Décodage des codes LDPC non binaires : un algorithme à très faible complexité

Adrian VOICILA[1,3], David DECLERCQ[1], Marc FOSSORIER[2], François VERDIER[1], Pascal URARD[3]

[1]ETIS ENSEA/UCP/CNRS UMR-8051, 95014 Cergy-Pontoise, (France),

[2]Dept. Electrical Engineering, Univ. Hawaii at Manoa, Honolulu, HI 96822, (USA),

[3]STMicroelectronics, Crolles, (France)

voicila@ensea.fr, declercq@ensea.fr
marc@aravis.eng.hawaii.edu, verdier@ensea.fr,pascal.urard@st.com

Résumé – Dans cet article, nous présentons un algorithme de décodage simplifié des codes LDPC non-binaires, appelé EMS. Cet algorithme est une simplification de l'algorithme somme-produit dans le domaine logarithmique, qui permet des gains en termes de complexité, de stabilité numérique et de réduction de l'espace de stockage des messages. Nous proposons également différentes variantes d'implantation de l'algorithme EMS. Les différentes variantes présentées sont comparées par des courbes de performances (FER) et par la complexité. Cet article présente aussi une comparaison intéressante entre l'algorithme de décodage proposé (EMS) et son homologue binaire Min-Sum en termes de performances de décodage et complexité.

Abstract – In this paper, we introduce a simplified log-domain decoding algorithm for LDPC codes over GF(q), named EMS. While this algorithm is a simplification of the conventional sum-product algorithm, the log-domain decoding has advantages in terms of implementation, computation complexity and numerical stability. Further, we propose some variants of implementation for the EMS algoritm, yielding a lower computational complexity. The different variants are compared with their binary homologous, the Min-Sum algorithm, both in terms of simulated FER performance and computational complexity.

1 Introduction

Les codes LDPC construits sur des corps de Galois non binaires GF(q) (d'ordre q) et décodés avec l'algorithme de propagation de croyances (BP), représentent une solution efficace pour la correction des erreurs dans le cas de trames courtes et/ou de modulations d'ordres élevés [2, 4]. Cependant, les décodeurs associés ont le désavantage d'avoir une complexité variant en $O(q^2)$ [2,3] et l'espace mémoire nécessaire au stockage des messages en $O(q)$. En conséquence, les corps d'ordre élevés ($q \geq 64$) ne peuvent être considérés pour une implantation matérielle. Le décodage BP des codes LDPC consiste en la mise à jour des messages (taille q) circulant sur les branches du graphe bipartite associé aux codes LDPC [2]. Les messages circulant sur les branches représentent les densités de probabilité des symboles du code et sont donc des vecteurs de taille q. Dans [6] nous avons présenté un algorithme de décodage simplifié pour les codes non binaires ayant une complexité dominée par $O({n_m}^2)$, basé sur l'utilisation des $n_m << q$ valeurs les plus fiables lors de la mise à jour des messages par la fonction de parité. Nous proposons dans cet article deux améliorations significatives de réduction de complexité par rapport à l'algorithme présenté en [6] : (*i*) dans [6], l'espace mémoire nécessaire n'était pas réduit et restait de l'ordre $O(q)$. Nous proposons une structure modifiée des messages permettant de réduire la complexité de stockage en $O(n_m)$ avec $n_m << q$, à l'aide d'une opération de troncature. Cette approche est décrite en détails dans la section 3 et résoud partiellement le problème de la taille mémoire nécessaire à l'implantation d'un décodeur LDPC non binaire. (*ii*) la structure tronquée des messages nous a conduit à proposer une réduction de la complexité calculatoire supplémentaire à celle proposée dans [6]. De la même façon que dans [3,6], on peut ramener l'étude de la complexité globale d'un décodeur basé sur le BP à l'étude d'une " étape élémentaire", correspondant à une équation de parité minimale ne faisant intervenir que 3 symboles $GF(q)$. Dans [6], la complexité de cette étape élémentaire variait en $O({n_m}^2)$, et nous proposons ici un algorithme optimisé permettant de réduire cette complexité à un ordre $O(n_m \log_2 n_m)$. Cet algorithme est décrit dans la section 4. Le principal intérêt du nouvel algorithme que nous présentons ici est de montrer par simulations que les simplifications (*i*) et (*ii*) apportent une perte de performance négligeable, en utilisant une compensation optimisée des messages. L'algorithme, appelé EMS, que nous proposons ici a une complexité trés faible et représente la première solution de décodeur non binaire pratique pouvant concurrencer les décodeurs binaires. Nous montrons dans la section 5 des simulations de taux d'erreur trame prouvant cette assertion.

