Titre: Title:	Analyse des performances de multidécodeurs pour le décodage séquentiel
Auteur: Author:	Mylène Toulgoat
Date:	1989
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Toulgoat, M. (1989). Analyse des performances de multidécodeurs pour le décodage séquentiel [Master's thesis, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/58287/

UNIVERSITE DE MONTREAL

ANALYSE DES PERFORMANCES DE MULTIDECODEURS POUR LE DECODAGE SEQUENTIEL

par

Mylène TOULGOAT
DEPARTEMENT DE GENIE ELECTRIQUE
ECOLE POLYTECHNIQUE

MEMOIRE PRESENTE EN VUE DE L'OBTENTION DU GRADE
DE MAITRE ES SCIENCES APPLIQUEES (M.Sc.A.)

décembre 1989

ISBN 0-315-58126-3

UNIVERSITE DE MONTREAL

ECOLE POLYTECHNIQUE

Ce mémoire intitulé:

ANALYSE DES PERFORMANCES DE MULTIDÉCODEURS POUR LE DÉCODAGE SÉQUENTIEL

Présenté par: MYLENE TOULGOAT

en vue de l'obtention du grade de: MAÎTRE ÈS SCIENCES APPLIQUÉES (M.Sc.A.)

a été dûment accepté par le jury d'examen constitué de:

Mme Catherine Rosenberg, D. Sc., présidente

M. Jean Conan, Ph.D., membre du jury

M. Yvon Savaria, Ph.D., directeur

M. David Haccoun, Ph.D., co-directeur

SOMMAIRE

Le décodage séquentiel des codes convolutionnels est une puissante technique de correction d'erreurs. Le principal problème de cette technique est la variabilité du nombre de calculs par bit décodé. Cette variabilité impose la présence de deux tampons, un devant et un après le décodeur séquentiel. Il peut arriver que le tampon à l'entrée du décodeur déborde entraînant la perte de la synchronisation ainsi que de plusieurs blocs de données lorsque le décodeur est occupé avec un bloc très bruité.

Dans le but d'apporter une solution à ce problème, nous proposons de mettre plusieurs décodeurs séquentiels en parallèle avec une file d'attente commune à l'entrée. On cherche à caractériser les performances de ce système comme le débit efficace et la répartition des blocs dans la file d'attente en fonction de divers paramètres comme la taille de la pile, le rapport signal-à-bruit dans le canal, la longueur des blocs de données.

Dans un premier temps, la loi de service d'un décodeur séquentiel est formulée en fonction de divers paramètres. Les performances d'un système à s serveurs sont ensuite déterminées à l'aide de simulations sur ordinateur.

Finalement, la possibilité d'utiliser des ressources de réserve lors d'un débordement de pile est étudiée.

ABSTRACT

Sequential decoding using the Zigangirov-Jelinek (STACK) algorithm is a powerful decoding technique for convolutional codes. The main drawback of this technique is its computationnal variability. On that ground, the decoder must have an input buffer and an output buffer. Computational variability will occasionally lead to input buffer overflow or stack overflow and consequently, information erasures.

To solve this problem, parallelism is examined as an approach to alleviate the computational time variability of the Zigangirov-Jelinek algorithm. Fast service time can be achieved by using a number of moderate speed sequential decoders working concurrently. We analyse the performances queues, throughput of this new architecture according to stack length, signal to noise ratio.

First, the service law of sequential decoding is determined according to some parameters. A study of a multiprocessor decoder is done with computer simulations. Finally, the possibility of using spare decoders after a stack overflow is examined.

REMERCIEMENTS

Je tiens tout d'abord à remercier mon directeur, Yvon Savaria, qui m'a permis de mener à terme mon projet ainsi que mon co-directeur, David Haccoun, pour ses remarques pertinentes sur ma recherche. Je remercie aussi mon conjoint Eric Lajoie pour le soutien moral et informatique qu'il m'a apporté durant mes études de maîtrise.

Je désire remercier aussi tous ceux qui de près ou de loin m'ont aidée à la réalisation du projet. Je tiens à souligner notamment le travail de Jean-François Huard pour la réalisation d'un simulateur de décodeurs séquentiels et pour ses conseils lors de la réalisation de mon simulateur. Je remercie M^{me} Rosenberg pour les discussions qui m'ont guidée au début de ce travail. Je remercie également Nicolas Arel, Yves Blaquière et Laurent Amar pour leurs conseils en programmation. De même, je remercie Claude Thibeault pour son aide précieuse lors de la réalisation de mes tableaux et graphiques.

Finalement, je tiens à souligner l'aide financière reçue par le biais d'une subvention du Conseil de Recherche en Sciences Naturelles et en Génie du Canada.

TABLE DES MATIERES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES SYMBOLES

K: longueur de contrainte du code

L: nombre de bits d'information dans un bloc

r: taux de codage

M: mémoire d'un codeur

U: séquence d'information

$X^{(U)}$: séquence codée

Y: séquence reçue

$P(y_j|x_i)$: probabilité de transition d'un canal DMC d'une entrée i vers une sortie j

q(i): probabilités d'entrée d'un DMC

τ_i: métrique de branche

Γ: métrique accumulée

B: biais de la métrique de branche

ß: constante multiplicatrice de la loi de Pareto

α: exposant Pareto

C_{bloc}: nombre de calculs par bloc

C_{bit}: nombre de calculs par bit

Cmax: nombre maximal de calculs que permet une pile de taille finie S

S: taille de la pile du décodeur séquentiel

C: capacité du canal

R_{comp}: taux de coupure d'un canal

$E_o(\alpha)$: fonction de Gallager

P_{ov}: probabilité de débordement du tampon d'entrée

B: capacité du tampon d'entrée en branches

U: gain de vitesse du décodeur séquentiel

P_d: probabilité de débordement de la pile du décodeur séquentiel

P(E): probabilité d'erreur sur l'ensemble des codes

W(y): distribution du temps d'attente (ou du nombre de blocs) dans le tampon d'entrée

t_m: temps d'interarrivée entre deux blocs

ρ: x_s/t_m

x_s: temps moyen de service

σ_a^2: variance du temps d'interarrivée

σ_b^2: variance de la loi de service

Wmoy: temps moyen d'attente dans le tampon d'entrée (dans notre cas, nombre moyen de blocs dans le tampon)

s: nombre de serveurs

$\%deb_i$: pourcentage de débordement de la pile S_i

K_1, K_2: constantes à déterminer dans l'expression de la fonction de densité

n_1, n_2: exposants à déterminer dans l'expression de la fonction de densité

fc(x): fonction de densité du nombre de calculs par bloc

x_m: abcisse du point maxiaml de la fonction de densité

$Cmoy_i$: nombre moyen de calculs par bloc du décodeur i

σ_i^2: variance du nombre de calculs par bloc du décodeur i

Eb/No: rapport signal-à-bruit dans le canal

n_d: nombre de blocs décodés

n_t: nombre de blocs transmis

n_i: nombre de blocs qui sont décodés par le décodeur i

n_{deb}: nombre de blocs qui débordent

s_1: nombre de décodeurs simples dans le multidécodeur avec décodeurs de réserve

s_2: nombre de décodeurs de réserve dans le multidécodeur avec décodeurs de réserve

$Cmax_1$: nombre maximal de calculs permis pour un décodeur simple

$Cmax_2$: nombre maximal de calculs permis pour un décodeur de réserve

$Cmoy_1$: nombre moyen de calculs permis pour un décodeur simple

$Cmoy_2$: nombre moyen de calculs permis pour un décodeur simple

P_{di}: probabilité de débordement de la pile i

$p_w(x=i)$: probabilité que la file d'attente soit d'une longueur i

Wmax: nombre maximal de blocs dans la file d'attente à l'entrée

w_{theo}: longueur moyenne théorique de la file d'attente à l'entrée

w_{sim}: longueur moyenne de la file d'attente tirée des simulations

C_1: coût de l'extenseur

C_2: coût de l'historique

C_3: coût du module bloc

C_4: coût de la file prioritaire

n_{ex}: nombre de transistors par extenseur

n_{bit}: nombre de transistors par bit d'historique

P_h: profondeur de l'historique

n_{bloc}: nombre de transistors par module bloc

n_{enr}: nombre de transistors par enregistrement

P_f: profondeur de la file prioritaire

e_{ci}: nombre d'enregistrement par circuit de file

c_{enr}: nombre de circuits de file nécessaire pour avoir des enregistrements complets

CHAPITRE 1

INTRODUCTION

Le but des communications numériques est la transmission de données avec un bon débit et une faible probabilité d'erreur. L'utilisation du codage contribue à atteindre cet objectif. Les codes convolutionnels sont particulièrement bien adaptés aux communications terrestres et par satellite car , utilisés avec un bon décodeur, ils permettent d'atteindre une probabilité d'erreur très faible. Leur utilisation peut se faire selon plusieurs techniques qui se divisent en trois groupes: FEC (Forward Error Correction), ARQ (Automatic Repeat Request), FEC/ARQ.

1.1 Techniques de transmission de l'information

La technique FEC (Forward Error Correction) consiste à utiliser un code correcteur d'erreur lors de la transmission du message. Un système de communication avec codage correcteur d'erreur est représenté à la figure 1.1.

Le taux de codage d'un code convolutionnel est donné par b/v bits/symbole. Il y a donc transmission de v symboles codés pour b bits d'information. Plus le taux de codage est faible (c.-à-d. v élevé), plus la redondance est élevée. Le codage augmente la vitesse de transmisssion car v>b. Donc, la

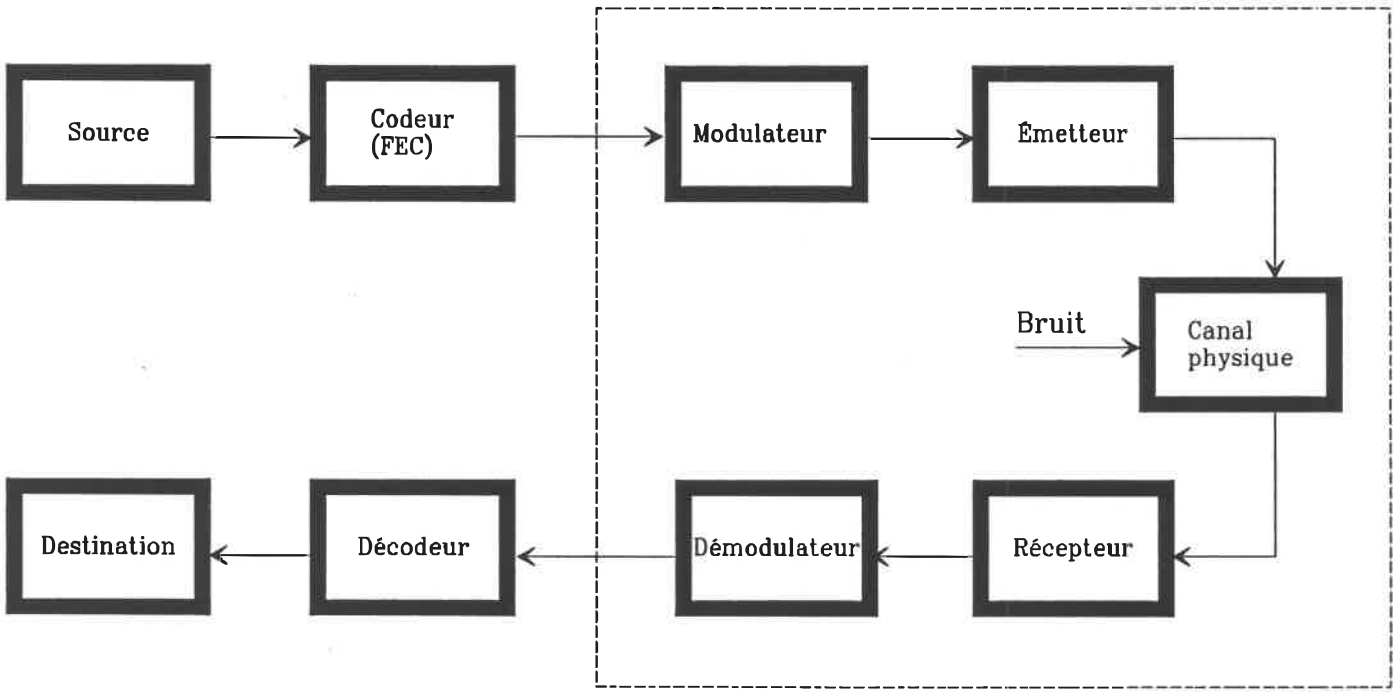


Figure 1.1 Système de communication utilisant la technique FEC

largeur de bande utilisée augmente. Par conséquent, en supposant une puissance de transmission constante, l'énergie par symbole est plus petite que l'énergie par bit, donc le récepteur reçoit plus de symboles codés erronés que le nombre de bits qui seraient en erreur si le codage n'était pas employé. Cependant, si le système est bien conçu, grâce au pouvoir de correction du code, le système codé peut donner une meilleure performance d'erreur générale que le système non codé.

La limite de la technique FEC est le pouvoir de correction du code. Lorsqu'il est dépassé, le message délivré à l'usager est erroné. Il est donc avantageux d'utiliser un code avec une grande redondance pour diminuer la probabilité d'erreur mais le débit efficace, qui est le rapport entre le nombre de bits d'information délivrés à l'usager et le nombre de symboles codés transmis sur le canal, est alors diminué.

Shannon a montré [1] que pour tout canal il existe des codes qui permettent une communication fiable si et seulement si le taux de codage est inférieur à un taux maximal C appelé capacité du canal. Par conséquent, pour des taux de codage supérieurs à C, il est certain que la probabilité d'erreur ne peut pas tendre vers zéro.

Il faut cependant noter que jusqu'à présent il a été impossible de fonctionner à un taux près de la capacité et

d'obtenir des transmissions sans aucune erreur. Il existe pourtant des applications comme les communications entre ordinateurs où les erreurs ne sont absolument pas tolérées. On fait alors appel à une autre technique , appelée ARQ (automatic repeat request)[2].

La technique ARQ demande la retransmission des données lorsqu'elles sont détectées en erreur au récepteur. Cette méthode nécessite de diviser l'information en blocs pour pouvoir déterminer quel bloc est détecté en erreur. De plus, le transmetteur doit conserver les blocs jusqu'à ce que le récepteur lui envoie un signal qui indique si oui ou non le bloc a été détecté en erreur. Si le bloc est détecté sans erreur, c'est un ACK (acknowledgement) qui est envoyé sinon c'est un NACK. Il doit donc aussi y avoir un canal de retour pour permettre l'envoi des ACK(NACK). Comme dans les systèmes ARQ on ne cherche qu'à détecter les erreurs, il n'est pas nécessaire d'utiliser un code très puissant. Il y a donc moins de symboles codés transmis pour un bit d'information d'où une augmentation du taux de codage. Par contre, la retransmission des blocs erronés diminue le débit efficace du système. Par conséquent, lorsque le canal se dégrade, le débit efficace peut diminuer considérablement ce qui n'est pas souhaitable. Un autre inconvénient de cette technique est le délai nécessaire à l'obtention de l'information par l'usager et le problème de tamponnage qui

en découle. Il peut donc s'avérer plus avantageux d'utiliser une troisième approche: l'approche hybride FEC/ARQ.

Un système hybride FEC/ARQ est composé d'un système FEC contenu dans un système ARQ. Le système ARQ retransmet les blocs détectés en erreur et permet ainsi d'atteindre une très faible probabilité d'erreur tandis que le système FEC diminue le nombre de retransmissions en corrigeant autant de blocs en erreur que possible. Cette technique combine les avantages des deux précédentes tout en minimisant leurs inconvénients. En effet, elle permet d'obtenir un bon compromis entre une probabilité d'erreur faible et un débit efficace élevé au coût d'une plus grande complexité.

1.2 Enoncé du problème

Ce mémoire étudie les systèmes FEC se servant des codes convolutionnels. La méthode de décodage utilisée est le décodage séquentiel qui est une technique de décodage probabiliste des codes convolutionnels. Le problème majeur du décodeur séquentiel est que le nombre de calculs qui lui est nécessaire pour décoder un bloc est variable.

Puisque le temps de calcul est variable, il doit y avoir un tampon à l'entrée du décodeur pour accumuler les blocs venant du canal si le décodeur séquentiel est déjà occupé. Ce tampon peut causer des problèmes car s'il déborde,

plusieurs blocs ainsi que la synchronisation sont perdus. Pau et Haccoun ont proposé une solution qui consiste à limiter les ressources du décodeur séquentiel pour diminuer la probabilité de débordement du tampon d'entrée [3]. Nous ajoutons une nouvelle dimension à cette solution. Nous proposons une nouvelle architecture consistant à mettre un nombre s de décodeurs séquentiels en parallèle tout en limitant les ressources de chacun d'eux. Le système est représenté à la figure 1.2. La vitesse de décodage de ce multidécodeur est environ s fois plus rapide que s'il n'y avait qu'un seul décodeur séquentiel et le coût est s fois plus élevé. C'est une approche très intéressante puisqu'en général le coût d'un décodeur s fois plus rapide qu'un décodeur de vitesse lente est beaucoup plus élevé que s fois le coût du décodeur de vitesse lente. Donc, du point de vue du coût versus augmentation de vitesse, ce système s'avère très prometteur. Etant donné que le temps de calcul des décodeurs séquentiels reste variable, il est possible que tous les décodeurs soient occupés et incapables d'accepter un nouveau bloc. Une file d'attente se forme à l'entrée comme dans le cas d'un seul serveur. Il reste à savoir si cette file d'attente se comporte de façon satisfaisante. Il semble que l'effet de moyennage devrait limiter le nombre de blocs dans le tampon à l'entrée et par conséquent limiter la probabilité de débordement de ce dernier. Ce mémoire se

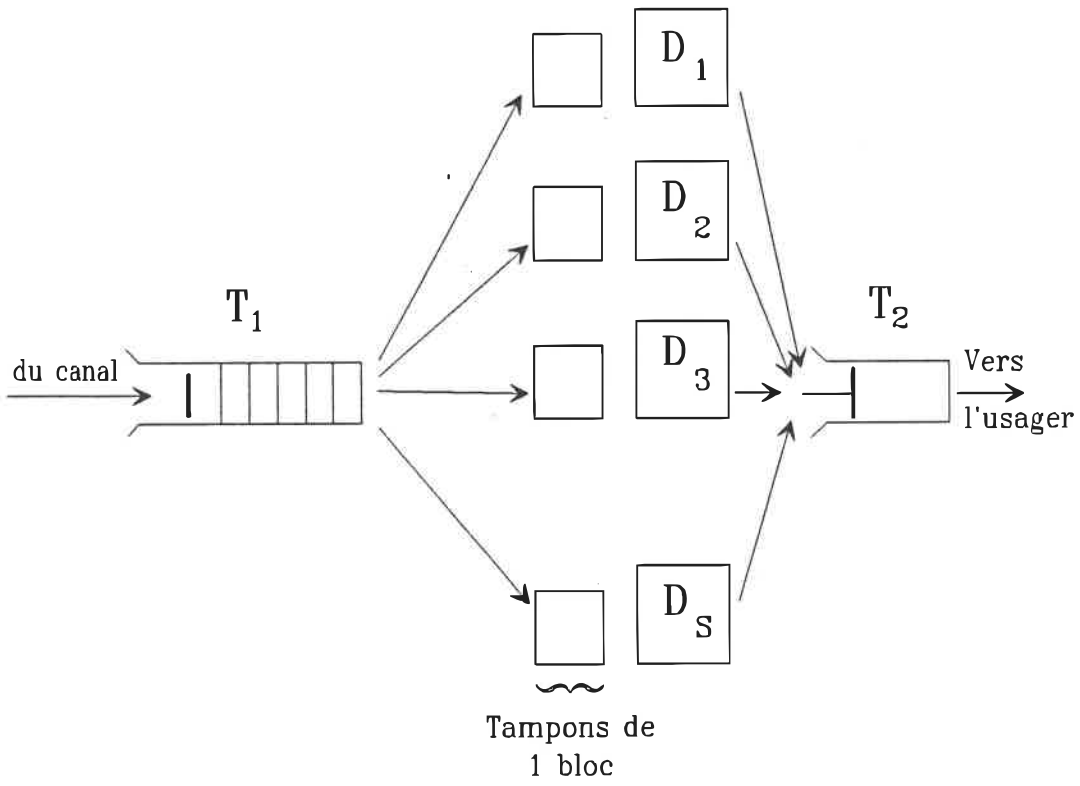


Figure 1.2 Multidécodeur

concentre sur l'analyse des performances (files d'attente, débit efficace, etc...) du multidécodeur. Le multidécodeur est une file d'attente pour laquelle le taux d'arrivée des blocs au système est constant et la loi de service des s serveurs reste à déterminer. Notre analyse porte d'abord sur la loi de service du serveur pour ensuite caractériser la file d'attente à l'entrée et le débit efficace du système de s décodeurs séquentiels mis en parallèle.

1.3 Plan du mémoire

Le chapitre 2 est consacré au codage convolutionnel ainsi qu'au décodeur séquentiel. Les conséquences de la variabilité du nombre de calculs par bit sont discutées pour ensuite terminer avec une courte introduction sur les files d'attente.

Le chapitre 3 traite essentiellement de la loi de service par bloc du décodeur séquentiel.

Quant au chapitre 4, il apporte quelques notions théoriques sur le multidécodeur étudié soit les conditions de stabilité et les conditions limites pour que la file d'attente à l'entrée soit d'une longueur maximale de un bloc. Ces conditions sont obtenues en faisant l'hypothèse que le tampon à l'entrée est infini. Dans la deuxième partie du chapitre, ces mêmes notions théoriques sont reprises mais

pour une version plus performante du système: le multidécodeur avec décodeurs de réserve.

Le multidécodeur de la figure 1.2 est caractérisé à l'aide de simulations dont les résultats sont présentés au chapitre 5. Nous caractérisons notamment les tailles des files d'attente à l'entrée et à la sortie de même que le débit efficace en fonction de divers paramètres.

Le chapitre 6 clôt la série de résultats par une analyse partielle de l'option multidécodeur avec décodeur de réserve et sa comparaison avec le multidécodeur simple. Finalement, le mémoire se termine par une conclusion générale.

1.4 Contributions spécifiques

Les contributions principales apportées par cette recherche sont les suivantes:

- Détermination d'une expression pour la loi de service par bloc du décodeur séquentiel.
- Application au multidécodeur simple et au multidécodeur avec décodeurs de réserve des conditions de stabilité données par la théorie des files d'attente.

- Généralisation à s serveurs de la condition limite (donnée par Haccoun[4] pour un seul serveur) pour que la file d'attente soit d'une longueur maximale de un bloc.
- Simulation du multidécodeur de la figure 1.2 à l'aide de la loi de service dont il a été question plus haut.
- Simulation d'une variante du multidécodeur simple, le multidécodeur avec décodeurs de réserve et comparaison avec le multidécodeur simple.

CHAPITRE 2

CODAGE CONVOLUTIONNEL

Il a été vu au chapitre précédent que le codage convolutionnel est particulièrement bien adapté aux communications terrestres et par satellite. Un système de communication avec codage correcteur d'erreur est représenté à la figure 1.1. Il est essentiellement constitué d'un encodeur, d'un canal discret sans mémoire (Dicrete Memoryless Channel) et d'un décodeur. Dans ce chapitre, il sera discuté de l'encodage convolutionnel et des diverses représentations du processus d'encodage, de la performance des codes, des principes de décodage optimal et de deux algorithmes de décodage probabiliste: l'algorithme de Viterbi et l'algorithme de Zigangirov-Jelinek.

2.1 L'encodeur convolutionnel

Dans ce mémoire, seuls les codes de taux 1/v sont utilisés. L'analyse du codeur est donc restreinte à ceux-ci.

Un codeur convolutionnel est une machine linéaire à états finis. Il est constitué d'un registre à décalage de longueur K, de v additionneurs modulo-2 connectés à certaines cellules du registre et d'un commutateur qui échantillonne la

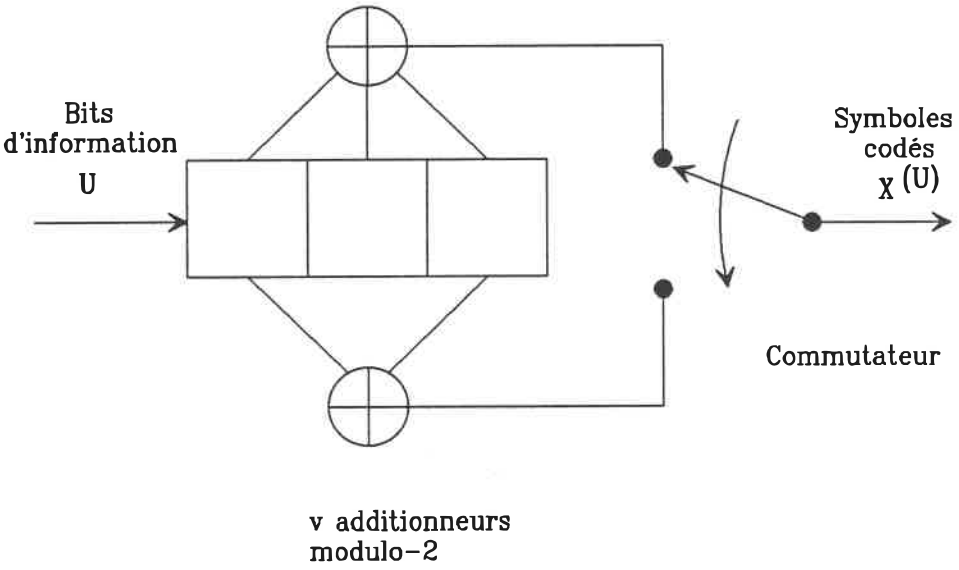
sortie des v additionneurs. Le schéma d'un codeur K=3 et v=2 est donné à la figure 2.1.

Le code est complètement spécifié par les connexions entre les cellules du registre à décalage et les v additionneurs modulo-2. Celles-ci sont exprimées par le vecteur rangée $G_j=(g_{1j},\ldots,g_{vj})$ j=1,2,...,K, appelé vecteur des connexions. L'élément g_{ij} est égal à 1 si la $j^{ième}$ cellule du registre est reliée au $i^{ième}$ additionneur modulo-2. Les vecteurs de connexion de la figure précédente sont $G_1=(1,1)$, $G_2=(1,0)$, $G_3=(1,1)$.

Une queue de K-1 bits de valeur 0 est ajoutée à la fin de chaque bloc de L bits d'information pour s'assurer qu'au début de l'encodage du bloc suivant, l'état du codeur est zéro. Par conséquent, une séquence de L bits d'information $U=(u_1,u_2,u_3,\ldots,u_L)$ produit la séquence codée de sortie $X=(x_1^{(1)},\ldots,x_1^{(v)},\ldots,x_{L+K-1}^{(1)},\ldots,x_{L+K-1}^{(v)})$. La longueur totale de la séquence de sortie est de (L+K-1)v symboles.

Le taux de codage r est le rapport entre le nombre de bits d'information et le nombre de symboles codés c.-à-d. r = L/(L+K-1)v bits/symbole. En général, L >> K ce qui permet d'écrire $r\approx 1/v$.

Les v symboles représentant un bit d'information dépendent du bit lui-même et des (K-1) bits précédents. La mémoire M du codeur est définie comme le nombre de bits précédents c.-à-d. M=K-1. Par ailleurs, le nombre de cellules



K=3
v=2
r=1/2
M=2

Figure 2.1 Codeur convolutionnel

du registre à décalage donne la longueur de contrainte K du code.

Le processus d'encodage peut être représenté de trois façons: par le diagramme d'états, la structure en arbre et la structure en treillis.

L'encodeur étant une machine linéaire à états finis, un diagramme d'états lui est associé. L'état est donné par les (K-1) bits précédant le bit d'information à coder. Le nombre d'états est déterminé par la mémoire du codeur . Il y a 2^{K-1} états différents possibles. Puisque pour chaque état le bit d'information entrant peut prendre la valeur 0 ou 1, deux séquences de sorties distinctes possibles sont associées à chaque état. Ces séquences sont déterminées par le code c'est-à-dire les vecteurs de connexion. Le diagramme d'états du codeur de l'exemple précédent est montré à la figure 2.2. Dans ce cas, K est égal à 3. Il y a donc quatre états d'encodeurs différents: 00, 01, 10, 11. Le changement d'un état à un autre se fait selon la valeur du bit d'entrée donnée entre parenthèses. Les symboles codés sont inscrits sur les transitions. Ce type de représentation est très compacte.

