Variabilité de calculs et débordements de décodeurs séquentiels à pile

Computational variability and overflows of stack sequential decoders

David HACCOUN

Département de Génie Électrique
École Polytechnique de Montréal, Montréal, QUÉBEC, CANADA

David Haccoun est diplômé en Génie Physique de l'École Polytechnique de Montréal (1965), a obtenu une Maîtrise en Sciences (Génie Électrique) du Massachusetts Institute of Technology, Cambridge, U.S.A., en 1966, et un Doctorat es Sciences (Ph. D.) de l'Université McGill, Montréal, en 1974.

Depuis 1966, il est professeur au Département de Génie Électrique de l'École Polytechnique de Montréal, où il fut chef de la section Communication et Informatique (1980-1982). En 1984-1985, il fut Professeur de Recherche Invité à l'Université Concordia (Montréal). Ses recherches portent sur la théorie des communications, la théorie de l'information et du codage, les communications mobiles et les communications par satellite. Il est le coauteur avec MM. Bhargava, R. Mathias et P. Nuspl de l'ouvrage *Digital Communications by Satellite*, John Wiley, New York, 1981, (ouvrage traduit en Japonais et publié à Tokyo en 1984).

M. Haccoun agit comme consultant auprès d'industries et d'agences gouvernementales canadiennes, est membre de l'Ordre des Ingénieurs du Québec, de Sigma Xi et membre sénior de IEEE. Il fut membre du comité d'organisation et trésorier du Symposium International de théorie de l'information (Saint-Jovite, Canada, 1983), et est président fondateur de la Société Canadienne de Théorie de l'Information (Montréal 1986).

RÉSUMÉ

Cet article traite de codage convolutionnel et de décodage séquentiel par l'algorithme à pile de Zigangirov-Jelinek et de certaines de ses variantes. Ces variantes ont toutes pour objectif la diminution de la variabilité de l'effort de calcul du décodage séquentiel. Utilisant la simulation sur ordinateur, on montre que cette variabilité peut être grandement diminuée au prix d'un accroissement de l'effort de calcul moyen, mais aussi sans dégradation de la performance d'erreur.

Le remplissage de la pile du décodeur est examiné ainsi que son impact sur la taille de la file d'attente dans le tampon d'entrée du décodeur. Une analyse simple de la file d'attente a montré qu'en choisissant judicieusement le gain de vitesse du décodeur, la taille de la pile et la longueur des blocs, on peut limiter la taille du tampon d'entrée à deux longueurs de bloc sans risque de débordements. Enfin on montre qu'en controlant les débordements de la pile et en utilisant une procédure de retransmission des blocs ayant fait déborder la pile, on peut controler la probabilité de débordement du tampon d'entrée et réduire sensiblement la probabilité d'erreur. Ces avantages sont obtenus au prix d'une faible réduction du taux de codage effectif et d'une légère augmentation de la complexité du décodeur.

MOTS CLÉS

Codage convolutionnel, décodage séquentiel, Zigangirov-Jelinek, Pareto, pile, tampon, retransmissions ARQ.

Cette recherche a été supportée en partie par une subvention du Conseil de recherches en sciences naturelles et en génie du Canada.

SUMMARY

This paper presents the Zigangirov-Jelinek (Stack) algorithm of sequential decoding and some of its variants. The objective of all the variants is to reduce the computational variability of the decoding effort. Using computer simulation we show that this variability can be substantially reduced at a cost of a larger average decoding effort, but with no degradation of the error performance.

The required decoder stack sizes and their impact on the waiting line at the input buffer is examined. It is shown that by properly choosing the decoder speed factor, stack size and length of the data blocks, the length of the waiting line cannot exceed two data blocks. Finally by containing the overflows in the stack and by using a retransmission procedure on the overflowed blocks, the input buffer overflows can be controlled and the error probability reduced. These advantages are obtained at a cost of a small decrease of the system throughput and a slight increase of the decoder complexity.

KEY WORDS

Convolutional coding, sequential decoding, stack, input buffer, Zigangirov-Jelinek, Pareto, ARQ retransmission.

TABLE DES MATIÈRES

1. Introduction

L'usage de plus en plus répandu de techniques de transmissions numériques dans les télécommunications terrestres et par satellite conduit à l'utilisation grandissante de procédures de correction d'erreur par codage de canal qui sont puissantes, fiables et pratiques. Aussi un problème important consiste à développer des techniques de codage et décodage délivrant de faibles probabilités d'erreur avec des décodeurs de complexité acceptable. Dans les canaux de communication sans mémoire, les systèmes utilisant le codage convolutionnel avec décodage probabiliste sont parmi les plus intéressants tant du point de vue de leur performance d'erreur que du point de vue de leur réalisation et implantation matérielle. Le décodage probabiliste comprend un ensemble de techniques où le message décodé est obtenu par des procédures probabilistes plutôt que par des opérations algébriques fixes, et où les codes utilisés n'ont pas, en principe, à satisfaire à une structure algébrique particulière comme pour les codes en blocs. Ces codes peuvent être choisis au hasard sans nuire à la technique de décodage, ce qui permet d'augmenter considérablement leur champ d'application.

Les deux principales techniques de décodage probabiliste des codes convolutionnels sont le décodage séquentiel [1] et le décodage de Viterbi [2]. Chacune de ces techniques consiste à trouver un chemin particulier (le message émis) dans un graphe orienté (arbre ou treillis), où on assigne aux branches des valeurs de vraisemblance γ_j (appelées « métriques ») entre les symboles reçus du canal de transmission et les symboles codés qui auriaent pu être émis. L'objectif général du décodeur est donc de déterminer le chemin ayant la métrique totale $\Gamma = \sum_j \gamma_j$ la plus élevée, et ce, avec un minimum d'effort et un maximum de fiabilité. Ce chemin de métrique maximale trouvé par le décodeur est la séquence décodée $\underline{\hat{X}}$, de laquelle on déduit la séquence d'information $\underline{\hat{U}}$ qui est alors transférée à l'utilisateur. Le décodeur commet une erreur de séquence non détectée si $\underline{\hat{U}} \neq \underline{U}$, où $\underline{U}$ est la séquence d'information émise par la source.

Les techniques de décodage séquentiel et de décodage de Viterbi sont nettement différentes l'une de l'autre, ont des performances d'erreur et des domaines d'application différents qui les distinguent nettement l'une de l'autre en plus de les distinguer des techniques de codage en bloc [3].

La figure 1 montre les courbes de performance et les gains de codage de plusieurs systèmes de codage utilisant une modulation de type PSK cohérente parfaite [3]. Le *gain de codage* d'un système de codage est égal à la différence en décibels des valeurs de E_b/N_0 requises pour une probabilité d'erreur donnée entre ce système de codage et la modulation PSK cohérente parfaite sans codage. Ici, E_b est l'énergie

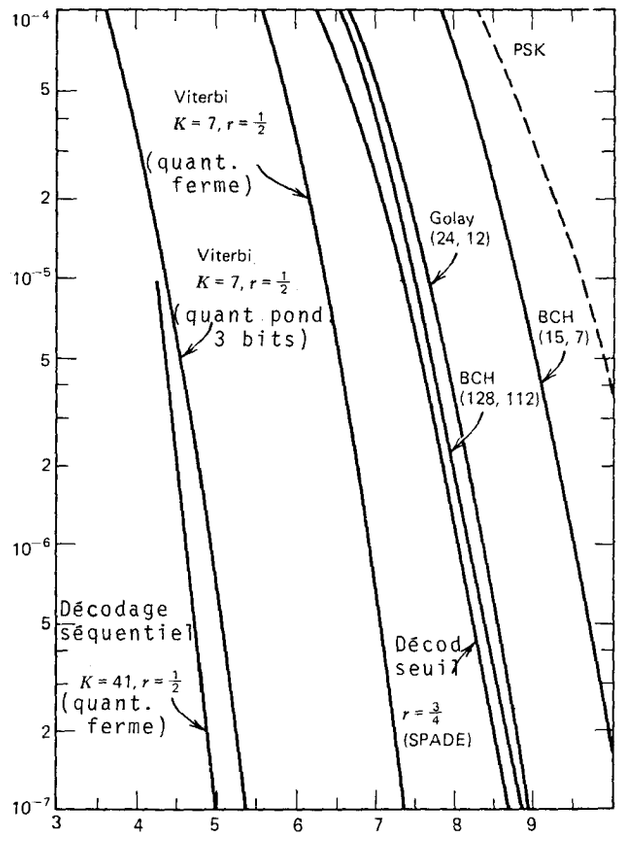

Fig. 1. – Courbes de performance de quelques systèmes codés.

recue par bit d'information, N_0 la densité spectrale du bruit, et le rapport signal-à-bruit par bit d'information E_b/N_0 sert de facteur de mérite pour comparer les performances de différents systèmes de modulation et de codage. Le problème de base des systèmes codés peut se résumer à déterminer le système qui fournira une performance d'erreur donnée avec la plus faible valeur de E_b/N_0. Se référant à la figure 1, par exemple, au taux d'erreur de 10^{-5}, le codage en bloc BCH (128,112) donne un gain de codage de 2 dB, alors que le décodage de Viterbi avec quantification pondérée ($K=7$, $R=1/2$) permet un gain de codage égal à 5,0 dB. On peut voir qu'un décodeur séquentiel avec quantification ferme permet un gain de codage égal à 5,2 dB, alors qu'en quantification pondérée à huit niveaux ou 3 bits, le gain de codage peut atteindre 7 à 8 dB. D'un point de vue pratique, un gain de 5,2 dB peut se traduire soit par une réduction de 5,2 dB de la puissance de transmission de l'émetteur, soit par une augmentation de la vitesse de transmission des données non codées par un facteur de $10^{0.52}=3,3$. Selon les applications, chacune de ces possibilités peut s'avérer particulièrement intéressante pour améliorer la conception d'un système, en particulier dans les liaisons numériques par satellite où chaque décibel d'énergie émise par le satellite est extrêmement coûteux.

Cet article traite essentiellement de codage convolutionnel et de décodage séquentiel en particulier de l'algorithme de Zigangirov-Jelinek (algorithme à pile) et de certaines de ses variantes. Après avoir brièvement rappelé la structure des codes convolutionnels et présenté l'algorithme à pile de base, quelques variantes du décodage séquentiel à pile qui permettent de diminuer la variabilité de l'effort de calcul sont présentées. Le comportement du remplissage de la pile est examiné ainsi que de son impact sur la dynamique de la file d'attente au tampon d'entrée. Enfin on présente une procédure d'utilisation de décodeurs séquentiels à pile où les blocs difficiles à décoder provoquent un débordement de la pile et sont retransmis. Cette procédure qui fait du décodeur séquentiel un décodeur hybride détecteur-correcteur d'erreur permet un échange de la mémoire et du gain de vitesse du décodeur. En particulier on montre qu'avec un choix judicieux de la taille de la pile et du gain de vitesse, un tampon de taille deux longueurs de bloc ne débordera jamais.

2. Codage convolutionnel et décodage séquentiel

Structure des codes convolutionnels

Un codeur convolutionnel de taux de codage $R=1/V$ peut être représenté par une machine linéaire à états finis composée d'un registre à décalage de K cellules, de V additionneurs modulo-2 connectés à certaines cellules du registre à décalage, et d'un commutateur qui balaye les V additionneurs modulo-2. L'ensemble des connexions entre le registre à décalage et les additionneurs modulo-2 spécifie le code. Par exemple, un codeur convolutionnel $K=3$, $R=1/2$ est montré sur la figure 2.