2 Notations et position du problème

Un code LDPC dans $GF(q)$ est défini par une matrice de parité H creuse de dimension $M \times N$ ($R = \frac{N-M}{N}$ est le rendement) dont les éléments non nuls appartiennent à $GF(q)$. Les multiplications effectuées dans le corps $GF(q)$ seront notées $\otimes$ et tout élément d'un corps GF(q) peut être décrit en fonction d'un élément primitif du corps α. Le corps de Galois est donc composé des éléments $\{0, \alpha^0, \alpha^1, ..., \alpha^{q-2}\}$.

La représentation graphique des codes LDPC non-binaires consiste à connecter l'ensemble des N nœuds de variables aux M nœuds de parité. Dans le cas de codes LDPC réguliers, les nœuds de variables sont connectés à d_v branches et les nœuds de parité à d_c branches.

L'algorithme de propagation de croyances consiste en deux étapes de calcul. Premièrement le passage par les nœuds de données est une multiplication termes à termes des $d_v - 1$ messages entrants. Deuxièmement, le passage par les nœuds de parité consite en une marginalisation des d_c messages entrant conditionnellement à l'équation de vérification de parité :

$$\sum_{t=1}^{d_c} h_t \otimes c_t = 0 \tag{1}$$

où h_t sont des éléments non-nuls de la matrice H et c_t les symboles correspondants du mot de code. Cette deuxième étape est très complexe et représente l'obstacle majeur au décodage de codes LDPC dans des corps d'ordre élévés. La complexité par nœud de parité est de l'ordre de $O(q^2)$, et peut être ramenée à une complexité moindre $O(q * log(q))$ lorsqu'on effectue le calcul dans le domaine de Fourier [2]. Cependant, sous cette forme l'algorithme de propagation de croyance requiert des multiplications et des divisions, ce qui est dommageable à toute implantation pratique. En vue d'une implantation matérielle des codes LDPC non-binaires, il est donc nécessaire de proposer un décodeur utilisant des logarithmes de rapports de vraisemblance comme messages, ne nécessitant que des additions. Nous allons utiliser, pour désigner le vecteur log-rapports de vraisemblance (LDR) d'une variable aléatoire $z \in GF(q)$, la notation suivante :

$$\mathbf{L}(z) = [L[0] \dots L[q-1]]^T$$

où

$$L[i] = \log \frac{P(z = \alpha_i)}{P(z = \alpha_0)} \tag{2}$$

avec $P(z = \alpha_i)$ la probabilité que la variable aléatoire z est egale à $\alpha_i \in GF(q)$. Avec cette définition $L[0] = 0, \quad L[i] \in \mathbb{R}$. En général, les vecteurs message à la sortie du canal sont des vecteurs de dimension $q-1$ et ils sont notés par $\mathbf{L}_{ch} = [L_{ch}[k]_{k\in\{0,\dots,q-1\}}]^T$. Dans ce cas, les probabilités $P(z = \alpha_i)$ dépendent de la statistique du canal.

3 Structure des messages tronqués

On note **U** (respectivement **V**) les vecteurs des messages entrants (respectivement sortants) dans un noeud de parité, représentés à l'aide de LDR, comme dans [3, 4, 6]. Les messages **V** et **U** sont limités à seulement n_m valeurs (les plus grandes). De plus, les valeurs d'un message sont triées par ordre décroisant. Ainsi $V[0]$ représente la valeur maximale et $V[n_m - 1]$ est la valeur minimale de **V**. Bien qu'intéressante en terme de réduction de mémoire et de calcul, l'opération de troncature de messages introduit une perte d'information qui conduit à une dégradation des performances du décodeur. Cette perte de performances peut être atténuée en employant une compensation appropriée de l'information qui a été tronquée. Dans le but de développer un décodeur à faible complexité, nous avons choisi de compenser les $(q - n_m)$ valeurs tronquées par une valeur scalaire simple γ, que l'on se propose de justifier dans la suite de cette section.

