Architecture de décodeur de code produit haut débit

Patrick ADDE et Ramesh PYNDIAH

ENST Bretagne, Technopôle Brest Iroise, BP 832, 29285 BREST Cedex , France
E-mail : Patrick.Adde@Enst-Bretagne.fr, Ramesh.Pyndiah@Enst-Bretagne.fr

Résumé - Ce papier propose une nouvelle architecture d'un turbo décodeur, permettant de traiter un débit d'information élevé et utilisant comme code correcteur d'erreurs un code produit. Elle est indépendante du code élémentaire choisi (convolutif ou en bloc linéaire) et de l'algorithme de décodage utilisé. Elle prend en compte le parallélisme lié aux propriétés de la matrice générée par un code produit permettant l'emploi d'un seul plan mémoire simple-port. Une complexité m fois plus grande du décodeur permet l'obtention d'une vitesse de décodage m^2 fois plus élevée.

Abstract *– This paper presents a new circuit architecture for turbo decoding which achieves very high data rates when using product codes as error correcting codes. This architecture is independent of the selected elementary codes used (convolutional or bloc) and of the decoding algorithm used for elementary decoders. It exploits the possibilities of parallel decoding which results from the properties of product codes and it requires only a single port memory. For an increase in data rate by a factor of m square, the circuit complexity of the elementary decoders increases by a factor of only m while the required memory is constant.*

1. Introduction

La transmission de l'information (données, image, parole…) repose de plus en plus sur les techniques numériques de transmission. Beaucoup d'efforts ont été faits ces dernières années en matière de codage de source pour réduire le débit numérique, tout en conservant une bonne qualité. Ces techniques nécessitent une meilleure protection des éléments binaires vis à vis de perturbations liées à la transmission. L'utilisation de codes correcteurs d'erreurs puissants dans ces systèmes de transmission se révélait indispensable. Le début des années 90 a vu la naissance d'une nouvelle technique de code correcteurs : les turbo codes. Ce concept a été introduit en 1991 par C. Berrou [1]. Le schéma de codage proposé repose sur la concaténation parallèle de deux codes convolutifs séparés par un entrelacement non uniforme. Le décodage utilise un processus itératif basé sur des décodeurs à entrées et sorties pondérées. Ces principes, connus sous le nom de turbo code convolutif TCC, ont des performances inégalées à ce jour et proches de la limite de Shannon.

Une solution alternative aux TCC sont les turbo codes en blocs TCB proposés en 1994 par R. Pyndiah [2][3][4]. Le schéma de codage est basé sur la concaténation série de codes en blocs (code produit, introduit en 1954 par P. Elias [5]) et le décodage reprend le processus itératif obtenu à partir de décodeurs élémentaires à entrées et sorties pondérées. La concaténation série garantit une grande distance de Hamming minimale (9, 16, 24, 36 voire plus) et ce pour des blocs de données relativement petits. L'information extrinsèque qui joue un rôle important dans le turbo décodage est calculée non seulement pour les bits de données mais aussi pour les bits de redondance à chaque itération. Les TCB deviennent très attractifs lorsque le rendement de codage est élevé et si le taux d'erreurs visé est faible.

Ces différentes techniques de turbo décodage intègrent progressivement tous les systèmes de communications numériques. Ces derniers traitent des débits de plus en plus élevés, nécessaires vu la quantité d'informations traitées et l'utilisation de canaux de transmission rapides à base de fibres optiques (Gigabits, voire Térabits pour les futurs systèmes).

Après avoir rappelé le principe des codes produits, ce papier présente les architectures possibles de turbo décodeurs. Une organisation spécifique des plans mémoires est ensuite proposée, permettant d'utiliser m turbo décodeurs élémentaires en parallèle pour traiter un débit d'information m^2 fois plus élevé.

2. Les codes produits et leur décodage itératif

Un code produit permet d'obtenir à partir de deux codes en blocs simples (faible distance de Hamming minimale δ) un code dont la distance de Hamming minimale est égale au produit des distances de Hamming des codes élémentaires utilisés et le rendement au produit des rendements élémentaires.