La structure en arbre associée au diagramme d'états précédent est donnée à la figure 2.3. Elle représente l'évolution dans le temps de l'encodage. Un état du codeur

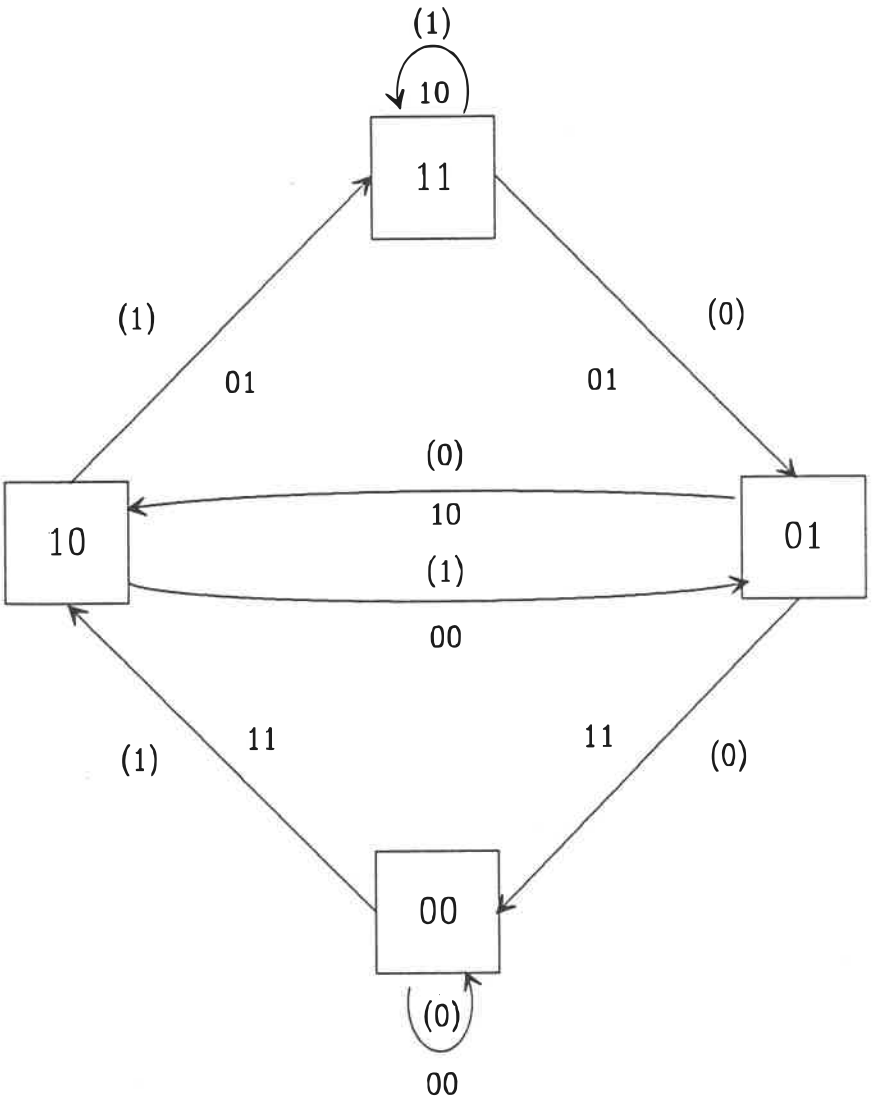


Figure 2.2 Diagramme d'état associé au codeur de la figure 2.1

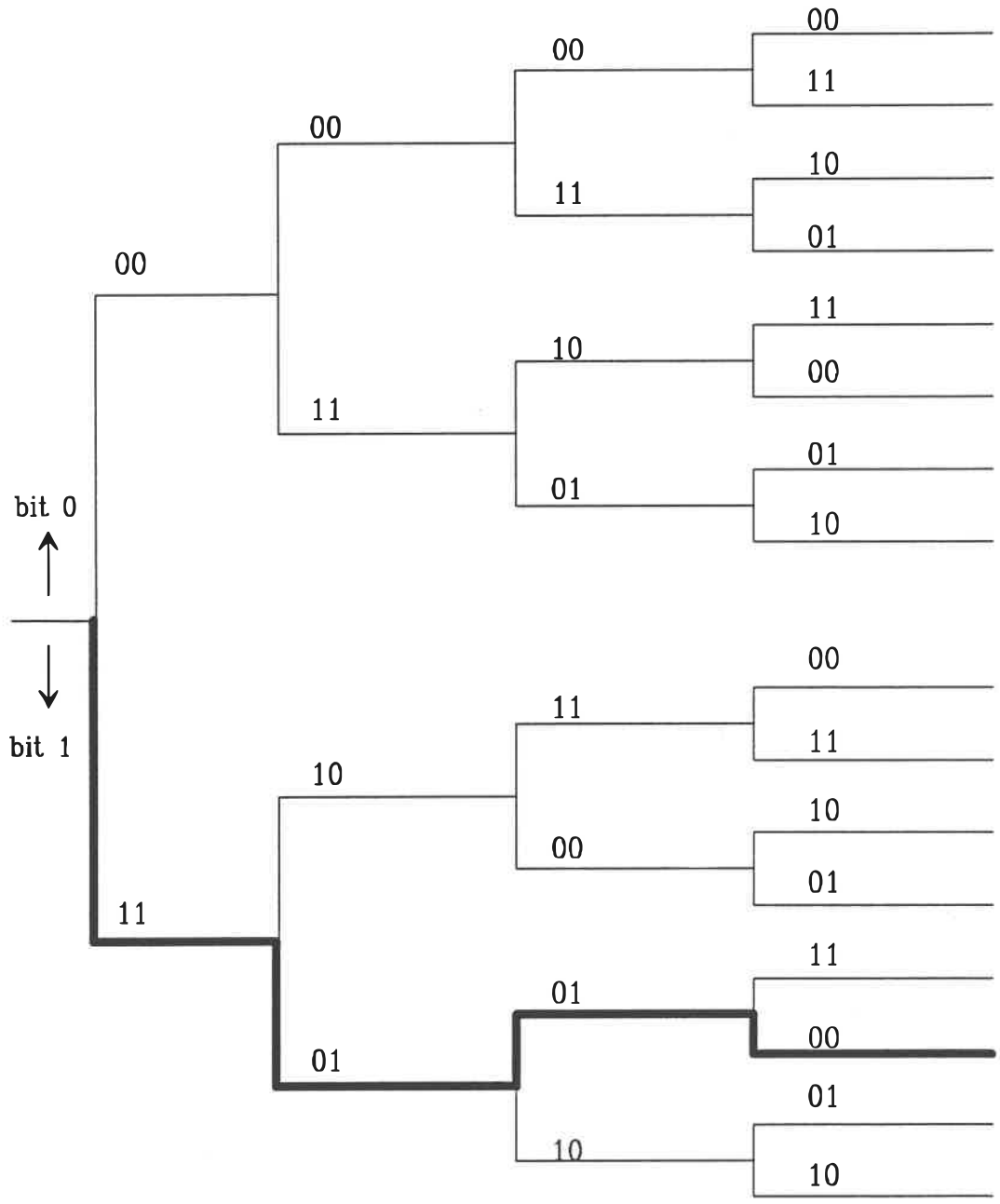


Figure 2.3 Arbre d'encodage associé au codeur de la figure 2.1

est associé à chacun des noeuds de l'arbre. L'état du noeud le plus à gauche dans l'arbre est zéro à cause de la queue de (K-1) zéros ajoutée à la fin du bloc de bits d'information avant l'encodage. De chaque noeud émergent deux branches. Chaque niveau de l'arbre correspond à l'entrée d'un bit d'information dans le codeur. Une branche vers le haut indique que le bit est 0 et une branche vers le bas indique que le bit est un 1. Chaque branche de l'arbre porte v symboles codés associés au bit d'information et à l'état du codeur lors de l'encodage du bit. Un chemin dans l'arbre spécifie une séquence codée associée à une séquence d'information U. Par exemple, le chemin en caractères gras à la figure 2.3 représente la séquence d'information U=(1,1,0,1). L'arbre a autant de niveaux qu'il y a de bits dans le bloc codé. Cette représentation est utilisée dans le décodage séquentiel.

La structure en treillis est une version plus compacte de l'arbre. En effet, dans l'arbre chaque noeud correspond à un état de l'encodeur. Or, pour une séquence de L bits, l'arbre comprend $2^{(L+K-1)}$ noeuds terminaux tandis qu'il y a $2^{(K-1)}$ états différents. Puisque L>>K, il est clair que les états de l'arbre ne sont pas tous distincts. Dans ce cas, si durant l'encodage deux séquences différentes se retrouvent au même niveau et au même état et si les bits suivants dans les deux séquences sont identiques, elles donneront les mêmes

symboles de sortie à partir de l'état de reconvergence. Ces deux parties de chemin dans l'arbre sont combinées pour obtenir un seul chemin dans le treillis. Le treillis associé à la structure en arbre de l'exemple précédent est donné à la figure 2.4. Le treillis comporte (L+K-1) niveaux. Chaque niveau a au maximum $2^{(K-1)}$ états. L'algorithme de Viterbi utilise cette représentation.

2.2 Principes de décodage optimal

Le canal utilisé pour ce mémoire est un canal discret sans mémoire (Discrete Memoryless Channel) à bruit blanc gaussien. Un tel canal a des entrées choisies parmi un alphabet de I lettres et des sorties ayant un alphabet de J lettres. Si I=2 et J=2, le canal appelé binaire symétrique (Binary Symetric Channel), est dit à quantification dure. De même, si I=2 et J>2, le canal est à quantification douce. Dans ce mémoire, un canal à quantification douce avec J=8 est utilisé. Ce canal est représenté à la figure 2.5. Le canal est caractérisé par ses probabilités de transition $P(y_j|x_i)$ qui dépendent du rapport signal-à-bruit par bit Eb/No et du taux de codage.

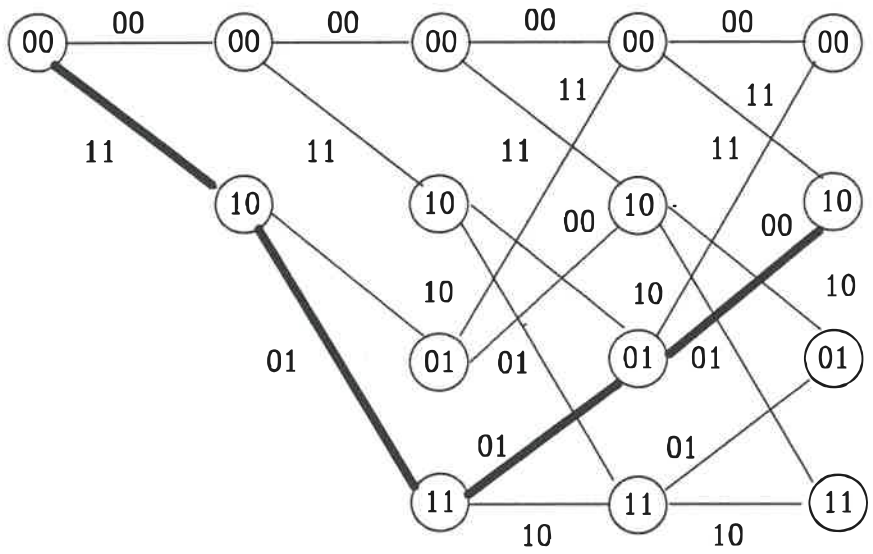


Figure 2.4 Treillis associé au codeur de la figure 2.1

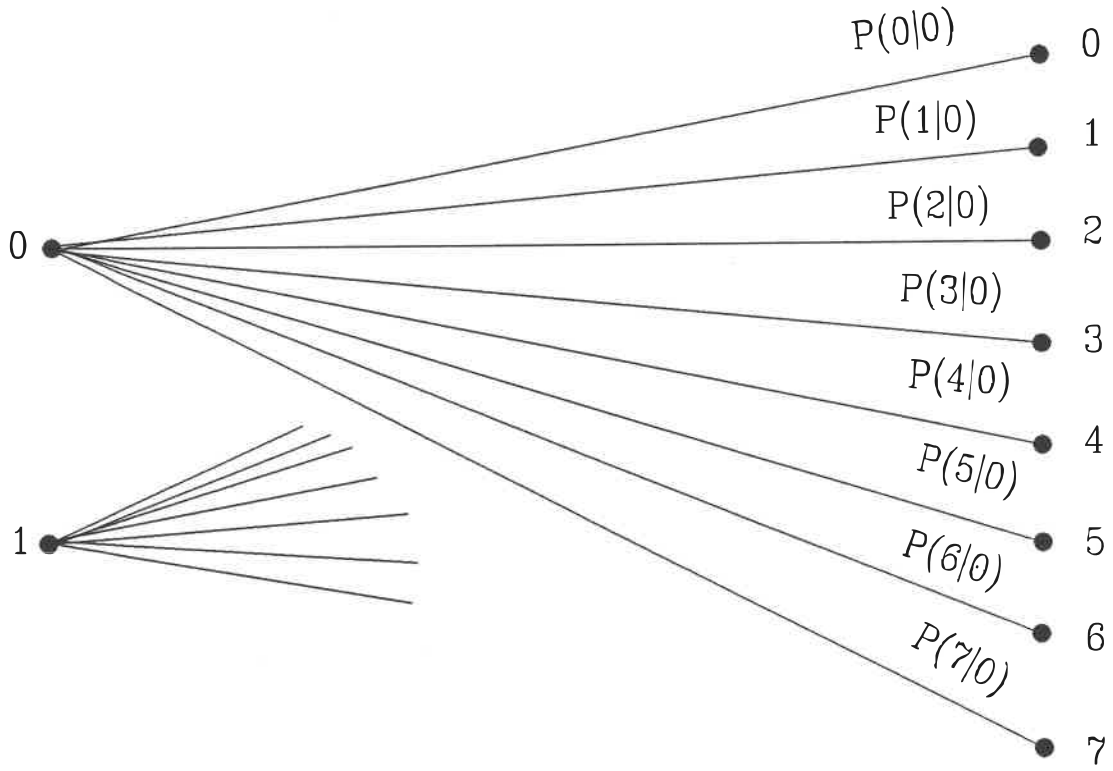


Figure 2.5 Canal discret sans mémoire
quantification 3 bits

Le problème à la réception consiste à déterminer lequel des 2^L messages d'entrée possibles a été transmis. Il est bien connu [5] que, si les entrées sont équiprobables, la probabilité d'erreur de décodage d'une séquence est minimisée si le critère de sélection du message m' est:

$$P(Y|X^{(m')}) \geq P(Y|X^{(m)}) \qquad \text{pour tout } m \neq m'$$

Le décodeur qui utilise ce critère est optimal et est appelé décodeur à maximum de vraisemblance.

Pour un canal DMC (Discrete Memoryless Channel), ce critère peut facilement s'évaluer. Supposons une séquence de bits d'information $U=(u_1,u_2,u_3,\ldots,u_L)$. Les symboles de sortie $x_i^{(U)}$ pour le i^{ieme} bit de la séquence d'entrée sont $x_i^{(U)}=(x_{i1}^{(U)},x_{i2}^{(U)},\ldots,x_{iv}^{(U)})$ tandis que la séquence reçue est donnée par $y_i=(y_{i1},y_{i2},\ldots,y_{iv})$. Dans ce cas, puisque le canal est sans mémoire, le critère s'écrit:

$$P(y_i|x_i^{(U)}) = \prod_{j=1}^{v} P(y_{ij} \mid x_{ij}^{(U)})$$

Il est plus facile d'utiliser le logarithme de la fonction $P(y_i|x_i^{(U)})$ car la mesure de vraisemblance devient additive.

Dans ce cas, la métrique de branche τ_i est définie par:

$$\tau_{i=} \sum_{i=1}^{v} \log_2 [P(y_{ij} \mid x_{ij}^{(v)})]$$

La métrique du chemin Γ est définie comme la somme de toutes les métriques de branche d'un chemin c'est-à-dire:

$$\Gamma = \sum_{i=1}^{L} \tau i$$

Le décodage optimal consiste à calculer Γ pour les 2^L chemins possibles et à choisir le message qui correspond à Γ_{max}.

2.3 Algorithme de Viterbi

L'algorithme de Viterbi est une technique de décodage optimal basée sur la structure en treillis.
Plutôt que de calculer les métriques de tous les 2^L messages possibles, il tient compte de la reconvergence des chemins. Le décodage de Viterbi consiste à considérer uniquement les chemins qui pourront peut-être donner la métrique accumulée la plus élevée. Tous les autres chemins sont éliminés par le décodeur. Par conséquent, si à un certain moment, deux séquences conduisent au même état et au même niveau, celle dont la métrique accumulée est la plus faible est éliminée. En effet, il est inutile d'examiner toutes les séquences possibles issues de cette dernière car, étant donné que les deux séquences partent du même état et du même niveau, la

différence entre la métrique la plus faible et celle plus élevée ne sera jamais compensée.

Cet algorithme procède comme suit. Soit les K-1 premières branches d'une séquence reçue, $\mathbf{Y}_{K-1}=(y_1,...,y_{K-1})$, le décodage commence par l'examen de 2^{K-1} chemins émergeant du noeud le plus à gauche et par le calcul des métriques accumulées de chaque chemin. Puisqu'il n'y a pas encore de reconvergence pour ces K-1 premières branches, il y a 2^{K-1} chemins distincts. Le décodage continue avec la réception de la $K^{ième}$ branche dont les métriques accumulées des 2^K chemins résultants sont calculées. Il y a donc 2 chemins qui reconvergent à chacun des états du treillis. Pour chaque état, le chemin ayant la métrique accumulée la plus faible est éliminé ce qui laisse 2^{K-1} chemins à prolonger. Le décodage se poursuit jusqu'à la réception de la queue de la séquence où l'extension des branches cesse puisque la valeur des bits de la queue est connue. Dans la queue, le nombre de chemins survivants diminue par deux à chaque niveau. Le décodage est fini lorsqu'il ne reste plus qu'un seul chemin qui est le chemin ayant la métrique accumulée la plus élevée. Cet algorithme nécessite $2^{(K-1)}$ comparaisons pour décoder un bit d'information. A cause de cette explosion exponentielle, l'utilisation pratique de l'algorithme de Viterbi se limite à des codes de longueur de contrainte $K \leq 7$.

2.4 Décodage séquentiel

Le décodage séquentiel est la plus puissante technique de décodage disponible en termes de performance d'erreur sur le DMC [2]. Il est dit séquentiel car il traite une seule branche à la fois. Basé sur la structure en arbre, il est sous-optimal car il n'examine qu'une faible partie de l'arbre.

Le but du décodage séquentiel est de déterminer le chemin ayant la métrique accumulée la plus élevée étant donné la séquence observée. Contrairement au décodage de Viterbi qui est une recherche exhaustive dans le treillis, le décodage séquentiel est une recherche heuristique qui fonctionne par essais et erreurs. Il étend le chemin ayant localement la métrique accumulée la plus élevée. Ce chemin est choisi parmi un ensemble de chemins situés à des niveaux différents.

Une idée fondamentale du décodage séquentiel est de biaiser la métrique de telle sorte que l'extension des chemins improbables soit minimisée. Le biais B est choisi pour que la métrique de branche soit positive le long du chemin correct et négative le long des chemins incorrects.

Fano[6] a déterminé que la valeur B=r minimise l'effort moyen de calcul par bit. La métrique utilisée dans le décodage séquentiel est donc la suivante:

$$\tau_i = \sum_{i=1}^{v} \left[\frac{\log_2 P(y_{ij} \mid x_{ij}^{(U)})}{f(y_{ij})} - B \right]$$

où $Y_j=(y_{j1},\ldots,y_{jv})$ est la séquence reçue associée à la branche transmise $X_j^{(U)}$ et f(j) est la probabilité nominale d'avoir la sortie j pour un DMC étant donné des probabilités d'entrée q(i),i=1,2,...,Q et des probabilités de transition P(j|i),j=1,2,...,J, J>Q. f(j) est donné par l'équation suivante:

$$f(j) = \sum_{i=1}^{Q} q(i)\ P(j|i)$$

Il existe deux principaux algorithmes de décodage séquentiel: l'algorithme de Fano [6] et l'algorithme de Zigangirov-Jelinek [7]. Seul ce dernier est décrit ici car c'est celui dont il est question dans ce mémoire.

Le décodeur utilisant l'algorithme de Zigangirov-Jelinek base sa recherche dans l'arbre sur une liste de chemins déjà explorés, ordonnés par ordre décroissant de leur métrique. Il est souvent appelé algorithme à pile (STACK). Le chemin ayant la métrique accumulée la plus élevée est étendu.

Chaque chemin exploré est identifié par le noeud de son extrémité.

L'algorithme peut se résumer de la façon suivante. Au début du décodage, la pile est chargée avec le noeud initial c.-à-d. le noeud le plus à gauche dans l'arbre. Sa métrique accumulée est fixée à zéro ou à une valeur arbitraire. Ensuite les étapes suivantes sont répétées:

1- Calcul de la métrique des deux successeurs du noeud ayant la métrique accumulée la plus élevée.

2- Ces successeurs sont placés au bon endroit dans la pile selon la valeur de leur métrique accumulée.

3- Le noeud qui vient d'être étendu est retiré de la pile.

4- Détermination du noeud ayant la métrique accumulée la plus élevée. Si ce noeud est un noeud terminal, le décodage est fini et le chemin décodé est récupéré à partir de ce noeud. Sinon, retour à 1.

Il faut remarquer que le chemin ayant la métrique accumulée la plus élevée à la fin du décodage n'a pas toujours été localement le chemin le plus probable. Ce

phénomène s'explique par une chute de métrique du chemin correct causée par du bruit sur le canal. Dans ce cas, tous les chemins incorrects ayant une métrique accumulée localement plus grande que celle du chemin correct sont étendus.

Un calcul est défini comme l'exécution des étapes 1 à 4 de l'algorithme. Il faut en général un calcul pour décoder un bit d'information. Cependant, dans les périodes de bruit élevé sur le canal, plusieurs calculs sont parfois nécessaires pour décoder un bit. La variabilité du nombre de calculs par bit et par bloc est due à l'extension des chemins incorrects. Il a été démontré, [8] à [12], que le nombre de calculs par bit, C_{bit}, suit asymptotiquement une loi de type Pareto :

$$P(C_{bit} > x) = \beta x^{-\alpha} \qquad 1 << x < \infty$$

où α est appelé l'exposant de Pareto et est donné par la relation:

$$r = E_o(\alpha)/\alpha$$

où $E_o(\alpha)$ est la fonction de Gallager donnée par:

$$E_o(\alpha) = -\log_2 \sum_{j=1}^{J} \left[\sum_{i=1}^{Q} q(i) \left[P(j|i) \right]^{1/(1+\alpha)} \right]^{1+\alpha}$$

ß est une constante multiplicatrice qui dépend du décodeur. La valeur de ß est déterminée à l'aide de simulations sur ordinateur. En effet, en faisant des simulations avec une pile très grande, le comportement Pareto est très bien observé pour de grandes valeurs du nombre de calculs. Lorsque la probabilité cumulative du nombre de calculs par bit est tracée sur échelle log-log en fonction du nombre de calculs, ß est l'ordonnée à l'origine de la droite qui représente le comportement pour de grandes valeurs de calcul. La figure 2.6 donne la valeur de ß pour le décodeur séquentiel utilisé (ß=.1).

Il est généralement admis [8], que le nombre de calculs par bloc, C_{bloc}, suit aussi asymptotiquement une loi de type Pareto pour de grandes valeurs de x:

$$P(C_{bloc} > x) \approx LP(C_{bit} > x) \qquad L << x < \infty$$

Il existe une probabilité non-nulle que le décodage d'un bloc nécessite l'exploration complète de l'arbre d'où la possibilité pour un décodeur d'effectuer un très grand nombre de calculs pour le décodage d'un bloc. Heureusement en pratique l'effort de calcul est limité par la taille de la pile du décodeur.

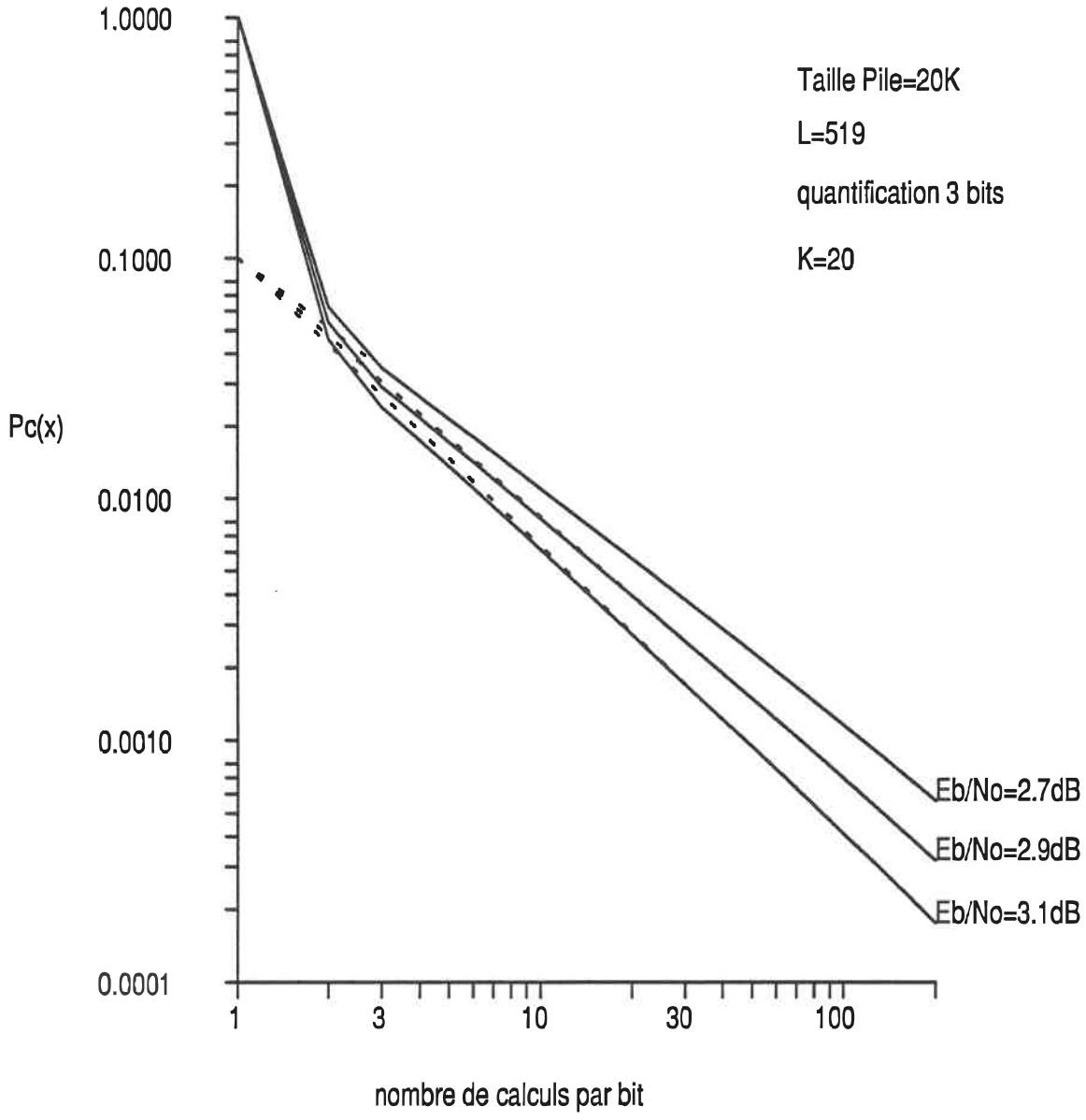


Figure 2.6 Cumulative du nombre de calculs par bit

Il a été démontré [3] que l'effort de calcul par bit suit une loi asymptotiquement Pareto bornée par le nombre maximal de calculs permis pour un bloc:

$$P(C_{bit}>x \mid C<Cmax) = \begin{cases} 1 & x<1 \\ \dfrac{\beta(x^{-\alpha} - Cmax^{-\alpha})}{(1-\beta Cmax^{-\alpha})} & 1<x<Cmax \\ 0 & x>Cmax \end{cases} \quad (2.1)$$

Rappelons que dans le cas de l'algorithme Z.-J., Cmax=(S/2) + K-1 où S est la taille de la pile en bits.

Les moments d'ordre i du nombre de calculs par bit ont été déduits de cette expression en utilisant les formules suivantes [3]:

$$Fc(x)=1-P(C>x \mid C<Cmax)$$

$$fc(x)=Fc'(x)$$

$$E[C^i]= \int x^i fc(x)dx$$

d'où

$$E[C^i]= \begin{cases} \dfrac{1}{1-\beta Cmax^{-\alpha}} \dfrac{\alpha\beta}{i-\alpha}(Cmax^{i-\alpha} - 1^{i-\alpha}) + 1-\beta & \alpha>i \\ Ln\ (Cmax) + 1-\beta & \alpha=i \end{cases}$$

Il a été démontré que si Cmax=∞, le nombre moyen de calculs par bit est borné si $\alpha>1$ tandis que la variance l'est pour

$\alpha>2$. Le taux de coupure Rcomp est défini comme étant le taux de codage pour lequel $\alpha=1$ c.-à-d. $Rcomp=E_0(1)$. Pour borner le nombre moyen de calculs du décodeur séquentiel, il faut donc travailler à un taux de codage $r < Rcomp$. Dans cette région, la moyenne est bornée mais la variance ne l'est pas. Par contre, pour $Cmax < \infty$ tous les moments sont bornés de sorte que la divergence de l'effort de calcul du décodeur est prévenue par un échec de décodage. Dans la procédure ARQ/FEC, cet échec de décodage est contourné par la retransmission des blocs qui débordent.

2.5 Problèmes liés à la variabilité du nombre de calculs

La variabilité du nombre de calculs par bit est la source de deux problèmes majeurs du décodeur séquentiel:

- le nombre moyen de calculs peut devenir excessif.
- le délai de décodage est variable.

Pour borner le nombre moyen de calculs du décodeur séquentiel, il suffit de travailler à un taux de codage $r < Rcomp$. Pour une efficacité maximale, r est choisi de quelques pourcents inférieur à Rcomp. Puisque l'algorithme de Viterbi ne souffre pas de la variabilité du temps de calcul du décodeur séquentiel , il peut fonctionner à un taux de

codage plus élevé. Dans le cas du décodage de Viterbi, le taux de codage est limité à la capacité du canal, C. La capacité est définie comme:

$$C = \lim_{\alpha \to 0} \frac{dEo(\alpha)}{d(\alpha)} \quad \text{bits/symbole}$$

En conclusion, le décodeur séquentiel fonctionne correctement si r<Rcomp<C tandis que le décodeur de Viterbi fonctionne correctement si r<C.

Pour s'accomoder du délai de décodage variable, il faut mettre un tampon à l'entrée et à la sortie du décodeur. Le tampon à l'entrée sert à accumuler les blocs venant du canal lorsque le décodeur séquentiel est déjà occupé. Le tampon à la sortie sert à régulariser le débit de l'information envoyée à l'usager. Il peut aussi servir à conserver certains blocs s'il y a une remise en ordre des blocs avant l'envoi de l'information à l'usager.

Malheureusement, la variabilité du temps de calcul cause des débordements soit de la pile du décodeur séquentiel (STACK OVERFLOW) ou du tampon d'entrée (BUFFER OVERFLOW). Ce dernier événement est catastrophique car il cause des effacements et une perte de synchronisation. C'est pour cette raison que l'information est divisée en blocs de quelques centaines à quelques milliers de bits. Malgré tout, le

débordement du tampon à l'entrée entraîne la perte de plusieurs blocs non décodés et une perte de synchronisation tandis que le débordement de la pile du décodeur séquentiel entraîne seulement la perte du bloc courant. Il existe un compromis entre ces deux phénomènes. En effet, il a été démontré [9] que si le gain de vitesse est suffisamment grand, la probabilité de débordement du tampon d'entrée, P_{∞}, lors du décodage du premier bit d'un bloc est donnée par:

$$P_{\infty} = P(C > BU)$$

où B:capacité du tampon d'entrée en branches
U: gain de vitesse du décodeur séquentiel

D'après ce résultat, Pau et Haccoun [3] ont montré à l'aide de l'équation 2.1, que la probabilité de débordement du tampon d'entrée augmente avec la taille de la pile. De plus la probabilité de débordement de la pile, P_d est donnée par:

$$P_d = P(C>Cmax) = \beta L(Cmax)^{-\alpha}$$

On constate que la probabilité de débordement de la pile diminue avec l'augmentation de Cmax. Il est clair que si l'on souhaite diminuer les débordements du tampon d'entrée, il faut limiter la taille de la pile et par conséquent accepter

les débordements de la pile. C'est un compromis entre le pourcentage de débordement du tampon d'entrée et le débit efficace.

2.6 Performance d'erreur

Une borne supérieure pour la probabilité d'erreur sur l'ensemble des codes a été donnée par Viterbi. Lorsque r < Rcomp:

$$P(E) < L2^{-KRcomp/r}$$

où L est la longueur en bits du bloc d'information

r est le taux de codage

Rcomp est le taux de coupure du canal $Rcomp=E_o(1)$.

Pour diminuer la probabilité d'erreur, P(E), il faut soit augmenter K ou diminuer r.

2.7 Décodage séquentiel versus décodage de Viterbi

A la lumière des résultats des sections précédentes, il est possible de comparer les décodeurs séquentiel et Viterbi en termes de variabilité du temps de calcul, de performance d'erreur et de complexité.