Définition [message compensé] :
Soit **A** *un message du graphe qui représente un vecteur LDR de taille* q*. Le message tronqué associé* **B** *contient les* n_m *plus grandes valeurs de* **A***, triées dans un ordre décroissant, plus une valeur supplémentaire* $\gamma_A \in \mathbb{R}$.

Une manière de fixer la valeur du scalaire γ_A est de supposer que le message tronqué doit représenter un vecteur de poids de probabilité dont la somme des éléments doit valoir 1 : $\sum_{k=0}^{n_m-1} P_B[k] + (q - n_m)P_{\gamma_A} = 1$, où $\{\mathrm{B[k]}\}_{k\in\{0,\dots,n_m-1\}}$ représente les valeurs des probabilités associées au vecteur LDR **B**. La normalisation du vecteur $\mathbf{P}_B$ est donnée par :

$$(q - n_m)P_{\gamma_A} = 1 - P_A[0] \sum_{k=0}^{n_m-1} e^{B[k]}$$

$$\Rightarrow \quad \frac{P_{\gamma_A}}{P_A[0]} = \frac{\frac{1}{P_A[0]} - \sum_{k=0}^{n_m-1} e^{B[k]}}{q - n_m}$$

$$\log \frac{P_{\gamma_A}}{P_A[0]} = \log \left(\sum_{k=0}^{q-1} e^{A[k]} - \sum_{k=0}^{n_m-1} e^{B[k]} \right) - \log(q - n_m)$$

$$\gamma_A = \log \left(\sum_{k=0, A[k]\notin B}^{q-1} e^{A[k]} \right) - \log(q - n_m)$$

On remarque que le calcul de la valeur supplémentaire γ_A requiert les $q - n_m$ valeurs ignorées du vecteur **A**, et l'utilisation d'une fonction non-linéaire. La fonction non-linéaire peut être exprimer en fonction de l'opérateur $\max^*(x_1, x_2)$ [3]. Ainsi, un utilisant l'approximation
$\max^*(x_1, x_2) = \log(e^{x_1} + e^{x_2}) \approx max(x_1, x_2)$ [3], nous obtenons :

$$\gamma_A \approx B[n_m] - \log(q - n_m) \tag{3}$$

où $B[n_m]$ est la plus grande valeur parmi les $(q - n_m)$ valeurs ignorées du vecteur **A**. Afin de compenser l'approximation de l'opération max^*, on ajoute une correction globale sur tous les messages dont la valeur est fixée par simulation. Pour quantifier la valeur de la correction globale, nous calculons le seuil δ de convergence du décodeur en utilisant un algorithme d'évolution de densité, outil désormais classique pour l'étude des codes LDPC [1]. Ce seuil correspond au (E_b/N_0) minimal au delà duquel un code de taille infinie atteint une probabilité d'erreur nulle. Ainsi, plus le seuil δ est faible, meilleures sont les performances.

4 Algorithme à complexité minimale

L'algorithme EMS peut être implanté d'une manière efficace en utilisant la méthode récursive proposée dans [2] pour la mise à jour des noeuds de variable et de parité. Cette méthode s'interprète comme une implantation type "Forward/Backward" du calcul des fiabilités. A chaque étape de récursion, appelée étape élémentaire, on considère seulement deux messages entrants pour effectuer le calcul des fiabilités.

4.1 Etape élémentaire : Noeud de parité

Comme précisé dans [3, 6], une implantation récursive de l'étape correspondant à l'équation de parité permet de ne considérer que seulement deux messages entrants (**U, I**) et le message de sortie est stocké dans le vecteur **V**. Par souci de clarté on note $\beta_{\mathbf{V}}$, $\beta_{\mathbf{I}}$ et $\beta_{\mathbf{U}}$ (of size n_m) les vecteurs index qui stockent les éléments $\alpha_k \in GF(q)$ associés aux valeurs des LDR des vecteurs **V**, **I** et **U**. Par exemple, $U[k]$ représente la valeur LDR qui correspond au symbole $\beta_U[k] \in GF(q)$. Nous définisons $S(\beta_V[i])$ comme étant l'ensemble des toutes les combinaisons possibles des symboles qui vérifient l'équation de parité réduite $\beta_V[i] \oplus \beta_U[j] \oplus \beta_I[p] = 0$, avec $\beta_V[i] \in GF(q)$ l'élément du corps de Galois correspondant à la valeur $V[i]$. Les valeurs du message de sortie sont obtenues à l'aide de l'équation :

$$V[i] = \max_{S(\beta_V[i])} (U[j] + I[p]) \qquad i \in \{0, \ldots, n_m - 1\} \quad (4)$$

