus, il est intéressant d'observer que sur chaque plage de d^* constant, A^* augmente en même temps que le rendement.

Le calcul de d^* et A^* permet d'estimer les performances d'un décodeur SCL pour de forts SNR. En effet, pour un SNR élevé, le décodeur SCL se comporte comme un décodeur à maximum de vraisemblance dont la probabilité d'erreur paquet peut être exprimée telle que [14] :

$$P_e = \sum_{d=d^*}^N A_d \mathcal{Q} \left(2dR \frac{E_b}{N_0} \right) \quad (7)$$

où A_d désigne le nombre de mots de code de poids d , $\frac{E_b}{N_0}$ est le rapport signal à bruit (SNR) et $\mathcal{Q}(u)$ la fonction définie par $\mathcal{Q}(u) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{u^2}{2}} du$. Cette expression peut être

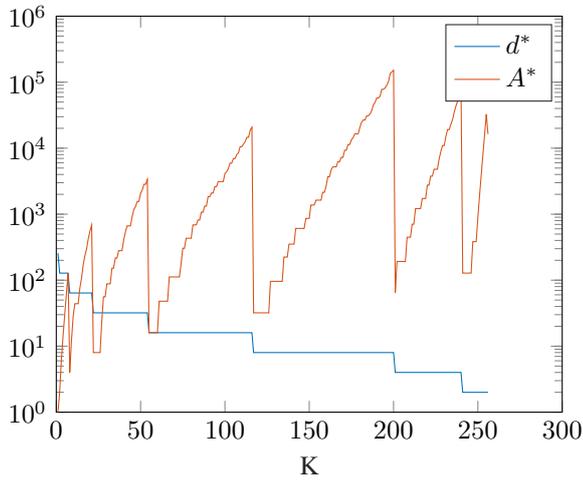


FIGURE 1 : d^* et A^* pour $N = 256$ en fonction de K

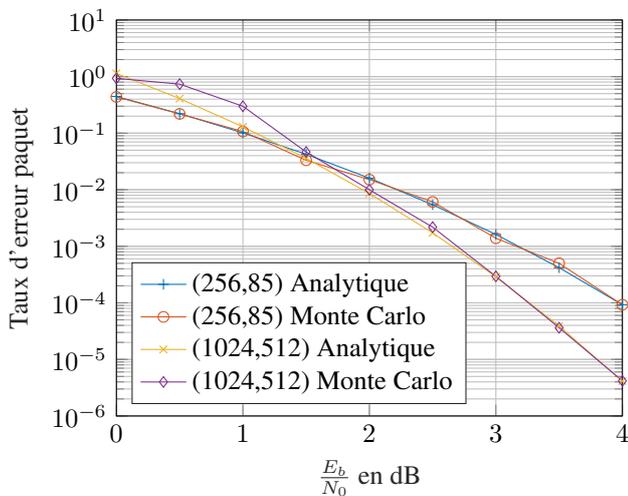


FIGURE 2 : Comparaison d'un taux d'erreur paquet pour les codes polaires (256,85) et (1024,512)

approximée par :

$$P_e \approx A^* \mathcal{Q} \left(2d^* R \frac{E_b}{N_0} \right) \quad (8)$$

Sur la figure 1, en prenant en compte l'approximation de l'équation (8), on peut observer que pour un même $\frac{E_b}{N_0}$ donné et sur une plage de d^* fixe, l'augmentation de A^* entraîne une augmentation du taux d'erreur paquet.

L'évolution du taux d'erreur paquet pour les codes polaires (256,85) et (1024,512) obtenue par le biais de l'équation 8 a été comparée à une simulation de Monte Carlo [4] obtenue pour un décodeur SCL ayant une taille de liste 8. La figure 2 montre qu'à partir de 2dB, les deux courbes se superposent pour les deux codes considérés. La même tendance a été observée sur tous les codes étudiés.