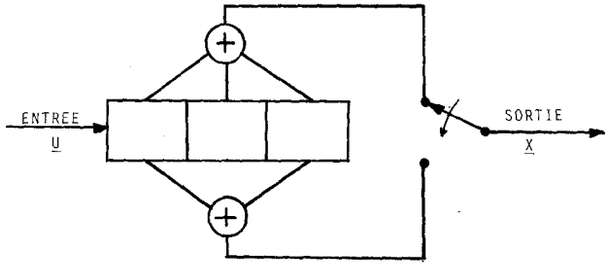

Fig. 2. – Codeur convolutionnel, $K=3$, $R=1/2$.

Un codeur convolutionnel fonctionne comme suit : les bits d'information sont introduits par la gauche, un bit à la fois, et après chaque décalage, les additionneurs modulo-2 sont échantillonnés en séquence par le commutateur, fournissant ainsi V symboles codés qui sont modulés et émis dans le canal. Le taux de codage est donc $R=1/V$. Pour ces codeurs binaires simples, la longueur K du registre à décalage s'appelle la *longueur de contrainte* du code. Un codeur peut

être facilement généralisé, et admettre non pas 1 mais n bits à la fois dans le codeur, avec $n < V$; le taux de codage devient alors $R = n/V$.

Arbre et treillis

Considérant seulement des codes convolutionnels de taux $R = 1/V$ et de longueur de contrainte K, à chaque bit d'information correspond deux branches d'un *arbre*, chacune portant V symboles codés. L'extrémité de chaque branche est un *nœud* caractérisé par un *état* du codeur. L'état du codeur est le contenu des $(K-1)$ premières cellules du registre à décalage, et donc le nombre d'états distincts est égal à $2^{(K-1)}$.

Un *chemin dans l'arbre* est spécifié par la séquence d'information qui est entrée dans le codeur et deux chemins reconvergent (*i. e.* ont le même état terminal) si leurs $(K-1)$ derniers bits d'information sont identiques [3]. Au-delà d'une profondeur égale à $(K-1)$ l'arbre d'encodage contient donc une énorme redondance qui peut être éliminée en ne gardant qu'un seul chemin au-delà de chaque nœud de reconvergence. L'arbre devient alors un *treillis* ayant 2^{K-1} états, et pour une séquence d'information de longueur L bits, les chemins dans l'arbre ou le treillis ont donc une longueur maximale égale à L branches. Un exemple d'arbre et de treillis correspondant au codeur de la figure 2 est donné sur les figures 3 et 4 respectivement.

Les notions de chemin, arbre et treillis sont essentielles à la compréhension du codage des codes convolutionnels. La séquence d'information étant représentée par un chemin (le chemin correct), la fonction de décodage consiste donc, connaissant la séquence reçue, à trouver le chemin dans l'arbre ou le treillis qui soit le

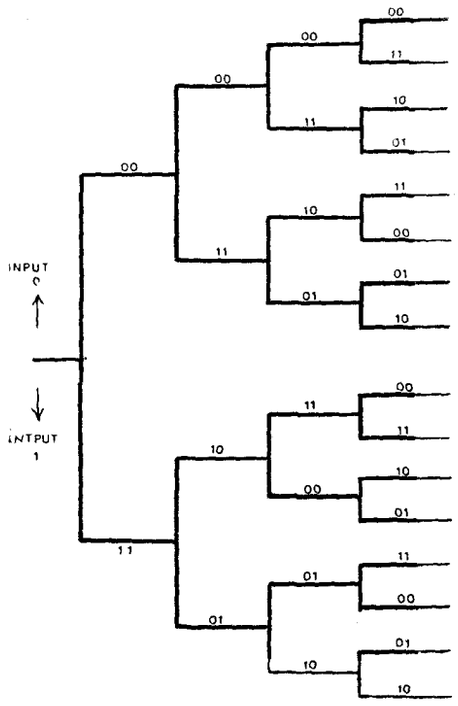

Fig. 3. – Arbre d'encodage du codeur de la figure 2.

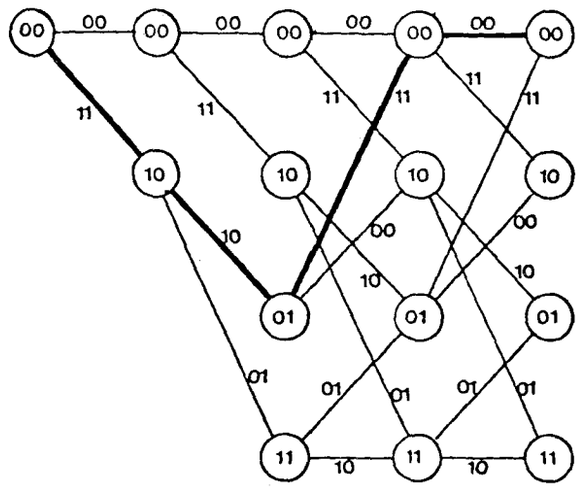

Fig. 4. – Treillis correspondant à l'arbre de la figure 3.

plus « vraisemblable », c'est-à-dire qui « ressemble » le plus à la séquence reçue. Le décodage séquentiel dont il est question dans le reste de cet article est une des techniques parmi les plus puissantes et les plus efficaces pour trouver ce chemin le plus vraisemblable.

3. Décodage séquentiel

Un décodeur séquentiel utilise la structure en arbre du code et n'explore, un chemin à la fois, que la partie de l'arbre qui paraît être la plus vraisemblable, sans cependant explorer l'arbre entier. C'est donc une procédure sous-optimale. Dans un canal sans mémoire, la fonction de vraisemblance utilisée, appelée aussi « métrique de symbole » est donnée par

$$\gamma_i = \log_2 \frac{P(y_i \mid x_i)}{P(y_i)} - R \tag{1}$$

où x_i est le symbole codé émis dans le canal, y_i est le symbole reçu correspondant, R est le taux de codage et $P(y_i \mid x_i)$ est la probabilité de transition du canal pour des symboles x_i et y_i. Par exemple pour un code de taux $R = 1/2$, un canal binaire symétrique de probabilité de transition p et des entrées équiprobables, la métrique (1) devient

$$\gamma_i = \begin{cases} \log_2(2p) - 1/2, & y_i \neq x_i \\ \log_2 2(1-p) - 1/2, & y_i = x_i. \end{cases} \tag{2}$$

Dans un canal sans mémoire la métrique est additive le long des symboles des branches d'un même chemin, de sorte que la métrique $\Gamma^{(U)}$ du nœud extrémité d'un chemin $\underline{U}$ de longueur m symboles est donnée par

$$\Gamma^{(U)} = \sum_{i=1}^{m} \gamma_i. \tag{3}$$

La métrique totale Γ tend à croître en moyenne le long du chemin correct, et tend à décroître, en moyenne le long de tous les chemins incorrects. Un exemple de cette métrique est donné sur la figure 5.

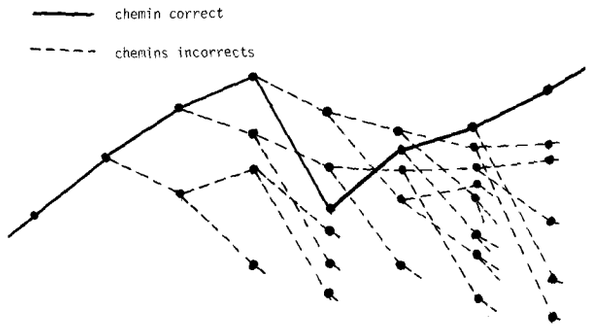

Fig. 5. – Exemple de métrique des chemins corrects et incorrects.

Sachant la séquence reçue du canal, l'objectif du décodeur séquentiel est d'explorer l'arbre d'encodage le long du chemin ayant la métrique la plus élevée parmi tous les chemins explorés. Cependant le bruit du canal provoquant occasionnellement des chutes locales de la métrique du chemin correct, le décodeur cesse alors de suivre le chemin correct pour explorer des chemins incorrects plus vraisemblables. Par conséquent, bien qu'en moyenne très faible, l'effort de décodage exprimé en nombre de calculs effectués par bit décodé est aussi très variable, avec une fonction de répartition du type de Pareto, c'est-à-dire :

$$(4) \qquad P(C \geqq N) \approx l N^{-\alpha}, \qquad N \gg 1$$

où l est une constante, et où le paramètre α, $\alpha \geqq 0$, appelé exposant de Pareto ne dépend que du taux de codage et du canal [3-6].

La variabilité de l'effort de calcul est l'inconvénient principal du décodage séquentiel et nécessite l'utilisation d'un tampon à l'entrée du décodeur pour y stocker les branches reçues du canal en attente d'être décodées. Le débordement de ce tampon constitue un événement d'erreur catastrophique entraînant un grand nombre de bits en erreur. Il est donc très important de réduire cette variabilité de l'effort de décodage et un certain nombre de procédures ont été élaborées à cette fin [7-8].

Algorithme à pile

Les deux principaux algorithmes de décodage séquentiel sont l'algorithme de Fano [4] et l'algorithme de Zigangirov-Jelinek (Z-J) [1, 3]. Le présent article ne traite que l'algorithme Z-J. Cet algorithme utilise une *pile* pour stocker toutes les caractéristiques des chemins explorés, et un *tampon* d'entrée pour stocker les séquences reçues en attente d'être décodées. Un schéma de principe du décodeur est montré sur la figure 6. La pile est une liste ordonnée où sont stockés les chemins explorés par ordre décroissant de leur métrique. Le *sommet* de la pile contient le chemin ayant la métrique maximale courante; ce chemin est

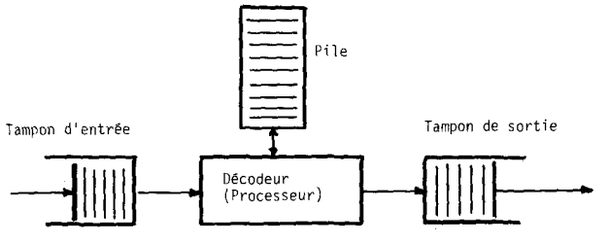

Fig. 6. – Schéma de principe d'un décodeur à pile.

donc celui qui sera prolongé. L'algorithme a donc pour objet de déterminer à chaque étape le sommet de la pile et d'en faire le prolongement jusqu'à atteindre un nœud terminal, c'est-à-dire un nœud de profondeur L, où L est la longueur de la séquence à décoder. Il se compose des trois étapes suivantes :

1. Calcul des métriques des deux chemins issus du sommet de la pile par prolongement d'une branche, et insertion dans la pile de ces deux chemins.
2. Élimination du sommet qui vient d'être prolongé.
3. Détermination du nouveau sommet. Si c'est un nœud terminal, stop. Sinon retour à 1.

Lorsque l'algorithme s'arrête, le sommet de la pile est le nœud terminal du chemin décodé, qui est alors facilement récupéré.

Bien que très simple, cet algorithme n'est pas pratique car le temps nécessaire à la mise en ordre *exacte* de la pile est beaucoup trop élevé. Cette difficulté est contournée en effectuant une mise en ordre *approximative :* les nœuds explorés sont insérés aléatoirement dans des sous-piles de la façon suivante : un nœud U de métrique $\Gamma^{(U)}$ est inséré au hasard dans la sous-pile Q si

$$(5) \qquad QH \leqq \Gamma^{(U)} < (Q+1)H$$

où H est une valeur arbitraire.

Avec cette modification, la recherche du sommet de la pile est réduite à la recherche de la sous-pile maximale non vide, et le chemin prolongé est choisi au hasard dans cette sous-pile, habituellement selon la procédure dernier-entré-premier-sorti (LIFO). Cette modification de l'algorithme facilite considérablement la procédure de recherche du nœud à prolonger et rend l'algorithme à pile applicable et pratique. La structure de données utilisée pour simuler l'algorithme sur ordinateur est fournie en annexe. Cette structure a servi de base à une réalisation matérielle d'un décodeur séquentiel à pile fonctionnant à environ 1 megabits/s [9].