Nous proposons une stratégie de parcours des deux vecteurs triés **U** et **I**, qui fournit un nombre minimum d'opérations pour le calcul des n_m valeurs de sortie triées du vecteur **V**. Pour la clarté de la présentation, nous utilisons une matrice M construite à partir des vecteurs **U** et **I**, chaque élément de M étant de la forme $M[j, p] = U[j] + I[p]$. Cette matrice contient les n_m^2 candidats pour la mise à jour du vecteur **V**. Le but de notre algorithme est d'explorer d'une manière efficace la matrice M afin de calculer itérativement les n_m plus grandes valeurs de sortie, en utilisant le fait que M est construite à partir des messages triés :

1. Initialisation : les valeurs de la première colonne de M sont introduites dans un trieur de taille n_m.
2. Sortie : la plus grande valeur est calculée et recopiée dans le vecteur **V**.
3. Évolution : Le voisin droit - en ce qui concerne la matrice de M - de la valeur remplie est introduit dans le trieur .
4. Aller à (2)

En utilisant cet algorithme la complexité associée d'un nœud de parité est dominée par $\mathcal{O}(n_m \log_2 n_m)$ qui correspond au nombre d'opérations max nécessaires pour l'insertion des n_m valeurs dans une liste déjà triée. Cette complexité est significativement plus faible que celle des algorithmes existant dans la littérature [3, 4, 6].

4.2 Etape élémentaire : Noeud de variable

L'étape élémentaire pour la mise à jour des noeuds de variable est caractérisée par les messages d'entrée **V** , **I** et le message de sortie **U**. Les vecteurs **V** , **I** et **U** sont triés par ordre décroisant. Soient $\beta_{\mathbf{V}}$, $\beta_{\mathbf{I}}$ et $\beta_{\mathbf{U}}$ les vecteurs index qui leur sont associés.
En utilisant les équations de l'algorithme BP dans le domaine logarithmique pour la mise à jour des noeuds de variable [3], le but d'une étape élémentaire est de calculer le vecteur de sortie **U** contenant les n_m plus grandes valeurs parmi les $2n_m$ valeurs candidates. Soit T un vecteur intermédiaire de taille $2n_m$, le traitement d'une étape élémentaire dans le cas de la mise à jour d'un noeud de variable est décrit ci-dessous :

$$\begin{aligned} T[k] &= V[k] + Y \qquad k \in \{0, \ldots, n_m - 1\} \quad (5) \\ T[n_m + k] &= \gamma_V + I[k] \qquad k \in \{0, \ldots, n_m - 1\} \end{aligned}$$

avec

$$Y = \begin{cases} I[l] & \text{if } \beta_I[l] = \beta_V[k] \\ \gamma_I & \text{if } \beta_I[l] \notin \beta_{\mathbf{V}} \end{cases} \qquad k, l \in \{0, \ldots, n_m - 1\}$$

La valeur de compensation γ est utilisée lorsque l'index requis n'est pas présent dans le message d'entrée.
Si le vecteur d'entrée **V** correspond au vecteur de canal ($\mathbf{L}_{ch}$) du symbole reçu, l'equation (6) devient :

$$\begin{aligned} T[k] &= V[k] + Y \qquad k \in \{0, \ldots, n_m - 1\} \\ T[n_m + k] &= L_{ch}[\beta_I[k]] + I[k] \qquad k \in \{0, \ldots, n_m - 1\} \end{aligned}$$