Le nombre de calculs nécessaires pour décoder un bit par l'algorithme de Viterbi est constant. Cependant, il est relié au nombre d'états 2^{K-1}. Par conséquent, le temps de

calcul augmente de façon exponentielle avec la longueur de contrainte du code. Pour cette raison, le décodeur de Viterbi est limité à des codes K<8 ce qui implique que la probabilité d'erreur est limitée au pouvoir de correction de codes de complexité faible.

Par opposition au Viterbi, le décodeur séquentiel a un temps de calcul variable. Le principal avantage du décodage séquentiel est que si le nombre moyen de calculs par bit est fini, il est largement inférieur à 2^{K-1}. Donc, en moyenne, l'effort de calcul est moins grand. De plus, le nombre moyen de calculs par bit est indépendant de la longueur de contrainte du code. Une probabilité d'erreur très petite peut être atteinte sans pour autant augmenter le nombre moyen de calculs par bit.

Du point de vue du coût de réalisation, la complexité du décodage de Viterbi augmente exponentiellement avec K tandis que la complexité du décodeur séquentiel varie linéairement avec K. Le choix entre un décodeur séquentiel et un décodage de Viterbi est donc un compromis entre la variabilité de temps de calcul et la performance d'erreur.

Il faut ajouter que la variabilité du nombre de calculs et une de ses conséqences, le débordement de la pile du décodeur séquentiel, peuvent être mis à profit. En effet, un bloc qui fait déborder la pile est un bloc très bruité qui aurait été décodé avec des erreurs par un décodeur optimal.

Le décodeur séquentiel permet donc non seulement de corriger les erreurs mais aussi de les détecter.

2.8 Introduction aux files d'attente

Le décodeur séquentiel avec son tampon à l'entrée peut être considéré comme une file d'attente dont le serveur est le décodeur séquentiel. Ce mémoire développe une analyse préliminaire d'un multidécodeur. Ce dernier est constitué d'un certain nombre s de décodeurs séquentiels en parallèle, d'un tampon à l'entrée et d'un tampon à la sortie. Le multidécodeur peut aussi être considéré comme étant une file d'attente à s serveurs. Les sections qui suivent donnent quelques résultats sur les files d'attente qui nous permettront de faire une analyse plus poussée du système.

2.8.1 Notation

Il existe une notation particulière aux files d'attente. Dans ce mémoire, cette notation est limitée aux cas où le tampon est infini et où la population est infinie. Les files d'attente sont notées avec les paramètres A/B/s [13] où

A représente la distribution du temps d'arrivée entre deux blocs; B représente la distribution de la loi de service. Il

est supposé que les temps d'arrivée sont indépendants. De même, il est supposé que les temps de service sont indépendants. A ou B prennent des valeurs notées M ce qui implique que le temps d'arrivée (de départ) entre deux blocs est distribué exponentiellement, alors que D indique que le temps entre deux blocs qui arrivent (qui partent) est constant, enfin G indique que la distribution est générale. Quant à s, il représente le nombre de serveurs du système.

D'après cette notation, le système étudié est une file d'attente D/G/s. C'est un cas particulier de la file d'attente G/G/s. Les sections suivantes donnent les résultats permettant de solutionner partiellement les files d'attente G/G/1 et G/G/s.

2.8.2 Files d'attente G/G/1 et G/G/s

Dans le but de solutionner les files d'attente G/G/1, le système est décrit par certaines équations à l'aide du diagramme de temps du système (évolution dans le temps du système) qui mènent à l'établissement de l'intégrale de Lindley's. Cette intégrale est du type Wiener-Hopf. Elle est solutionnée par une méthode spectrale. Cependant, cette méthode nécessite la connaissance de la transformée de Laplace de la loi de service. Or, comme il sera vu au

chapitre 3, cette transformée n'existe pas pour la loi de service du décodeur séquentiel.

Par contre, on peut obtenir une approximation pour la distribution de la file d'attente à l'entrée, appelée "Heavy Traffic Approximation". Dans cette approximation, la distribution du temps d'attente, W(y), est donnée par [13]:

$$W(y) \approx 1-\exp\left[\frac{-2t_m(1-\rho)y}{\sigma_a^2 + \sigma_b^2}\right]$$

où t_m=temps interarrivée entre deux blocs

$\rho = x_s/t_m$ où x_m est le temps moyen de service

σ_a^2= variance du temps d'interarrivée

σ_b^2= variance de la loi de service

Cette expression est valide pour $\rho < 1$ mais $\rho \approx 1$ et lorsque

$$2(1-\rho)\ t_m << \sigma_a^2 + \sigma_b^2$$

Un résultat similaire est donné par l'approximation de Kingman-Kollerstom pour la file d'attente G/G/s [14, 15]:

$$W(y) \approx 1-\exp\left[\frac{-y}{Wmoy}\right] \qquad (2.2)$$

où

$$Wmoy = \frac{\sigma_a^2 + \sigma_b^2/s^2}{2tm(1-\rho)} \qquad (2.3)$$

avec s est le nombre de serveurs

Wmoy: temps moyen d'attente dans le tampon d'entrée

Il faut noter que W(y) donne le temps d'attente dans le tampon d'entrée plutôt que le nombre de blocs qui s'y trouvent. Cependant, si l'unité de temps est choisie comme étant l'intervalle entre l'arrivée de deux blocs au tampon d'entrée alors W(y) donnera le nombre de blocs dans le tampon d'entrée. En effet, dans ce cas, la probabilité que le temps d'attente soit x fois le temps entre l'arrivée de deux blocs est égale à la probabilité qu'il y ait x blocs dans le tampon d'entrée.

La principale limitation de cette approximation est qu'elle est obtenue en faisant l'hypothèse que le tampon à l'entrée est infini. Pour cette raison, il est impossible de calculer le pourcentage de débordement du tampon d'entrée. Cette expression nous sert plutôt à déterminer la répartition des blocs dans le tampon d'entrée.

CHAPITRE 3

LOI DE SERVICE DU DECODEUR SEQUENTIEL

Le but de ce mémoire est d'analyser les performances d'un multidécodeur constitué de plusieurs décodeurs séquentiels travaillant en parallèle. Ce système est une file d'attente D/G/s. Il faut considérer les blocs qui arrivent dans le système plutôt que les bits car tous les bits d'un même bloc doivent aller au même décodeur.

L'analyse de ce système nécessite la connaissance de la loi de service du décodeur séquentiel ou du moins de ses deux premiers moments, la moyenne et la variance. Notre connaissance de cette loi étant très partielle, ce chapitre est consacré à l'étude de la loi de service du décodeur séquentiel.

3.1 Généralisation de la loi de service par bit

Il a été vu au chapitre précédent que le nombre de calculs par bit suit asymptotiquement une loi de Pareto étant bornée par le nombre maximal de calculs donnée par (2.1).

Il semble raisonnable que le nombre de calculs par bloc suive une loi similaire à celle par bit à un facteur L près. En généralisant l'expression de Pau [3], la fonction de répartition par bloc est donnée par:

$$P(C_{bloc}>x \mid C<Cmax) = \begin{cases} 1 & x<L \\ \dfrac{\beta L(x^{-\alpha} - Cmax^{-\alpha})}{(1-\beta L Cmax^{-\alpha})} & L\leq x\leq Cmax \\ 0 & x>Cmax \end{cases} \qquad (3.1)$$

tandis que pour $\alpha > i$, les moments d'ordre i du nombre de calculs par bloc sont donnés par:

$$E[C^i]= \frac{1}{1-\beta L Cmax^{-\alpha}} \frac{\alpha L \beta}{i-\alpha} (Cmax^{i-\alpha} - L^{i-\alpha}) + (1-\beta L^{i-\alpha}) L^i \qquad (3.2)$$

et comme il a été vu au chapitre précédent, la probabilité de débordement, P_d, peut être exprimée par:

$$P_d = P(C_{bloc} > Cmax) = \beta L (Cmax)^{-\alpha} \qquad (3.3)$$

et par conséquent le pourcentage de débordement, %deb:

$$\%deb = P_d \times 100 = \beta L (Cmax)^{-\alpha} \times 100 \qquad (3.4)$$

En regardant (3.1) de plus près, on constate la présence d'une discontinuité à x=L tandis que la courbe est continue pour x>L et en particulier pour x=Cmax. Or, étant donné qu'en pratique le nombre de calculs est limité à Cmax à cause de la pile, tous les blocs occasionnant Cmax calculs ou plus, devraient être considérés dans le modèle comme faisant Cmax

calculs. Par conséquent, il devrait y avoir une discontinuité à Cmax plutôt qu'à L. L'équation 3.1 ne rend pas compte de ce phénomène. Il y a donc de fortes chances pour que (3.1) ne soit pas une bonne expression pour la cumulative du nombre de calculs par bloc.

Une série de simulations à l'ordinateur ont donné des valeurs empiriques de la moyenne et de la variance du nombre de calculs par bloc ainsi que le pourcentage de débordement d'un décodeur particulier pour différentes tailles de pile et différents rapports signal-à-bruit. Chacune des simulations comprenant 125000 blocs de 519 bits a été effectuée avec un code de Johannesson de taux 1/2 ayant K=20 en quantification 3 bits [16]. Les moyennes, variances et pourcentages de débordement sont donnés au tableau 3.1. Ces résultats peuvent être comparés aux moyennes et aux variances par bloc calculées par (3.2) et aux pourcentages de débordement calculés par (3.4). Ces résultats sont présentés au tableau 3.2. Il faut noter que les moyennes et les variances présentées dans les tableaux de ce mémoire sont renormalisées. Ainsi, les moyennes inscrites dans les tableaux sont égales aux moyennes calculées divisées par L tandis que les variances sont égales aux variances calculées divisées par L^2. La normalisation a aussi été faite pour les résultats de simulation. Cette normalisation implique que le calcul n'est plus défini comme le calcul d'un bit mais plutôt

Taille pile (bits)	Paramètres d'intérêt	Rapport signal-à-bruit Eb/No (dB)		
		2.7 $\alpha = 1.03$	2.9 $\alpha = 1.145$	3.1 $\alpha = 1.265$
2000	moyenne	1.41	1.32	1.25
	%deb	15.4	9.3	5.3
	variance	.093	.076	.0564
	Cmax	1019	1019	1019
3000	moyenne	1.51	1.38	1.28
	%deb	7.39	4.06	2.13
	variance	0.278	0.193	0.126
	Cmax	1519	1519	1519
4000	moyenne	1.56	1.41	1.29
	%deb	4.8	2.52	1.3
	variance	0.487	0.313	0.192
	Cmax	2019	2019	2019
5000	moyenne	1.6	1.43	1.30
	%deb	3.5	1.81	0.91
	variance	0.7	0.432	0.256
	Cmax	2519	2519	2519
20000	moyenne	1.81	1.53	1.35
	%deb	0.736	0.33	0.15
	variance	4.27	2.18	1.07
	Cmax	10019	10019	10019

Tableau 3.1 Moyennes, variances et pourcentages de débordement obtenus par les simulations, L=519, K=20

Taille pile (bits)	Paramètres d'intérêt	Rapport signal-à-bruit Eb/No (dB)		
		2.7 $\alpha = 1.03$	2.9 $\alpha = 1.145$	3.1 $\alpha = 1.265$
2000	moyenne	1.02	1.008	1.004
	%deb	4.14	1.87	.813
	variance	.0089	.0045	.0022
	Cmax	1019	1019	1019
3000	moyenne	1.04	1.02	1.008
	%deb	2.74	1.18	0.49
	variance	.0362	.0180	.0085
	Cmax	1519	1519	1519
4000	moyenne	1.052	1.025	1.012
	%deb	2.05	.853	.342
	variance	.0747	.0362	.0169
	Cmax	2019	2019	2019
5000	moyenne	1.066	1.032	1.0147
	%deb	1.62	.6619	.259
	variance	0.1202	.0573	.0261
	Cmax	2519	2519	2519
20000	moyenne	1.164	1.0724	1.031
	%deb	.3929	0.1362	.0457
	variance	1.0395	.4375	.1732
	Cmax	10019	10019	10019

Tableau 3.2 Moyennes, variances et pourcentages de débordement obtenus par (3.2) et (3.4), L=519, K=20

comme le calcul d'un bloc. Ce dernier est égal à L fois le calcul d'un bit. D'après ces deux tableaux, il est clair que la loi de Pareto ne peut pas être considérée comme une bonne approximation de la loi de service par bloc pour de faibles valeurs du nombre de calculs. Il faut donc trouver une autre solution.

3.2 Fonction de densité du nombre de calculs par bloc

Une autre avenue possible est d'obtenir la fonction de densité du nombre de calculs par bloc grâce à des simulations et de réaliser une régression avec écart quadratique minimal sur ces courbes. La connaissance du comportement asymptotique de la loi peut donner une très bonne idée de l'expression générale de la fonction de densité par bloc. Les figures 3.1 et 3.2 donnent les fonctions de densité tirées de la simulation de 125000 blocs pour L=319, L=519 et 3 rapports signal-à-bruit différents. Ces simulations ont été faites avec un code de Johannesson dont les vecteurs générateurs sont $G^{(1)}$=2142513 et $G^{(2)}$=3276177 de taux 1/2 ayant K=20 en quantification 3 bits [16], pour différentes tailles de pile. Les moyennes, variances et pourcentage de débordement pour L=319 sont donnés au tableau 3.3. Les résultats pour L=519 ont déjà été présentés au tableau 3.1.

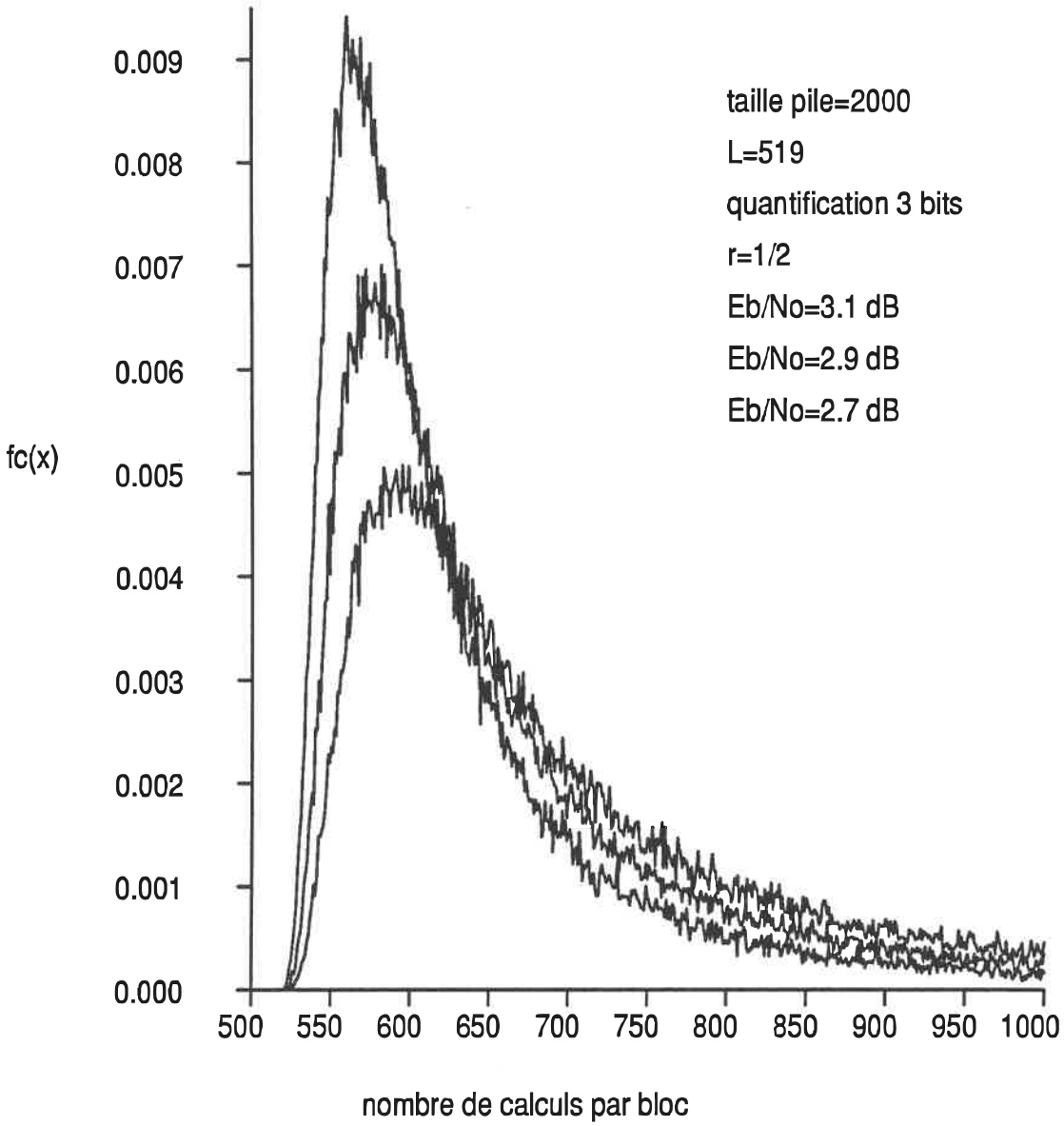


Figure 3.1 Fonction de densité du nombre de calculs par bloc pour L=519

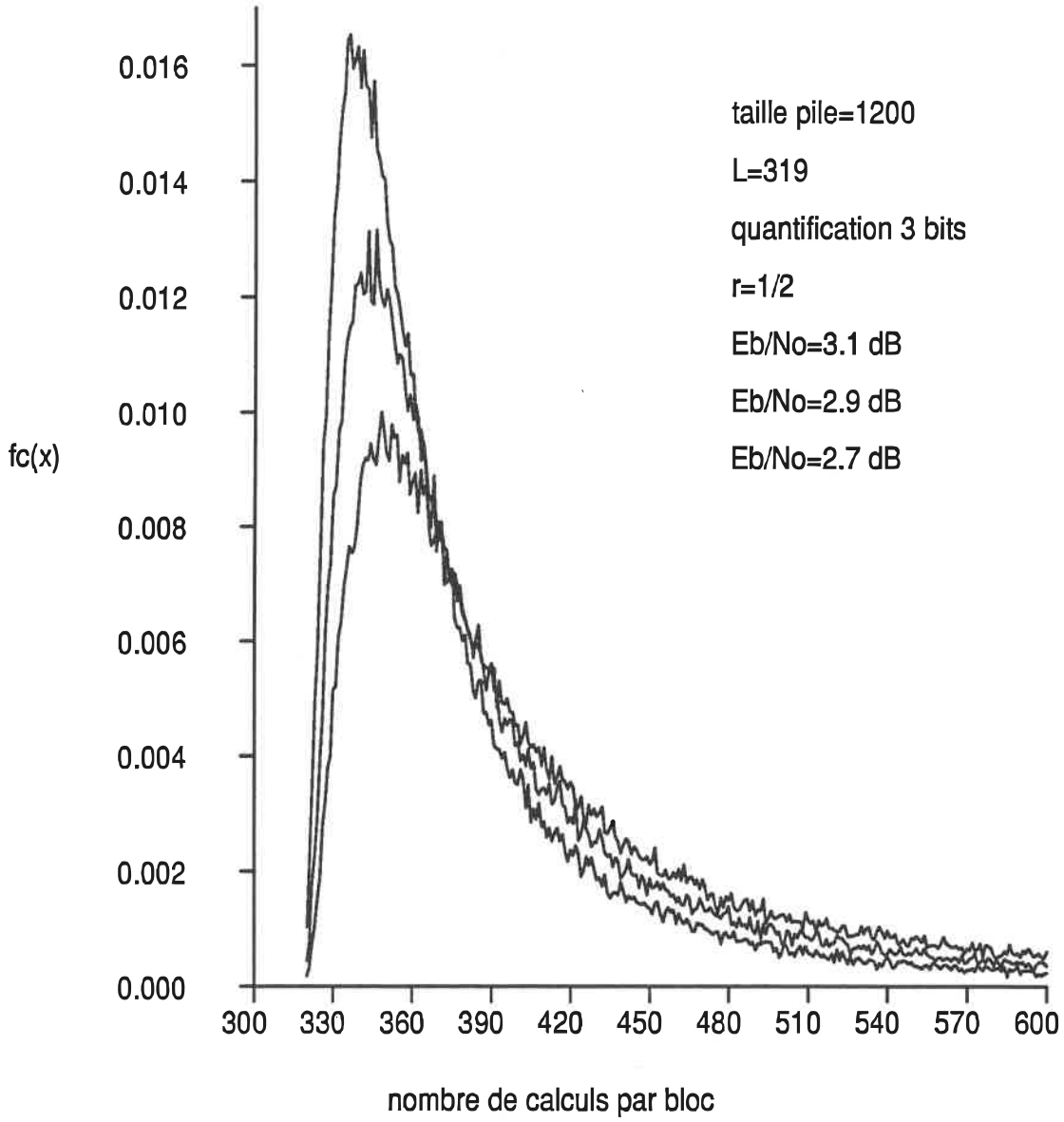


Figure 3.2 Fonction de densité du nombre de calculs par bloc pour L=319

Taille pile (bits)	Paramètres d'intérêt	Rapport signal-à-bruit Eb/No (dB)		
		2.7 $\alpha = 1.03$	2.9 $\alpha = 1.145$	3.1 $\alpha = 1.265$
2400	moyenne	1.49	1.36	1.27
	%deb	4.48	2.5	1.36
	variance	.447	.298	.191
	Cmax	1219	1219	1219
12000	moyenne	1.72	1.48	1.33
	%deb	.688	.346	.166
	variance	3.93	2.09	1.12
	Cmax	6019	6019	6019

Tableau 3.3 Moyennes, variances et pourcentages de débordement obtenus par les simulations, L=319, K=20

Les premières constatations sont les suivantes:

- les fonctions de densité varient selon le rapport signal-à-bruit Eb/No, la longueur des blocs L, la longueur de contrainte du code K et la taille de la pile S.

- la taille de la pile n'affecte pas la forme de la fonction de densité. Elle n'affecte que la borne supérieure de la courbe et la valeur de l'impulsion de normalisation.

- la valeur de cette impulsion représente la probabilité de débordement de la pile. Plus la pile est grande, plus la valeur de l'impulsion est petite.

3.2.1 Expression générale de la fonction de densité

L'expression générale de la fonction de densité est déduite à partir des résultats précédents. Grâce à la fonction de densité, les moyennes, les variances et les probabilités de débordement pourront être calculées et

comparées avec celles obtenues par simulation. A partir des résultats des figures 3.1 et 3.2, il apparaît que:

1- La fonction de densité du nombre de calculs par bloc commence à x=L.

2- Une expression analytique de la fonction de densité contenant une puissance de (x-L) au numérateur semble possible car la fonction est nulle à x=L.

3- A Cmax, la fonction a une impulsion dont la valeur donne la probabilité de débordement de la pile du décodeur.

De plus, l'inclusion au dénominateur de l'expression analytique d'une puissance de x permet de reproduire le comportement asymptotiquement Pareto pour x très grand. Les observations précédentes conduisent à une forme générale de la fonction de densité exprimée par l'équation suivante:

$$fc(x) = \frac{K_1 (x-L)^{n1}}{(K_2 + (x-L)^{n2})} \qquad L \leq x < Cmax$$

$$= \int_{Cmax}^{\infty} \frac{K_1 (x-L)^{n1}}{(K_2 + (x-L)^{n2})} dx \qquad x = Cmax$$

$$= 0 \qquad \text{ailleurs}$$

où les paramètres K_1, K_2, n_1 et n_2 sont à déterminer.

Les résultats obtenus à la section suivante sont déduits de cette expression générale. Ils sont basés sur une détermination heuristique de certains paramètres. Les régressions avec écart quadratique minimal ont été faites à partir de résultats de simulations obtenus avec une pile de 20000 bits pour L=519 et une pile de 12000 bits pour L=319. Le critère de l'écart quadratique minimal a été choisi pour évaluer la validité des résultats.

3.2.2 Détermination des paramètres

Puisque le comportement asymptotique de la fonction de répartition est de type Pareto, il semble logique de poser $n_2=n_1+\alpha+1$. Il faut noter que K_1 n'est pas nécessairement égal à $\alpha\beta L$ comme le comportement asymptotique porterait à le penser. Ce dernier déduit par la théorie ne tient pas compte du fait que la fonction de densité réelle commence à x=L.

En effet,

$$fc(x) \approx \alpha\beta L \left(1 / x^{\alpha+1} \right) \qquad \text{en théorie}$$

plutôt que

$$fc(x) \approx \alpha\beta L \left(1 / (x-L)^{\alpha+1} \right)$$

Il reste trois paramètres à déterminer: K_1, K_2 et n_1. Les paramètres K_2 et K_1 sont évalués en fonction de n_1. K_2 est déterminé grâce à l'abscisse du point maximal, x_m, pour une valeur donnée n_1. En effet, en cherchant l'abscisse du point maximal de fc(x), on a:

$$\frac{dfc(x)}{dx}\Big|_{x_m} = 0$$

alors

$$K_1 n_1 (x_m - L)^{n1-1} (K_2 + (x_m - L)^{n1+\alpha+1} =$$
$$K_1 (x_m - L)^{n1} (n_1+\alpha+1) (x_m - L)^{n1+\alpha}$$

d'où l'expression suivante peut être obtenue pour K_2:

$$K_2 = \left[\frac{n_1+\alpha+1}{n_1} - 1 \right] (x_m - L)^{n1+\alpha+1}$$

Le paramètre K_1 est déterminé à l'aide d'une propriété fondamentale des fonctions de densité: l'aire sous la courbe

doit être égale à 1. Les tables d'intégrales nous donnent l'intégrale suivante et sa solution:

$$I = \int_0^\infty \frac{x^m\,dx}{x^n + a^n} = \frac{\pi\, a^{m+1-n}}{n \sin[(m+1)\pi/n]} \qquad 0<m+1<n$$

En posant x=z-L, on a:

$$I = \int_L^\infty \frac{(z-L)^m\,dz}{(z-L)^m + a^m}$$

Or, fc(x) étant une fonction de densité on peut écrire:

$$I \times K_1 = \int_L^\infty fc(x)\,dx = 1$$

où $m = n_1$

$n = n_2 = n_1+\alpha+1$

$a = (K_2)^{1/(n1+\alpha+1)}$

Cette expression nous permet d'isoler K_1 en fonction des paramètres K_2 et n_1 (i.e x_m et n_1 car $K_2=f(x_m,n_1)$):

$$K_1 = \frac{(n_1+\alpha+1)}{\pi}\, K_2^{\,\alpha/(n1+\alpha+1)} \sin\left[\frac{(n_1+1)}{n_1+\alpha+1}\,\pi\right]$$

L'écart quadratique entre les résultats théoriques et les résultats de simulation a été minimisé en fonction de n_1. Plus précisément, l'écart quadratique entre la probabilité théorique et celle obtenue par simulation a été calculé pour chaque valeur du nombre de calculs. Ces valeurs ont ensuite

été additionnées de x=L à x=20L. La somme a été minimisée en fonction de n_l. Les paramètres des régressions pour L=519 et L=319 en fonction de différents rapports signal-à-bruit sont donnés au tableau 3.4 tandis que le tableau 3.5 donne les paramètres pour Eb/No=2.7dB et L=319, 419, 519 et 619. Dans le cas L=519, la valeur n_l=1.25 s'est avérée celle qui minimise l'écart quadratique pour les trois rapports signal-à-bruit examinés. Les figures 3.3 à 3.5 donnent le résultat de la régression (trait plein) en comparaison avec le résultat de simulation (trait haché). L'échelle du graphique est limitée à Cmax=1000 et l'impulsion à Cmax n'a pas été représentée pour des raisons de visibilité. En effet, pour Cmax=1000, P_d=.154 ce qui est plus de trois fois l'ordonnée du point maximal de la courbe représentée. Les moyennes, variances et pourcentage de débordement sont donnés au tableau 3.6. Ils sont calculés à partir des équations suivantes:

$$Cmoy = \int_{L}^{Cmax_l} x \; fc(x)dx + Cmax_l \; P_d \qquad (3.5)$$

$$\sigma^2 = \int_{L}^{Cmax_l} x^2 \; fc(x)dx + (Cmax_l)^2 \; P_d - Cmoy^2 \qquad (3.6)$$

Paramètres	L=519 (bits)			L=319 (bits)		
	Eb/No (dB)			Eb/No (dB)		
	2.7 $\alpha = 1.03$	2.9 $\alpha = 1.145$	3.1 $\alpha = 1.265$	2.7 $\alpha = 1.03$	2.9 $\alpha = 1.145$	3.1 $\alpha = 1.265$
xmax	598	581	568	354	347	341
n1	1.25	1.25	1.25	.81	.76	.65
K1	91.33	127.54	172.29	44.62	59.72	77.90
K2	2721466	2087791	1581967	60837	45145	28531
écart	1.9E-5	2.2E-5	2.5E-5	3.4E-5	4.3E-5	5.3E-5

Tableau 3.4 Paramètres des rgressions pour L=519 et L=319 pour différents rapports signal-à-bruit

Paramètres	Eb/No=2.7dB $\alpha = 1.03$			
	L (bits)			
	319	419	519	619
xmax	354	474	598	724
n1	.81	1.03	1.25	1.5
K1	44.62	66.13	91.33	117.59
K2	60837	417031	2721466	18.458E6
écart	3.4E-5	2.7E-5	1.9E-5	1.7E-5