Un décodeur séquentiel pratique tient compte du délai de décodage variable qui découle de la variabilité de l'effort de décodage en utilisant un *tampon d'entrée* et un *tampon de sortie* (voir *fig.* 6). Le tampon de sortie régularise le débit de sortie des séquences décodées alors que le tampon d'entrée sert à stocker les données provenant du canal et qui attendent d'être décodées. Aussi un problème important concerne le débordement de ces tampons. On peut montrer que, quelle que soit sa taille, il existe une probabilité non

nulle pour que le tampon d'entrée déborde entraînant une perte de données et une rupture de la liaison. L'analyse des débordements et des procédures de redémarrage du système sont parmi les problèmes importants de décodage séquentiel [10-11].

Afin de réduire les conséquences d'un débordement et d'une rupture de la communication, les données sont généralement organisées en « blocs » comportant quelque 500 à 2000 branches, chaque bloc se terminant par une séquence connue appelée *queue du message*. La queue de longueur égale à $(K-1)$ branches permet de remettre à zéro le registre à décalage du codeur local du décodeur, et de resynchroniser le système. En cas de débordement, le bloc en question est éliminé, le système est remis à zéro et le décodage peut se poursuivre pour les blocs suivants.

Effort de calcul du décodeur séquentiel

Quel qu'en soit l'algorithme, le décodage séquentiel implique toujours la possibilité pour le décodeur de revenir en arrière dans l'arbre et de changer une décision antérieure, c'est-à-dire de prendre une autre alternative que celle qui semblait être la meilleure. D'un point de vue théorique et analytique, chaque opération de prolongement d'un chemin est définie comme étant un *calcul*. Comme le nombre d'extensions effectuées est aléatoire, le nombre de calculs effectués par bloc décodé est donc également aléatoire. Aussi, contrairement aux algorithmes de décodage déterministe, l'analyse du décodage séquentiel concerne aussi bien la performance d'erreur que la distribution de l'effort de calcul. Cette variabilité de l'effort de calcul est l'un des principaux inconvénients du décodage séquentiel et le problème de sa diminution fait l'objet d'une grande activité de recherche [7-11].

Une analyse théorique relativement complexe [4-6] a montré que le nombre de calculs C effectué par bit décodé a une fonction de répartition qui suit asymptotiquement une loi de Pareto, donnée par (4) et répétée ci-dessous :

$$P(C \geqq N) \approx l N^{-\alpha}, \qquad N \gg 1 \tag{6}$$

où l et α dépendent du canal et du taux de codage R. *L'exposant de Pareto* α est un des paramètres clef pour évaluer la performance et la conception de décodeurs séquentiels.

La fonction de répartition (6) indique que l'effort de calcul suit une décroissance algébrique et non pas exponentielle avec N, représentant ainsi un autre inconvénient du décodeur séquentiel. Par conséquent, il devient impératif de s'assurer que le nombre *moyen* de calculs par bit soit fini, et également de faire face, en pratique, au retard qui découle de la variabilité de l'effort du décodage.

Comme on l'a mentionné plus haut, le retard de décodage se règle par l'utilisation de tampons d'entrée et de sortie. La réponse au problème de borner le nombre moyen de calculs réside dans l'analyse théorique de la variabilité de l'effort de calcul. Une analyse des moments de la cumulative [6] a montré que si l'exposant de Pareto α est inférieur à 2, la variance de l'effort de décodage diverge, et si $\alpha \leqq 1$, la *moyenne* de cet effort n'est plus bornée, c'est-à-dire que le nombre moyen de calculs devient théoriquement infini. Ceci se traduit en pratique par un décodage erratique, avec de très longues recherches arrière dans l'arbre et des débordements des tampons et de la pile. Le taux de codage qui correspond à la valeur limite $\alpha = 1$ est appelé *taux de coupure* (*Computational Cut Off Rate*) et est noté R_{comp}.

Ce taux R_{comp} ne dépend que du canal et se calcule facilement [3, 5, 12], mais d'un point de vue pratique, ce paramètre représente la limite extrême d'utilisation de décodeurs séquentiels. Aussi un important paramètre de design est le rapport R/R_{comp}, que l'on désire aussi près que possible de 1 mais sans l'atteindre. En pratique on choisit des points de fonctionnement où R/R_{comp} est compris entre 0,80 et 0,99, et on évalue approximativement l'exposant de Pareto par

$$\alpha \approx \frac{R_{comp}}{R}. \tag{7}$$

On peut noter en passant que la valeur de E_b/N_0 correspondant à R_{comp} peut être calculée pour différents modèles de canaux et de niveaux de quantification, avec en général une amélioration de 2 dB lorsque la quantification du canal passe de deux niveaux (1 bit) à huit niveaux (3 bits) [3, 12].

L'analyse théorique des moments de l'effort de calcul en général, et du nombre moyen de calculs $\bar{C}$ en particulier est réputée être difficile, et ne donne que des bornes asymptotiques et relativement peu serrées sur des ensembles de codes. D'un point de vue pratique, ces bornes sont donc très peu utiles pour des fins de conception impliquant un code particulier. On doit donc avoir recours à de longues simulations sur ordinateur pour obtenir des résultats utilisables. Cependant, utilisant une approche théorique totalement différente et basée sur les processus de ramification, une analyse récente de $\bar{C}$ a donné des résultats considérablement plus précis [13]. De plus, les résultats de cette analyse sont directement applicables à des cas particuliers car ils utilisent les paramètres du code ainsi que les valeurs particulières des métriques utilisées par le décodeur.

La nature Pareto de la cumulative de l'effort de calcul a été confirmée pour toutes sortes de canaux. Un raisonnement simple permet d'expliquer un tel comportement. Lorsque le bruit dans le canal devient assez fort pour provoquer une chute de la métrique du chemin correct, le décodeur entre dans une phase de recherche arrière et prolonge les nœuds des chemins incorrects en conformité avec l'algorithme. Le nombre de ces chemins incorrects croît de façon exponentielle avec la profondeur de la chute de métrique du chemin correct. Cependant, pour des canaux sans mémoire, tout intervalle de bruit qui provoque une chute de métrique apparaît avec une probabilité qui décroît exponentiellement avec la durée de cet intervalle. Le comportement de Pareto n'est rien d'autre que l'effet combiné de ces deux comportements exponentiels.

Problèmes de débordement

Il est clair qu'un débordement du tampon d'entrée aurait des conséquences catastrophiques. Il est donc impératif de prévoir une taille de tampon adéquate pour minimiser, voire éliminer un tel événement.

Cependant on montre que quelle que soit sa taille, il existe toujours une probabilité non nulle pour que le tampon d'entrée déborde [4-6]. Cette probabilité est assez grande (nettement plus grande que la probabilité d'erreur) et un débordement conduit à des effacements (en anglais « erasures »). Ces effacements ne sont pas des erreurs mais sont considérés plutôt comme étant des incertitudes sur la valeur des bits décodés.

Pour des séquences de longueurs L bits, l'expression approchée de la probabilité de débordement du tampon d'entrée est donnée par la relation :

$$(8) \qquad P(\text{débordement}) \approx L(GB)^{-\alpha}$$

où B est la taille du tampon, α est l'exposant de Pareto et G est le *gain de vitesse* du décodeur [3]. Ce gain de vitesse est le rapport entre la durée des bits provenant du canal et le temps d'un calcul par le décodeur. L'expression (8) indique encore une distribution de Pareto et une probabilité de débordement qui ne varie que lentement avec la taille du tampon, donc très difficile à combattre. Aussi en pratique, on choisit des tampons d'entrée très grands, et on sectionne les séquences d'information en blocs de longueurs variant de 500 à 2 000 ou 3 000 bits. De plus, on adopte toujours certaines procédures de récupération en cas de débordement. Ces procédures peuvent consister en une demande de retransmission des blocs qui sont sur le point de déborder [14-17], ou en un arrêt de décodage proprement dit et en une estimation de la séquence transmise. Ces procédures de récupération, qui peuvent varier selon les applications, sont une partie essentielle du décodage séquentiel et ne peuvent être ignorées dans une réalisation pratique.

4. Variantes de l'algorithme Z-J

Des recherches antérieures [7, 18] ont montré que la variabilité de l'effort de calcul d'un décodeur séquentiel peut être réduite en faisant non pas l'extension d'un seul chemin (le sommet de la pile), mais l'extension simultanée d'un certain nombre M de chemins les plus vraisemblables, c'est-à-dire ayant les métriques les plus élevées. Ce nombre M peut ne pas être fixe, et peut même s'adapter aux besoins courants imposés par le bruit dans le canal de transmission. Ces variantes du décodage séquentiel à extension multiple appartiennent à la classe d'algorithmes généralisés à pile qui unifient les techniques de décodage séquentiel et de Viterbi [7]. Dans toutes ces variantes, la réduction de la variabilité de calcul est toujours obtenue au prix d'un effort moyen plus grand, mais aussi n'entraîne aucune dégradation de la performance d'erreur. De plus, étant donné la structure de la pile, leur mise en œuvre est particulièrement facile, n'impliquant par rapport à l'algorithme de base qu'une complexité additionnelle minime. Les cas particuliers des variantes à plusieurs chemins de l'algorithme Z-J considérées ici sont brièvement décrits ci-dessous.

(i) *Algorithme à* M *chemins*

Il s'agit tout simplement de faire l'extension simultanée des M chemins occupant les plus hautes positions dans la pile. Cet algorithme peut être très facilement rendu adaptatif en imposant des restrictions sur M, ou en modifiant M en fonction du comportement général de la métrique du sommet de la pile [7].

(ii) *Algorithme à* S *sous-piles*

Cet algorithme fait l'extension simultanée de tous les chemins contenus dans les S sous-piles de métrique maximale courante [18]. Cet algorithme est auto-adaptatif car le nombre des chemins résidant dans les sous-piles varie en fonction du bruit dans le canal. Lorsque le canal est calme, les sous-piles maximales ne contiennent que quelques chemins et sont souvent vides, alors que pendant des périodes de bruit intense les métriques ayant une tendance à avoir des valeurs très rapprochées les unes des autres, les populations de chemins dans les mêmes sous-piles augmentent considérablement. Là encore S peut ne pas être maintenu constant tout au long du décodage, et pratiquement on impose toujours un nombre maximal, M_{max}, de chemins à prolonger simultanément. La

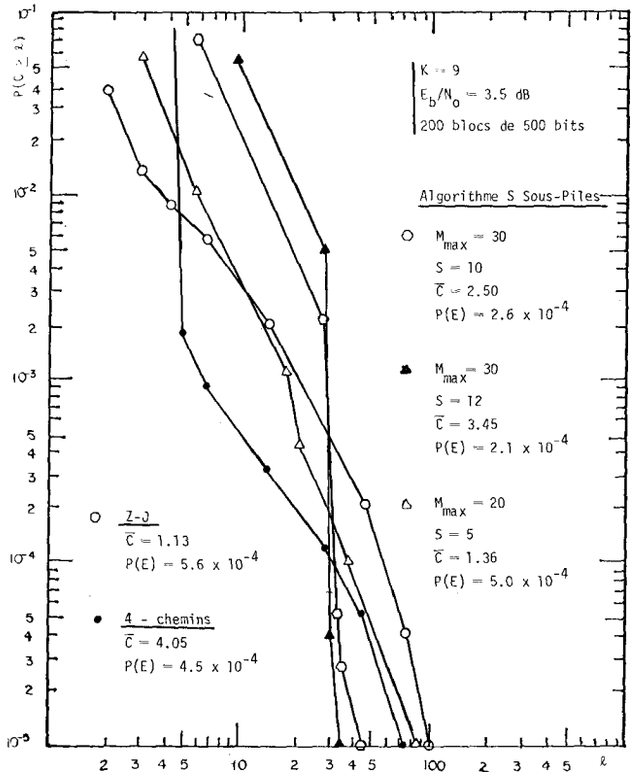

Fig. 7. — Fonctions de répartition du nombre de calculs par bit décodé des algorithmes Z-J, à quatre chemins et à S sous-piles pour un code K = 9, R = 1/2, à E_b/N_0 = 3,5 dB. S = Nombre de sous-piles; M_{max} = Nombre maximal de chemins prolongés simultanément; P(E) = Probabilité d'erreur par bit; $\bar{C}$ = Nombre moyen de calculs par bit décodé.