Si on considère deux codes en blocs élémentaires $C_1(n_1,k_1,\delta_1)$ et $C_2(n_2,k_2,\delta_2)$, le code produit se présente sous forme de matrice **C** à n_1 lignes et n_2 colonnes où :

-les échantillons binaires d'informations sont représentés par une sous-matrice **M** à k_1 lignes et k_2 colonnes,

-chacune des k_1 lignes de la matrice **M** est codée par le code C_2,

-chacune des n_2 colonnes de la matrice **C** est codée par le code C_1.

Si le code C_1 est linéaire, les $(n_1–k_1)$ lignes construites par C_1 sont des mots du code de C_2 et peuvent donc être décodés comme les k_1 premières lignes. Dans ce cas, le code produit se caractérise par n_1 mots de code de C_2 suivant les lignes, et par n_2 mots de code de C_1 suivant les colonnes. Les codes C_1 et C_2 peuvent être obtenus à partir de codes élémentaires convolutifs ou de codes en blocs linéaires.

Le turbo décodage d'un tel code consiste à effectuer un décodage à entrées et sorties pondérées de toutes les lignes puis de toutes les colonnes de la matrice **C** (FIG. 1).

De manière générale, les informations échangées d'une demi-itération à une autre sont définies par le schéma de la FIG.2. R_k correspond à l'information reçue du canal, R'_k à l'information qui vient de la demi-itération antérieure et R'^+_k à l'information envoyée à la demi-itération suivante. La sortie de chaque demi-itération est donc égale à R_k plus

l'information extrinsèque, W_k, multipliée par *alpha*. Cette information extrinsèque correspond à l'apport du décodeur courant. Elle est obtenue par différence entre la sortie pondérée F_k et l'entrée pondérée de ce décodeur.

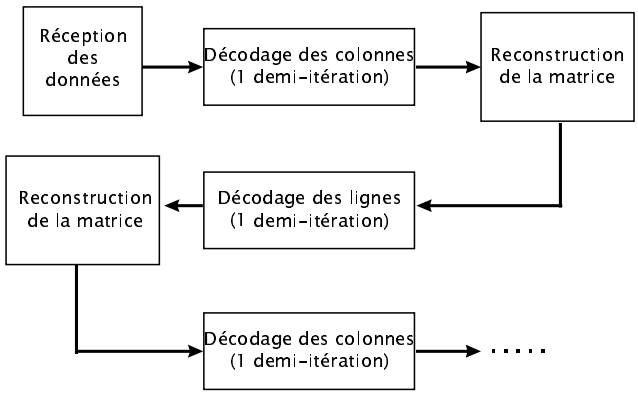

FIG. 1 : Processus itératif du turbo décodage.

On considère par la suite le décodeur à entrées et sorties pondérées comme un bloc ayant R_k et W_k (échantillons codés sur q bits) comme entrées, délivrant R_k^+ et W_k^+ (échantillons codés sur q bits) à la sortie avec une certaine latence L (retard nécessaire pour mettre en œuvre l'algorithme de décodage). Il prend le nom d'Unité de Traitement (UT).

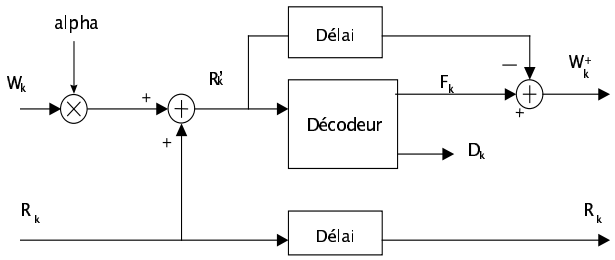

FIG. 2 : Schéma-bloc d'une demi-itération (UT).

En considérant une autre découpe de ce schéma bloc, W_k peut être remplacé par R'_k qui devient entrée-sortie du bloc : W_k est alors variable interne.

3. Architectures de turbo décodeur de code produit

L'analyse fonctionnelle de l'algorithme de turbo décodage a permis d'identifier deux architectures possibles [6][7] pour un circuit turbo décodeur de code produit (l'une modulaire et l'autre s'apparentant à une machine dite de Von Neumann. Ces deux structures sont maintenant décrites.