Tableau 3.5 Paramètres des régressions pour Eb/No=2.7dB et L=319, 419, 519, 619

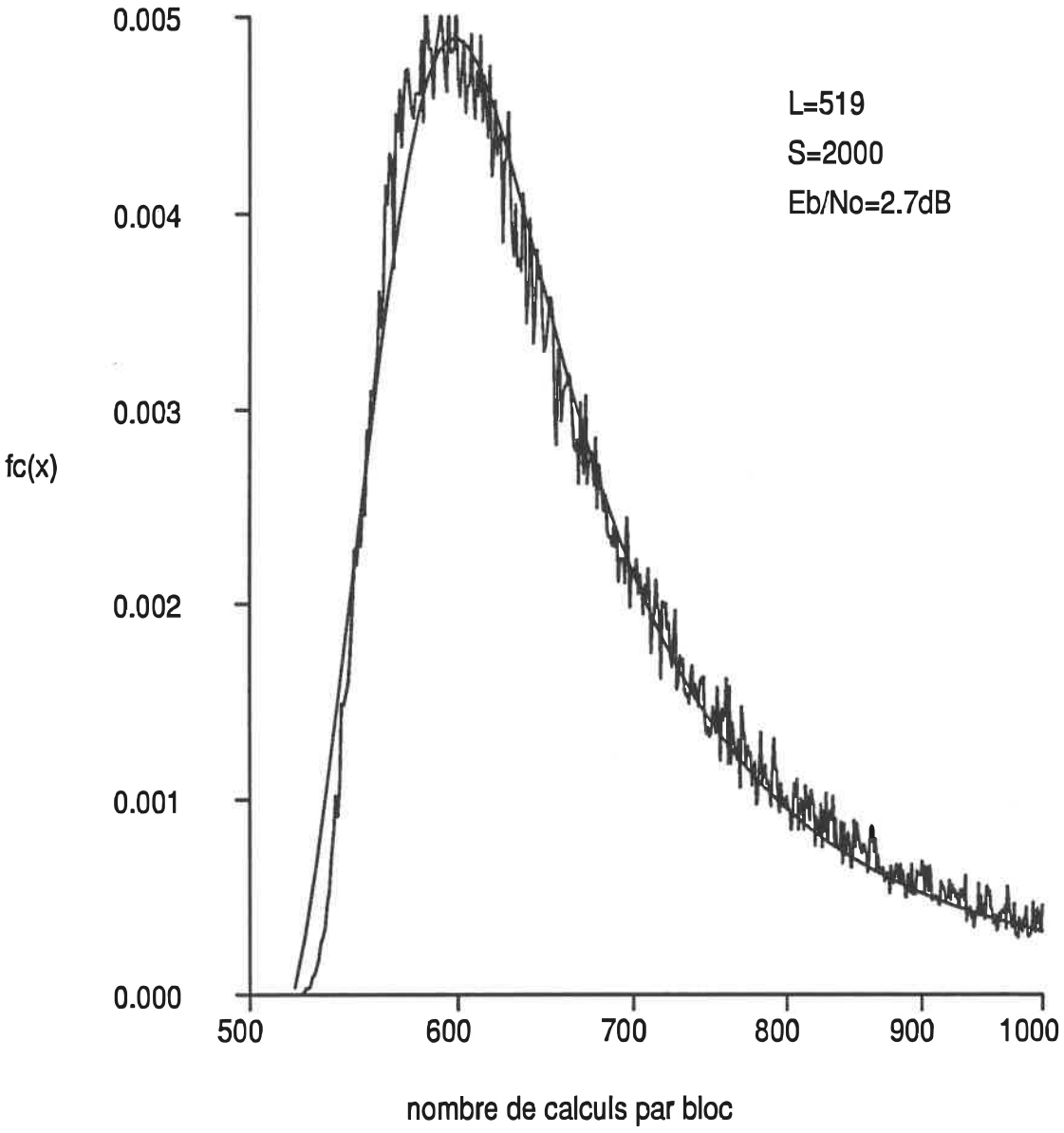


Figure 3.3 Comparaison de la fonction de densité par bloc tirée des simulations et de celle obtenue par la régression, n_1=1.25, x_m= 598

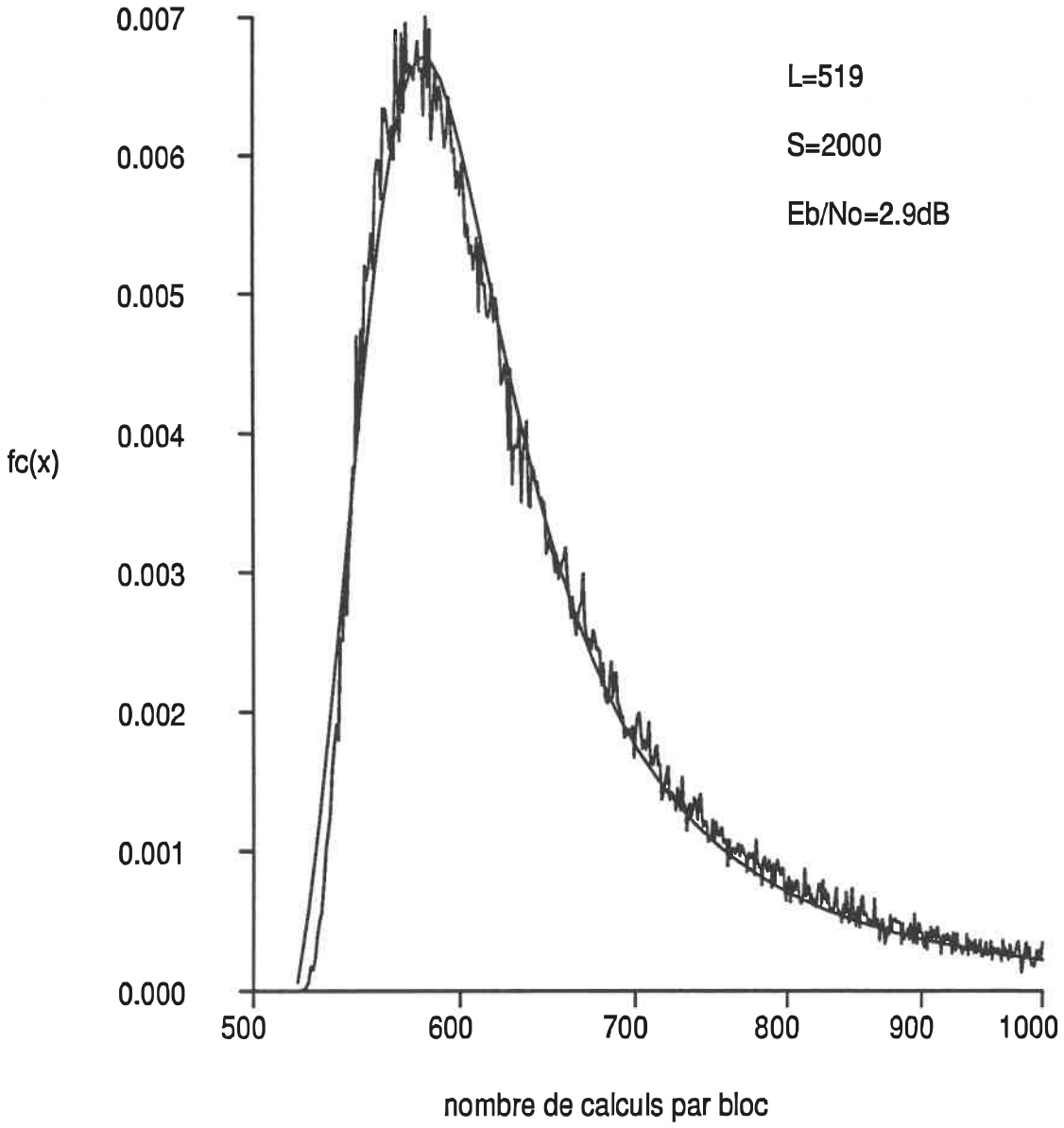


Figure 3.4 Comparaison de la fonction de densité par bloc tirée des simulations et de celle obtenue par la régression, n_1=1.25, x_m=581

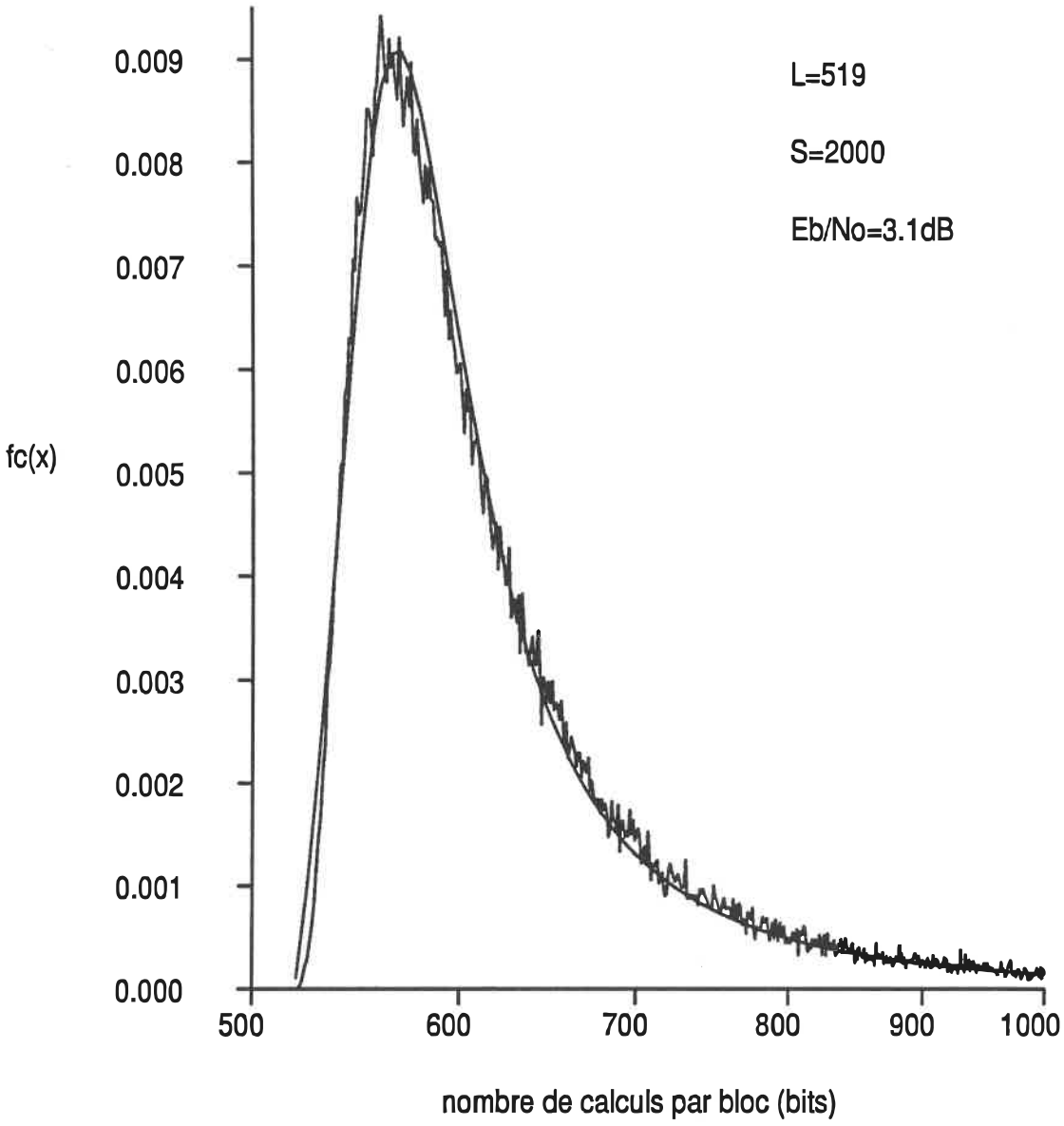


Figure 3.5 Comparaison de la fonction de densité par bloc tirée des simulations et de celle obtenue par la régression, n_1=1.25, x_m=568

Taille pile (bits)	Paramètres d'intérêt	Rapport signal-à-bruit Eb/No (dB)		
		2.7 $\alpha = 1.03$	2.9 $\alpha = 1.145$	3.1 $\alpha = 1.265$
2000	moyenne	1.4	1.31	1.24
	%deb	14.7	9.04	5.25
	variance	0.092	0.075	0.056
	Cmax	1019	1019	1019
3000	moyenne	1.49	1.37	1.25
	%deb	7.2	4.09	2.18
	variance	0.275	0.195	0.128
	Cmax	1519	1519	1519
4000	moyenne	1.55	1.4	1.29
	%deb	4.74	2.57	1.31
	variance	0.482	0.318	0.196
	Cmax	2019	2019	2019
5000	moyenne	1.59	1.42	1.30
	%deb	3.53	1.85	0.91
	variance	0.7	0.44	0.259
	Cmax	2519	2519	2519
20000	moyenne	1.80	1.52	1.34
	%deb	0.71	0.312	0.13
	variance	4.23	2.14	1.01
	Cmax	10019	10019	10019

Tableau 3.6 Moyennes, variances et pourcentages de débordement calculés par (3.5), (3.6) et (3.7), L=519, K=20

où P_d est donné par:

$$P_d = \int_{Cmax_l}^{\infty} fc(x)dx$$

et le pourcentage de débordement, %deb est donné par:

$$\%deb = P_d \times 100 \qquad (3.7)$$

Dans le but d'étudier l'effet du rapport signal-à-bruit sur les différents paramètres de l'expression analytique, nous avons réalisé des simulations pour le cas L=319. Notez que n_l prend trois valeurs différentes pour les trois rapports signal-à-bruit examinés. Les figures 3.6 à 3.8 comparent le résultat de la régression (trait plein) avec celui de la simulation (trait haché). Dans ce cas aussi l'échelle est limitée à Cmax=600 et l'impulsion n'est pas montrée. Les moyennes, variances et pourcentages de débordement pour L=319 sont donnés au tableau 3.7. Il est aussi remarquable que les régressions pour L=519 donnent de meilleurs résultats que pour L=319.

Il faut rappeler que le calcul des paramètres de la fonction de densité par bloc nécessite la connaissance des résultats de simulation. Néanmoins, la connaissance d'une expression pour fc(x) peut grandement nous faciliter la tâche lors de la simulation de la file d'attente. D'autant plus que

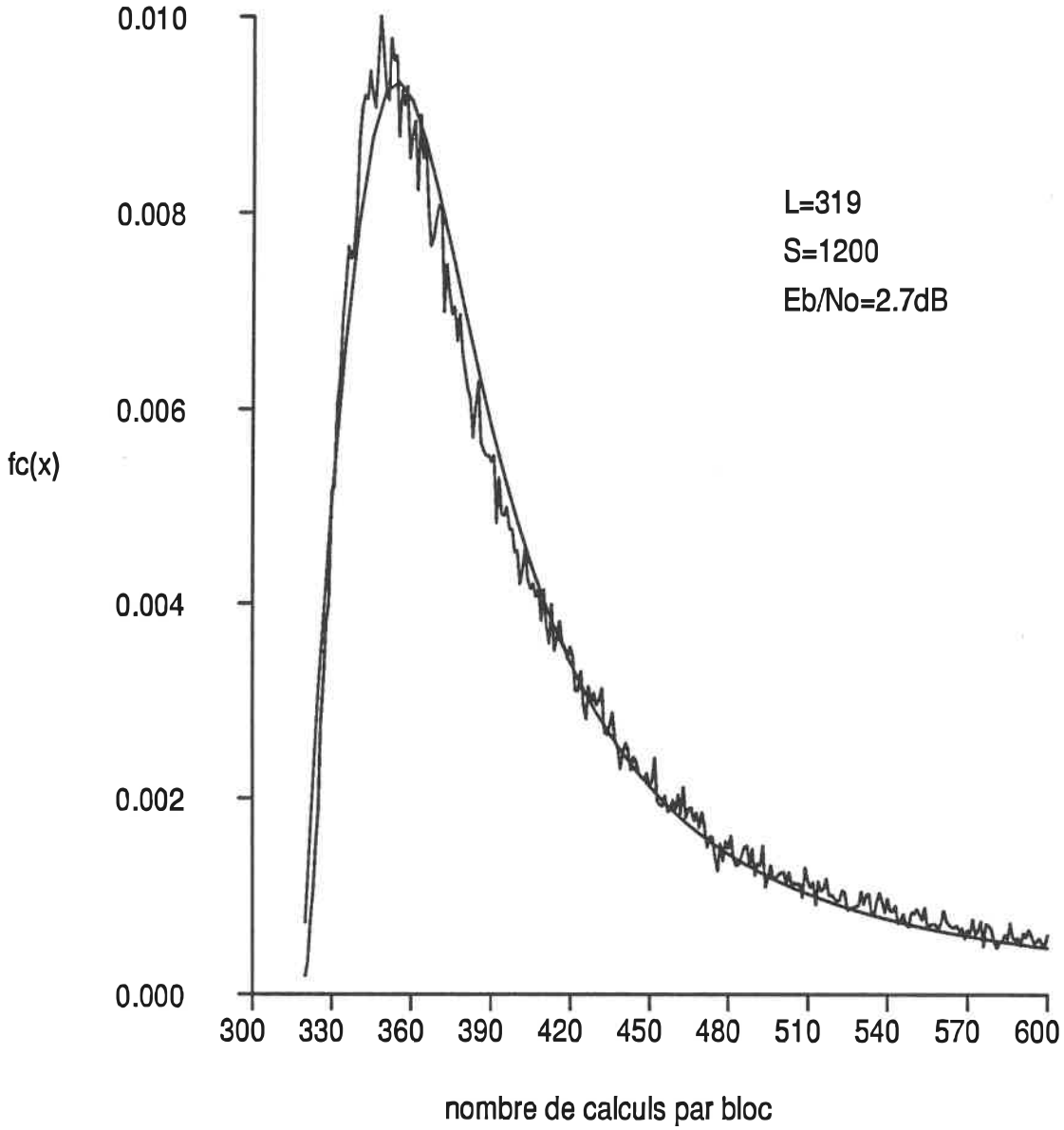


Figure 3.6 Comparaison de la fonction de densité par bloc tirée des simulations et de celle obtenue par la régression, n_1=.81, x_m=354

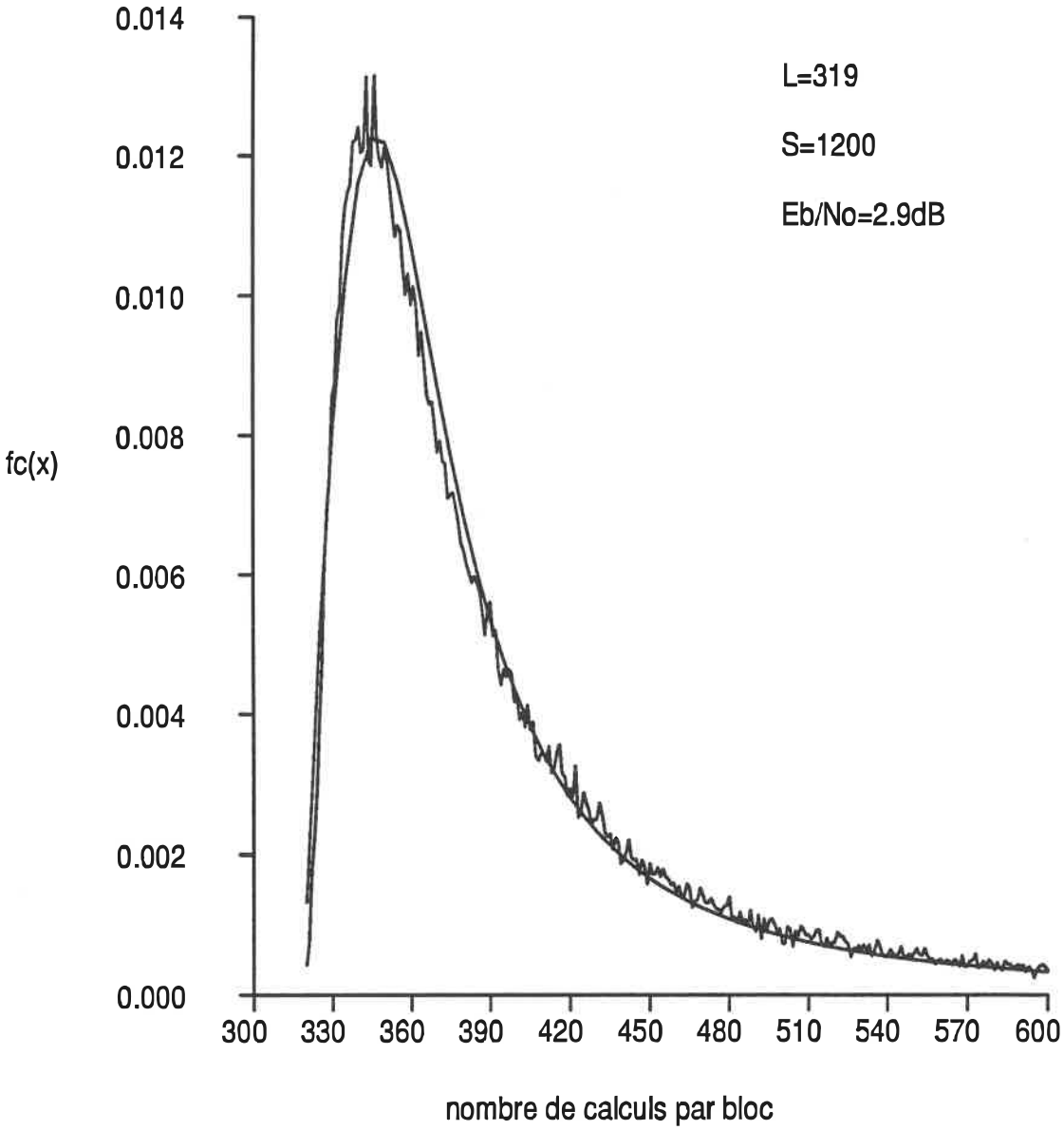


Figure 3.7 Comparaison de la fonction de densité par bloc tirée des simulations et de celle obtenue par la régression, n_1=.76, x_m=347

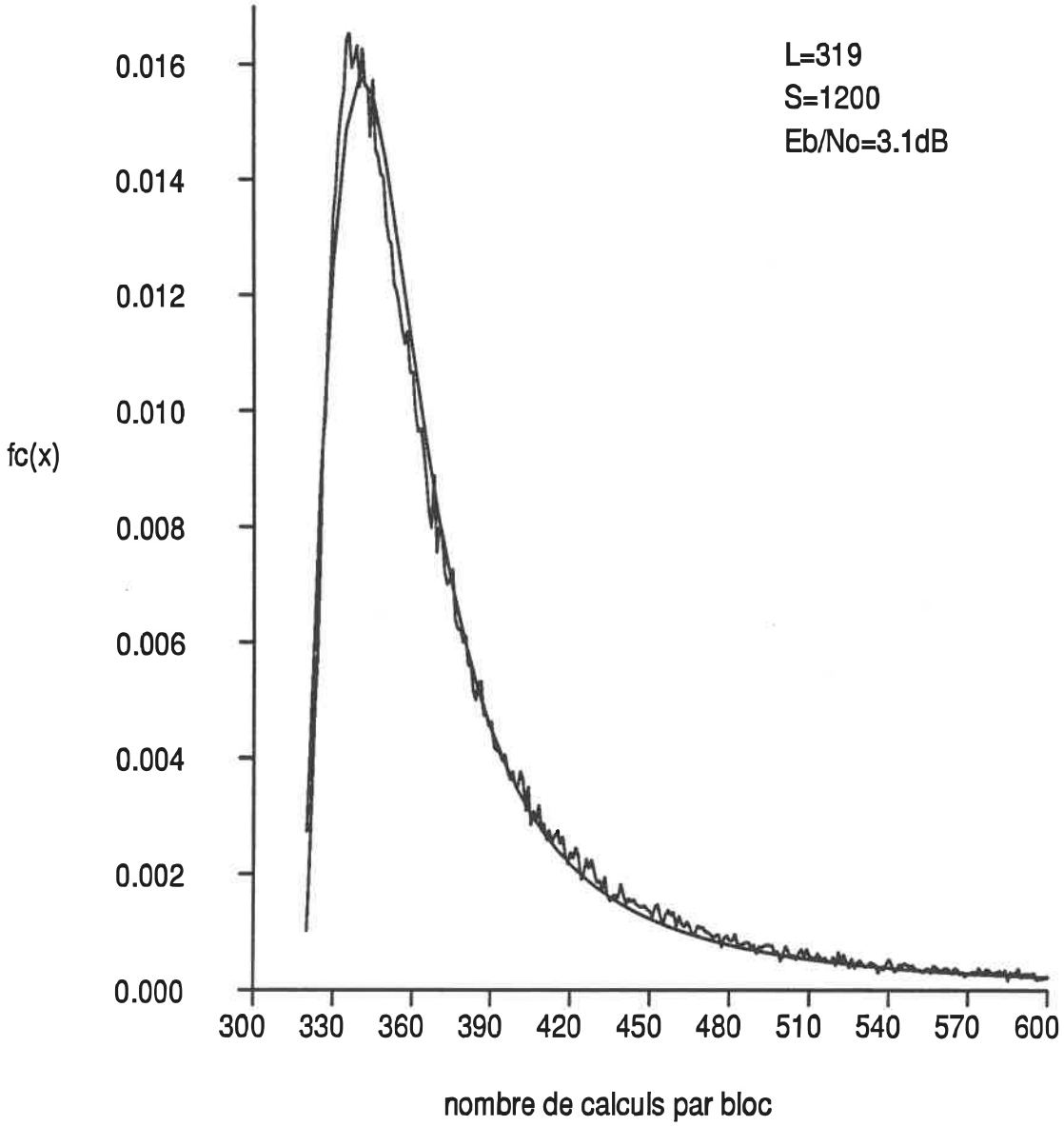


Figure 3.8 Comparaison de la fonction de densité par bloc tirée des simulations et de celle obtenue par la régression, n_1=.65, x_m=341

Taille pile (bits)	Paramètres d'intérêt	Rapport signal-à-bruit Eb/No (dB)		
		2.7 $\alpha = 1.03$	2.9 $\alpha = 1.145$	3.1 $\alpha = 1.265$
2400	moyenne	1.46	1.34	1.25
	%deb	3.93	2.2	1.17
	variance	.409	.271	.174
	Cmax	1219	1219	1219
12000	moyenne	1.66	1.44	1.30
	%deb	.596	.280	.153
	variance	3.43	1.79	.982
	Cmax	6019	6019	6019

Tableau 3.7 Moyennes, variances et pourcentages de débordement calculés par (3.5), (3.6) et (3.7), L=319, K=20

lorsque la taille de la pile varie, seules la valeur de la borne supérieure de la fonction et la valeur de l'impulsion changent.

3.2.3 Discussion

L'intérêt de l'expression analytique de fc(x) est indéniable. Elle permet le calcul de la moyenne et de la variance du nombre de calculs par bloc et la probabilité de débordement de la pile. Cependant, parce que les exposants n_1 et n_2 ne sont pas entiers, il est impossible de calculer les moyennes et les variances théoriques à l'aide d'une solution analytique. Elles ont plutôt été calculées à l'aide d'une intégration numérique utilisant la méthode du trapèze. De même, la transformée de Laplace de la fonction de densité ne peut être calculée.

La probabilité de débordement (valeur de l'impulsion) a été calculée de la même façon. Il faut comparer les résultats avec ceux obtenus par l'approximation de Pareto: les moyennes et variances sont calculées avec (3.2) et le pourcentage de débordement est donné par (3.4). Rappelons que les résultats de l'approximation de Pareto sont présentés au tableau 3.2. Les résultats obtenus par les régressions sont bien meilleurs que ceux calculés par l'approximation de Pareto tronquée (voir les tableaux 3.1, 3.2 et 3.3).

La valeur de Cmax semble poser quelques problèmes. La valeur de Cmax observée lors des simulations peut pour certains blocs être considérablement plus grande que la moitié de la taille de la pile. Pourtant, Cmax est défini comme (S/2)+K-1. Ce phénomène s'explique de la façon suivante: chaque calcul nécessaire au décodage d'un bit d'information entraîne l'entrée de deux noeuds dans la pile; cependant, lors du décodage de la queue, puisque le récepteur sait qu'il se trouve dans la queue, seuls les noeuds portant des symboles zéros sont entrés dans la pile. Donc, chaque calcul pour le décodage de 1 bit dans la queue ne demande qu'une entrée dans la pile. L'observation d'un Cmax très élevé (supérieur à S/2) lors du décodage d'un bloc signifie que, pour ce bloc, beaucoup de retours en arrière se sont faits à partir de la queue. Lors du calcul des moyennes, variances et pourcentages de débordement théoriques et des résultats de simulation, les cas où Cmax > (S/2 + K-1) sont négligés puisqu'ils ne correspondent pas à la définition de Cmax choisie et qu'ils sont relativement rares.

3.3 Conclusion

L'expression analytique de la fonction de densité par bloc est utile pour prévoir la moyenne , la variance du nombre de calculs par bloc ainsi que le pourcentage de

débordement pour différentes tailles de pile. La connaissance de ces paramètres nous permet de faire une analyse théorique du débit efficace et de la file d'attente du système D/G/s. L'expression analytique de la fonction de densité par bloc peut aussi être utilisée dans un simulateur de file d'attente pour analyser par des simulations le comportement du système.

CHAPITRE 4

CONDITIONS DE STABILITE ET CONDITIONS LIMITES

Le multidécodeur a pour but d'apporter une solution au problème de la variabilité du nombre de calculs par bloc du décodeur séquentiel. En effet, mettre plusieurs décodeurs séquentiels en parallèle augmente les possibilités d'avoir un décodeur libre pour accepter le bloc courant même si un ou plusieurs blocs précédents ont un temps de service très long. Avec cette architecture, la vitesse de décodage est très rapide (≈s fois la vitesse d'un seul décodeur où s est le nombre de décodeurs séquentiels en parallèle). Cependant, étant donné la variabilité du nombre de calculs par bloc, il peut arriver que tous les décodeurs soient occupés et incapables d'accepter un nouveau bloc. Il faut stocker ce bloc dans un tampon à l'entrée du système en attendant qu'un décodeur se libère. Il se peut même que plusieurs blocs aient le temps d'arriver avant que cet événement se produise. La taille de la file d'attente à l'entrée dépend de la taille de chacune des piles des décodeurs séquentiels et de leur gain de vitesse. En effet, plus un décodeur séquentiel a une grande pile, plus il a de possibilités pour faire des recherches arrières (ou une possibilité d'un plus grand nombre de calculs) avant que sa pile déborde. S'il cherche beaucoup, un grand nombre de bits vont s'accumuler dans le

tampon d'entrée. Il est donc avantageux de diminuer la taille de la pile pour diminuer la file d'attente à l'entrée. D'un autre côté, plus la taille de la pile est petite, plus le pourcentage de blocs qui débordent est grand, d'où une diminution du débit efficace du système.

La file d'attente dépend aussi du gain de vitesse, U, du décodeur. Ce dernier est défini comme le nombre de calculs [en blocs] qu'un décodeur peut faire pendant le temps de réception d'un nouveau bloc. Le décodeur doit être capable d'effectuer plus de calculs que le nombre moyen de calculs nécessaires pour décoder un bloc pour que la file d'attente ait une chance de se libérer.

Ce chapitre analyse les conditions limites qui gouvernent la file d'attente à l'entrée du système. On cherche aussi à déterminer le débit efficace du système. L'analyse de ces paramètres se fera en fonction de la taille de la pile, du gain de vitesse du décodeur, de la longueur des blocs et du niveau de bruit sur le canal c.-à-d. le rapport signal-à-bruit Eb/No. La deuxième partie de ce chapitre est consacrée à l'analyse d'un système un peu plus performant que le précédent: un multidécodeur avec décodeur de réserve. Les conditions limites de la file d'attente et le débit efficace de ce système seront aussi déduits. Il est important de remarquer que ces conditions sont déterminés en faisant l'hypothèse que le tampon à l'entrée est infini.

4.1 Organisation du système

Le système est essentiellement constitué d'un tampon d'entrée, T_1, d'un tampon de sortie, T_2 et de s décodeurs séquentiels en parallèle dénotés D_i, $1 \leq i \leq s$ disposant chacun d'un tampon interne de un bloc. Ce multidécodeur est illustré à la figure 4.1. Le multidécodeur peut être subdivisé en n classes $1 \leq n \leq s$ de décodeurs séquentiels. Chaque décodeur a une pile de taille S_i et un gain de vitesse U_i $1 \leq i \leq s$. Deux décodeurs D_i et D_j appartiennent à la même classe si et seulement si ils ont des piles et des gains identiques c.-à-d. $S_i = S_j$ et $U_i = U_j$.