4.3 Extension à d'autres codes polaires

Une extension de cette contribution est possible. Dans un premier temps, l'algorithme proposé peut être appliqué quelques soient les valeurs et les positions des bits gelés. Cet algorithme peut donc être appliqué à des codes polarization-adjusted convolutional (PAC) introduits dans [8], ou plus généralement

à des codes polaires avec des bits gelés dynamiques [13] où la valeur des bits gelés évolue durant le processus de décodage. En outre, cette approche peut être généralisée aux codes polaires multi-noyaux [5] si les graphes de décodage sont en arbre. Une extension aux codes polaires non-binaires est également possible.

5 Conclusion

Un algorithme faible complexité qui explore un nombre limité de cosets avec une contrainte sur leur poids minimal a été introduit dans cet article. Il permet d'obtenir une estimation précise de la distance minimale ainsi que le nombre d'occurrences associé. La méthode ainsi proposée évite l'exploration exhaustive de tous les cosets dont le nombre croît de manière exponentielle avec la taille du code. Les résultats expérimentaux montrent que la distance minimale et le nombre d'occurrences associé pour des codes du standard 5G ($N \leq 1024$) peuvent être obtenus, ce qui n'est pas possible avec les travaux de la littérature.

Références

- [1] 3GPP TS 38.212 V17.4.0 : 5G ; NR ; multiplexing and channel coding. 2023.
- [2] Erdal ARIKAN : Channel polarization : A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels. *IEEE Trans. on Inf. Theory*, 55(7):3051–3073, 2009.
- [3] Magali BARDET, Vlad DRAGOI, Ayoub OTMANI et Jean-Pierre TILLICH : Algebraic properties of polar codes from a new polynomial formalism. *In 2016 IEEE International Symposium on Information Theory (ISIT)*, pages 230–234, 2016.
- [4] Adrien CASSAGNE, Olivier HARTMANN, Mathieu LÉONARDON, Kun HE, Camille LEROUX, Romain TAJAN, Olivier AUMAGE, Denis BARTHOU, Thibaud TONNELIER, Vincent PIGNOLY, Bertrand LE GAL et Christophe JÉGO : Aff3ct : A fast forward error correction toolbox ! *SoftwareX*, 10:100345, 2019.
- [5] Frédéric GABRY, Valerio BTOGLIO, Ingmar LAND et Jean-Claude BELFIORE : Multi-kernel construction of polar codes. *In 2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, pages 761–765, 2017.
- [6] Nadine HUSSAMI, Satish Babu KORADA et Rudiger URBANKE : Performance of polar codes for channel and source coding. *In 2009 IEEE International Symposium on Information Theory*, pages 1488–1492, 2009.
- [7] Bin LI, Hui SHEN et David TSE : An adaptive successive cancellation list decoder for polar codes with cyclic redundancy check. *IEEE Communications Letters*, 16(12):2044–2047, 2012.
- [8] Mohsen MORADI, Amir MOZAMMEL, Kangjian QIN et Erdal ARIKAN : Performance and complexity of sequential decoding of PAC codes. *CoRR*, abs/2012.04990, 2020.
- [9] Kai NIU et Kai CHEN : CRC-aided decoding of polar codes. *IEEE Communications Letters*, 16(10):1668–1671, 2012.
- [10] Rina POLYANSKAYA, Mars DAVLETSHIN et Nikita POLYANSKII : Weight distributions for successive cancellation decoding of polar codes. *IEEE Transactions on Communications*, 68(12):7328–7336, 2020.
- [11] Mohammad ROWSHAN et Jinhong YUAN : Fast enumeration of minimum weight codewords of PAC codes. *In 2022 IEEE Information Theory Workshop (ITW)*, pages 255–260, 2022.
- [12] Ido TAL et Alexander VARDY : List decoding of polar codes. *In 2021 IEEE International Symposium on Information Theory (ISIT)*, pages 1–5, 2011.
- [13] Peter TRIFONOV et Vera MILOSLAVSKAYA : Polar codes with dynamic frozen symbols and their decoding by directed search. *In 2013 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2013.
- [14] A. VITERBI : Convolutional codes and their performance in communication systems. *IEEE Transactions on Communication Technology*, 19(5):751–772, 1971.
- [15] Hanwen YAO, Arman FAZELI et Alexander VARDY : A deterministic algorithm for computing the weight distribution of polar codes. *In 2021 IEEE International Symposium on Information Theory (ISIT)*, pages 1218–1223, 2021.