3.1 Architecture Possibles

3.1.1 Structure modulaire

A partir du schéma de fonctionnement de l'algorithme, on peut imaginer pour le turbo décodeur une structure modulaire dans laquelle chaque sous-circuit réalise une demi-itération de décodage (i.e. un décodage des lignes ou des colonnes d'une matrice de données, **[R]** ou **[R']**). Il faut mémoriser **[R]** et **[W]** (ou **[R']**, suivant le schéma bloc retenu).

3.1.2 Structure de Von Neumann

La seconde architecture s'apparente à une machine séquentielle de Von Neumann. Elle utilise une seule et même unité de traitement pour réaliser plusieurs itérations. Par rapport à la précédente, cette solution vise principalement à réduire l'encombrement du turbo décodeur. Elle présente en outre l'avantage de limiter à $2.n_1n_2$ échantillons au maximum la latence globale introduite par le circuit, indépendamment du nombre d'itérations effectué (n_1n_2 pour remplir une matrice et n_1n_2 supplémentaires pour le décodage).
Chaque échantillon est traité séquentiellement et doit être décodé en un temps ne dépassant pas l'inverse du produit du débit des données par le nombre de demi-itérations à effectuer. Ainsi, pour quatre itérations, le débit des données ne peut se faire qu'à un rythme au moins huit fois inférieur à celui de leur traitement. Ceci implique qu'entre les architectures modulaire et de Von Neumann, le débit maximal d'émission des données est divisé d'un facteur au moins égal au nombre de demi-itérations utilisé. La latence est moindre pour la structure de Von Neumann (2 n_1n_2 échantillons au maximum contre $(n_1n_2+L).it$ dans l'autre, it étant le nombre de demi-itérations) mais le débit est plus faible pour une même vitesse de traitement des données. Le nombre maximal d'itérations que l'on peut intégrer dans le circuit se trouve limité par le débit que l'on souhaite atteindre et par la fréquence maximale de fonctionnement qu'autorise la technologie utilisée pour l'UT.

3.2 Mémorisation des données et des résultats

L'encombrement du circuit provient essentiellement de la taille et du nombre des mémoires utilisées. Indépendamment de l'architecture générale retenue, il est en effet indispensable de mémoriser les matrices **[R]** et **[W]** (ou **[R']**) pour toute la durée de la demi-itération en cours (une demi-itération correspond à un décodage des lignes ou des colonnes d'une matrice de données). Le traitement des données en lignes puis en colonnes oblige à prévoir une première mémoire pour recevoir les données et une seconde pour les traiter. Ces deux mémoires travaillent alternativement en mode écriture et lecture, un automate gérant le séquencement. Chaque mémoire est organisée de manière matricielle et se compose, pour un code de longueur n_1n_2 et une quantification des données sur q bits, de q plans mémoires de n_1n_2 bits chacun.

3.2.1 Structure modulaire

Dans le cas de la structure modulaire, l'organisation générale du circuit pour une demi-itération est celle de la FIG. 3. Les données codées sur q bits qui parviennent au module de décodage sont rangées suivant les lignes d'une première mémoire A fonctionnant en mode écriture. Parallèlement, les données de la matrice reçue précédemment sont prélevées suivant les colonnes d'une seconde mémoire B fonctionnant elle en mode lecture. Une fois la mémoire A remplie et le contenu de la mémoire B lu, il y a inversion des modes de fonctionnement. La mémoire A passe en lecture - on commence alors à la décoder - tandis que la mémoire B passe en mode écriture afin de stocker les données correspondant au

mot de code suivant. En cascadant deux modules, l'un pour le décodage des colonnes et l'autre pour celui des lignes d'une matrice codée, on réalise une itération complète.

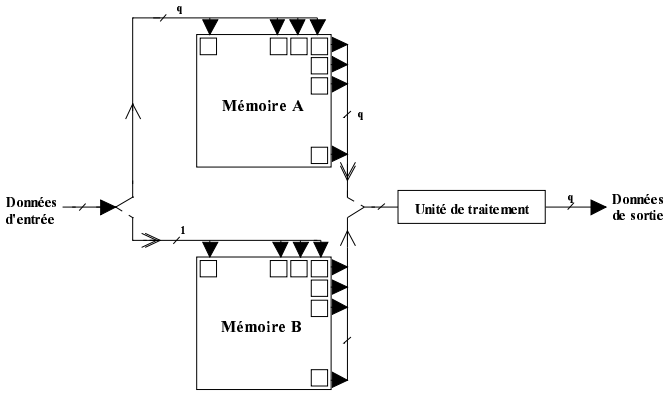

FIG. 3 : Schéma d'un module de décodage.