4.1.1 Condition de stabilité du multidécodeur

Il a été vu au chapitre 2 que le système est une file d'attente D/G/s. Il existe une condition de stabilité pour celui-ci. En effet, supposons que le taux d'arrivée des blocs au tampon d'entrée soit plus grand que le taux moyen de sortie des décodeurs. Il est clair dans ce cas que le système n'aura aucune chance de récupérer. La file d'attente à l'entrée augmentera indéfiniment. Il est nécessaire d'imposer la condition suivante au système pour qu'il soit stable: le taux d'arrivée des blocs au tampon d'entrée doit être plus petit que le taux moyen de sortie des blocs

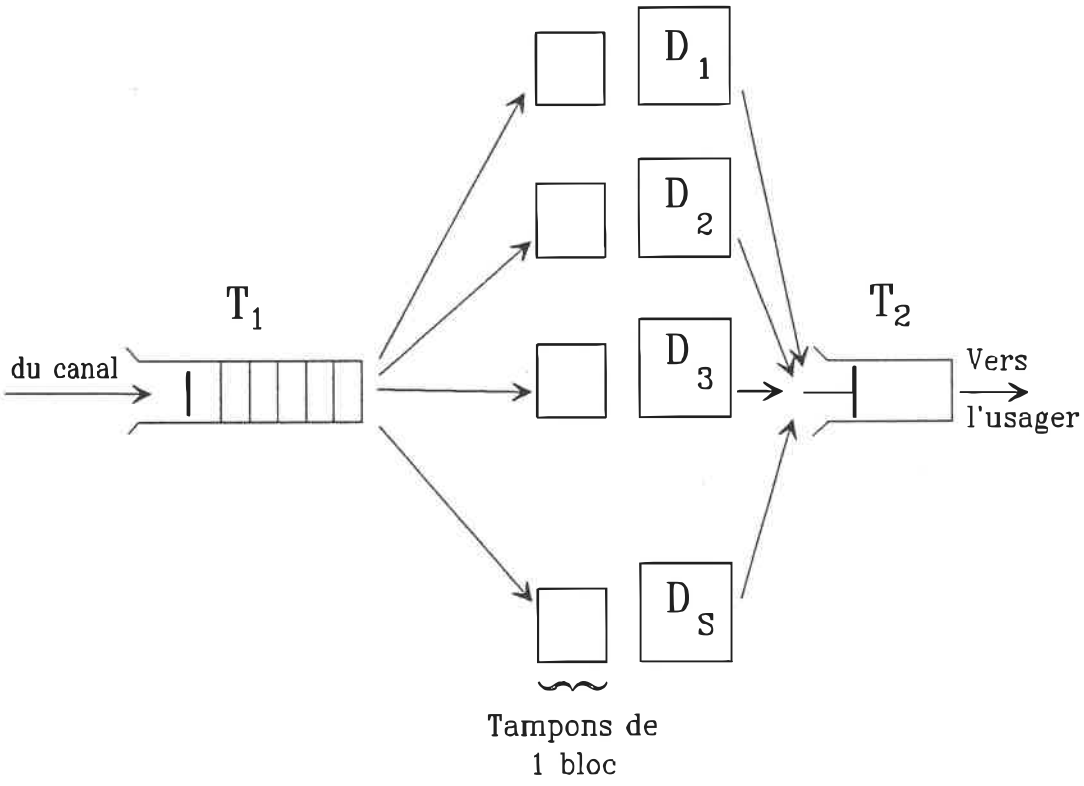


Figure 4.1 Multidécodeur

(c.-à-d. le taux moyen de service des blocs dans les décodeurs). Cette condition n'est nulle autre que < 1.

Au tampon d'entrée, l'unité de temps peut être le bloc car l'intervalle de temps entre l'arrivée de 2 blocs consécutifs est constant. Par contre, l'unité de temps au serveur est le calcul car on fait l'hypothèse que le temps pour faire un calcul (calcul d'un bloc comme expliqué au chapitre 3) est constant. Le paramètre qui permet de relier les deux taux est le gain de vitesse du décodeur séquentiel. Vu du tampon d'entrée, s'il y a un seul serveur les taux moyens s'écrivent:

temps moyen arrivée= 1 unité de temps/bloc

temps moyen de service=Cmoy/U unités/bloc

où Cmoy est obtenu par (3.5) divisée par L avec Cmax=S/2+K-1. d'où

taux arrivée= 1 bloc /unité de temps

taux de sortie= U/Cmoy blocs/unité de temps

Finalement, la condition de stabilité devient U > Cmoy.

Supposons maintenant qu'il y ait s serveurs ayant chacun un gain de vitesse U_i $1 \leq i \leq s$. Posons $Cmoy_i$, le nombre de calculs moyen mesuré en blocs que fait le $i^{ème}$ décodeur,

$1 \leq i \leq s$. Dans ce cas les taux d'arrivée et de sortie s'écrivent:

$$\text{taux arrivée} = 1$$

$$\text{taux de sortie} = \frac{U_1}{\overline{Cmoy}_1} + \ldots + \frac{U_s}{\overline{Cmoy}_s}$$

D'où la condition de stabilité:

$$\frac{U_1}{\overline{Cmoy}_1} + \ldots + \frac{U_s}{\overline{Cmoy}_s} > 1 \qquad (4.1)$$

Cette condition détermine la borne inférieure d'opération du système. En deçà de cette borne, le système est instable et la file d'attente augmente à l'infini. Au dessus, le système est stable mais il est tout de même avantageux d'augmenter le gain de vitesse pour diminuer la taille de la file d'attente.

4.1.2 Longueur maximale de la file d'attente à l'entrée

Il a été vu précédemment que la taille de la file d'attente dépend du gain de vitesse et de la taille de la pile du décodeur. L'analyse qui suit permet de montrer que sous certaines conditions, la taille de la file d'attente à l'entrée ne sera jamais plus grande que 1 bloc. Cette analyse

est une généralisation de celle faite par Haccoun pour un seul décodeur séquentiel [4].

Soient s décodeurs séquentiels mis en parallèle ayant chacun une pile de taille S_i $1 \leq i \leq s$, et une file d'attente commune à l'entrée. Soient U_i, le gain de vitesse de chacun des décodeurs et L, le nombre de bits d'information dans un bloc. On peut garantir que la file d'attente ne sera jamais plus grande qu'un bloc si le taux minimal de sortie des blocs est plus grand que le taux maximal d'entrée des blocs. Ces taux s'écrivent:

$$\text{taux maximal d'arrivée} = 1$$

$$\text{taux minimal de sortie} = \sum_{i=1}^{s} (U_i L)/Cmax_i$$

où $Cmax_i = S_i/2 + K - 1$.

D'où la condition

$$\sum_{i=1}^{s} \frac{U_i L}{Cmax_i} > 1 \qquad (4.2)$$

Si U_i et $Cmax_i$ sont constants pour tout i, la condition devient:

$$U > \frac{Cmax}{sL}$$

Il faut noter que cette analyse suppose que chaque décodeur dispose d'un tampon interne d'une longueur de 1 bloc. En réalité le tampon est d'une longueur de (s+1) blocs où chaque décodeur a un tampon interne de 1 bloc et un tampon de 1 bloc est commun aux s décodeurs.

4.1.3 Régions d'opération

Sous l'hypothèse que le gain est constant pour tous les décodeurs et que ces derniers sont tous identiques, les deux conditions sur le gain (stabilité et tampon maximal) donnent trois régions d'opération du système:

- si $U < Cmoy/s$ le système est instable et la taille de la file d'attente tend vers l'infini.

- si $Cmoy/s \leq U \leq Cmax/sL$, la stabilité est possible mais la taille de la file d'attente n'est pas garantie.

- si $U \geq Cmax/sL$, la stabilité est garantie et le tampon à l'entrée ne contiendra jamais plus d'un bloc à la fois.

La connaissance de la loi de service est utile pour analyser la deuxième région d'opération, celle où le système est stable mais où la taille de la file d'attente n'est pas garantie.

4.1.4 Débit efficace du multidécodeur

Le débit efficace est défini comme le rapport entre le nombre de bits d'information délivrés à l'usager et le nombre de symboles transmis dans le canal. Une façon équivalente de le définir est:

$$\text{débit efficace} = \frac{n_d\ L}{n_t\ L/r}$$

où n_d est le nombre de blocs décodés

L est le nombre de bits d'information dans un bloc

n_t est le nombre de blocs transmis dans le canal

r est le taux de codage

Par conséquent, le débit est donné en bits/symbole.

Dans le cas du multidécodeur, le débit efficace dépend fortement de la probabilité de débordement de chacune des piles des décodeurs séquentiels. Si toutes les piles du multidécodeur sont identiques, le débit efficace est donné par (1-probabilité de débordement de la pile)r. Par contre,

si les tailles de pile sont différentes, il est plus ardu de le déterminer car le flux de blocs venant du canal ne se répartit pas également dans les différents décodeurs. Pour évaluer le débit efficace, il faut connaître le nombre de blocs qui va à chacun des décodeurs. Une analyse complète nécessite la connaissance des probabilités d'occupation des décodeurs séquentiels. Or, la probabilité qu'il y ait i décodeurs occupés, $1 \leq i \leq s$, est égale à la probabilité qu'il y ait i consommateurs dans le système. Etant donné que dans une file d'attente G/G/s cette probabilité ne se calcule pas analytiquement, il devient difficile de faire une analyse du débit sans poser une hypothèse simplificatrice. Dans l'analyse qui suit, seul le cas où chacun des serveurs est toujours occupé est considéré. Cette simplification est justifiée parce qu'en général, les paramètres du système sont choisis pour une utilisation maximale des ressources.

En moyenne, le nombre de blocs, n_i, qui passent par le décodeur D_i est inversement proportionnel au nombre de calcul moyen par bloc que fait ce décodeur. On a donc:

$$n_i = \frac{n_t U_i / Cmoy_i}{\sum_{i=1}^{s} \left[\frac{U_i}{Cmoy_i} \right]}$$

où n_t est le nombre de blocs transmis.

Or, de ce nombre n_i, un certain pourcentage va déborder. Par conséquent, le nombre total de blocs qui débordent, n_{deb}, est donné par:

$$n_{deb} = \sum_{i=1}^{s} P_{di}\, n_i$$

où P_{di} est la probabilité de débordement de la pile S_i donnée par (3.7). D'après les deux formules précédentes, le débit efficace est donné par:

$$\text{débit efficace} = \left[1 - \frac{\sum_{i=1}^{s} \frac{P_{di} U_i}{Cmoy_i}}{\sum_{i=1}^{s} \frac{U_i}{Cmoy_i}}\right] r \qquad (4.3)$$

Cette dernière expression donne le débit efficace moyen sous l'hypothèse que tous les décodeurs séquentiels sont toujours occupés. En effet, étant donné la variabilité du nombre de calculs par bloc, le nombre de blocs envoyés à chaque décodeur ne peut être connu exactement. Il faut se contenter d'une mesure moyenne. Sur un grand nombre de blocs, le débit efficace calculé doit se rapprocher du débit efficace du système. Il apparaît aussi que plus la taille de la pile est petite, plus l'expression du débit efficace sera exacte car la variance du nombre de calculs par bloc diminue avec la diminution de la taille de la pile.

4.1.5 Conclusion

Le principal inconvénient du multidécodeur est qu'il existe un compromis entre la file d'attente à l'entrée et le débit efficace. En effet, pour obtenir un débit efficace élevé, il faut travailler avec de grandes piles ce qui entraîne une augmentation de la file d'attente à l'entrée ainsi qu'une augmentation du coût du système . Une nouvelle architecture avec un décodeur de réserve permet d'obtenir un débit efficace élevé sans toutefois être obligé d'avoir plusieurs décodeurs séquentiels avec de grandes piles. Cette nouvelle approche est examinée ci-après.

4.2 Organisation du système avec des décodeurs de réserve

Le système avec décodeurs de réserve est organisé de la façon suivante. Au lieu de mettre s décodeurs séquentiels en parallèle, il suffit de créer deux classes de décodeurs. La première est constituée de s_1 ($s_1 < s$) décodeurs simples en parallèle (petites tailles de pile) qui reçoivent leurs blocs du canal. La deuxième est constituée de s_2 ($s_2=s-s_1$) décodeurs puissants (taille de pile élevée). Ces derniers reçoivent leurs blocs des s_1 décodeurs simples. Plus précisément, ils reçoivent tous les blocs qui ont fait débordé les s_1 décodeurs simples. Le décodage de ces blocs est soit recommencé à zéro,

ou encore si toute l'information est donnée au décodeur de réserve, poursuivi à partir de l'endroit où le décodeur simple avait débordé. Le choix parmi ces deux possibilités se fait lors du design du système. Cette nouvelle organisation introduit une deuxième file d'attente, devant les s_2 décodeurs puissants.

Pour maximiser l'utilisation du système, un décodeur de la deuxième classe peut recevoir un bloc du canal s'il est libre et qu'aucun bloc n'ayant déjà débordé n'attend dans la deuxième file d'atente.

Le multidécodeur avec décodeurs de réserve est illustré à la figure 4.2

4.2.1 Condition de stabilité des deux files d'attente

Il importe ici de faire une hypothèse importante. Nous supposons qu'en régime permanent, tout le flux venant du canal ira aux s_1 décodeurs simples tandis que les s_2 décodeurs complexes ne recevront que les blocs qui ont déjà débordé. De cette façon, les conditions limites des deux files d'attente peuvent être facilement calculées. Pour simplifier l'analyse, on suppose que tous les décodeurs de la première classe sont identiques avec un gain de vitesse U_1, une pile S_1 et un temps de service moyen $Cmoy_1$ tandis que tous les décodeurs de la deuxième

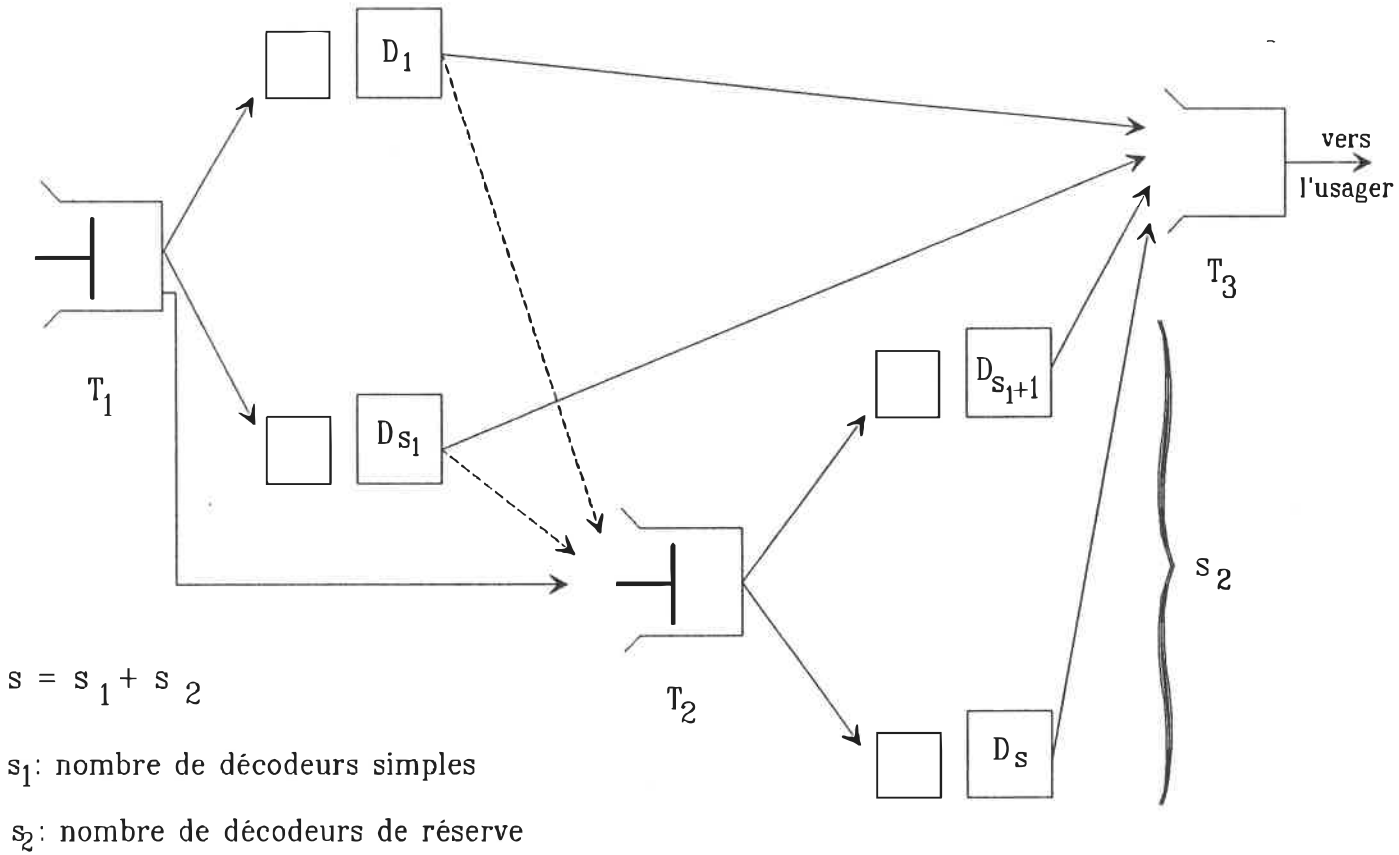


Figure 4.2 Multidécodeur avec décodeurs de réserve

classe ont un gain de vitesse U_2 , une pile de taille S_2 et un temps de service moyen $Cmoy_2$.

La condition limite pour la première file est la même que dans le système précédent sauf qu'au lieu d'avoir s décodeurs, il y en a s_1. Elle s'écrit donc:

$$U_1 > \frac{Cmoy_1}{s_1}$$

Pour l'autre file d'attente, il faut déterminer les taux d'arrivée et de sortie. La probabilité de débordement des blocs pour une taille de la pile donnée étant connue, le taux d'arrivée à la deuxième file d'attente peut s'écrire:

$$\text{taux arrivée moyen} = \text{taux de sortie des décodeurs simples} \quad Pd1$$

$$= \frac{s_1 \; U_1 \; P_{d1}}{Cmoy_1}$$

$$\text{taux de sortie moyen} = \frac{s_2 \; U_2}{Cmoy_2^{\cdot}}.$$

où $Cmoy_2^{\cdot}$ est défini comme étant le nombre moyen de calculs fait par les décodeurs de réserve. Il existe deux choix pour la procédure de décodage des décodeurs de réserve: le décodage est soit recommencé à zéro, soit poursuivi à partir de l'endroit où les décodeurs simples l'ont laissé. Si le décodage est recommencé à zéro alors:

$$Cmoy_2^{\bullet} = E[x \mid x>(Cmax_1)]$$

$$= \frac{\int_{(Cmax_1)}^{(Cmax_2)} x\, f_c(x)\, dx}{P(x> Cmax_1)}$$

Par contre, si les décodeurs complexes poursuivent le décodage là où les décodeurs simples l'ont laissé, alors $Cmoy_2^{\bullet} = E[x \mid x>(Cmax_1)] - (Cmax_1)$. Cette dernière option entraînerait une grande complexité et des délais supplémentaires avec le type d'architecture proposé. Quelques valeurs de $Cmoy_2^{\bullet}$ pour différents $Cmax_1$ et pour $Cmax_2$=10 000 sont données au tableau 4.1. Comme dans le chapitre 3, elles ont été renormalisées. La condition de stabilité est déduite de ces deux taux:

$$\frac{U_2}{U_1} > \frac{Cmoy_2^{\bullet}\ s_1\ P_{d1}}{s_2\ Cmoy_1} \qquad (4.4)$$

D'après cette expression, il apparaît que le gain de vitesse de la deuxième classe doit être choisi en fonction de celui de la première classe.

$Cmax_2$	$Cmax_1$	$Cmoy_2^*$
10000	1000	4.57
	1500	7.04
	2000	9.01
	2500	10.67
	4000	14.23
50000	1000	6.23
	1500	10.54
	2000	14.29
	2500	17.79
	4000	27.31

Tableau 4.1 Valeurs de $Cmoy_2^*$ en fonction de $Cmax_1$ et $Cmax_2$, L=519 et Eb/No=2.7 dB

4.2.2 Discussion sur la file d'attente maximale

Comme pour le système précédent, la taille de la file d'attente à l'entrée peut être limitée à un bloc. L'hypothèse selon laquelle tous les blocs venant du canal vont aux décodeurs simples tandis que les décodeurs de réserve ne reçoivent que les blocs qui ont déjà débordé tient toujours. La situation est la même que dans le cas du multidécodeur sauf qu'il y a s_I décodeurs au lieu de s. On a donc:

$$U_I > \frac{Cmax_I}{s_I L} \qquad (4.5)$$

Dans ce cas l'expression est simplifiée car le gain et la taille de la pile sont constants pour les s_I décodeurs. De plus, à cause de l'hypothèse posée, il n'y a pas de contribution des décodeurs de réserve. L'expression (4.5) est une conditon suffisante pour limiter la longueur de la file d'attente à un bloc mais pas nécessaire. En effet, si s_I est grand et/ou $Cmax_I$ élevé, la probabilité que tous les décodeurs soient occupés avec un bloc ayant au moins $Cmax_I$ calculs tend vers zéro. Dans ce cas, il peut exister un gain inférieur à la condition (4.5) pour lequel la file d'attente à l'entrée soit de un bloc.

Pour ce qui est de la deuxième file d'attente, il est plus difficile de déterminer un gain minimal qui garantisse la taille de la file d'attente à l'entrée. En

effet, les blocs arrivent à un taux constant au système. Une certaine partie de ceux-ci (ceux qui débordent) vont se rendre à la deuxième file d'attente. Cependant, il est impossible de garantir que les blocs arrivent à un taux constant au deuxième tampon. La condition de la file d'attente maximale ne peut donc pas être calculée avec ce taux d'arrivée. Par contre, dans le pire cas, tous les blocs qui arrivent au système débordent et par conséquent se rendent à la deuxième file d'attente. Dans ce cas, le taux d'arrivée à la deuxième file d'attente est constant et est donné par:

$$\text{taux arrivée maximal} = s_1 U_1 L / Cmax_1$$

$$\text{taux de sortie} = s_2 U_2 L / Cmax_2$$

D'où la condition:

$$\frac{U_2}{U_1} > \frac{Cmax_2 \; s_1}{s_2 \; Cmax_1} \qquad (4.6)$$

Respecter cette condition, assure que la deuxième file d'attente ne sera jamais plus grande que un bloc. Cependant, c'est une borne très large qui en pratique est presqu'inutile.

4.2.3 Débit efficace du multidécodeur avec décodeur de réserve

Le débit efficace de ce système est très facile à calculer. En effet, puisque tous les blocs qui font déborder les décodeurs simples sont envoyés aux décodeurs de réserve, le débit efficace dépend de la probabilité de débordement des décodeurs de réserve. Puisqu'ils sont tous identiques, le débit efficace s'écrit:

$$\text{débit efficace} = \left[1 - P_{d2}\right] r \qquad (4.6)$$

Contrairement à l'expression du débit efficace pour le multidécodeur simple, cette expression donne une valeur exacte car tous les décodeurs de réserve sont identiques.

4.3 Conclusion

Grâce à la théorie des files d'attente, des régions d'opération ont été déterminées. Ces bornes sont très utiles lors de la simulation pour prédire le comportement du système.

Le multidécodeur avec décodeur de réserve a un avantage sur le multidécodeur simple. En effet, la longueur de la file d'attente à l'entrée et le débit efficace ne sont

plus en conflit puisqu'ils ne dépendent plus du même paramètre. Les chapitres 5 et 6 analysent les performances de ces deux systèmes à l'aide de simulations.

CHAPITRE 5

ANALYSE DES PERFORMANCES DU MULTIDECODEUR

Ce chapitre analyse les performances du multidécodeur, c.-à-d. la distribution de la file d'attente à l'entrée et le débit efficace dans la deuxième région d'opération, celle où le système est stable et la taille de la file d'attente n'est pas garantie. Le chapitre débute par une brève description du simulateur réalisé. Les paramètres importants qui servent de critères de comparaison des performances sont définis. Des méthodes de solution du problème de la variabilité du nombre de calculs sont introduites. L'effet de l'augmentation du nombre de processeurs, de la variation de la longueur de la pile et du gain de chacun des processeurs est analysé pour les distributions des files d'attente à l'entrée et à la sortie. Les fonctions de répartition obtenues sont comparées avec celles obtenues par les équations (2.2) et (2.3). Finalement, la validité de l'expression du débit efficace trouvée au chapitre 4 est vérifiée.

5.1 Description du simulateur

Le simulateur a été écrit en langage C et s'exécute sur des ordinateurs SUN. La structure de données du simulateur est constituée essentiellement d'une liste chaînée qui contient deux types d'événements: arrivée d'un bloc dans le tampon et départ d'un bloc du système. L'arrivée d'un bloc dans le tampon se fait à tous les L+K-1 bits. Le départ d'un bloc se fait lorsque le décodage de celui-ci est fini. Ces deux types d'événements génèrent chacun des événements qui appartiennent aussi à l'un ou l'autre des deux types. L'organigramme de la figure 5.1 présente le fonctionnement du simulateur. La simulation consiste à retirer l'événement en haut de la liste. Selon le type de cet événement, des événements sont générés et sont à leur tour insérés dans la liste au bon endroit. La simulation continue avec le prochain événement placé en haut de la liste.

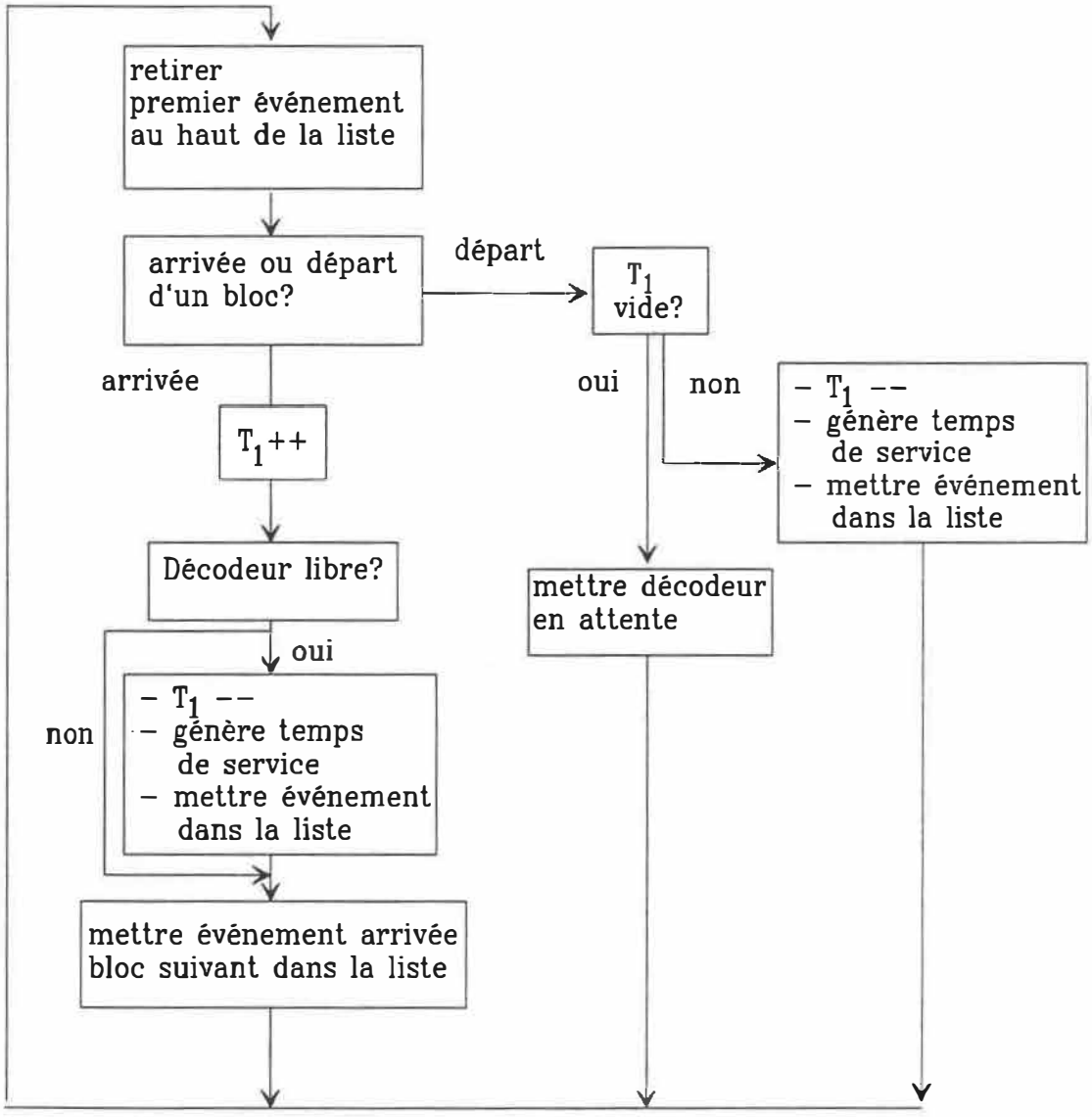


Figure 5.1 Organigramme du simulateur du multidécodeur

Les événements générés dépendent de l'événement en haut de la pile. Si l'événement retiré de la pile est l'arrivée d'un bloc au tampon d'entrée, alors

1- Le tampon est augmenté de 1

2- Si un décodeur est en attente, on lui donne ce bloc:

   a) le tampon est diminué de 1

   b) un temps de service est généré

   c) cet événement est placé dans la liste au bon endroit

3- L'événement "arrivée du bloc suivant" est généré et placé dans la liste.

Si l'événement est le départ d'un bloc, alors,

1- Si le tampon est vide le décodeur est mis en attente

2- Sinon:

   a) le tampon est diminué de 1

   b) un temps de service est généré

   c) cet événement est placé dans la liste

La mise en ordre de la pile se fait à l'aide du temps auquel se produit chaque événement. Les temps d'arrivée des blocs sont des multiples de L+K-1. A tous les L+K-1 bits, un

bloc arrive au système. Donc, temps arrivée du bloc suivant = temps arrivée du bloc courant + L+K-1 (le bloc courant étant l'événement en haut de la pile).

La génération des temps de service se fait de la façon suivante:

1- Un nombre aléatoire est généré entre 0 et 1
2- Le temps de service du décodeur séquentiel est déterminé à l'aide d'une table elle-même obtenue avec la loi de service trouvée au chapitre 3.
3- Du point de vue du tampon d'entrée, le temps de service obtenu en 2 doit être divisé par le gain de vitesse.
4- Finalement, le temps de la fin du décodage d'un bloc est donné par le temps de l'événement en haut de la liste + temps déterminé en 3.