figure 7 donne les distributions obtenues par simulation à l'ordinateur de l'effort de calcul des algorithmes Z-J, de l'algorithme à quatre chemins et de l'algorithme à S sous-piles dans lequel S est égal à 5, 10 et 12 avec des valeurs maximales de chemins à prolonger, M_{max}, égales à 30, 30 et 20 respectivement [7, 18]. On peut voir que l'algorithme à quatre chemins améliore de façon substantielle la distribution de l'algorithme Z-J de base, et que la variabilité de l'effort de calcul est encore réduite par l'algorithme à S sous-piles. A mesure que S augmente la cumulative se rapproche d'une fonction échelon, et en particulier, la figure 7 montre que pour S = 12 le nombre maximal de calculs pour décoder un bit se chiffre à 34, et que cet événement est très rare, n'étant apparu qu'avec une probabilité de 1.10^{-5}. Quant à l'effort moyen de calculs il augmente proportionnellement avec S, ne valant que 3,45 pour S = 12, alors qu'il est égal à 1,13 et 4,05 pour les algorithmes Z-J à quatre chemins respectivement. Enfin on remarque que l'augmentation du nombre d'extensions simultanées améliore aussi la performance d'erreur, ce qui bien sûr était prévisible.

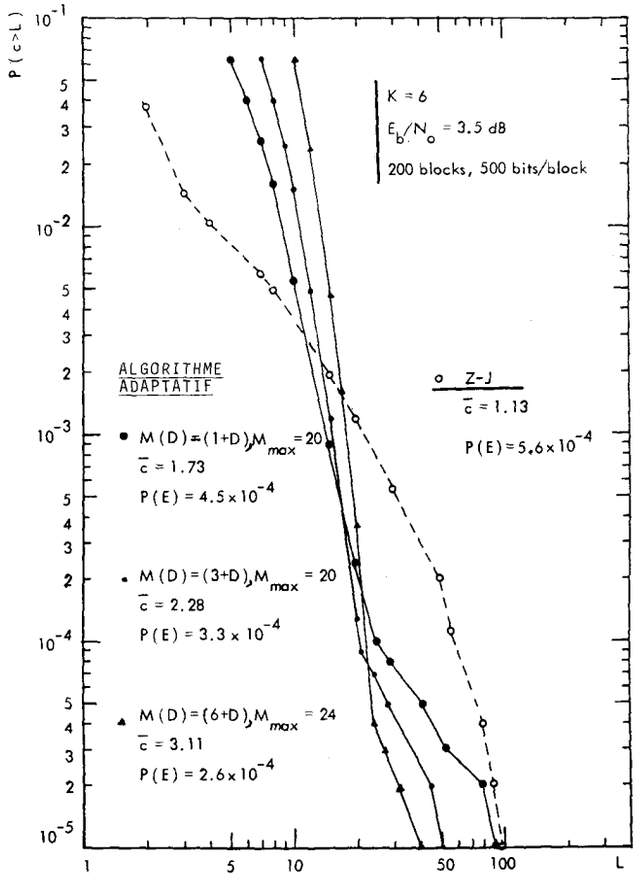

Fig. 8. — Fonctions de répartition du nombre de calculs par bit décodé des algorithmes Z-J et Adaptatif pour un code K = 6, R = 1/2, à E_b/N_0 = 3,5 dB. M(D) = Nombre de chemins à prolonger simultanément; M_{max} = Nombre maximal de chemins prolongés simultanément; $\bar{C}$ = Nombre moyen de calculs par bit décodé; P(E) = Probabilité d'erreur par bit.

(iii) *Algorithme adaptatif*

Cet algorithme tend à adapter le nombre de chemins prolongés simultanément avec le comportement de la métrique maximale, c'est-à-dire du bruit courant dans le canal. Pour cela on observe la croissance de cette métrique maximale. Si elle croît, un seul chemin est prolongé. Si elle chute d'une valeur D, on prolonge simultanément un nombre M(D), où M(D) est une fonction non décroissante de D [7, 18]. La figure 8 donne les fonctions de répartition obtenues par simulation à l'ordinateur du nombre de calculs par bit pour quelques fonctions linéaires de M(D). Là encore on peut voir l'effet de l'augmentation du nombre d'extensions simultanées sur la fonction de répartition de l'effort de calcul et sur la probabilité d'erreur.

(iv) *Algorithme prédicteur*

Il s'agit ici de déterminer à l'avance la chute D de la métrique afin de mieux faire varier S ou M(D) des algorithmes (ii) ou (iii) [18]. Pour cela la chute maximale de la métrique est déterminée en pénétrant l'arbre le long du chemin le plus vraisemblable sur une fenêtre d'observation de P branches. Utilisant les variations de la métrique dans cette fenêtre, l'algorithme prolonge alors un nombre de chemins qui tient compte non seulement du comportement passé de la métrique mais aussi de son comportement futur. La figure 9 donne les résultats de simulations obtenus

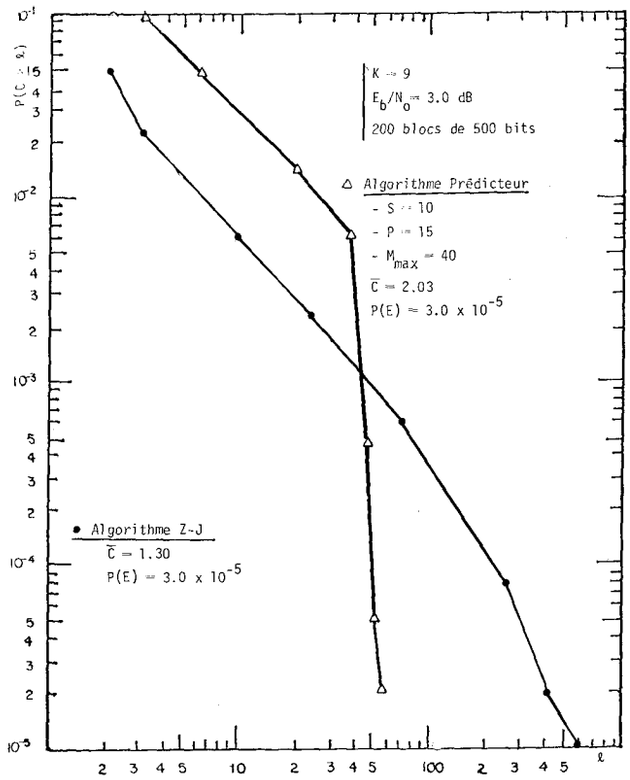

Fig. 9. — Fonctions de répartition du nombre de calculs par bit décodé des algorithmes Z-J et Prédicteur pour un code K = 9, R = 1/2, à E_b/N_0 = 3.0 dB. S = Nombre maximal de sous-piles; P = Profondeur de la fenêtre de prédiction; M_{max} = Nombre maximal de chemins prolongés simultanément; $\bar{C}$ = Nombre moyen de calculs par bit; P(E) = Probabilité d'erreur par bit.

avec cet algorithme utilisé en conjonction avec l'algorithme à S sous-piles [18]. La distribution de l'effort de calculs est encore améliorée par rapport à l'algorithme Z-J, mais par rapport à l'algorithme S sous-piles seul, le principal avantage de l'algorithme prédicteur réside dans la diminution du nombre moyen de calculs. Cependant, la mise en œuvre de la fonction de prédiction de l'algorithme est quelque peu plus complexe [18].

Comme on l'a vu, dans toutes ces variantes la réduction de l'effort est obtenue au prix d'un effort moyen plus élevé. Aussi faut-il encore évaluer l'impact de ces algorithmes sur les tailles et sur les comportements de la pile et de la file d'attente à l'entrée du décodeur. Ces problèmes sont examinés ci-après.

5. Impact des différentes variantes sur les tailles de la pile et du tampon d'entrée

Dans le but d'examiner les espaces mémoire requis par les différents algorithmes, l'algorithme Z-J et ses variantes décrites ci-dessus ont été simulés sur ordinateur, et les informations pertinentes sur le remplissage de la pile et sur la file d'attente à l'entrée du décodeur ont été recueillies. Ces simulations ont été effectuées en utilisant un code de Johannesson [19] de taux $R = 1/2$ et de longueur de contrainte $K = 24$. Dans ces simulations les données sont encore transmises sous forme de blocs de longueur $L = 500$ bits, auxquels on ajoute une queue de 23 bits à des fins de synchronisation. Les symboles transmis sont reçus en présence de bruit blanc gaussien, en quantification ferme avec des rapports signal à bruit E_b/N_0 égaux à 4,64 et 5,60 dB, correspondant à des rapports R/R_{comp} égaux à 0,99 et 0,85 respectivement.

Remplissage de la pile

Le nombre de nœuds stockés dans la pile, le nombre de débordements de la pile et le nombre d'erreurs sont observés à la fin du décodage de chaque bloc. La fonction de répartition du nombre d'entrées dans la pile pour l'algorithme Z-J est montrée sur la figure 10, et indique une distribution type de Pareto. Ce même type de distribution a été observé pour tous les algorithmes étudiés et est la conséquence directe de la distribution Pareto de l'effort de calcul du décodeur séquentiel. En effet comme chaque extension, simple ou multiple implique, au moins une entrée dans la pile, il est clair que la distribution du nombre d'entrées devra suivre celle du nombre de calculs compte tenu

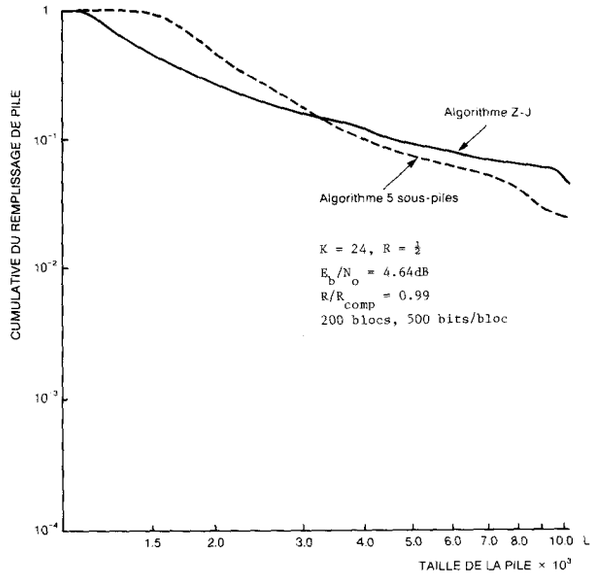

Fig. 10. — Fonctions de répartition du remplissage de la pile pour les algorithmes Z-J et à 5 sous-piles.

TABLEAU I

Remplissage de la pile

$R/R_{comp} = 0{,}85$, $E_b/N_0 = 5{,}60$ dB, $K = 24$, $R = 1/2$.

500 blocs simulés, 523 bits/bloc.