Les mémoires utilisées peuvent être conçues sans difficulté à partir de RAM (Random Access Memory) classiques, adressables en ligne et en colonne. D'autres solutions peuvent être envisagées (registres à décalage, par exemple) mais elles sont plus encombrantes.

D'un point de vue pratique, la solution modulaire a pour avantages de permettre une fréquence de fonctionnement élevée et d'être d'une grande souplesse d'utilisation. En contrepartie, la mise en cascade de plusieurs modules entraîne un accroissement de la latence et de l'encombrement du circuit. Ces paramètres deviennent rapidement rédhibitoires quand le nombre d'itérations et/ou la longueur du code augmente(nt).

3.2.2 Structure de Von Neumann

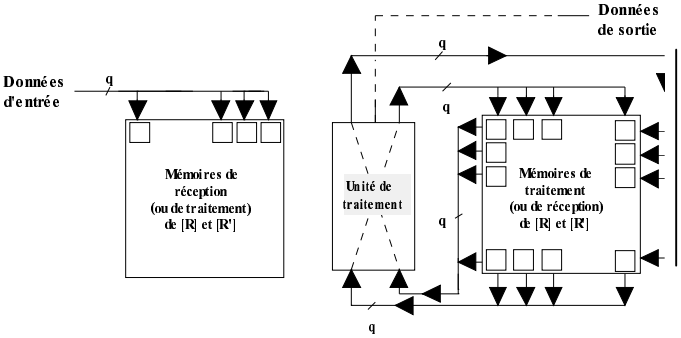

FIG. 4 : Structure de Von Neumann à traitement séquentiel.

Le circuit réalise cette fois plusieurs itérations en utilisant une seule unité de mémorisation et une seule unité de traitement pour l'ensemble des itérations. On vient reboucler sur lui-même un module de décodage. Avec cette architecture, le circuit complet ne comprend que six mémoires indépendamment du nombre d'itérations effectuées. Ces mémoires doivent cependant pouvoir être lues et écrites aussi bien en lignes qu'en colonnes. Les mémoires utilisées sont des RAM classiques et dans lesquelles on peut lire ou écrire une donnée repérée par son adresse. Comme on accède directement à chaque échantillon, il est possible de décoder la matrice indifféremment suivant ses lignes ou ses colonnes. Ces mémoires sont similaires à celles retenues pour la solution modulaire mais, le circuit complet n'en comportant que six (FIG. 4), le gain en surface est considérable (70% pour quatre itérations). Il faut toutefois remarquer que cette réduction de la surface est obtenue au détriment du débit des données (divisé par au moins *it* pour *it*/2 itérations : il faut en effet tenir compte dans ce calcul de la latence de chaque décodage élémentaire).

4. Architectures pour les hauts débits

Le chemin critique dans les architectures des turbo décodeurs se situe dans les unités de traitement. Pour une technologie cible donnée, on va considérer que les unités de traitements fonctionnent à une fréquence maximale F_{UTmax}, cette dernière étant donnée par la fréquence maximale de fonctionnement des décodeurs élémentaires. Si la fréquence débit $F_{Débit}$ est supérieure, il va falloir dupliquer les turbo décodeurs afin de traiter un plus grand nombre de données à la fois. Cela revient à paralléliser le flot de données (FIG. 5).

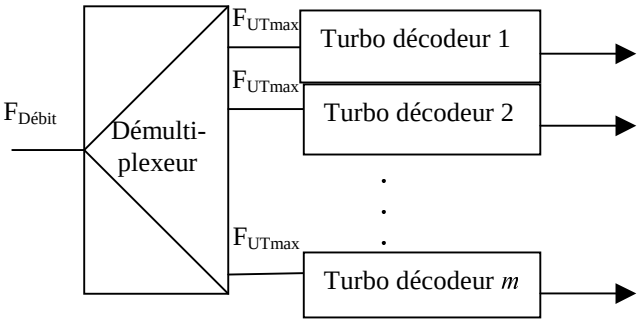

FIG. 5 : Architecture haut débit ($F_{Débit} = m.\ F_{UTmax}$)

On a vu que les codes produits avaient la propriété d'avoir des mots de code sur toutes les lignes (ou les colonnes) de la matrice C initiale. On peut donc, de la même façon que l'on vient de le voir, paralléliser le décodage des échantillons d'une matrice en dupliquant le nombre de décodeurs élémentaires du code C_1 (ou C_2).