Le simulateur tient aussi compte de la mise en ordre à la sortie. Une autre liste chaînée est ajoutée au simulateur à cette fin. Chaque fois qu'un bloc est décodé, il est inséré dans cette liste qui remet les blocs par ordre de leur arrivée au système. Ces blocs sont délivrés en ordre à l'usager chaque fois qu'il y a deux ou plusieurs blocs

consécutifs dans la liste. Cette remise en ordre n'est pas représentée à la figure 5.1.

5.2 Définition des paramètres d'intérêt

Le but de cette section est d'apporter quelques définitions et précisions sur les paramètres qui caractérisent la file d'attente à l'entrée et le débit efficace du multidécodeur.

La file d'attente peut être décrite de plusieurs points de vue: probabilité que la file d'attente , W, soit d'une longueur x , longueur moyenne de la file d'attente, longueur maximale de la file d'attente etc...

La probabilité, $p_w(x)$, que la file d'attente soit de longueur x est définie comme étant la proportion de temps pendant lequel la file d'attente est d'une longueur x. Avec cette probabilité, la longueur moyenne de la file d'attente, Wmoy, est obtenue par:

$$\sum_{i=0}^{\infty} p_w(x=i)\,i$$

La longueur maximale de la file d'attente, Wmax, est définie comme le nombre maximal de blocs qui a été observé dans la file d'attente pendant toute la simulation. Quant au débit efficace, il est obtenu à partir de l'équation 4.3.

Ces paramètres servent de critères de comparaison des performances du multidécodeur en fonction du nombre de processeurs, de la taille de la pile et du gain de vitesse des décodeurs.

5.3 Méthodes de solution de la variabilité du nombre de calculs

Le principal problème du décodage séquentiel est le comportement asymptotique de type Pareto de son temps de service. Le but de cette section est de décrire des moyens d'apporter une solution à ce problème. Le comportement de type Pareto du décodeur séquentiel provoque une augmentation du nombre de blocs dans la file d'attente à l'entrée. Or, il est avantageux de limiter la file d'attente à l'entrée pour éviter un débordement du tampon d'entrée. Une façon connue de le faire est de limiter la taille de la pile de sorte que le nombre de calculs maximal pour un bloc soit borné. Le seul inconvénient de cette méthode est la perte d'un certain nombre de blocs. Cette perte est d'autant plus grande que la taille de la pile est limitée. Pour améliorer la situation on peut songer à retransmettre les blocs qui ont débordé. Cette solution présente l'inconvénient que la retransmission des blocs augmente les files d'attente à l'entrée et à la sortie et diminue le débit efficace du système.

Une autre façon de diminuer la file d'attente à l'entrée consiste à augmenter le gain de vitesse du décodeur pour opérer dans une zone proche de la zone de gain où la taille de la file d'attente est garantie. Cette solution a le désavantage que la majorité du temps le système est vide d'où une perte d'efficacité. Ces deux approches ont déjà été explorées par Haccoun et Pau [3].

Une troisième approche consiste à augmenter le nombre de processeurs en parallèle de sorte que la vitesse de décodage est augmentée et le comportement de la file d'attente à l'entrée n'est plus Pareto. C'est la validité de cette approche qui est analysée en détail dans ce chapitre. Tous les résultats sont appuyés par des simulations faites avec 50000 blocs de 519 bits, avec un code de Johannesson de taux 1/2 ayant K=20, et un rapport signal-à-bruit de 2.7 dB. Le tableau 5.1 donne quelques valeurs de moyennes et de variances en fonction de Cmax pour un rapport signal-à-bruit de 2.7 dB et L=519. Ce sont ces valeurs qui ont été utilisées pour les calculs. De plus, le tableau 5.2 donne quelques valeurs des gains de vitesse minimaux pour garantir la stabilité et garantir la taille du tampon en fonction de Cmax pour différents nombres de processeurs.

Taille pile (bits)	Paramètres d'intérêt	Rapport signal-à-bruit Eb/No (dB) 2.7 $\alpha = 1.03$
3000	moyenne %deb variance Cmax	1.49 7.35 .268 1500
4000	moyenne %deb variance Cmax	1.55 4.81 .475 2000
5000	moyenne %deb variance Cmax	1.59 3.56 0.6922 2500
6000	moyenne %deb variance Cmax	1.62 2.83 .917 3000
8000	moyenne %deb variance Cmax	1.66 2.0 1.38 4000
20000	moyenne %deb variance Cmax	1.796 0.7114 4.226 10000

Tableau 5.1 Moyennes, variances et pourcentages de débordement en fonction de Cmax, L=519 et Eb/No=2.7dB

Cmax	Nombre de processeurs	Gains de vitesse minimaux	
		pour stabilité	pour tampon garanti
1500	s=1	1.49	2.89
	s=5	.298	.578
	s=10	.149	.289
2000	s=1	1.55	3.85
	s=5	.31	.771
	s=10	.155	.385
2500	s=1	1.59	4.82
	s=5	.318	.963
	s=10	.159	.482
3000	s=1	1.62	5.78
	s=5	.324	1.156
	s=10	.162	.578
10000	s=1	1.796	19.27
	s=5	.359	3.854
	s=10	.180	1.927
50000	s=1	2.052	96.33
	s=5	.410	19.268
	s=10	.205	9.633

Tableau 5.2 Gains de vitesse minimaux en fonction de Cmax avec L=519 et Eb/No=2.7dB

5.4 Effet de l'augmentation du nombre de processeurs

Il importe avant tout de dissocier deux phénomènes qui agissent sur la file d'attente à l'entrée: le nombre de processeurs et la taille de la pile de chacun des processeurs. Pour bien isoler l'effet sur la file d'attente de l'augmentation du nombre de processeurs, il faut utiliser des piles très grandes de telle sorte qu'il n'y aura presque pas de blocs qui débordent. Les simulations ont été faites avec des piles de taille S_i=100K bits (Cmax=50K) pour $1 \leq i \leq s$. Dans ces cas, il y a environ un bloc sur mille qui déborde. On peut donc négliger l'effet de la limitation de la taille de la pile sur la file d'attente à l'entrée.

Pour mettre en évidence l'impact du parallélisme sur la longueur moyenne de la file d'attente pour s=1 et s=10 et une longueur de pile donnée, il faut trouver une base de comparaison uniformisée. Les plages de gains pour s=1 et s=10 sont différentes. En effet, lorsque s=1, $2.05 \leq U \leq 96$, tandis que pour s=10, $.2 \leq U \leq 9.6$. Ces résultats sont obtenus pour S=100K et L=519. Le gain ne peut donc pas être utilisé. Par contre, il appert d'après les équations 4.1 et 4.2, que les plages du produit Us sont indépendantes de s. Pour les valeurs précédentes, la plage est de $2.05 \leq Us \leq 96$ pour s=1 et s=10. Par conséquent, le produit Us s'avère être le paramètre idéal pour la comparaison de la distribution (ainsi

que pour Wmoy et Wmax) des files d'attente obtenues avec un nombre différent de processeurs.

Il faut noter que le gain de vitesse peut être fractionnaire. En effet, dans le chapitre 4, le gain de vitesse a été défini comme étant le nombre de calculs de bloc qu'un décodeur peut faire pendant le temps de réception d'un bloc. Or, lorsqu'il y a un seul décodeur, la condition de stabilité U>Cmoy implique que U est plus grand que 1 car Cmoy est toujours plus grand que 1. Par contre, lorsqu'il y a plusieurs décodeurs, la condition de stabilité U>Cmoy/s ne limite pas le gain à une valeur supérieure à 1. Le gain peut donc prendre des valeurs fractionnaires qui sont justifiées comme suit: puisque le nombre de calculs qu'un décodeur peut faire pendant un intervalle de temps donné est fixé par la technologie, ce paramètre ne peut être varié. Par contre, le temps de réception d'un nouveau bloc n'est pas fixé par la technologie du décodeur séquentiel mais plutôt par le taux de transmission de l'information dans le canal. Dans ce cas, si le taux de transmission dans le canal est très élevé, un seul décodeur ne peut rendre le système stable et alors son gain de vitesse est fractionnaire par rapport au taux de transmission. Par contre le système peut être stable, s'il y a plusieurs décodeurs en parallèle ayant chacun un gain de vitesse fractionnaire.

Plusieurs simulations de la file d'attente à l'entrée ont été faites pour différents produits Us. La longueur moyenne de la file d'attente en fonction de Us est tracée à la figure 5.2 pour s=1 et s=10 et Cmax=50 000. L'augmentation du nombre de processeurs tend à diminuer la longueur moyenne de la file d'attente. De plus, la longueur moyenne tend vers zéro beaucoup plus rapidement lorsque le nombre de processeurs augmente. En fait, si la pile était de taille infinie, il serait impossible d'obtenir un gain minimal qui garantisse que la file d'attente maximale soit limitée à un bloc. Dans ce cas, la longueur moyenne de la file d'attente à l'entrée aurait le même comportement Pareto que le décodeur séquentiel. Par conséquent, une augmentation du gain pour un nombre de processeurs fixe n'améliorerait pas sensiblement le comportement moyen de la file d'attente. Puisqu'en pratique la taille de la pile est limitée, le comportement de la file d'attente moyenne n'est Pareto qu'au milieu de la région d'opération comme l'indique la droite de la figure 5.2 pour s=1. Par contre, cette droite est absente de la deuxième région d'opération du multidécodeur pour s=10. Cette observation indique qu'une augmentation du nombre de processeurs atténue le comportement Pareto de la file d'attente moyenne à l'entrée.

L'augmentation du nombre de processeurs diminue non seulement la longueur moyenne de la file d'attente mais

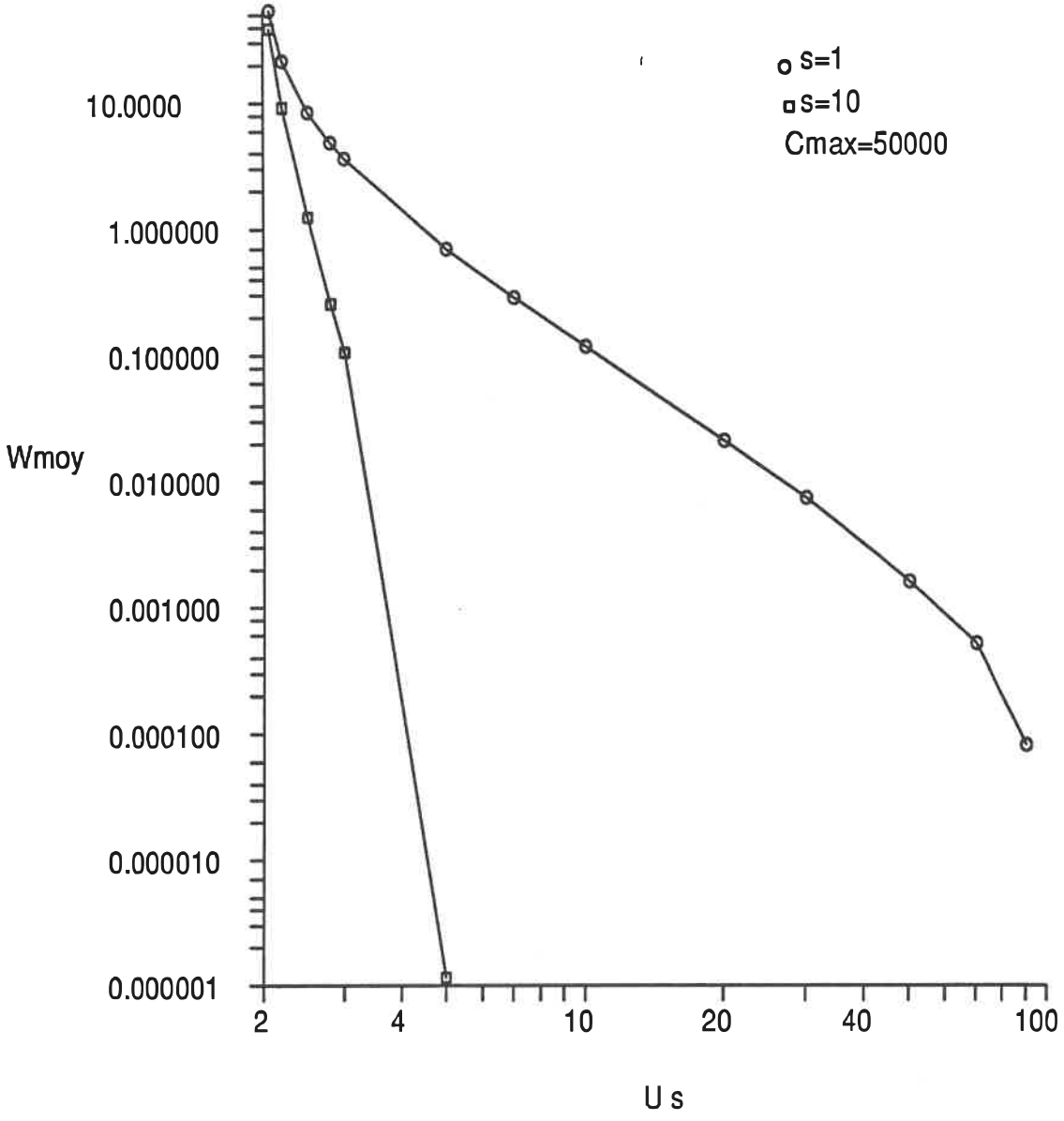


Figure 5.2 Longueur moyenne de la file d'attente à l'entrée en fonction du produit Us pour L=519 et Eb/No=2.7dB

diminue aussi la longueur de la deuxième région d'opération. Plus particulièrement, c'est le gain de vitesse minimal garantissant la longueur de la file d'attente qui diminue. Ce phénomène est facile à expliquer. En effet, l'obtention de l'équation 4.2 nécessite de poser une hypothèse majeure: tous les décodeurs ont un temps de service égal à (Cmax/L) calculs. Or, si s=1, la probabilité que le décodeur ait un temps de service égal à (Cmax/L) calculs est égale à la probabilité de débordement du décodeur. Par contre, avec s décodeurs, la probabilité que tous les décodeurs en parallèle aient un temps de service de (Cmax/L) calculs est égale à (probabilité de débordement)^s si tous les blocs sont indépendants. A mesure que s augmente, cette probabilité tend vers zéro. C'est pourquoi la deuxième région d'opération est plus petite que prévu lorsqu'il y a plusieurs décodeurs car le pire cas est surévalué. En réalité, il est plus plausible de calculer le gain minimal en supposant que seulement 2 ou 3 décodeurs séquentiels ont un temps de service de (Cmax/L) calculs tandis que les autres ont un temps de service de 3 ou 4 fois Cmoy calculs. Cette hypothèse donne une valeur plus réaliste du gain nécessaire à l'obtention d'une file d'attente d'un bloc. Par contre, elle ne garantit pas que le gain est minimal parce qu'il existe une probabilité non nulle que plus de 3 décodeurs aient un temps de service égal à (Cmax/L) calculs.

Il est intéressant de caractériser l'effet de l'augmentation du nombre de processeurs. La figure 5.3 donne la file d'attente moyenne à l'entrée en fonction du nombre de processeurs. Les simulations ont été faites pour Cmax=10K et Us=2. Cette figure confirme les résultats donnés plus tôt à savoir que la longueur moyenne de la file d'attente diminue avec l'augmentation du nombre de processeurs. Cette diminution est assez rapide au début mais plus le nombre de processeurs augmente, plus la diminution de la longueur moyenne de la file d'attente se fait petite. Il ne semble donc pas avantageux du moins du point de vue de la longueur moyenne de la file d'attente d'avoir un trop grand nombre de processeurs.

5.5 Effet de la variation de la taille des piles et des gains de vitesse

Cette section a pour but de présenter les effets combinés de l'augmentation du nombre de processeurs avec celle du gain de vitesse et de la diminution de la taille de la pile. Plusieurs résultats de simulations sont présentés ici.

Les figures 5.4, 5.5 et 5.6 donnent la longueur moyenne de la file d'attente à l'entrée en fonction de Cmax pour s=1,5 et 10 respectivement. Les courbes sont tracées

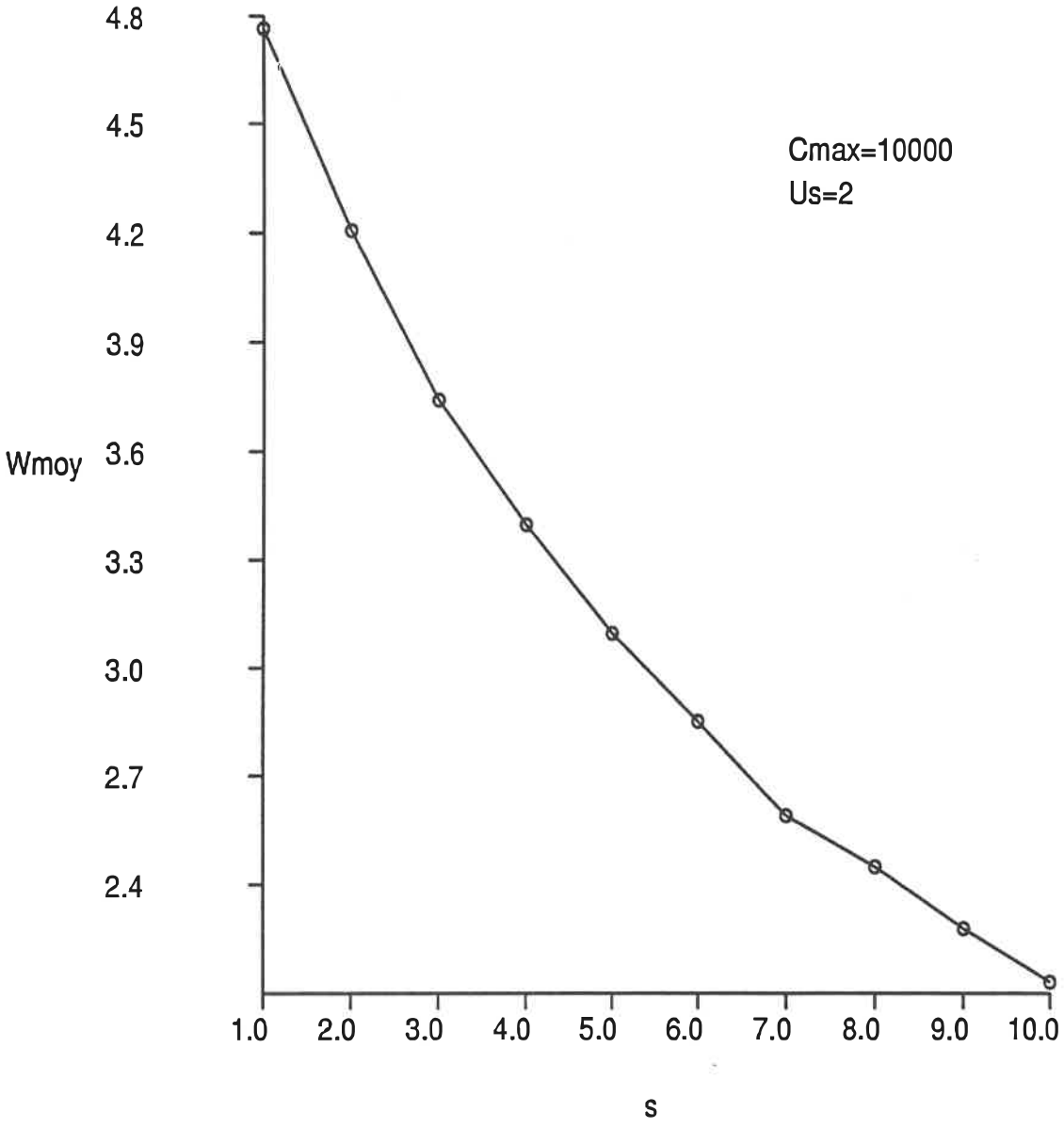


Figure 5.3 Longueur moyenne de la file d'attente à l'entrée en fonction du nombre de processeurs, L=519 et Eb/No=2.7dB

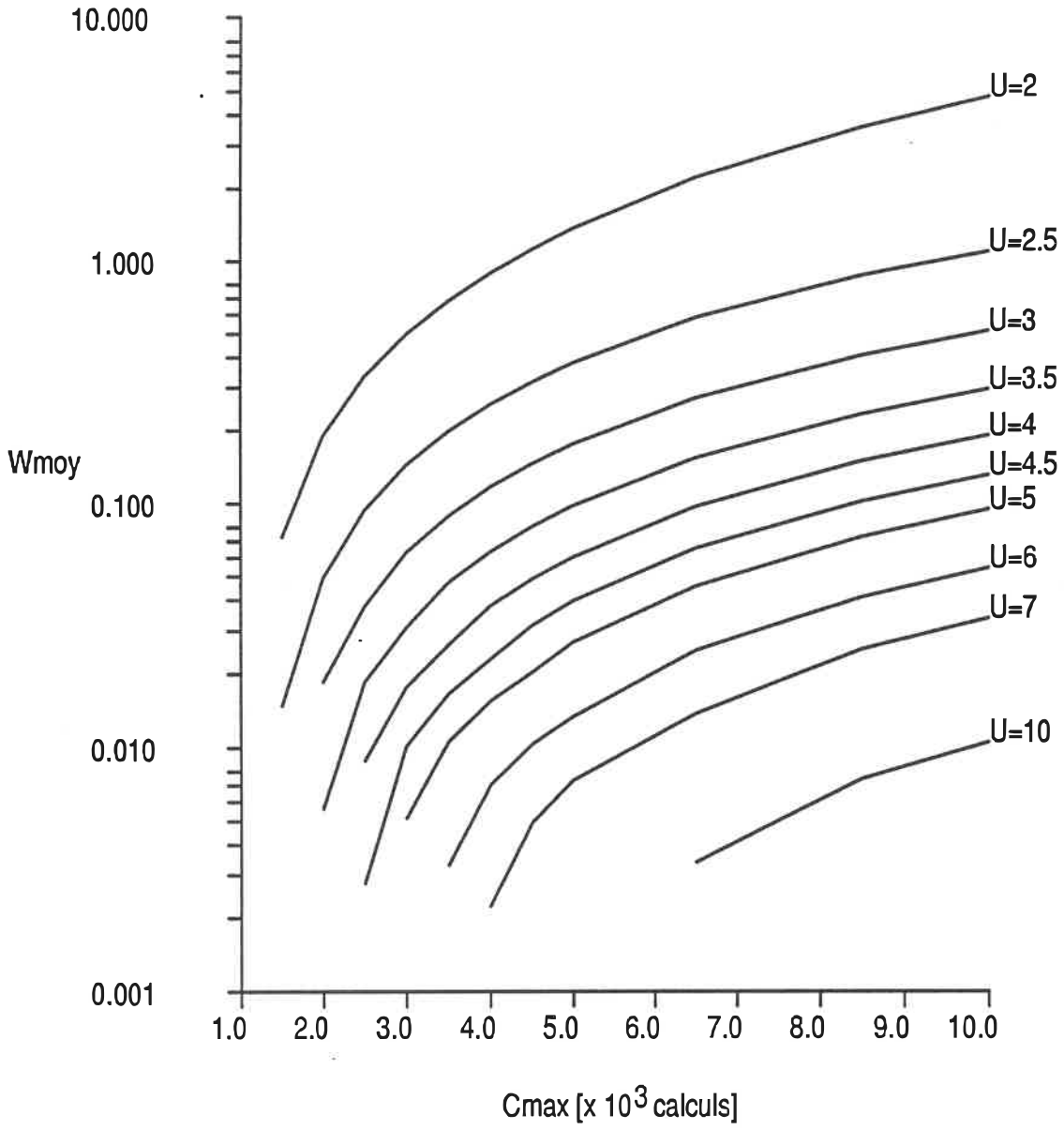


Figure 5.4 Longueur moyenne de la file d'attente à l'entrée en fonction de Cmax pour s=1, L=519 et Eb/No=2.7dB

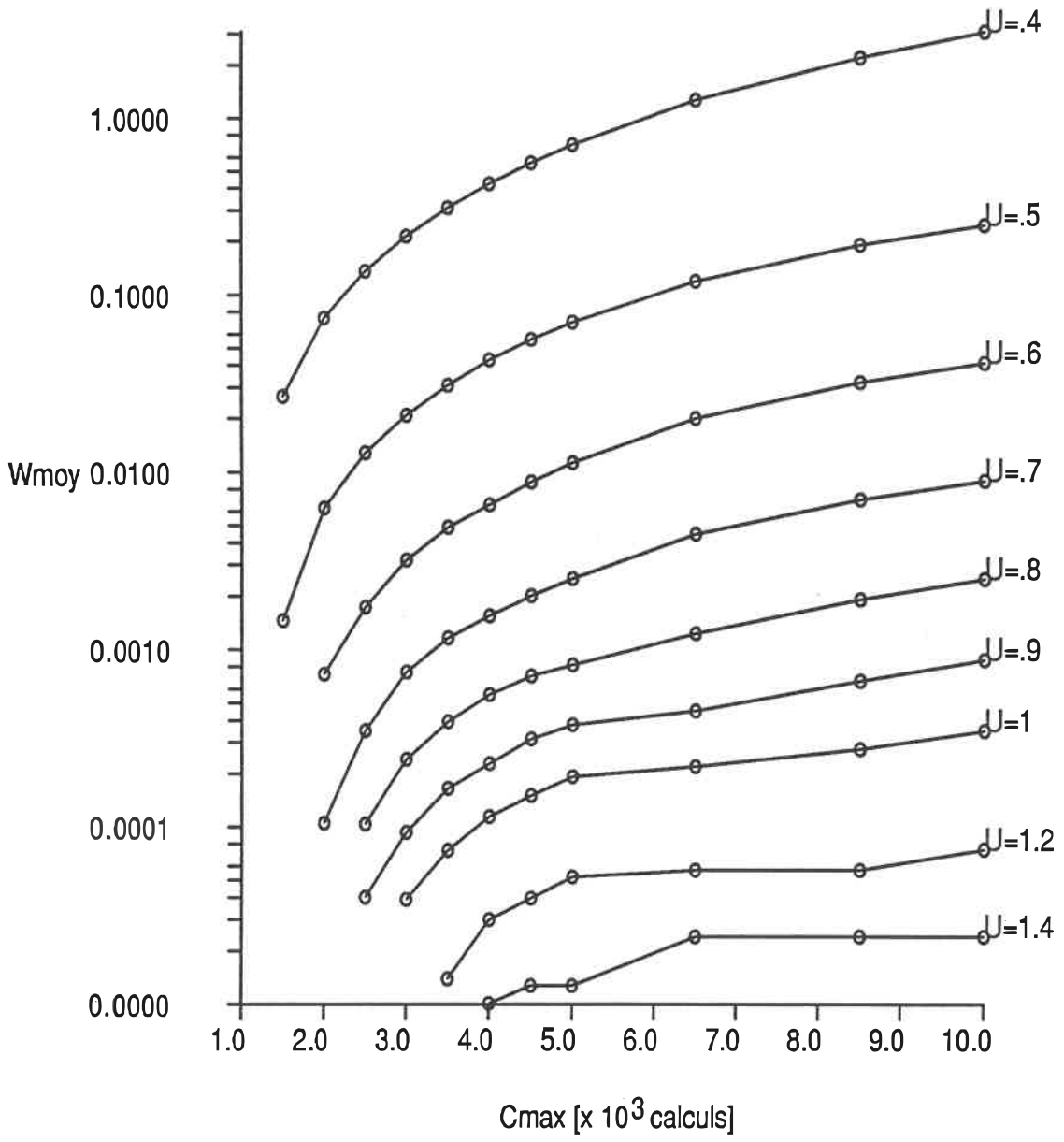


Figure 5.5 Longueur moyenne de la file d'attente à l'entrée en fonction de Cmax pour s=5, L=519 et Eb/No=2.7dB

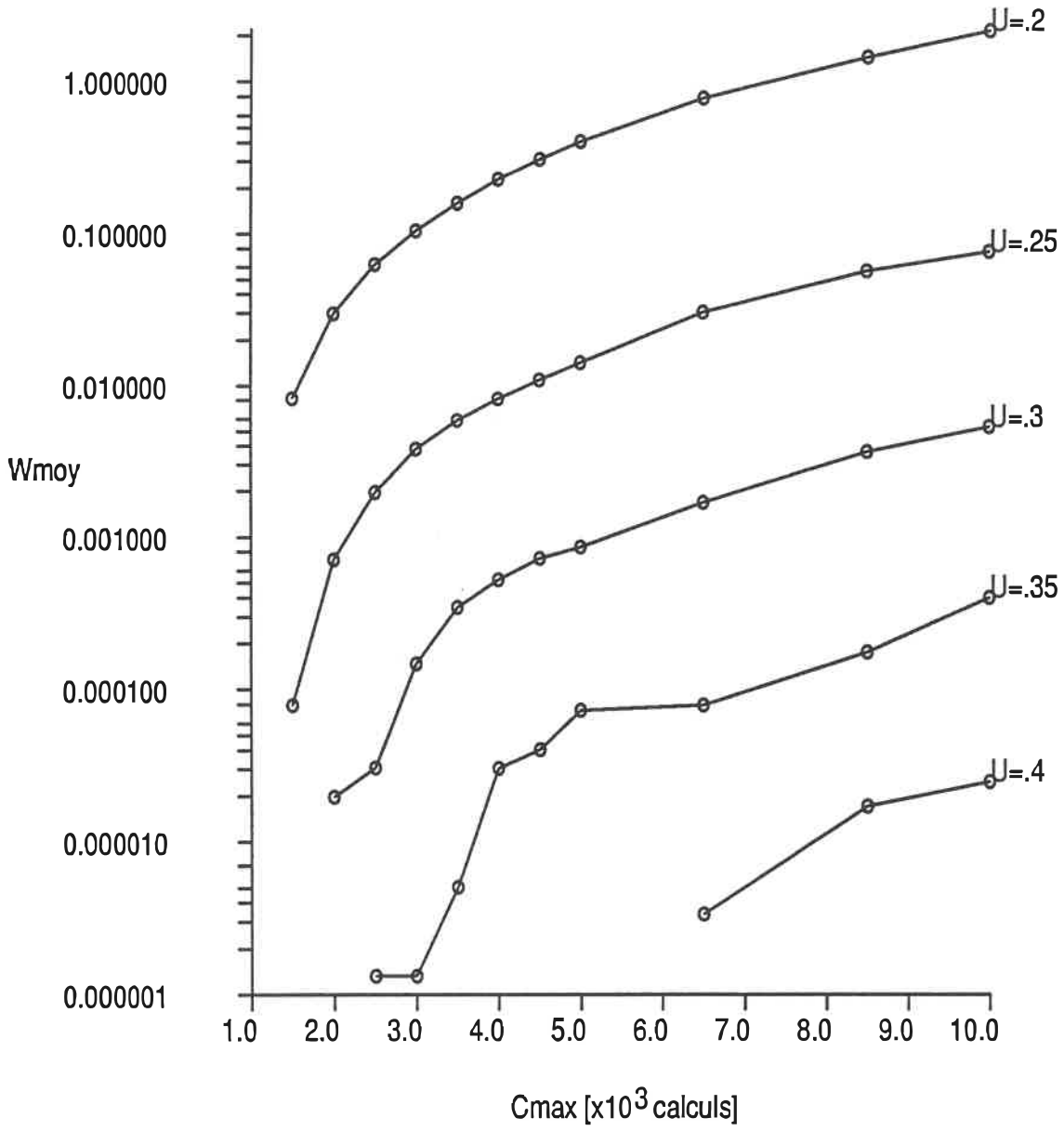


Figure 5.6 Longueur moyenne de la file d'attente à l'entrée en fonction de Cmax pour s=10, L=519 et Eb/No=2.7dB

pour différents gains. Même si les gains pour s=1, 5 et 10 sont différents, le produit Us est constant de telle sorte que ces figures peuvent être comparées entre elles. Comme il a été vu à la section 5.2, il apparaît que l'augmentation de la taille de la pile accroît la longueur moyenne de la file d'attente. Par contre, l'augmentation du gain de vitesse diminue la longueur moyenne de la file d'attente.