Le message de sortie **U** contient les n_m plus grandes valeurs du vecteur T :

$$\mathbf{U} = \max_{n_m}(\mathbf{T})$$

4.3 Evaluation de la réduction de complexité et de l'espace mémoire de l'algorithme EMS

La complexité totale de l'algorithme EMS pour un code régulier (d_v, d_c) est donnée dans le tableau 1. On remarque que la complexité de décodage de l'algorithme proposé varie en $O(n_m \log_2 n_m)$ pour la mise à jour des noeuds de parité ainsi que pour les noeuds de variable. En un sens, les deux principales étapes de décodage ont des complexités équivalentes, ce qui peut constituer un avantage pour une implantation materielle basée sur un modèle de processeur générique. De plus, on peut remarquer que la complexité de décodage ne dépend pas de q, l'ordre de corps sur lequel le code est défini.
Le principal avantage de l'algorithme EMS est qu'il conduit à une réduction massive de la complexité de décodage (en $O(n_m \log_2(n_m))$) comparé aux algorithmes classiques (BP) qui sont de complexité variant en $O(q^2)$.
Discutons à présent de l'espace mémoire nécessaire à un décodeur LDPC non-binaire, cet espace pouvant être partagé en deux sous-ensembles indépendants. Le premier servant au stockage des messages représentant les vecteurs L_{ch} et l'autre pour les messages extrinsèques sous la forme des vecteurs **V** et **U**. Sous l'hypothèse que chaque valeur réelle de fiabilité, respectivement du symbole GF(q), est stockée sur $Nbits$ (resp. $\log_2(q)$) bits, l'espace mémoire occupé par les messages extrinsèques est égual à $n_m * N * d_v * (Nbits + \log_2 q)$. Le deuxième avantage de l'algorithme proposé est que son espace mémoire varie donc linéairement avec n_m avec $n_m \ll q$.

5 Comparaison avec les codes et décodeurs binaires

Nous présentons à présent une comparaison en termes de compromis complexité/performances entre l'algorithme EMS que nous proposons et son équivalent binaire, l'algorithme Min-Sum corrigé. La complexité de l'algorithme Min-Sum (MS) corrigé, par itération et pour un nœud de parité de degré d_{c_b}, est égale à : $3(d_{c_b} - 2)$ opérations `min`, $3(d_{c_b} - 2)$ opérations `XOR` dédiées au calcul du signe de la valeur de sortie et 2 additions réelles pour la correction. La complexité associée à un calcul de nœud de variable de degré d_{v_b} est, elle, de $2d_{v_b} - 1$ additions réelles. La complexité totale de l'algorithme EMS pour un

	Nb. max	Nb. add. réelles	Nb. add. GF(q)
Nœud de parité	$3(d_c-2)n_m \log_2 n_m$	$3(d_c-2)(2n_m)$	$3(d_c-2)(2n_m)$
Nœud de variable	$(d_v+2(d_v-2))n_m \log_2(2n_m)$	$(d_v+2(d_v-2))(2n_m)$	0
Node de permutation	0	0	$n_m(d_v+d_c)$

TAB. 1 – La complexité de l'algorithme EMS

code régulier (d_v, d_c) est donnée dans le tableau 1. Nous avons testé le comportement des deux algorithmes sur deux type de canaux, BI-AWGN et QAM-AWGN. Les codes binaires que nous utilisons sont des codes LDPC proposés dans [5] et sont connus pour être très compétitifs aux tailles considérées. Nous avons représenté dans les figures Fig.1 et Fig.2 une comparaison des performances à taille finie de l'algorithme EMS pour diverses complexités ($n_m = 6, 12, 18, 36$) et de l'algorithme MS pour un code binaire équivalent sur le canal QAM256-AWGN et le canal BI-AWGN. La taille des mots de code est $N_b = 1008$ bits pour le rendement R=0.5 (Fig.1) et $N_b \approx 1057$ bits pour R=0.77 (Fig.2). Les codes non binaires considérés sont des codes réguliers *"ultra-creux"* dans GF(256) (Fig.1) et GF(64) (Fig.2), dont tous les nœuds de variable ont un degré 2, ces derniers ayant été identifiés comme performants à ces tailles [2, 4].

L'écart entre les deux algorithmes est fortement dépendant de la valeur du paramètre n_m. Lorsque n_m se rapproche de la valeur de l'ordre du corps q, les performances du décodeur s'améliorent et dépassent celles du code binaire. Un des aspects les plus intéressants est qu'avec $n_m = 6$, l'algorithme EMS présente une complexité équivalente de l'algorithme MS binaire, avec une perte négligeable dans le cas de la figure Fig.2. Cependant, une complexité cinq fois plus grande est nécessaire ($n_m = 18$) pour atteindre les performances du décodeur binaire dans le cas d'une transmission sur canal QAM256-AWGN (Figure 2) ce qui reste tout à fait raisonnable, surtout en regard des complexités proposées dans la littérature pour les décodeurs non binaires [2, 3]. Dans les deux cas, une augmentation de la complexité de décodage permet d'atteindre de meilleures performances que les solutions code/décodeurs binaires, ce qui nous permet d'affirmer que les codes LDPC non binaires dans des corps d'ordres élevés sont désormais des concurrents potentiels sérieux des codes LDPC binaires lorsqu'ils sont décodés à l'aide de l'algorithme EMS.