Paramètre \ Algorithme	Z-J	2-Chemins	N Sous-piles		Adaptatif M (D) = 1 + (1/2) D
			N = 5	N = 7	
Erreurs	0	0	0	0	0
Débordements pile, $S = 10^4$	0	0	0	0	0
Exp. Pareto α (pile)	2,8	3,5	3,5	4,2	3,5
Calculs moyens $\bar{C}$	1,10	1,82	1,24	1,58	1,40
Pile moyenne $\bar{S}$	1148	1905	1288	1660	1480
Pile minimale $S_{min}^{(1)}$ (*)	1260	2000	1480	1950	1620
Pile minimale $S_{min}^{(2)}$ (*)	2500	3160	2570	2950	2750
Pile minimale $S_{min}^{(3)}$ (*)	5600	6000	4900	5250	5250

(*) Pile minimale $S_{min}^{(k)} = S_{min}$: $\Pr(S \geq S_{min}) = 10^{-k}$, $k = 1, 2, 3$.

du nombre d'extensions simultanées effectuées par l'algorithme particulier. Ainsi, il a été observé que pour une même valeur de E_b/N_0, plus le nombre d'extensions simultanées, et donc le nombre moyen de calculs $\bar{C}$, est élevé, plus l'exposant α de Pareto (c'est-à-dire la pente de la courbe) est élevé. Les résultats des simulations concernant la pile sont résumés au tableau I pour R/R_{comp} valant 0,85. On y voit en particulier la corrélation très nette entre l'augmentation du nombre moyen de calculs $\bar{C}$, l'exposant de Pareto de la pile et la taille moyenne de la pile. Le tableau I donne aussi pour chaque algorithme la taille minimale de la pile pour une probabilité de débordement égale à 1.10^{-1}, 1.10^{-2} et 1.10^{-3}. Les résultats du tableau I indiquent que dans des conditions de bruit faible, les algorithmes à plusieurs chemins ne présentent pas d'avantage marqué par rapport à l'algorithme Z-J à un seul chemin. Cependant, dans des conditions de canaux très bruités les simulations ont indiqué que pour atteindre des probabilités de débordement faibles ($\leqq 10^{-3}$), l'algorithme Z-J requiert considérablement plus de mémoire pour sa pile que les variantes à plusieurs chemins. Naturellement cette situation n'est que la conséquence d'un exposant de Pareto plus faible pour l'algorithme Z-J.

A la lumière de ces résultats et de ceux obtenus avec des canaux très bruités correspondant à des valeurs de R/R_{comp} égales à 0,95 et 0,99, on peut tirer les conclusions suivantes [17] : pour des probabilités de débordement de pile de l'ordre de 10^{-2} à 10^{-3}, et un canal faiblement bruité ayant une valeur R/R_{comp} de l'ordre de 0,85, les algorithmes à plusieurs chemins multiples ne sont pas très avantageux, et un choix judicieux serait l'algorithme Z-J. Cependant comme le montre la figure 11, pour des canaux fortement bruités avec R/R_{comp} très près de 1, les algorithmes à chemins multiples sont préférables à l'algorithme Z-J. On choisira probablement l'algorithme fournissant l'exposant de Pareto le plus élevé sous les contraintes pratiques imposées par l'application particulière.

Si d'autre part on ne dispose que d'une pile de petite taille (environ 5 000 entrées), alors il est préférable de choisir un algorithme dont le nombre de calculs est relativement faible (par exemple l'algorithme Z-J) bien que ce choix puisse entraîner une variabilité de calcul appréciable. Par contre si la pile est de grande taille, par exemple supérieure à 10 000 ou 20 000 nœuds, les algorithmes à chemins multiples sont alors plus indiqués que l'algorithme Z-J.

File d'attente dans le tampon d'entrée

Quel que soit l'algorithme de décodage séquentiel la variabilité de l'effort de calcul nécessite l'utilisation d'un tampon à l'entrée du décodeur afin d'y stocker les branches reçues du canal pendant les recherches arrière du décodeur. Un comportement typique de la file d'attente dans ce tampon est montré sur la figure 12. Lorsque le décodeur entre dans une phase de recherche arrière d'un meilleur chemin, les symboles codés reçus du canal s'accumulent dans le tampon d'entrée et forment une file d'attente. Dès que le décodeur trouve le bon chemin sa progression en avant doit être rapide afin de rattraper en quelque sorte le « temps perdu »; un décodeur séquentiel doit donc bénéficier d'un gain de vitesse par rapport au taux de réception des branches du canal. Ce gain de vitesse G qui est le nombre de calculs que le décodeur peut effectuer durant le temps de réception d'une nouvelle branche du canal, doit nécessairement être supérieur à l'effort de calcul moyen $\bar{C}$. Dans le cas contraire, en moyenne la file d'attente croîtra sans

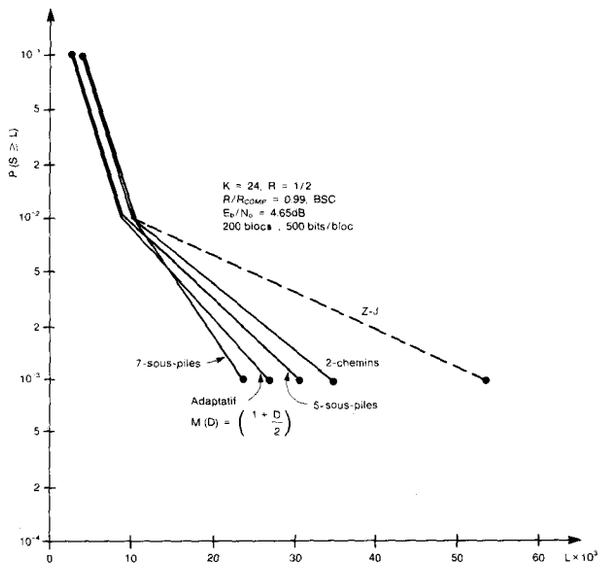

Fig. 11. — Fonctions de répartition empiriques du remplissage de la pile de plusieurs algorithmes pour des canaux fortement bruités.

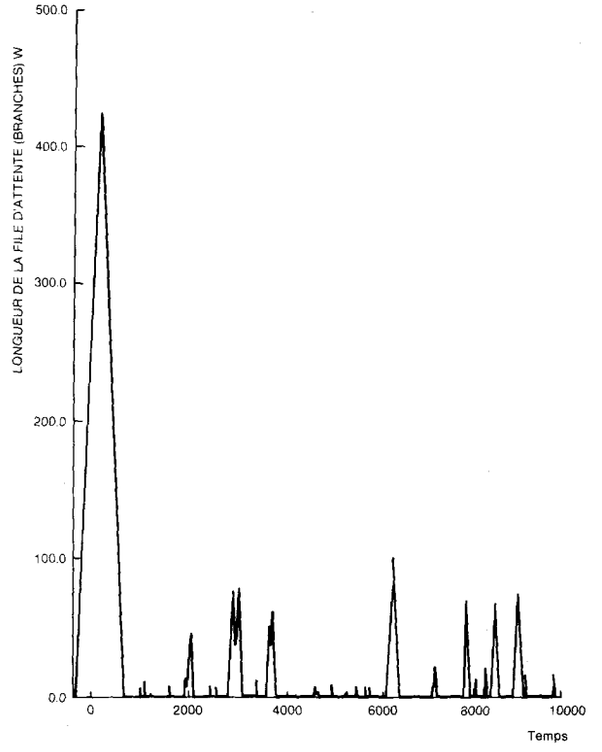

Fig. 12. — Dynamique de la file d'attente dans le tampon d'entrée. (Le temps est en unités de durée des branches.)

cesse et conduira avec certitude à un débordement du tampon d'entrée. Par exemple sur la figure 11, en fonction du temps la file d'attente apparaît sous la forme de pics disjoints, indiquant un gain de vitesse adéquat.

Dans les décodeurs séquentiels à pile la dynamique de la file d'attente dépend de la taille de la pile. En effet une pile de très grande taille permet des recherches arrière importantes sans débordements de la pile, ce qui peut entraîner un accroîssement considérable de la file d'attente dans le tampon d'entrée, voire même un débordement de ce tampon. De plus, la taille de la file d'attente dépend aussi du gain de vitesse du décodeur. Par conséquent le comportement de la file d'attente dépend de la taille de la pile et du gain de vitesse. Un raisonnement simple permet de voir que sous certaines conditions un tampon d'entrée de taille maximale égale à deux longueurs de bloc ne débordera jamais [17].

En effet, soit G le gain de vitesse du décodeur, S la taille de la pile et L la longueur des blocs en nombre de branches. Supposant un tampon d'entrée initialement vide, si le décodage du premier bit d'un bloc fait déborder la pile, alors le nombre de branches qui doivent être stockées dans le tampon d'entrée pendant le temps nécessaire à ce débordement est égal à S/2 G. La pile ayant débordé, le décodage du bloc est interrompu; l'algorithme vide sa pile et élimine du tampon toutes les branches de ce bloc avant de commencer le décodage du bloc suivant. Il est clair que la file d'attente au tampon d'entrée dépendra du nombre de branches qui doivent être stockées dans ce tampon pendant le temps requis pour un débordement de la pile.

Soit

$$\beta = \frac{S}{2GL} \tag{8}$$

le paramètre de remplissage du tampon d'entrée. On peut alors montrer [17] que si $\beta \geqq 1$ la file d'attente dans le tampon d'entrée peut augmenter indéfiniment avec des débordements successifs de la pile, alors que si $\beta \leqq 1$ cette file d'attente ne peut avoir une longueur supérieure à 2 L branches. Naturellement sous cette condition la distribution de la file d'attente ne sera pas asymptotiquement de type de Pareto.

Pour la réalisation matérielle de décodeurs séquentiels, un tampon d'entrée ne pouvant contenir que deux blocs est une solution fort intéressante. L'analyse ci-dessus montre qu'on peut échanger taille de la pile et gain de vitesse pour atteindre un compromis adéquat pour une probabilité de débordement donnée. Un compromis judicieux entre la taille de la pile, la longueur des blocs et le gain de vitesse doit donc être effectué dans une réalisation matérielle.

Des résultats de simulations avec l'algorithme Z-J et ses variantes sont donnés sur le tableau II pour des piles de taille 10 000 nœuds, des blocs de longueur 523 bits, $R/R_{comp} = 0{,}85$ et des valeurs de β variant de 4,78 à 1,91. On voit que pour des gains de vitesse faibles, (G = 2) et donc un β élevé ($\beta = 4{,}78$) la taille moyenne de la file d'attente augmente avec le nombre d'extensions simultanées de l'algorithme (ou encore avec le nombre moyen de calculs par bit). Cependant les tailles de tampon correspondant à des probabilités de débordement de 10^{-2} et 10^{-3} sont sensiblement les mêmes pour tous les algorithmes, indiquant encore une réduction de la variabilité de l'effort de calcul pour un effort moyen plus grand.

Comme on pouvait le prévoir, les valeurs moyennes de la file d'attente et les tailles du tampon correspondant à des probabilités de débordement de 1.10^{-1} à 1.10^{-3} diminuent toutes avec une augmentation du gain de vitesse. En particulier pour G = 5 et $\beta = 1{,}91$, les valeurs moyennes sont réduitent à deux branches environ et les faibles valeurs de la taille du tampon d'entrée correspondant à une probabilité de débordement de 10^{-3} peuvent laisser croire qu'effectivement la file d'attente ne divergera que très lentement. D'ailleurs au cours de simulations extensives, portant sur

TABLEAU II

File d'attente au tampon d'entrée

Code K = 24, R = 1/2.

$R/R_{com} = 0{,}85$, $E_b/N_0 = 5{,}60$ dB.