Des courbes de Wmax en fonction de Cmax ont été tracées pour différents gains et pour un nombre de processeurs donné. Les figures 5.7, 5.8 et 5.9 donnent quelques résultats pour s=1, 5 et 10 respectivement. La même tendance est observée sur ces courbes à savoir que Wmax augmente lui aussi avec Cmax et qu'une augmentation du gain diminue Wmax pour un Cmax donné. Il est important de constater que l'augmentation du nombre de processeurs permet de diminuer le produit Us pour lequel Wmax est atteint. Cette caractéristique a déjà été discutée à la section précédente et peut être utilisée pour déterminer un seuil minimal pour le gain.

En effet, il est intéressant de tracer sur un même graphique les courbes de Wmax en fonction de Cmax pour un produit Us fixe et s=1, 5 et 10. Le résultat est donné à la figure 5.10 pour Us=2 et Us=3.5. On constate que dépendant de la valeur de Us, le comportement de Wmax varie. En effet, lorsque Us est faible, il n'y a rien à gagner à augmenter le

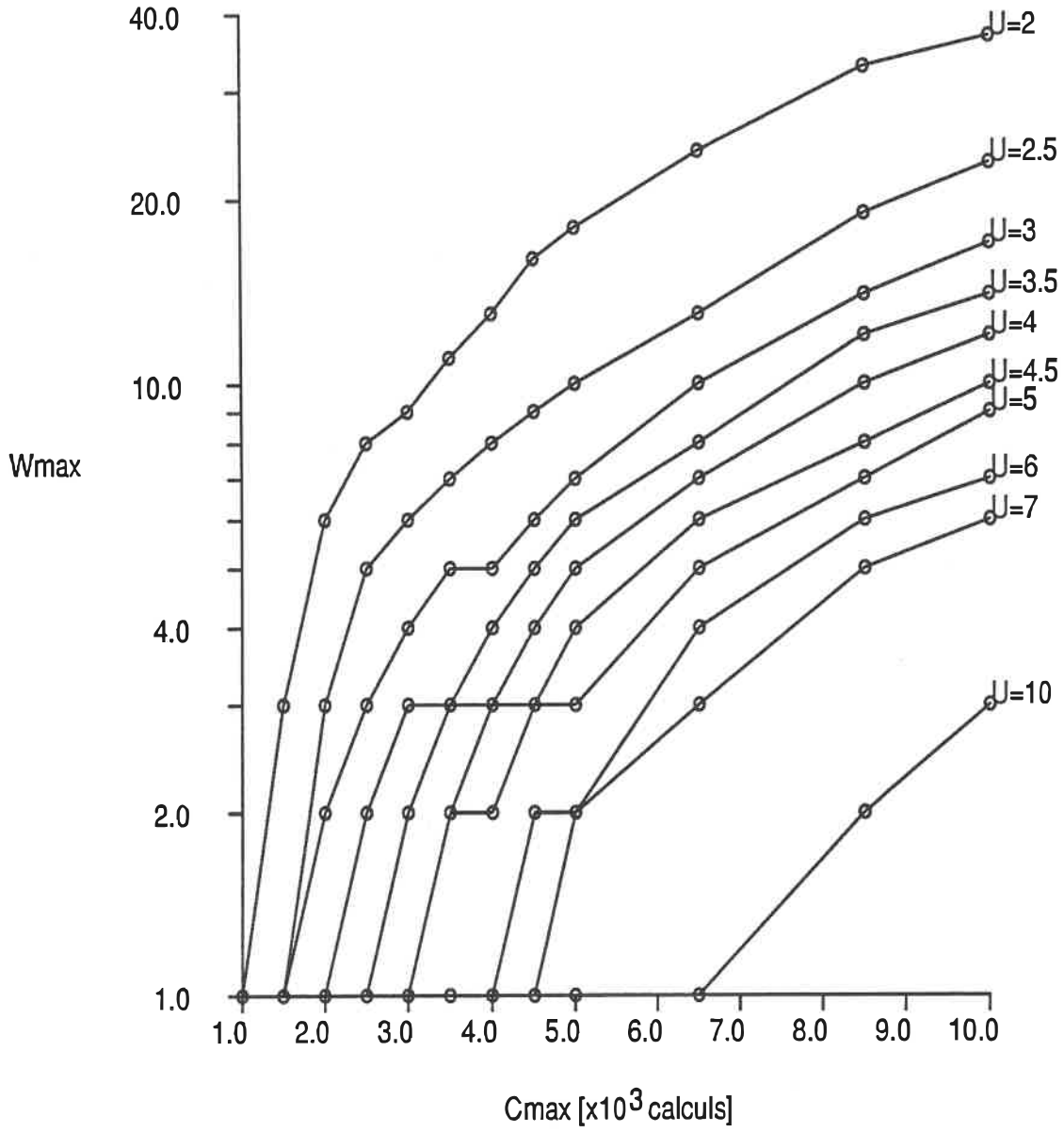


Figure 5.7 Longueur maximale de la file d'attente à l'entrée en fonction de Cmax pour s=1, L=519 et Eb/No=2.7dB

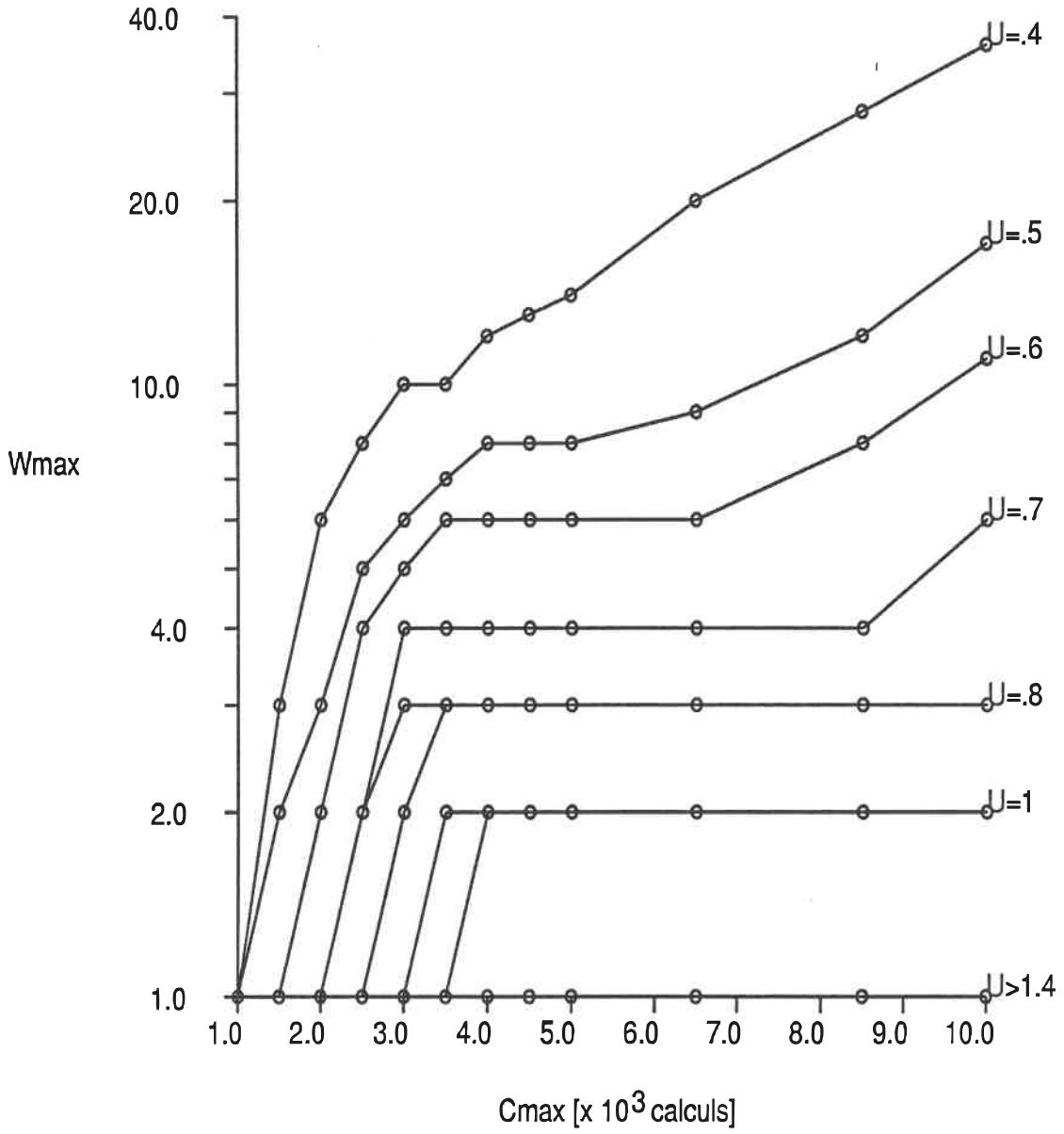


Figure 5.8 Longueur maximale de la file d'attente à l'entrée en fonction de Cmax pour s=5, L=519 et Eb/No=2.7dB

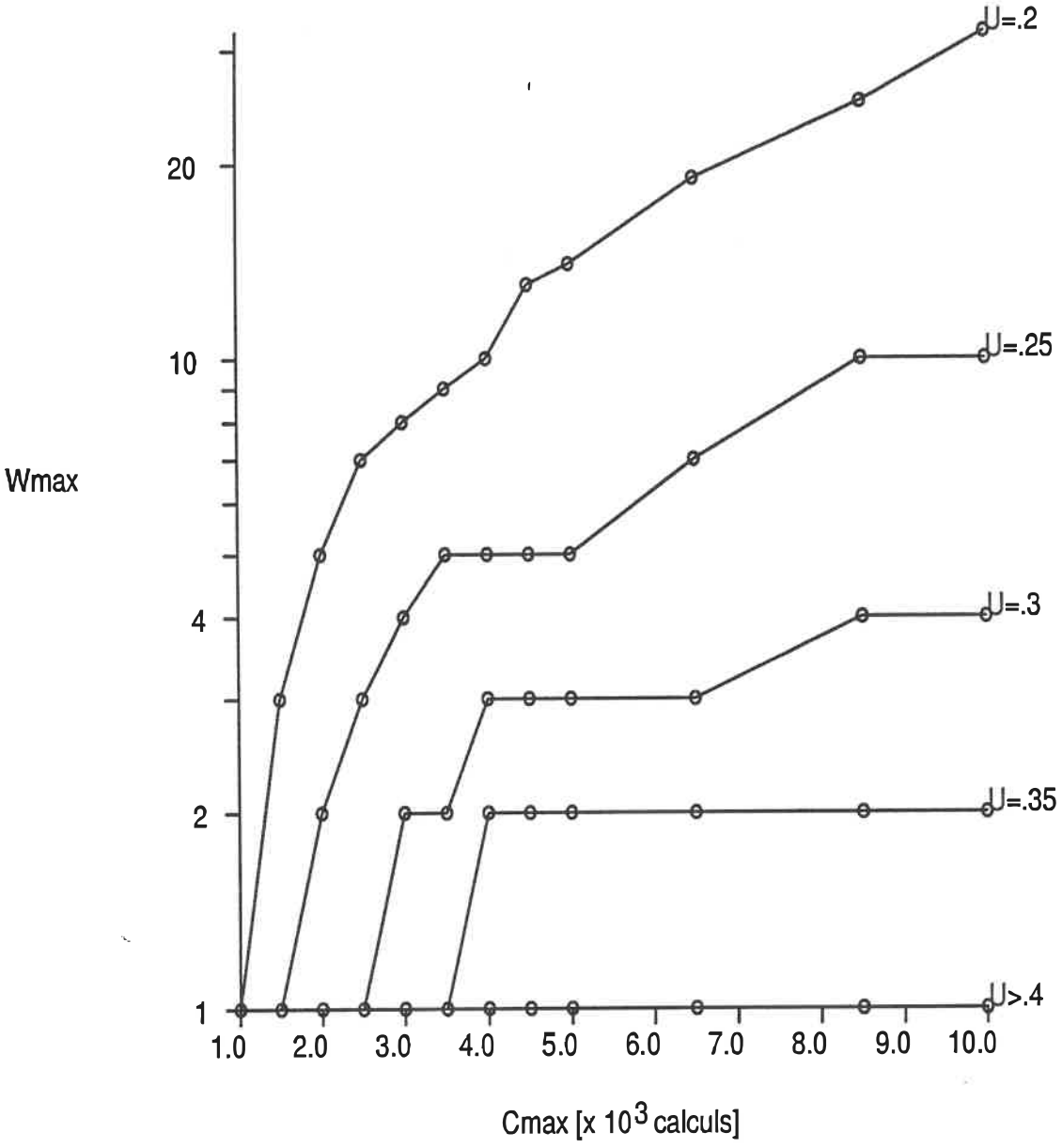


Figure 5.9 Longueur maximale de la file d'attente à l'entrée en fonction de Cmax pour s=10, L=519 et Eb/No=2.7dB

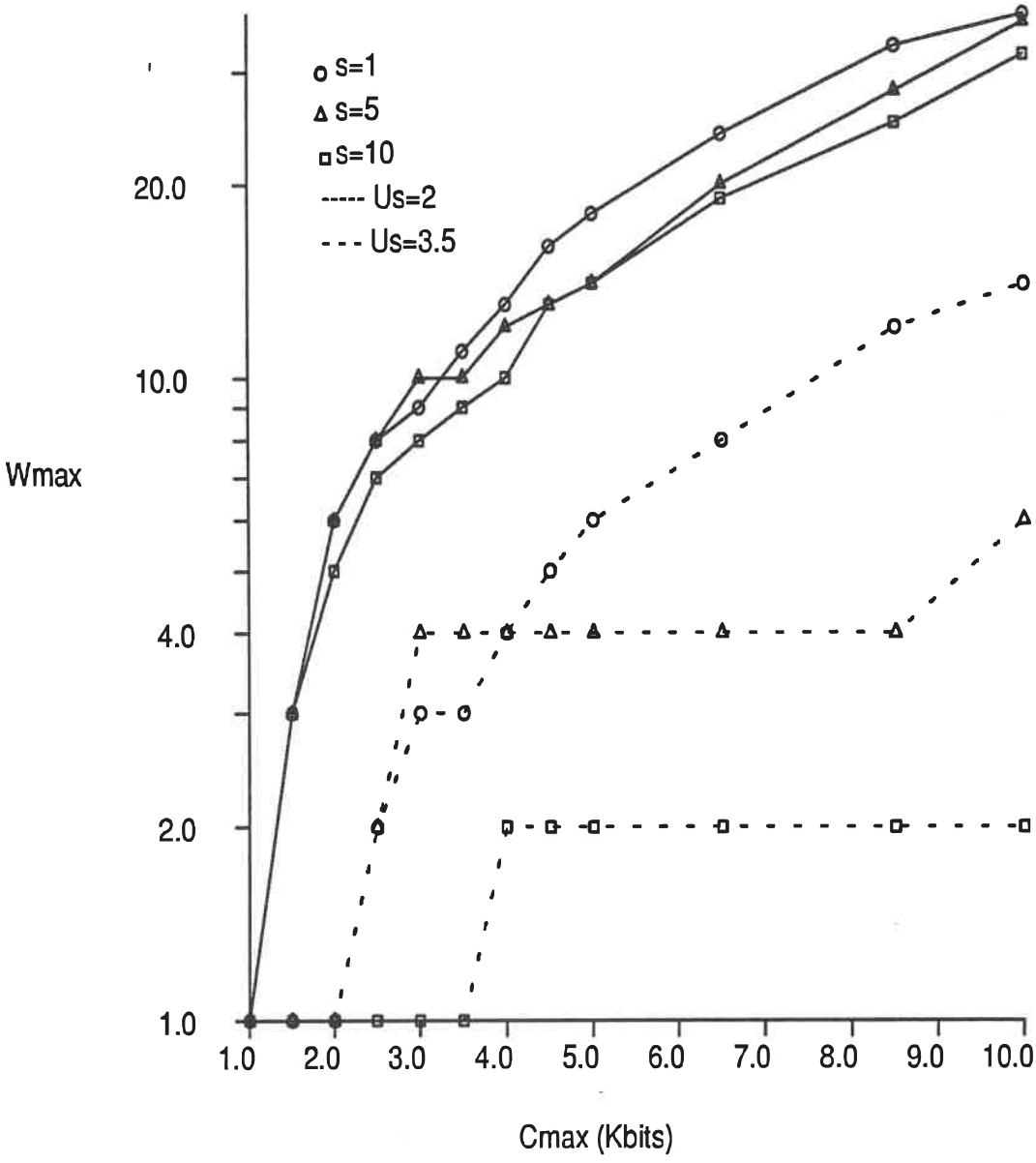


Figure 5.10 Longueur maximale de la file d'attente à l'entrée en fonction de Us=2 et Us=3.5 pour L=519 et Eb/No=2.7dB

nombre de processeurs: même si Wmoy diminue, Wmax est à peu près le même pour s=1, 5 et 10. Par ailleurs, lorsque Us est trop élevé, dans tous les cas on est proche de la région de stabilité garantie et alors Wmax est petit pour tous les systèmes. Il existe cependant une zone du produit Us où si s est petit, Wmax est assez élevé et si s est grand, Wmax est petit ou même égal à 1 bloc. Pour déterminer le seuil minimal pour le produit Us, il n'y a qu'à tracer Wmax en fonction de Us pour Cmax donné et différents nombres de processeurs. Par exemple, pour Cmax=5K et s=1, 5 et 10, la figure 5.11 donne un seuil minimal pour le produit Us de 4. Puisque le nombre de processeurs est fixé (1, 5 ou 10), on peut parler de seuil minimal pour le gain. Avec un produit Us=4, si s=10 alors Wmax=1 bloc tandis que si s=1 ou s=5, Wmax > 1. Le gain de chacun des décodeurs est différent dans chacun des cas mais comparable car Us est une constante.

C'est un phénomène intéressant lorsque le gain de vitesse d'un décodeur n'est pas suffisamment élevé pour être dans la zone de file d'attente garantie. Ce manque peut être compensé en augmentant le nombre de processeurs en parallèle. Cependant, dans ce cas, il est clair que le gain absolu du système augmente.

Il faut noter cependant qu'il est possible que le Wmax observé ne soit pas la limite supérieure de la taille du tampon. En effet, en simulant un nombre de blocs plus grand

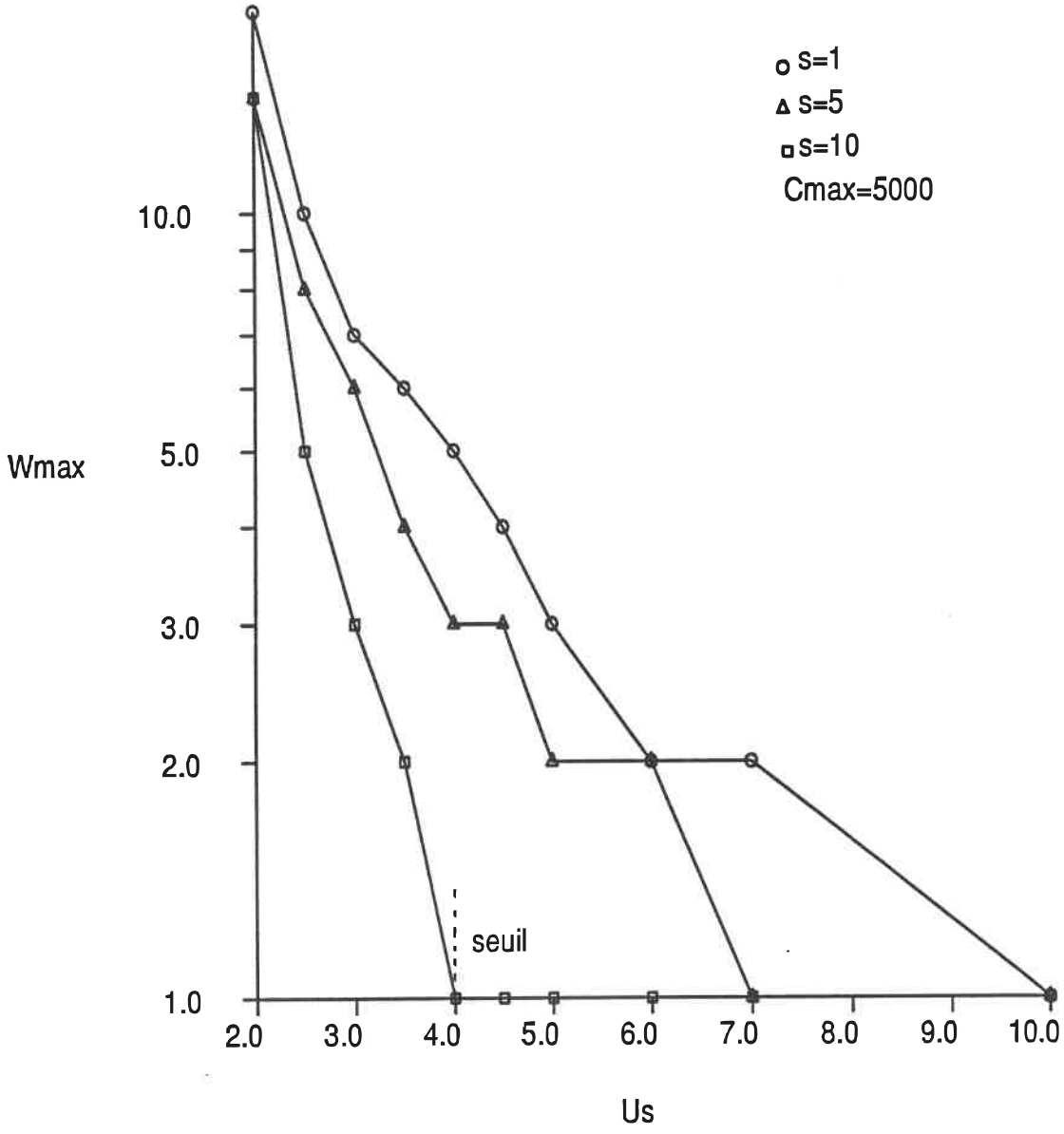


Figure 5.11 Détermination du seuil minimal du produit Us pour Cmax=5000, L=519 et Eb/No=2.7dB

que 50000, il se peut que les valeurs de Wmax soient plus élevées. Seule la condition sur le gain donnée par (4.4) peut garantir la taille du tampon. Une approche plus pratique de ce prooblème est donnée au chapitre 7 dans les recherches futures.

5.6 Fonction de répartition

Une autre caractéristique importante de la file d'attente est la fonction de densité du nombre de blocs dans le tampon d'entrée. La fonction de densité rend compte de la distribution du nombre de blocs autour de la moyenne. La figure 5.12 donne les fonctions de répartition de deux multidécodeurs, s=1 et s=10, avec Cmax=50K et un produit gain*s=2.06. D'après cette figure, l'augmentation du nombre de processeurs diminue l'étalement de la fonction de répartition.

Ces résultats de simulation sont comparés aux équations 2.2 et 2.3. L'équation 2.3 donne une borne supérieure sur la longueur moyenne de la file d'attente à l'entrée d'un système G/G/m. Cette borne est une bonne approximation de la moyenne lorsque le gain de vitesse est proche de la zone d'instabilité. Le tableau 5.3 donne les moyennes de simulation et les moyennes calculées avec l'équation 2.3.

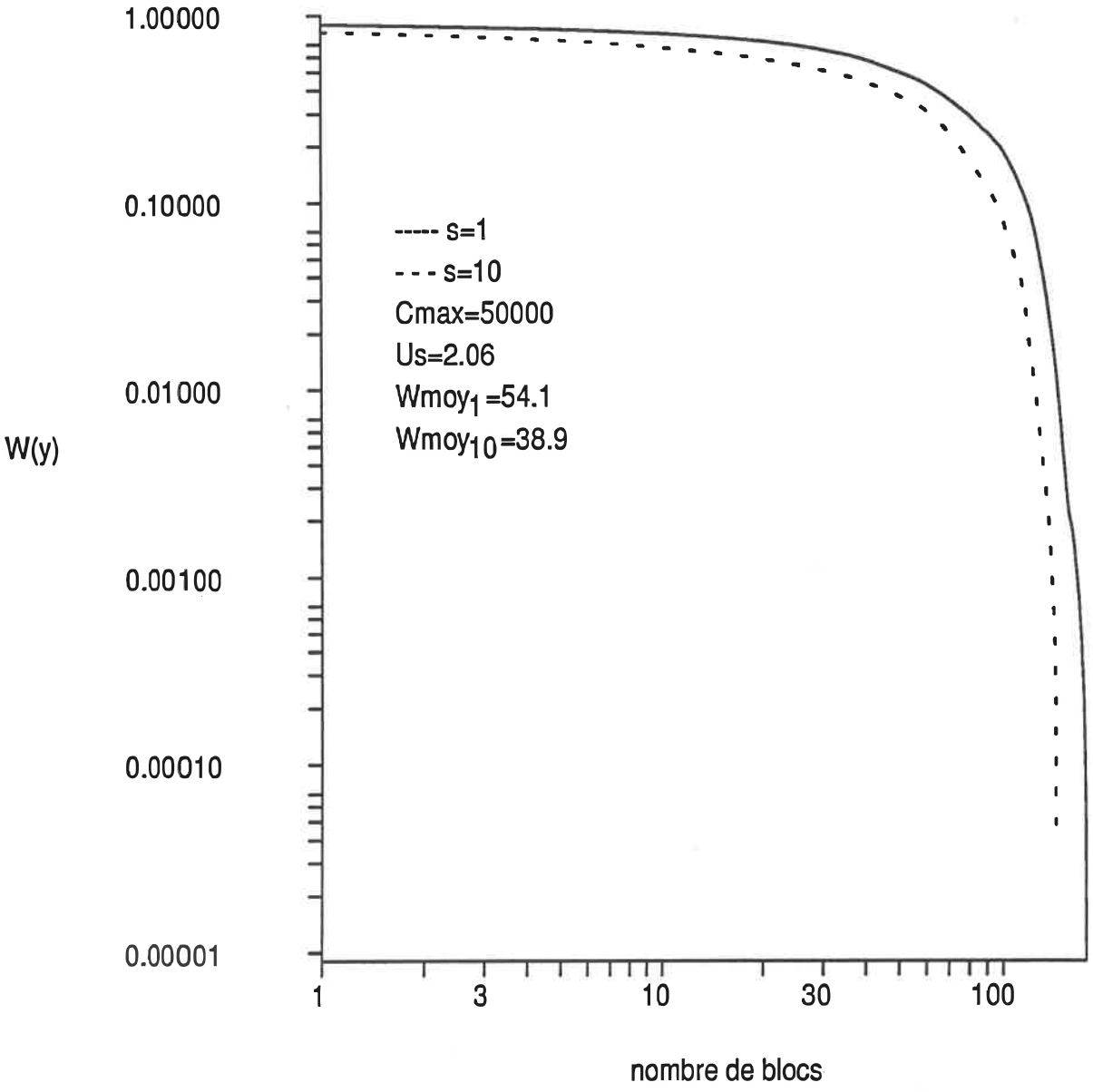


Figure 5.12 Cumulative du nombre de blocs de la file d'attente à l'entrée, L=519 et Eb/No=2.7 dB

Nombre processeurs	Cmax	Produit Us			
		2		3	
		w_{theo}	w_{sim}	w_{theo}	w_{sim}
s=1	1500	.131	.072748	.0296	0
	2500	.4221	.333452	.0818	.0377277
	4000	1.0147	.888041	.1716	.116914
s=5	1500	.131	.0267529	.0296	0
	2500	.4221	.135797	.0818	.00173408
	4000	1.0147	.425126	.1716	.00654019
s=10	1500	.131	.00823763	.0296	0
	2500	.4221	..625603	.0818	3.06358E-5
	4000	1.0147	.22785	.1716	.000523918

Tableau 5.3 Comparaison du nombre moyen de blocs dans la file d'attente à l'entrée calculé par (2.3) et les moyennes de simulation

Les paramètres sont donnés par:

$\sigma_a^2 = 0$

σ_b^2 = variance/ U^2

ρ= Cmoy/(Us)

t_m=1

La variance et Cmoy sont obtenus à l'aide de la fonction de densité de la loi de service obtenue au chapitre 3. Quelques valeurs de la variance et de Cmoy en fonction de Cmax sont données au tableau 5.1. Il faut remarquer ici que σ_b^2 n'est pas égal à la variance mais à variance/U^2. En effet, vu du tampon d'entrée, le temps de service, la moyenne et l'écart-type du serveur sont divisés par le gain de vitesse, U.

D'après le tableau 5.3, il apparaît que les résultats théoriques sont indépendants de s. En réalité, l'équation 2.3 dépend du produit Us. Or, comme il a été expliqué au début du chapitre, la comparaison de systèmes ayant un nombre différent de processeurs nécessite de maintenir le produit Us constant. C'est pourquoi, les résultats du tableau 5.3 sont identiques pour différents nombres de processeurs.

L'équation 2.3 étant valide proche de l'instabilité, Wmoy est mieux approximée quand le gain de vitesse est faible. C'est pourquoi, pour un gain donné (Us fixe), les résultats sont meilleurs lorsque Cmax augmente. De plus, les résultats

du tableau 5.3 indiquent que l'augmentation du nombre de processeurs diminue la justesse de l'approximation de Wmoy.

L'équation 2.2 permet de caractériser la fonction de répartition du nombre de blocs dans la file d'attente à l'entrée. D'après cette équation, autour de Us=Cmoy, la distribution du nombre de blocs dans la file d'attente suit une loi exponentielle. Puisque les moyennes calculées par l'équation 2.3 ne collent pas toujours aux résultats de simulation, l'équation 2.2 sera calculée à l'aide des moyennes de simulation. Les figures 5.13 et 5.14 donnent les résultats pour s=1 et s=10 respectivement. Dans les deux cas, l'approximation exponentielle constitue une borne de la distribution du nombre de blocs dans la file d'attente à l'entrée pour un nombre élevé de blocs.