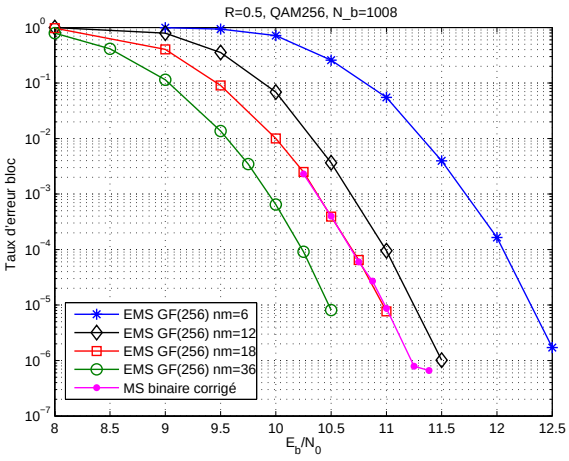

FIG. 1 – Comparaison EMS vs. Min-Sum binaire, canal QAM256-AWGN

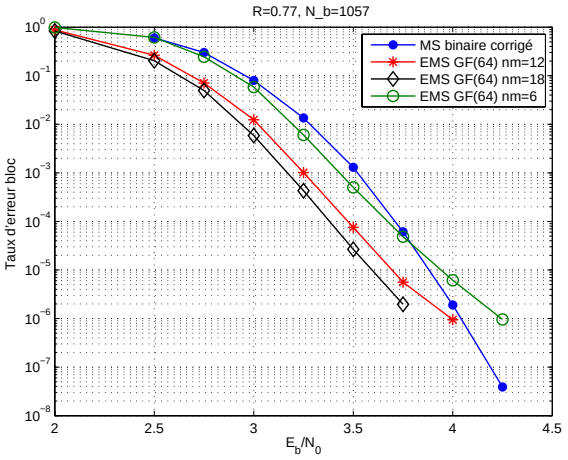

FIG. 2 – Comparaison EMS vs. Min-Sum binaire, canal BI-AWGN

6 Conclusion

Dans cet article, nous avons proposé un algorithme de décodage non-binaire à complexité réduite, l'algorithme EMS. Celui-ci est un bon candidat pour une éventuelle implantation matérielle d'un décodeur LDPC non-binaire en raison de sa faible complexité et d'une quantité de mémoire pour le stockage des mesages inférieure à celle des autres algorithmes existants. De plus la dégradation des performances constatée est négligable.

Références

[1] T.J. Richardson, M.A. Shokrollahi and R.L. Urbanke, "Design of Capacity-Approaching Low-Density Parity Check Codes," *IEEE Trans. Inform. Theory*, vol. 47, pp. 619-637, Feb. 2001.

[2] M. Davey and D.J.C. MacKay, "Low Density Parity Check Codes over GF(q)," *IEEE Commun. Lett.*, vol. 2, pp. 165-167, June 1998.

[3] H. Wymeersch, H. Steendam and M. Moeneclaey, "Log-Domain Decoding of LDPC Codes over GF(q)," *The Proc. IEEE Intern. Conf. on Commun.*, Paris, France, June 2004, pp. 772-776.

[4] D. Declercq and M. Fossorier, "Decoding Algorithms for Nonbinary LDPC Codes over GF(q)", *IEEE Trans. on Comm.*, vol. 55, pp. 633-643, Avril 2007.

[5] D.J.C. MacKay, "Online database of low-density parity check codes", http ://www.inference.phy.cam.ac.uk/mackay/codes/data.html.

[6] A. Voicila, D. Declercq, M. Fossorier et F. Verdier, "Algorithmes simplifiés pour le décodage de codes LDPC non-binaires", GRETSI, Louvain-la-Neuve, Belgique, September, 2005.