L = 523 bits/bloc, Pile S = 10 000, 200 blocs simulés.

β	Algorithme / Paramètre	Z-J	2-Chemins	S sous-piles		Adaptatif M(D) = 1 + (1/2) D
				S = 5	S = 7	
4,78	File moyenne $\bar{W}$	18,2	82	22	45	24
	Tampon min. $B_{min}^{(1)}$ (*)	1	250	12	80	16
	Tampon min. $B_{min}^{(2)}$ (*)	500	1 260	630	1 000	631
	Tampon min. $B_{min}^{(3)}$ (*)	1 580	2 240	1 585	1 780	–
3,19	File moyenne $\bar{W}$	6,1	9	6,8	9	6,5
	Tampon min. $B_{min}^{(1)}$ (*)	1	1	1	6	1
	Tampon min. $B_{min}^{(2)}$ (*)	160	250	160	200	160
	Tampon min. $B_{min}^{(3)}$ (*)	1 000	1 000	1 000	1 000	890
1,91	File moyenne $\bar{W}$	2,1	2,3	2,2	2,6	1,9
	Tampon min. $B_{min}^{(1)}$ (*)	1	1	1	1	1
	Tampon min. $B_{min}^{(2)}$ (*)	12	20	16	25	13
	Tampon min. $B_{min}^{(3)}$ (*)	250	500	445	500	400

(*) Tampon min. $B_{min}^{(k)} = B$: $Pr(W \geqq B) = 10^{-k}$, $k = 1, 2, 3$.

des milliers de blocs, des longueurs de file d'attente dépassant 1 000 branches n'ont jamais été atteintes.

La figure 13 donne les fonctions de répartition des longueurs des files d'attente au tampon d'entrée pour une pile de taille $S = 5\,000$ et pour des gains de vitesse G variant de 2,0 à 15, correspondant à des valeurs de β variant de 2,44 à 0,326. Tel que prévu, on observe que toute augmentation du gain de vitesse ou diminution du paramètre de remplissage du tampon β se traduit par une amélioration du comportement de la file d'attente. De plus, conformément à ce qui a été établi, on peut voir que pour toutes les valeurs de β inférieures à 1 la longueur de la file d'attente demeure inférieure à sa valeur maximale égale à deux longueurs de bloc, soit $W_{max} = 2 \times 523 = 1\,046$ branches. Cependant, cette longueur maximale a été largement dépassée dans tous les cas où β est supérieur à 1. De plus, la figure 13 et toutes les autres simulations effectuées avec d'autres valeurs de S et de E_b/N_0 montrent que la probabilité d'atteindre une longueur de file d'attente $W_{max} = 2L$ est d'autant plus grande que la valeur du paramètre de remplissage du tampon β est élevée, avec bien sûr $\beta \geqq 1$.

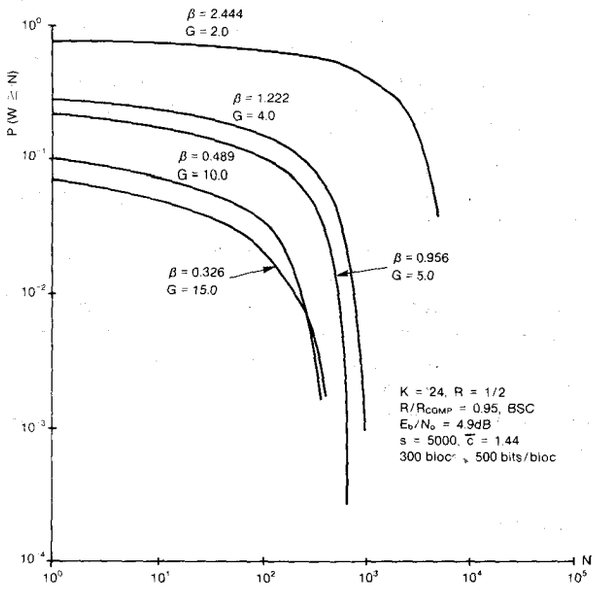

Fig. 13. — Fonctions de répartition des longueurs des files d'attente dans le tampon d'entrée pour plusieurs gains de vitesse et une pile de taille S = 5 000. Algorithme Z-J.

La figure 13 illustre bien l'interdépendance qui existe entre la taille de la pile S, le gain de vitesse G et la longueur de la file d'attente W, donc la taille du tampon d'entrée B requise. Ainsi pour maintenir β inférieur à 1 et donc se suffire d'un tampon d'entrée de longueur 2 L branches, toute augmentation de la taille de la pile entraîne une augmentation proportionnelle de G_{min}, le gain de vitesse minimum du décodeur. Par exemple, avec $L = 523$ branches. G_{min} est égal à 1,43, 4,78 et 9,56 pour des piles de tailles $S = 1\,500$, 5 000 et 10 000 entrées respectivement. Ainsi toute réduction des débordements de la pile pour un choix de piles de plus grandes tailles entraîne l'exigence de gains de vitesse plus élevés (ce qui se traduit en pratique par une électronique plus rapide) si on veut limiter la taille du tampon d'entrée à 2 L branches.

L'analyse ci-dessus montre que les paramètres clefs d'un décodeur séquentiel : tailles de la pile et du tampon, gain de vitesse et longueurs des blocs sont fortement liés les uns aux autres. Un échange est donc possible entre ces différents paramètres. Dans une application donnée, le choix final des paramètres du décodeur dépendra de cette application et de ses contraintes propres.

A première vue, les résultats ci-dessus semblent être nettement différents des résultats classiques du décodage séquentiel, où la fonction de répartition de la file d'attente au tampon d'entrée suit une loi asymptotiquement de Pareto [6-12]. Cependant ces résultats classiques ont été obtenus avec l'algorithme de Fano [4] qui ne comporte pas de pile mais seulement un tampon d'entrée. L'absence de pile de l'algorithme de Fano correspond à une pile qui ne peut jamais déborder pour l'algorithme Z-J, c'est-à-dire à une pile de taille infinie. Ceci revient donc à avoir un paramètre β infiniment grand et comme on l'a vu plus haut à une file d'attente non bornée. Il n'y a donc aucune ambiguité avec les résultats classiques de l'algorithme de Fano, mais comme ici les débordements peuvent survenir aussi bien dans la pile que dans le tampon d'entrée, on peut donc comme on le verra plus loin, contrôler avec avantage ces débordements pour qu'ils apparaissent seulement dans la pile.

6. Procédure de retransmission

Dans certaines liaisons telles les communications entre ordinateurs, une probabilité d'erreur inférieure à 10^{-10} est souvent requise. Dans ces conditions même le décodage séquentiel s'avère insuffisant et une procédure de retransmission devient nécessaire. Donc, en principe les blocs provoquant un débordement de la pile ou du tampon devraient être retransmis. Il s'agit donc de déterminer si ces débordements devraient se produire dans la pile ou le tampon d'entrée.

Un débordement du tampon d'entrée entraîne toujours des conséquences sérieuses pour le maintien de la liaison : perte de synchronisation de bloc et élimination possible de plusieurs blocs de données. Cependant le choix d'un tampon de grande taille entraîne nécessairement l'utilisation d'une grande pile, alors qu'un débordement de la pile oblige le décodeur à expurger du tampon le bloc qui a fait déborder la pile et le force à décoder un nouveau bloc. Si le bruit accompagnant ce nouveau bloc et les blocs suivants est normal, alors ces blocs seront facilement décodés, permettant au décodeur de réduire, voire même éliminer la file d'attente qui aurait pu s'accumuler dans le tampon d'entrée. Il apparaît donc déjà préférable de confiner les débordements dans la pile plutôt que dans le tampon. De plus comme la distribution du

remplissage de la pile est de type Pareto, un grand accroissement de la taille de la pile n'a que peu d'effets sur sa probabilité de débordement. Enfin, d'un point de vue pratique chaque entrée d'un nœud dans la pile requiert quelque 75 à 100 bits alors qu'une entrée dans le tampon d'entrée ne nécessite que 2 à 6 bits, de sorte que toute réduction de la taille de la pile peut se traduire par une réduction de l'espace mémoire et une simplification substantielles du décodeur.

Par conséquent il s'avère avantageux d'utiliser une pile de taille modeste afin d'y détecter très tôt par débordement les blocs difficiles à décoder, et de demander leur retransmission. Ces blocs étant expurgés assez tôt, le tampon d'entrée devient peu susceptible à des débordements et donc sa taille peut être réduite sans grande conséquence. Comme il a été montré plus haut, en choisissant le gain de vitesse G tel que $G \geqq (S/2L) \geqq \bar{C}$, un tampon de taille (2 L) branches ne débordera jamais. La taille minimale de la pile S pourra donc être déterminée en fonction de la longueur des blocs.

Un autre avantage à retransmettre les blocs qui débordent provient du fait que le décodage des blocs nécessitant un très grand effort de calcul est souvent erroné. En retransmettant ces blocs, ces erreurs peuvent donc être évitées. Par conséquent une telle procédure de retransmission permet de réduire au minimum la taille du tampon, et de réduire la probabilité d'erreurs non détectées, au prix d'une faible détérioration du taux de codage effectif. Ainsi le décodeur séquentiel servirait à la correction et à la détection des erreurs.

La technique de retransmission, a été simulée sur ordinateur utilisant la procédure ARQ sélectif [3], c'est-à-dire, que seuls les blocs causant un débordement de la pile sont retransmis (voir *fig.* 14). Utilisant encore le même code de Johannesson [19] de taux $R = 1/2$ et de longueur de contrainte $K = 24$, et des blocs de longueur $L = 523$ bits, on a fait varier le gain de vitesse G du décodeur de 2 à 20 et la taille de la pile S de 1 000 à 10 000 entrées. Sauf exceptions le nombre de blocs à transmettre est égal à 200 avec des valeurs de E_b/N_0 dans le canal direct égales à 4,64 et 4,89 dB, correspondant à $R/R_{comp} = 0{,}99$ et 0,95 respectivement. Le canal de retour est supposé sans bruit.

Pour chaque taille de la pile les 200 blocs sont d'abord transmis dans un premier cycle. Tous les blocs ayant

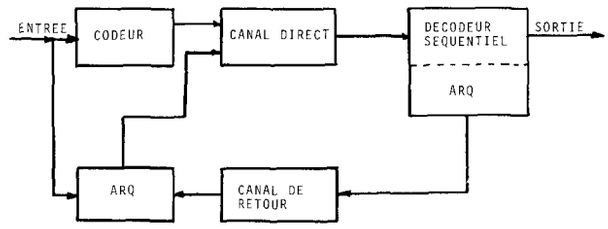

Fig. 14. – Schéma de principe du décodeur séquentiel avec ARQ.

débordés sont ensuite retransmis dans un deuxième cycle. Si au cours de ce deuxième cycle des blocs débordent encore, ils sont retransmis dans un troisième cycle et ainsi de suite jusqu'au décodage complet des 200 blocs.