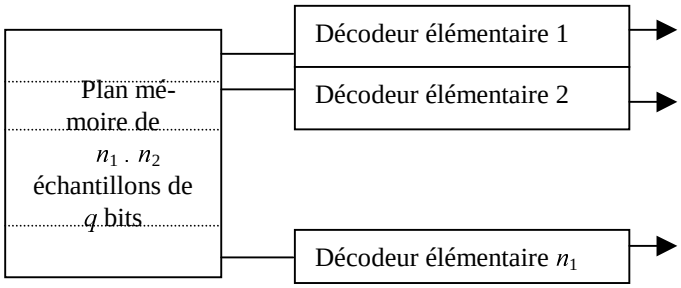

FIG.6 : Architecture haut débit décodant plusieurs lignes à un même instant.

On peut ainsi traiter un nombre maximal n_1 (ou n_2) mots de code à condition toutefois que les accès mémoire, en lecture ou en écriture, aient lieu à des instants différents (plusieurs points mémoire d'une matrice ne peuvent être lus ou écrits en même temps, à moins d'utiliser des RAM « multi-ports »). Cette contrainte étant respectée, il est possible de gagner un facteur n_2 (ou n_1) dans le rapport $F_{Débit}/F_{UTmax}$ puisqu'il peut y avoir n_2 (ou n_1) échantillons traités à un instant donné.

L'inconvénient majeur de cette architecture est que la mémoire doit fonctionner à une fréquence m. F_{UTmax}, si on a m décodeurs élémentaires en parallèle.
Si on veut augmenter le débit pour une même vitesse de fonctionnement de la mémoire, on peut mémoriser à une même adresse plusieurs données. Il faut, cependant, pouvoir utiliser ces données aussi bien en ligne ou en colonne. D'où l'organisation suivante : à cette adresse vont se trouver des données adjacentes en lecture (ou écriture), aussi bien en ligne ou en colonne.
Considérons deux lignes adjacentes et deux colonnes adjacentes de la matrice initiale.

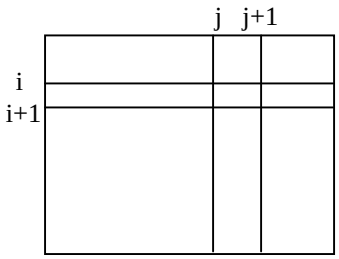

FIG. 7 : Obtention de la nouvelle matrice

Les 4 échantillons (i,j), (i,j+1), (i+1,j) et (i+1,j+1) constituent un mot de la nouvelle matrice qui a 4 fois moins d'adresses (I,J) mais des mots 4 fois plus grands. Si n_1 et n_2 sont pairs,
alors si $1 \leq I \leq n_1/2$, $i=2*I-1$.
De même, si $1 \leq J \leq n_2/2$, $j=2*J-1$
Pour le décodage ligne, les échantillons (i,j), (i,j+1) sont affectés à une unité de traitement UT1, (i+1,j) et (i+1,j+1) à une unité de traitement UT2. Pour le décodage colonne, il faut prendre (i,j), (i+1,j) pour UT1 et (i,j+1), (i+1,j+1) pour UT2. Si les unités de traitement savent traiter en entrée (lecture de la RAM) et en sortie (écriture de la RAM) ces couples d'échantillons dans le même temps $1/F_{UTmax}$, le temps de traitement de la matrice sera 4 fois plus rapide que pour la matrice initiale (FIG.8).