A la fin de la simulation la taille maximale de la file d'attente W_{max} est observée ainsi que le nombre de blocs retransmis, le nombre de cycles de retransmissions, le nombre d'erreurs, le nombre moyen de calculs par bit, et le taux effectif de codage ou débit effectif (en nombre de bit décodé par symbole effectivement transmis, où l'effet de la queue est pris en considération). Un exemple de résultats est fourni au tableau III où la taille de la pile varie de 1 300 à 10 000 entrées. Comme on pouvait le prévoir, une pile de petite taille donne lieu à de nombreuses retransmissions et même plusieurs cycles de retransmissions. La taille de la pile augmentant, le nombre de retransmissions diminue, et très vite un seul cycle de retransmissions suffit. Sur le tableau III, on peut voir qu'au-delà d'une pile de taille 2 000 entrées tous les débordements sont résolus en un seul cycle de retransmission, et pour une pile de 10 000 entrées seulement 3 blocs sur 500 ont débordé. Les retransmissions nécessitant un effort de calcul supplémentaire, le nombre moyen de calculs $\bar{C}$ a tendance à augmenter à mesure que la taille de la pile diminue, passant de 1,73 pour une pile de 1 300 entrées à 1,49 pour une pile de 10 000 entrées. Naturellement retransmissions et effort de calcul se répercutent sur le débit effectif qui passe de 0,329 pour une pile de 1 300 entrées à 0,475 pour une pile de 10 000 entrées. (Tenant compte de la queue, le débit effectif maximum est égal à 0.478.)

Tel que mentionné plus haut, les blocs difficiles à décoder sont souvent décodés en erreur. Par consé-

TABLEAU III

Décodage séquentiel avec retransmissions par débordement de la pile
$R/R_{comp} = 0{,}95$, $E_b/N_0 = 4{,}89$ dB, $K = 24$, $R = 1/2$
$L = 523$ bits/bloc; Quantification ferme.

Pile (entrées)	Calcul moyen $\bar{C}$	Débit effectif R_{eff}	Débordements/ Nombre blocs	Cycles de retransmission	Erreurs
1 300	1,73	0,329	91/200	4	0
1 500	1,47	0,400	39/200	3	0
2 000	1,37	0,447	14/200	1	0
2 500	1,37	0,455	10/200	1	0
5 000	1,44	0,469	6/300	1	22
10 000	1,49	0,475	3/500	1	67

quent une pile de très grande taille permettant de gros efforts de calculs pour décoder un bloc peut délivrer ce bloc entaché d'erreurs, alors qu'une petite pile aura vite débordé et nécessite une retransmission de ce bloc. Le tableau III illustre bien ce phénomène où une pile de 5 000 entrés a délivré 22 erreurs sur 300 blocs et une pile de 10 000 entrées a délivré 67 erreurs sur 500 blocs. Par contre aucune erreur n'a été observée pour des piles de tailles comprises entre 1 300 et 2 500 entrées.

Tout comme pour les systèmes sans retransmissions, la fonction de répartition du nombre d'entrées dans la pile suit une loi de Pareto indépendemment du gain de vitesse du décodeur. De plus conformément à ce qui a été établi plus haut, l'échange entre gains de vitesse et tailles de la pile et du tampon se maintient. En particulier pour toutes les valeurs de β inférieures à 1, la file d'attente est restée inférieure à sa longueur maximale théorique égale à 2 L branches, et l'a dépassée dans tous les cas où β était supérieur à 1.

On peut être intéressé à une configuration minimale du décodeur : pile minimale, gain de vitesse minimal et taille de tampon minimale ($\beta \leqq 1$). Sachant que le gain de vitesse doit être au moins égal à l'effort de calcul moyen $\bar{C}$, la taille minimale de la pile S_{min} est donc

$$S_{min} = 2 L \bar{C} \tag{9}$$

avec

$$G_{min} = \bar{C}. \tag{10}$$

Naturellement avec ces valeurs minimales les débordements de la pile seront très fréquents. Par exemple au tableau III, pour $\bar{C} = 1{,}73$ la relation (9) donne $S_{min} = 1\,810$. Se référant au tableau III pour cette valeur de $\bar{C}$ la taille de la pile utilisée n'étant que 1 300, il n'est donc pas surprenant que le nombre de blocs ayant débordé soit si élevé : 91 débordements sur 200 blocs à transmettre, nécessitant quatre cycles de retransmissions, et correspondant à environ 31 % de blocs ayant débordé. Par comparaison pour $\bar{C} = 1{,}44$, $S_{min} = 1\,506$, et donc une pile de taille $S = 5\,000$ ne débordera que très rarement. Ceci est encore bien illustré sur le tableau III où seulement 6 blocs sur 300 ont débordé, soit une proportion de 2 %. En contrepartie, tel que mentionné plus haut les gains de vitesse devront être supérieurs à 1,73 et 4,78 si on veut limiter les files d'attente à une longueur maximum de $2 L = 1\,046$ branches.

Dans la technique de retransmission présentée ici, la retransmission d'un bloc n'est demandée qu'après l'apparition d'un débordement de la pile. Naturellement d'autres procédures de retransmission peuvent être envisagées. Mentionnons celles basées essentiellement sur une anticipation des difficultés de calcul (et donc des débordements de la pile) développées par Drukarev et Costello [15], et Haccoun et Pau [16]. Dans toutes ces techniques l'inconvénient de la variabilité de l'effort de calcul est mis à profit pour faire du décodeur séquentiel avec retransmission un système hybride de correction-détection des erreurs des plus puissants.

7. Conclusions

Dans cet article nous avons présenté un certain nombre de variantes de l'algorithme de décodage séquentiel de Zigangirov-Jelinek. Toutes ces variantes ont pour objectif la diminution de la variabilité de l'effort de calcul du décodage séquentiel. Les résultats de simulations extensives ont démontré que cette variabilité de l'effort de calcul peut être grandement réduite au prix d'un accroissement de l'effort de calcul moyen, mais aussi sans dégradation de la performance d'erreur. La fonction de répartition du remplissage de la pile dans toutes ces variantes étant le reflet de l'effort de calcul, cette fonction est toujours du type de Pareto, avec un exposant de Pareto qui croît avec le nombre moyen de calculs effectués. Le choix de l'algorithme à utiliser dépendra de la taille de la pile disponible. La taille de la pile limitant l'effort de calcul maximal par bloc, une analyse simple de la file d'attente dans le tampon d'entrée a montré qu'avec un gain de vitesse suffisant on peut limiter la taille du tampon d'entrée à deux longueurs de bloc sans débordement. Enfin en contrôlant les débordements de la pile et en utilisant une procédure de retransmission des blocs ayant débordé, on peut contrôler la probabilité de débordement du tampon et réduire sensiblement la probabilité d'erreur au prix d'une faible réduction du taux de codage effectif. Pour finir on a montré qu'il existe un échange possible entre la taille de la pile, la longueur des blocs, la taille du tampon et le gain de vitesse du décodeur, et qu'un choix judicieux de la taille de la pile et du gain de vitesse ne fera jamais déborder un tampon de taille deux longueurs de bloc. Ces résultats pourront être avantageusement mis à profit dans une réalisation matérielle de décodeur pour réduire substantiellement l'espace mémoire requis, et faire du décodeur séquentiel à pile une possibilité pratiquement attrayante pour la correction des erreurs dans des voies de communication très bruitées.

BIBLIOGRAPHIE

[1] F. Jelinek, A Sequential Decoding Algorithm Using a Stack, *IBM Journal of Research and Development*, vol. 13, novembre 1969.

[2] A. J. Viterbi, Convolutional Codes and Their Performance in Communication Systems, *IEEE Trans. on Com. Tech.*, vol. COM-19, octobre 1971.

[3] V. Bhargava, D. Haccoun, R. Matyas et P. Nuspl, *Digital Communications by Satellite*, John Wiley, New York, octobre 1981.

[4] R. M. Fano, A Heuristic Discussion of Probabilistic Decoding, *IEEE Trans. on Inform. Theory*, vol. IT-9, p. 64-73, avril 1963.

[5] R. G. GALLAGER, *Information Theory and Reliable Communication*, John Wiley, New York, 1968.

[6] I. M. JACOBS et E. R. BERLEKAMP, A Lower Bound to the Distribution of Computation for Sequential Decoding, *IEEE Trans. on Inform. Theory*, vol. IT-13, avril 1967, p. 167-174.

[7] D. HACCOUN et M. FERGUSON, Generalized Stack Algorithms for the Decoding of Convolutional Codes, *IEEE Trans. on Inform. Theory*, vol. IT-21, novembre 1975, p. 638-651.

[8] P. R. CHEVILLAT et D. J. COSTELLO, A Multiple Stack Algorithm for Erasure-free Decoding of Convolutional Codes, *IEEE Trans. on Comm.*, vol. COM-25, décembre 1977, p. 1460-1470.

[9] D. HACCOUN, Décodeur séquentiel haute vitesse; réalisation et tests préliminaires, Rapport Technique n° EP 84-R-19, École Polytechnique de Montréal, juin 1984, 88 pages.

[10] D. HACCOUN, M. DUFOUR, Stack and Input Buffers Overflows of Stack Decoding Algorithms, Book of Abstracts, *IEEE International Symposium on Information Theory*, Santa Monica, California, février 1981.

[11] D. HACCOUN, Problèmes de débordement de décodeurs séquentiels à pile, 8e *Colloque GRETSI sur le traitement du signal et ses applications*, Nice, France, juin 1981, p. 919-923.

[12] J. M. WOZENCRAFT et I. M. JACOBS, *Principles of Communications Engineering*, J. Wiley, New York, 1965.

[13] D. HACCOUN, A Branching Process Analysis of the Average Number of Computations of the Stack Algorithm, *IEEE Trans. on Inform. Theory*, vol. IT-30, n° 3, mai 1984, p. 497-508.

[14] P. Y. PAU et D. HACCOUN, Sequential Decoding with ARQ, Book of Abstracts, *Proc. IEEE International Symposium on Information Theory*, Saint-Jovite, Québec, septembre 1983.

[15] A. DRUKAREV et D. J. COSTELLO Jr., Hybrid ARQ Error Control Using Sequential Decoding, *IEEE Trans. Inform. Theory*, vol. IT-29, juillet 1983, p. 521-535.

[16] D. HACCOUN et P. Y. PAU, Analysis of Sequential Decoding With Retransmission Procedures, soumis pour publication dans *IEEE Trans. on Inform. Theory*.

[17] D. HACCOUN, Efforts de calcul et débordements de décodeurs séquentiels à pile, Rapport Technique EPM/RT-85-15, École Polytechnique de Montréal, mai 1985, 55 pages.

[18] A. JANELLE et D. HACCOUN, Décodage adaptatif des codes convolutionnels, Rapport Technique EP 78-R-18, École Polytechnique de Montréal, 1978, 156 pages.

[19] R. JOHANNESSON, Robustly Optimal Rate One-half Binary Convolutional Codes, *IEEE Trans. on Inform. Theory*, vol. IT-21, juillet 1975, p. 464-468.

[20] D. HACCOUN et CHEN NAIYUN, Variants of the Stack Algorithm for the Decoding of High Rate Codes By Sequential Decoding, 1*st Canadian Domestic and International Satellite Communications Conference*, Ottawa, p. 21.4.1-21.4.5, juin 1983.

[21] D. HACCOUN, A Markov Chain Analysis of the Sequential Decoding Metric, *IEEE Trans. on Inform. Theory*, vol. IT-26, n° 1, p. 109-113, janvier 1980.

ANNEXE

Structure des données de l'algorithme Z-J

Pour nos simulations, tous les progiciels comprenant émetteur, canal, récepteur et décodeur ont été écrits en FORTRAN et exécutés sur un ordinateur IBM 4341 de l'École Polytechnique de Montréal. Selon l'algorithme utilisé, le décodage de 100 000 bits requiert 6 à 8 minutes de temps de calcul de l'Unité Centrale de Traitement. Dans cette annexe, nous présentons la structure des données utilisée pour simuler sur ordinateur les différentes variantes de l'algorithme à pile du décodeur séquentiel, en particulier l'organisation de la pile [3]. Cette même structure a aussi été utilisée dans la réalisation matérielle d'un décodeur séquentiel haute vitesse [9].

Une « entrée » dans la pile (c'est-à-dire un nœud) doit comporter *trois* entités nécessaires à l'extension du nœud, à savoir la *métrique cumulée* du nœud, sa *profondeur* dans l'arbre, et *l'état* correspondant du codeur. De plus, on utilise deux *pointeurs d'adresses* qui servent à la mise en ordre de la pile et à la récupération de la séquence décodée. Toute cette information est stockée dans cinq mots contigus ayant tous la même adresse, et appelés respectivement META, ETAT, PROF, SUIV et PERE. Un espace-mémoire de plusieurs milliers d'entrées doit être prévu. Enfin, un vecteur auxiliaire (les *sous-piles*) dénoté SPIL est aussi utilisé pour trouver facilement la sous-pile maximale et le nœud à prolonger tel qu'expliqué ci-après.

Initialement, tout l'espace-mémoire de la pile et du vecteur auxiliaire SUIV sont mis à zéro, et on assigne au nœud origine une métrique cumulée de valeur positive et arbitraire afin d'éviter la manipulation de métriques accumulés négatives. Les entrées successives dans la pile se font à des adresses consécutives et, une fois dans la pile, une entrée n'est jamais détruite. Toute entrée ayant une métrique cumulée Γ donnant une même valeur entière de $Q = \Gamma/H$ appartient à la même sous-pile Q.

D'un point de vue structurel, tous les nœuds de la pile appartenant à une même sous-pile Q_m, sont liés entre eux par leur chaîne de pointeurs SUIV. Le pointeur SUIV du premier nœud de la sous-pile Q_m contient la valeur 0, le pointeur SUIV du second nœud de Q_m contient l'adresse du premier, le pointeur SUIV du troisième nœud de Q_m contient l'adresse du second, et ainsi de suite jusqu'au dernier nœud de Q_m. L'adresse du dernier nœud de Q_m (c'est-à-dire le début de la chaîne) est contenue dans le Q_m-ième mot du vecteur SPIL. En d'autres termes, toutes les entrées de la pile appartenant à une même sous-pile Q sont liées entre elles par les pointeurs SUIV, et le début de la chaîne se trouve à la Q-ième adresse du vecteur auxiliaire SPIL.

Soit un nœud de métrique Γ et de sous-pile Q_m à stocker dans la pile à l'adresse courante N. Les valeurs des métriques, état, et profondeur, sont

d'abord stockées dans leur registre respectif à l'adresse N. Ensuite, l'adresse contenue dans SUIV (Q_m) est stockée dans le pointeur SUIV (N) et la valeur N est stockée dans SUIV (Q_m), ce qui préserve l'intégrité de la chaîne des entrées de sous-pile Q_m. Pour toute sous-pile vide Z, la valeur correspondante de SUIV (Z) est zéro.

Lorsqu'un nœud doit être prolongé par l'algorithme, son « élimination » consiste simplement à le retirer de la chaîne des pointeurs SUIV, ce qui a pour effet qu'il ne soit plus jamais prolongé. La sous-pile maximale courante notée Q_{top} étant toujours connue par l'algorithme, pour déterminer le nœud à prolonger, il suffit de trouver la première sous-pile non vide, en commençant bien sûr par Q_{top} et, de là, de trouver l'adresse du dernier nœud appartenant à cette sous-pile et inséré dans la pile. Naturellement, la chaîne du pointeurs SUIV et le vecteur SPIL doivent être correctement modifiés pour « éliminer » ce nœud prolongé.

A titre d'exemple, considérons les nœuds stockés dans la pile aux adresses 75, 90 et 100, et dont les métriques accumulées sont respectivement 641, 647 et 645. Soit $H=8$ de sorte que les sous-piles correspondant à ces métriques sont toutes égales à $641/8=80$. Ces trois entrées appartenant toutes à la même sous-pile 80, elles sont toutes liées par une même chaîne de pointeurs SUIV dont le début réside dans SPIL (80), comme indiqué ci-dessous et par le schéma de la figure A1 (*a*).

$$\text{SUIV}\ (75) = 0$$
$$\text{SUIV}\ (90) = 75$$
$$\text{SUIV}\ (100) = 90$$

et

$$\text{SPIL}\ (80) = 100$$

Supposons à présent que la sous-pile maximale Q_{top} soit égale à 80. Par conséquent, selon la procédure LIFO décrite plus haut, c'est le dernier nœud de la sous-pile 80 qui doit être prolongé, c'est-à-dire celui dont l'adresse est contenue dans SPIL (80), donc le nœud à l'adresse 100. Une fois ce nœud prolongé, on le retire de la chaîne des pointeurs en posant simplement

$$\text{SPIL}\ (80) = \text{SUIV}\ (100) = 90.$$

La figure A1 (*b*) illustre cette situation.

A la fin du décodage d'un bloc, il faut récupérer la séquence d'information décodée. Ceci est effectué à l'aide de la chaîne de pointeurs PERE. Le pointeur PERE de chaque nœud de la pile contient l'adresse du nœud « père » duquel il est issu. Ainsi, partant du dernier nœud décodé, par la chaîne de pointeurs, PERE, on remonte branche par branche au nœud origine de la séquence décodée.

Simplification : utilisation de fanion

A l'examen de l'algorithme, il apparaît que pour les codes binaires de taux $R=1/V$, (1/2, 1/3, ...), chaque extension de chemin implique un accès à SPIL et

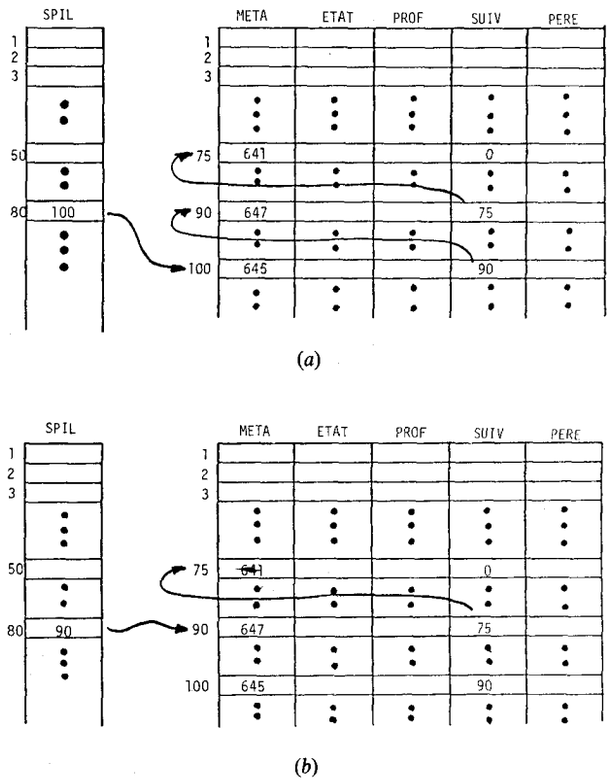

Fig. A.1. – Structure des données de la pile : (*a*) Insertion d'un nœud dans la pile. (*b*) Extension d'un nœud de la pile.

deux nouvelles entrées dans la pile. Pour des codes de taux $R=U/V$, chaque extension implique 2^U entrées dans la pile. Comme une entrée dans la pile nécessite quelque 80 à 100 bits, naturellement on cherche toujours à éliminer toute entrée jugée inutile. Pour les codes de taux $R=U/V$, $U=(V-1)$, (par ex. 2/3, 3/4, ...), une procédure d'élimination de branche basée sur l'utilisation de seuils de rejets a été proposée par l'auteur et a donné d'excellents résultats [20].

Dans le cas des codes de taux 1/V, une procédure très simple, appelée *procédure de fanion* (en anglais « Flag »), permet de réduire considérablement le nombre d'entrées dans la pile. Cette procédure est basée sur le fait que si une des deux branches issues d'un nœud prolongé a une métrique positive, l'autre branche aura une métrique négative. (Ce fait découle du choix de la métrique biaisée (1) en décodage séquentiel [21].) Par conséquent, si l'une des extensions d'un nœud qui vient d'être prolongé a une métrique positive, cette extension aura nécessairement la métrique accumulée maximale, c'est-à-dire sera au sommet de la pile et devra dont être prolongée à son tour. Il est donc tout à fait inutile de l'insérer dans la pile pour l'en extraire immédiatement après. Cependant, si cette branche qui n'a pas été insérée faisait partie du chemin correct, alors bien sûr on ne pourrait pas la récupérer. Donc la seule information utile à préserver pour ce nœud non inséré est le bit d'information qui y correspond. Sachant que ce bit d'in-

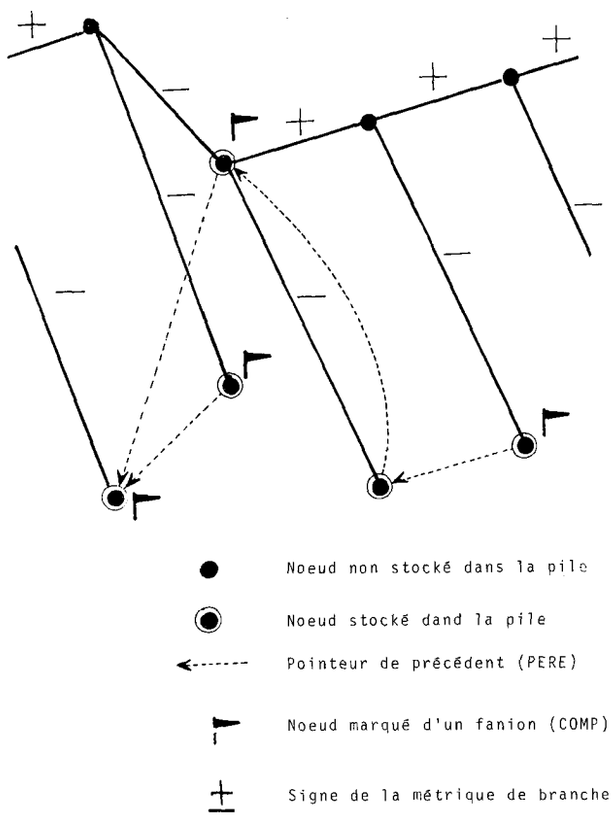

Fig. A.2. — Illustration de la procédure du fanion.

formation est le complément du bit d'information correspondant à la branche négative issue du même nœud que la branche positive, il suffit donc de l'indiquer par un indicateur binaire, ou fanion. Si l'indicateur est activé (fanion levé), on complémente le bit d'information de la branche, sinon on ne le complémente pas. La structure de données de la pile contient donc une entité supplémentaire portant un bit d'information dans laquelle on indique si le fanion est levé ou baissé. La figure A 2 donne un exemple de l'utilisation de fanion qui est noté COMP.

D'un point de vue pratique, l'usage de ce fanion est-il rentable? En effet, pour que ce soit rentable d'un point de vue espace-mémoire, il faut que l'économie du nombre d'entrées dans la pile compense l'augmentation de mémoire de la pile entière requise par le fanion. Une analyse théorique originale du comportement de la métrique a montré que selon les valeurs du rapport signal à bruit, plus de 70% des extensions du chemin correct donnent lieu à des métriques de branche positives [21], cette proportion augmentant avec le rapport signal à bruit. Sachant qu'une entrée dans la pile implique de 80 à 100 bits d'espace-mémoire, il est clair que l'usage de fanion conduit à une économie de mémoire appréciable. Une analyse détaillée de cette procédure a confirmé cette économie, même dans le cas d'un décodeur séquentiel utilisé en conjonction avec une procédure de retransmission, et donc tous les algorithmes à pile pour les codes de taux 1/V utilisent la procédure du fanion. Cette procédure a bien sûr été implantée dans notre réalisation matérielle du décodeur séquentiel [9].

Manuscrit reçu en septembre 1985
version révisée le 15 *mai* 1986