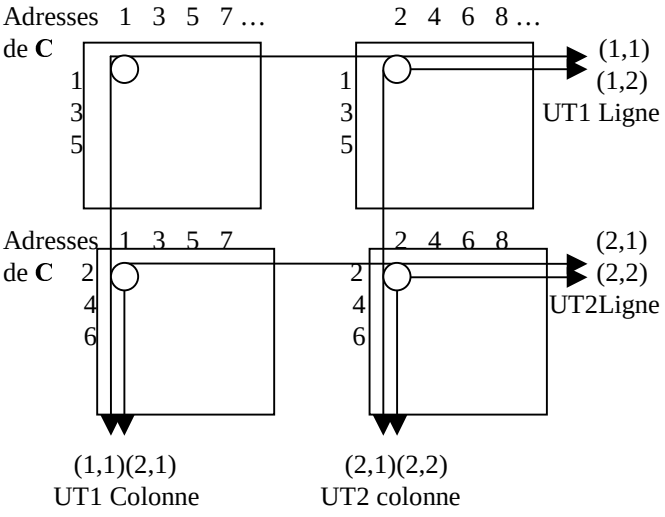

FIG. 8 : Exemple de « découpe » de la mémoire en 4 parties.

En généralisant, si un mot de la nouvelle matrice contient m échantillons d'une ligne et l échantillons d'une colonne, le temps de traitement de la matrice est $m.l$ fois plus rapide avec seulement m unités de traitement du décodage « ligne » et l unités de traitement du décodage « colonne ».
Dans le cas où les codes C_1 et C_2 sont identiques, les UT « ligne » et les UT « colonnes » le sont aussi : alors $m=l$ et m unités de traitement sont nécessaires. Cette organisation des matrices de données ne requiert pas d'architectures de mémoires particulières, ni une rapidité plus grande. Par ailleurs, si la complexité de l'UT reste inférieure à m fois celle de l'UT précédente, la complexité totale est moindre pour une vitesse m^2 fois plus élevée (ce dernier résultat aurait pu être obtenu en utilisant m^2 UT, comme indiqué FIG. 5).

5. Conclusions

Nous avons proposé une architecture de décodage des codes produits, fonctionnant à haut débit. Ils peuvent être obtenus à partir de codes convolutifs ou de codes en blocs linéaires. Elle consiste à modifier l'organisation initiale de la mémoire C afin d'accélérer le temps de décodage. Pendant un temps $1/F_{UTmax}$, m échantillons sont traités dans chacun des m décodeurs élémentaires, ce qui permet un gain m^2 en débit. Dans le cas où le traitement de ces m échantillons n'augmente pas de manière importante la surface du décodeur élémentaire, le gain en surface est proche de m, lorsqu'on compare cette solution à celle nécessitant m^2 décodeurs (FIG. 5). Le gros avantage de cette architecture est qu'elle n'utilise qu 'un plan mémoire en entrée (au lieu de m^2 FIG. 5). Par ailleurs, le nombre de mots de la mémoire étant moindre, le temps d'accès à cette mémoire est diminué.

Références

[1] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding : Turbo-codes (1), " *IEEE Int. Conf. on Comm.ICC' 93*, vol 2/3, May 1993, pp. 1064-1071.
[2] R. Pyndiah, A.Glavieux, A. Picart and S.Jacq, "Near optimum decoding of product codes, "in proc. of *IEEE GLOBECOM '94 Conference*, vol. 1/3, Nov.- Dec. 1994, San Francisco, pp. 339-343 .
[3] R. Pyndiah, "Near optimum decoding of product codes : Block Turbo Codes," *IEEE Trans. on Comm.*, vol 46, n° 8,August 1998, pp. 1003-1010.
[4] R. Pyndiah, "Iterative decoding of product codes : block turbo code," *Int. Symposium on turbo codes and related topics*, Brest, Sept. 1997, pp. 71-79.
[5] P. Elias, "Error-free coding," *IRE Trans. on Inf. Theory*, vol. IT-4, pp. 29-37, Sept. 1954.
[6] P. Adde, R. Pyndiah, O. Raoul, " Performance and complexity of block turbo decoder circuits " - Third International Conference on Electronics, Circuits and System ICECS'96, pp. 172-175, 13-16 October 1996 - Rodos, Greece.
[7] P. Adde, R. Pyndiah, O. Raoul and J.R. Inisan, "Block turbo decoder design," *Int. Symposium on turbo codes and related topics*, Brest, Sept. 1997, pp.166-169.