5.7. File d'attente à la sortie

Cette section veut illustrer le comportement de la file d'attente à la sortie du système lors de l'augmentation du nombre de processeurs. Un tampon à la sortie est nécessaire pour régulariser le débit des blocs envoyés à l'usager mais aussi pour stocker les blocs si l'usager exige une mise en ordre avant de les recevoir. Il semble probable que la mise en ordre des blocs crée une file d'attente non-négligeable à la sortie.

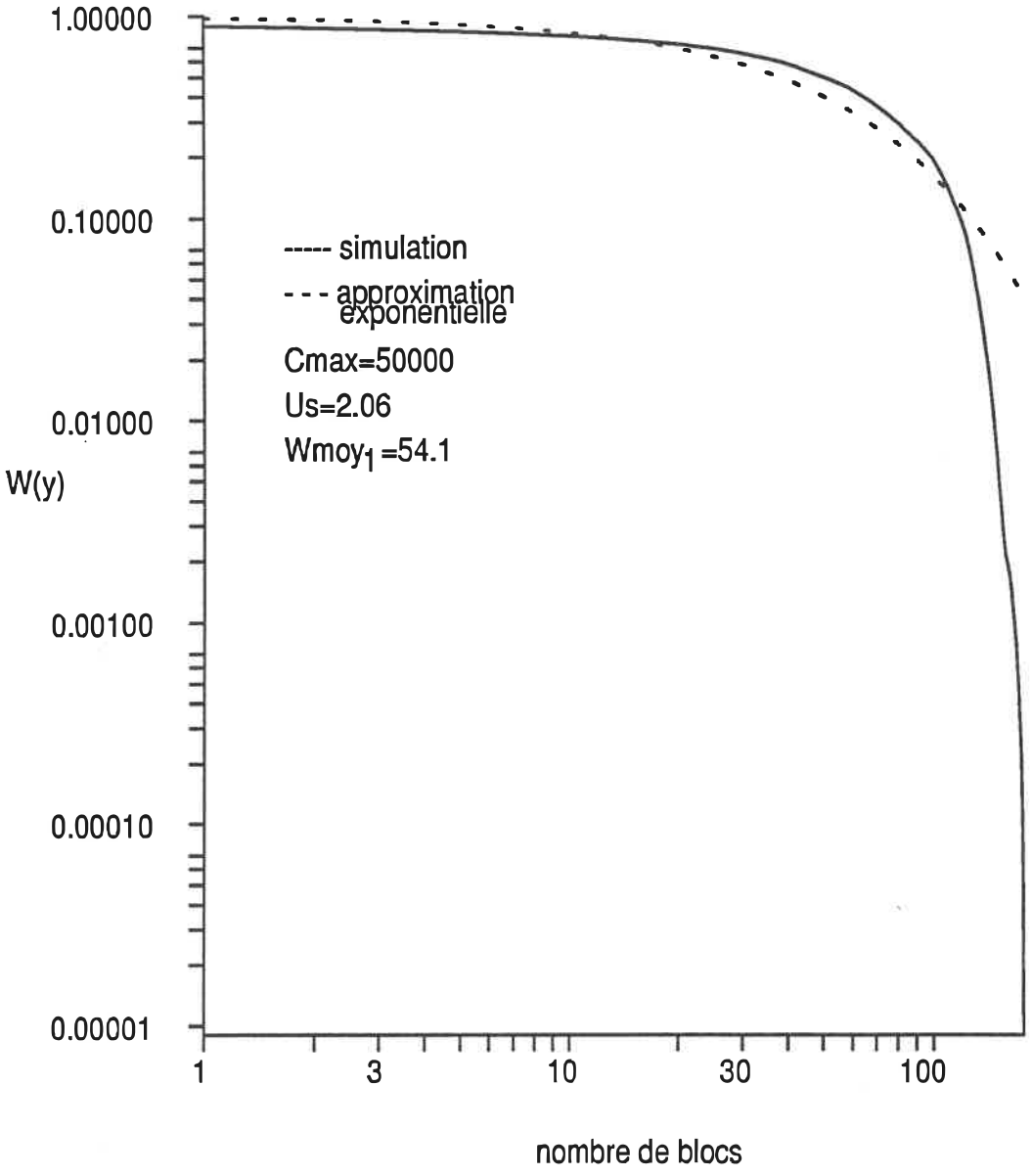


Figure 5.13 Comparaison entre la cumulative obtenue par simulation et celle obtenue avec (2.2) et (2.3) pour s=1, L=519 et Eb/No=2.7 dB

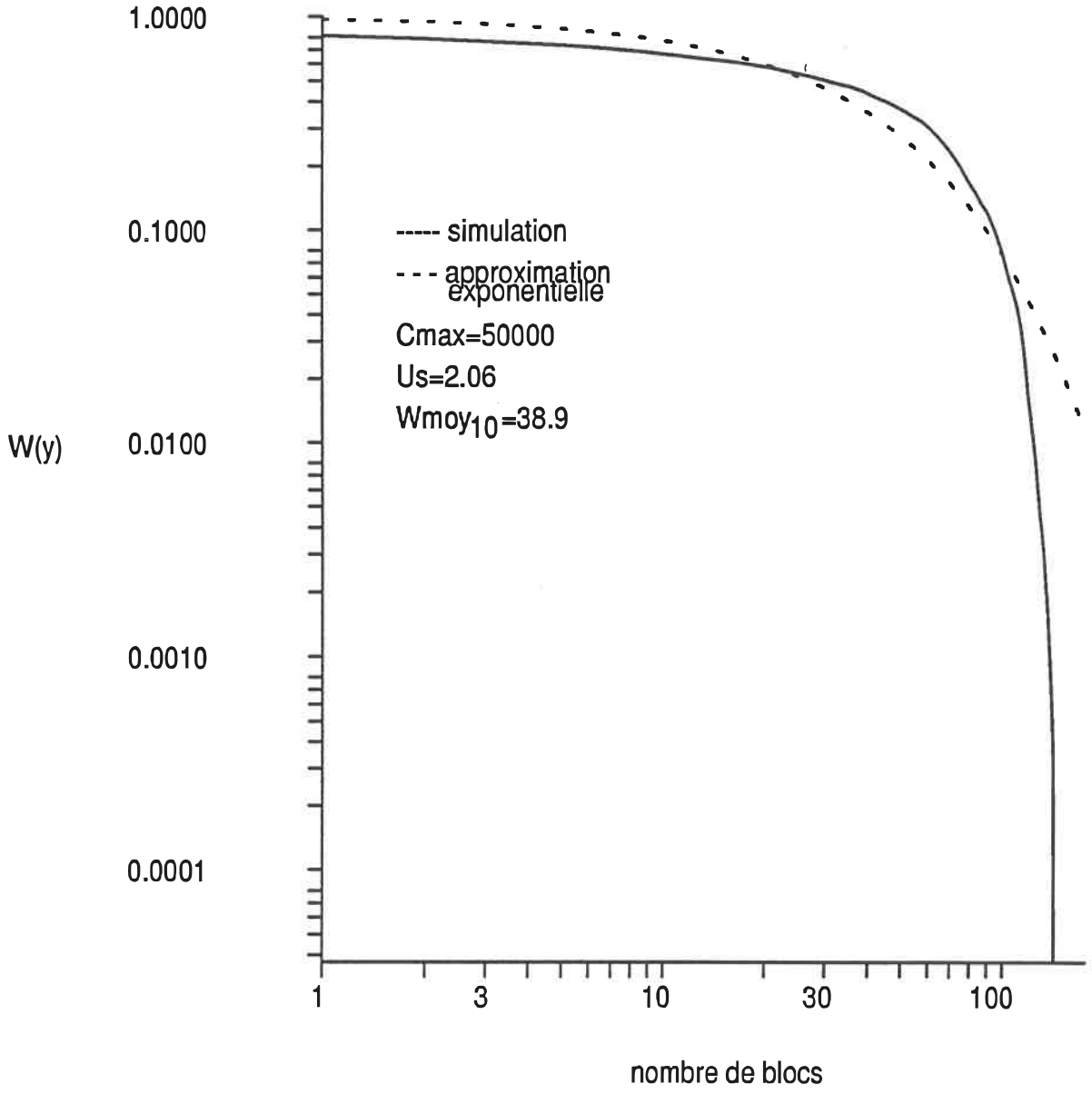


Figure 5.14 Comparaison entre la cumulative obtenue par simulation et celle obtenue avec (2.2) et (2.3) pour s=10, L=519 et Eb/No=2.7 dB

La figure 5.15 montre la longueur moyenne des files d'attente à l'entrée et à la sortie pour Cmax=10K et différents produits Us. Pour Cmax=10k, le seuil minimal du produit Us pour avoir Wmax=1 avec s=10 et Wmax> 1 pour s=5 ou 1 est de 4. Il est clair que même si l'augmentation du nombre de processeurs diminue la file d'attente à l'entrée, la file d'attente à la sortie a tendance à augmenter même pour la région du seuil et beaucoup plus fortement que la file d'attente à l'entrée peut diminuer.

Donc, lors d'une remise en ordre, l'augmentation du nombre de processeurs déplace le problème du tampon d'entrée à la sortie et l'amplifie.

5.8. Débit efficace du multidécodeur

L'expression du débit efficace obtenue à l'équation 4.3 a été vérifiée pour un gain constant et pour quatre multidécodeurs différents. En plus du débit efficace, le tableau 5.4 donne les différents paramètres qui caractérisent chacun des multidécodeurs: le nombre total de processeurs, le nombre de classes, le nombre de processeurs dans chaque classe et la taille des piles utlisées dans chaque classe. Le gain de vitesse étant le même pour tous les processeurs, il est inutile de l'inscrire dans le tableau 5.4 car il se

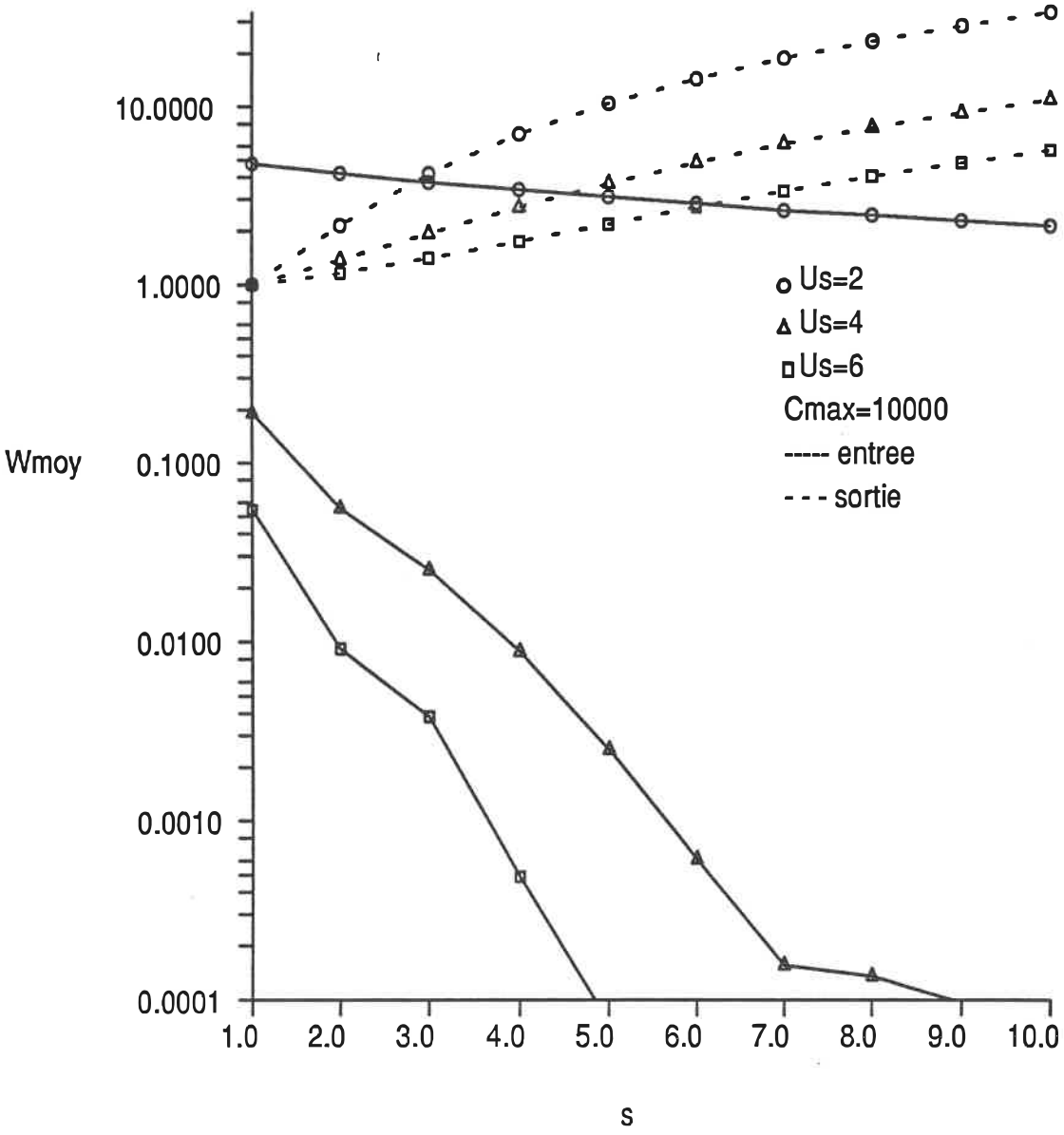


Figure 5.15 Longueur moyenne des files d'attente à l'entrée et à la sortie en fonction de s pour L=519 et Eb/No=2.7 dB

	Nombre de processeurs par classe	Cmax permis dans chaque classe	débit efficace bits/symbole	
			théo.	simul.
système 1 4 processeurs 2 classes	1^{ere} 3 2^{eme} 1	1^{ere} 1000 2^{eme} 4000	.438	.447
système 2 8 processeurs 4 classes	1^{ere} 2 2^{eme} 2 3^{eme} 2 4^{eme} 2	1^{ere} 4000 2^{eme} 3000 3^{eme} 2500 4^{eme} 1000	.469	.470
système 3 15 processeurs 2 classes	1^{ere} 5 2^{eme} 10	1^{ere} 1000 2^{eme} 10000	.468	.469
système 4 5 processeurs 5 classes	1^{ere} 1 2^{eme} 1 3^{eme} 1 4^{eme} 1 5^{eme} 1	1^{ere} 1000 2^{eme} 1500 3^{eme} 2000 4^{eme} 2500 5^{eme} 3000	.465	.465

Tableau 5.4 Comparaison du débit efficace de quelques systèmes avec ceux obtenus par (4.3)

simplifie dans l'expression 4.3. Les résultats sont comparés aux valeurs données par les simulations. Les résultats indiquent que l'expression 4.3 donne des valeurs de débit efficace proches des valeurs de simulation.

5.9. Conclusion

Les résultats présentés dans ce chapitre nous permettent de constater que l'augmentation de vitesse obtenue grâce à l'architecture multidécodeur ne se fait pas aux dépens des autres performances du système. En effet, l'augmentation du nombre de processeurs diminue la file d'attente moyenne à l'entrée. De plus, l'augmentation du nombre de processeurs nous permet d'abaisser le seuil minimal pour le gain. Ce seuil détermine la valeur minimale du gain pour lequel la file d'attente maximale est de 1 bloc.

Cependant, l'augmentation du nombre de processeurs peut causer un problème lorsqu'il est nécessaire de remettre les blocs en ordre avant de les délivrer à l'usager. En effet, dans ce cas, même si la file d'attente à l'entrée diminue, la file d'attente à la sortie augmente dans des proportions beaucoup plus grandes que la diminution de la file d'attente à l'entrée. Il reste donc à savoir si le coût du tampon nécessaire à la sortie du système est excessif par rapport au

coût du système. Cette question est discutée au chapitre 6 où le coût du système est évalué.

Finalement, malgré les avantages du multidécodeur, il faut toujours faire face au compromis entre la longueur de la file d'attente à l'entrée et le débit efficace du système. Le chapitre 6 analyse une approche permettant d'éviter ce compromis: le multidécodeur avec décodeurs de réserve.

CHAPITRE 6

PERFORMANCES DU MULTIDECODEUR AVEC DECODEURS DE RESERVE

Le but de ce chapitre est d'analyser le multidécodeur avec décodeurs de réserve représenté à la figure 4.2 et de le comparer avec le multidécodeur simple de la figure 4.1. Ce chapitre ne traite que du cas où le décodeur de réserve recommence le décodage du bloc en entier. De plus l'analyse se limite au cas où il n'y a qu'un seul décodeur de réserve.

Le chapitre débute par une description des modifications apportées au simulateur du multidécodeur simple pour inclure l'utilisation de décodeurs de réserve. Les paramètres d'intérêt ainsi que les critères d'analyse du système sont donnés dans les sections qui suivent. Ensuite, quelques résultats concernant le multidécodeur avec décodeurs de réserve sont présentés: nous avons notamment modélisé l'effet de l'augmentation du nombre de processeurs s_1 pour $Cmax_1$ fixe ainsi que l'effet de la variation de $Cmax_1$ pour des nombres de processeurs s_1 et s_2 fixes. Dans la deuxième partie du chapitre, le système est comparé avec le multidécodeur simple: les critères de comparaison et les résultats sont présentés. Toutes les simulations ont été faites avec 50000 blocs de 519 bits, un rapport signal-à-bruit de 2.7dB et un code de Johannesson de taux 1/2 ayant une longueur de contrainte K=20.

6.1 Modifications apportées au simulateur

Le simulateur a été écrit en langage C et s'exécute sur des ordinateurs SUN. La structure de données du simulateur du multidécodeur avec décodeur de réserve est constituée de 3 listes chaînées L_1, L_2 et L_3. La première liste L_1 est une liste dont les éléments sont des événements. Les éléments de la deuxième liste L_2 sont des blocs qui doivent être décodés par le décodeur de réserve tandis que ceux de la troisième liste, L_3, sont tous les blocs transmis qui attendent pour la remise en ordre avant la livraison à l'usager.

On est intéressé à connaître les distributions des files d'attente à l'entrée, devant les décodeurs de réserve, ainsi qu'à la sortie. Les tampons d'entrée, devant le décodeur de réserve et à la sortie sont respectivement notés T_1, T_2 et T_3. La longueur de T_1 est calculée chaque fois qu'un événement se produit. La longueur de T_2 est donnée par le nombre d'événements de la liste L_2 tandis que la longueur de T_3 est donnée par le nombre d'élements de la liste L_3. La première liste L_1 contient deux types d'événements: arrivée d'un bloc dans le tampon et départ d'un bloc du système. L'arrivée d'un bloc dans le tampon se fait à tous les L+K-1 bits. Le départ d'un bloc se fait lorsque le décodage de celui-ci est fini. Ces deux types d'événements génèrent

chacun des événements qui appartiennent aussi à l'un ou l'autre des deux types. La simulation consiste à retirer l'événement en haut de la liste L_I. Selon le type de cet événement, des événements sont générés et sont à leur tour insérés dans la liste L_I au bon endroit. La simulation continue avec le prochain événement placé en haut de la liste.

Les événements générés dépendent de l'événement en haut de la liste L_I. Si l'événement retiré de la liste L_I est l'arrivée d'un bloc au tampon d'entrée, alors

1- Le tampon T_I est augmenté de 1

2- Si un décodeur est en attente (les décodeurs simples ont priorité sur les décodeurs de réserve), on lui donne ce bloc:

   a) le tampon T_I est diminué de 1

   b) un temps de service est généré par la méthode I

   c) cet événement est placé dans la liste L_I au bon endroit

3- L'événement "arrivée du bloc suivant" est généré et placé dans la liste.

Si l'événement est le départ d'un bloc, alors,

1- Si le départ s'est fait d'un décodeur simple, alors

   a) on teste si le bloc a débordé

      - si oui, alors,

         - on ajoute ce bloc à L_2 et T_2 est augmenté de 1

- on teste si un décodeur de réserve est libre. Si oui
  1) le premier élément de L_2 est retiré et T_2 est diminué de 1
  2) un temps de service est généré par la méthode II.
  3) l'événement est placé dans L_1
- sinon (le bloc n'a pas débordé),
  - le bloc est inséré dans L_3 selon son ordre d'arrivée au récepteur et T_3 est augmenté de 1
  - si c'est possible des blocs sont délivrés à l'usager.

b) on vérifie l'état de T_1
- si T_1 est vide, le décodeur qui a fini est mis en attente.
- si T_1 n'est pas vide, il est diminué de 1 et un temps de service est généré par la méthode I. Ce nouvel événement est mis dans la liste L_1.

2- Sinon (le bloc provient d'un décodeur de réserve),

a) insérer le bloc dans L_3 et augmenter T_3 de 1

b) vérifier si T_2 est vide
- si non,
  1) retirer le bloc au début de L_2 et diminué T_2 de 1
  2) le temps de service est généré par la méthode II

3) l'événement est placé dans L_I

- si oui,
  - si T_I n'est pas vide
    1) T_I est diminué de 1
    2) un temps de service est généré par la méthode I
    3) l'événement est placé dans L_I
  - si T_I est vide, le décodeur est mis en attente.

Finalement, le processus est recommencé en retirant le premier élément de L_I.

La mise en ordre de la pile se fait à l'aide du temps auquel se produit chaque événement. Les temps d'arrivée des bloc sont des multiples de L+K-1. A tous les L+K-1 bits, un bloc arrive au système. Donc temps arrivée du bloc suivant = temps arrivée du bloc courant + L+K-1 (le bloc courant étant l'événement en haut de la pile).

Il y a deux possibilités pour la génération des temps de service: les méthodes I et II. La méthode I qui est utilisée pour les blocs qui n'ont pas encore été décodés se fait de la façon suivante:

1- Un nombre aléatoire est généré entre 0 et 1.

2- Le temps de service du décodeur séquentiel est déterminé à l'aide d'une table elle-même obtenue avec la loi de service trouvée au chapitre 3.

3- Du point de vue du tampon d'entrée, le temps de service obtenu en 2 doit être divisé par le gain de vitesse.

4- Finalement, le temps de la fin du décodage d'un bloc est donné par le temps de l'événement en haut de la liste + temps déterminé en 3.

La génération des temps de service par la méthode II est plus simple. Elle est utilisée pour les blocs qui ont déjà débordé. Il suffit de conserver le temps de service généré à l'étape 3 de la méthode I et de l'associer au bloc qui est décodé par un décodeur simple. Si le bloc fait déborder la pile du décodeur simple, il est inséré avec son temps de service dans L_2. Lorsque ce bloc est retiré de L_2, la fin du décodage est calculée en faisant la somme entre son temps de service associé et le temps de l'événement en haut de la pile.

Le simulateur tient aussi compte de la mise en ordre à la sortie. Chaque fois qu'un bloc est décodé, il est inséré dans une troisième liste, L_3, qui remet les blocs par ordre de

leur arrivée au système. Ces blocs sont délivrés en ordre à l'usager chaque fois qu'on le peut.

6.2 Paramètres d'intérêt

Plusieurs paramètres du multidécodeur avec décodeurs de réserve interagissent entre eux. Le but de cette section est de décrire leur interdépendance. Les paramètres du système sont les suivants:

s_1: nombre de décodeurs simples

s_2: nombre de décodeurs de réserve

U_1: gain de vitesse d'un décodeur simple

U_2: gain de vitesse d'un décodeur de réserve

$Cmax_1$: nombre de calculs maximal permis pour un décodeur simple

$Cmax_2$: nombre de calculs maximal permis pour un décodeur de réserve

Le paramètre le plus difficile à cerner est le gain de vitesse. Le gain de vitesse U_1 dépend de la technologie des décodeurs simples ainsi que du taux d'arrivée des blocs venant du canal. Il n'est pas affecté par les paramètres suivants: s_1, $Cmax_1$ et L. En fait, c'est plutôt le gain de vitesse minimal nécessaire à la stabilité du système qui

dépend de s_1, $Cmax_1$ et L. De façon plus claire, étant donné un décodeur simple faisant un nombre de calculs par seconde fixé par la technologie et étant donné un taux d'arrivée des blocs au système, il faut choisir les paramètres $Cmax_1$, s_1 et L de telle sorte que le gain de vitesse minimal nécessaire à la stabilité du système soit plus petit que le gain de vitesse du décodeur.

Le gain de vitesse, U_2, dépend de la technologie des décodeurs de réserve et du taux d'arrivée des blocs à la file d'attente devant le décodeur de réserve. Or, ce taux d'arrivée dépend de L, $Cmax_1$, s_1 et U_1. Par conséquent, U_2 dépend des paramètres des décodeurs simples. Il est clair cependant que U_2 ne dépend pas de $Cmax_2$ et s_2. De plus, comme dans le cas précédent, il faut respecter la condition de gain minimal pour la stabilité de la deuxième file d'attente.

Les autres paramètres, $Cmax_1$, $Cmax_2$, s_1, s_2, ne dépendent d'aucun paramètre. Ils sont choisis pour rencontrer les conditions de stabilité du système et pour en fixer le débit efficace.

6.3 Critères d'analyse des performances

L'analyse des performances du multidécodeur avec décodeurs de réserve peut se faire de plusieurs façons car il y a beaucoup plus de paramètres qui varient que dans le cas

du multidécodeur simple. L'analyse faite dans ce chapitre suit les mêmes lignes que celle faite pour le multidécodeur simple:

- distribution des files d'attente pour s_1 qui varie avec U_1s_1 et $Cmax_1$ donnés.
- distribution des files d'attente pour $Cmax_1$ qui varie avec U_1 et s_1 fixés.

Cette analyse consiste à déterminer les files d'attente moyennes en fonction de U_1s_1 pour un nombre variable de processeurs ou pour $Cmax_1$ variable. Pour normaliser la comparaison, U_1s_1 est maintenu constant lorsque s_1 ou $Cmax_1$ varie. Puisqu'ici il y a deux classes de décodeurs, il faut déterminer U_2 et s_2. Le paramètre s_2 a été fixé arbitrairement à 1 pour faciliter l'analyse. Quant au paramètre U_2, deux possibilités s'offrent à nous: puisque les valeurs de U_1s_1 sont balayées, il faut soit fixer U_2 pour tous les points, soit le faire varier avec la variation de U_1s_1.

Fixer U_2s_2 consiste à fixer U_2 car s_2 a été choisi égal à 1 pour faciliter l'analyse. Dans ce cas, U_2 est calculé pour satisfaire la condition de stabilité avec la valeur maximale de U_1. D'un autre côté, U_2s_2 peut varier avec U_1s_1. Dans ce cas, la condition de gain minimal pour U_2 doit être revérifiée pour

tous les points. De plus, on veillera à maintenir un rapport constant entre U_1s_1 et U_2s_2.

Cette dernière approche a été retenue car dans ce cas, il est plus facile dissocier les effets sur les files d'attente de la variation de divers paramètres. Par conséquent, dans les résultats qui suivent, $U_1s_1+U_2s_2$ est constant pour U_1s_1 donné quelque soit s_1 ou $Cmax_1$. Par contre, lorsque U_1s_1 varie, U_2s_2 varie aussi. On s'assure cependant que pour toutes les valeurs de U_1s_1, $U_2s_2=kU_1s_1$ où k est une constante.

Evidemment , ce choix est discutable. Nous avons identifié d'autres façons de voir le problème, mais leur analyse déborde le cadre de ce mémoire. Les sections qui suivent présentent des résultats préliminaires.

6.4 Effet de l'augmentation du nombre de processeurs s_1

Il est important de s'assurer que les hypothèses, posées au chapitre 4, permettant de calculer les conditions limites des files d'attente tiennent en tout temps. En effet, il a été posé au chapitre 4 qu'en régime permanent et si le système est bien balancé, tout le flux venant du canal va aux s_1 décodeurs simples tandis que les s_2 décodeurs complexes ne reçoivent que les blocs qui ont déjà débordé. Ces hypothèses

sont valides si le nombre s_1 est élevé et si s_2 est faible par rapport à s_1.

Pour alléger le texte une notation a été introduite. Par exemple, le multidécodeur simple du chapitre 5 est noté système 1 tandis que le multidécodeur avec le décodeur de réserve est noté système 2. De plus, la file d'attente moyenne à l'entrée est appelée F_1, celle devant le décodeur de réserve, F_2 et celle à la sortie F_3.

Cette section a pour but de déterminer l'effet sur les files d'attente moyennes, F_1, F_2 et F_3, de l'augmentation du nombre de processeurs s_1 tout en maintenant s_2, $Cmax_1$ et $Cmax_2$ fixes. Comme il a été montré au chapitre 5, lorsqu'il y a un nombre différent de processeurs, il faut normaliser le produit Us pour être en mesure de comparer les diverses solutions. Puisqu'il y a deux classes de décodeurs, la somme $U_1s_1 + U_2s_2$ est maintenue constante lorsque s_1 varie. La file d'attente moyenne est tracée en fonction du produit U_1s_1. En effet, d'après la condition de stabilité de la deuxième file d'attente (équation 4.4), on constate que U_2s_2 dépend du produit U_1s_1. Or, même si s_1 varie, le terme U_1s_1 est maintenu constant pour la comparaison et donc le produit U_2s_2 est constant pour tout s_1 étant donné un produit U_1s_1 fixe. Par conséquent, les points des figures peuvent être comparés entre eux de deux façons. Dans la première, les points qui sont sur la même colonne c.-à-d. pour s_1 qui varie et U_1s_1 fixe

sont comparés entre eux. Dans ce cas, la somme $U_1s_1+U_2s_2$ est constante. Dans la deuxième, les points qui sont sur la même courbe sont comparés entre eux c.-à-d. pour s_1 fixe et U_1s_1 qui varie. Dans ce cas, $U_1s_1+U_2s_2$ n'est pas constant mais $U_2s_2=kU_1s_1$ où k est une constante. La figure 6.1 donne les résultats de simulation pour $Cmax_1=2000$ avec $s_2=1$ et $Cmax_2=10000$. Dans la simulation présentée à la figure 6.1, U_2 varie de 0.81 à 2.012 pour respecter la condition de stabilité de la deuxième file d'attente tandis que U_1s_1 varie de 1.6 à 4. La constante k est donc égale à .505. Comme dans le cas du multidécodeur simple, l'augmentation du nombre de processeurs diminue la file d'attente moyenne F_1, et augmente la file d'attente moyenne F_3. Ce phénomène s'explique facilement. Le fait d'augmenter le nombre de processeurs n'enlève pas les blocs difficiles. Il doivent toujours être décodés par le système. Mais, lorsqu'une remise en ordre à la sortie est exigée, le temps pour décoder un bloc devient proportionnel au nombre de serveurs dans le système. En effet, pendant qu'un bloc bloque le décodeur de réserve, les autres serveurs continuent à décoder des blocs de telle sorte qu'il faudra accumuler ces blocs dans un tampon à la sortie en attendant la fin du décodage du bloc difficile. La file d'attente à la sortie est donc fortement reliée au nombre de serveurs. De plus, il est clair qu'elle est aussi fortement dépendante du gain de vitesse du décodeur de réserve. Donc, elle dépend de la

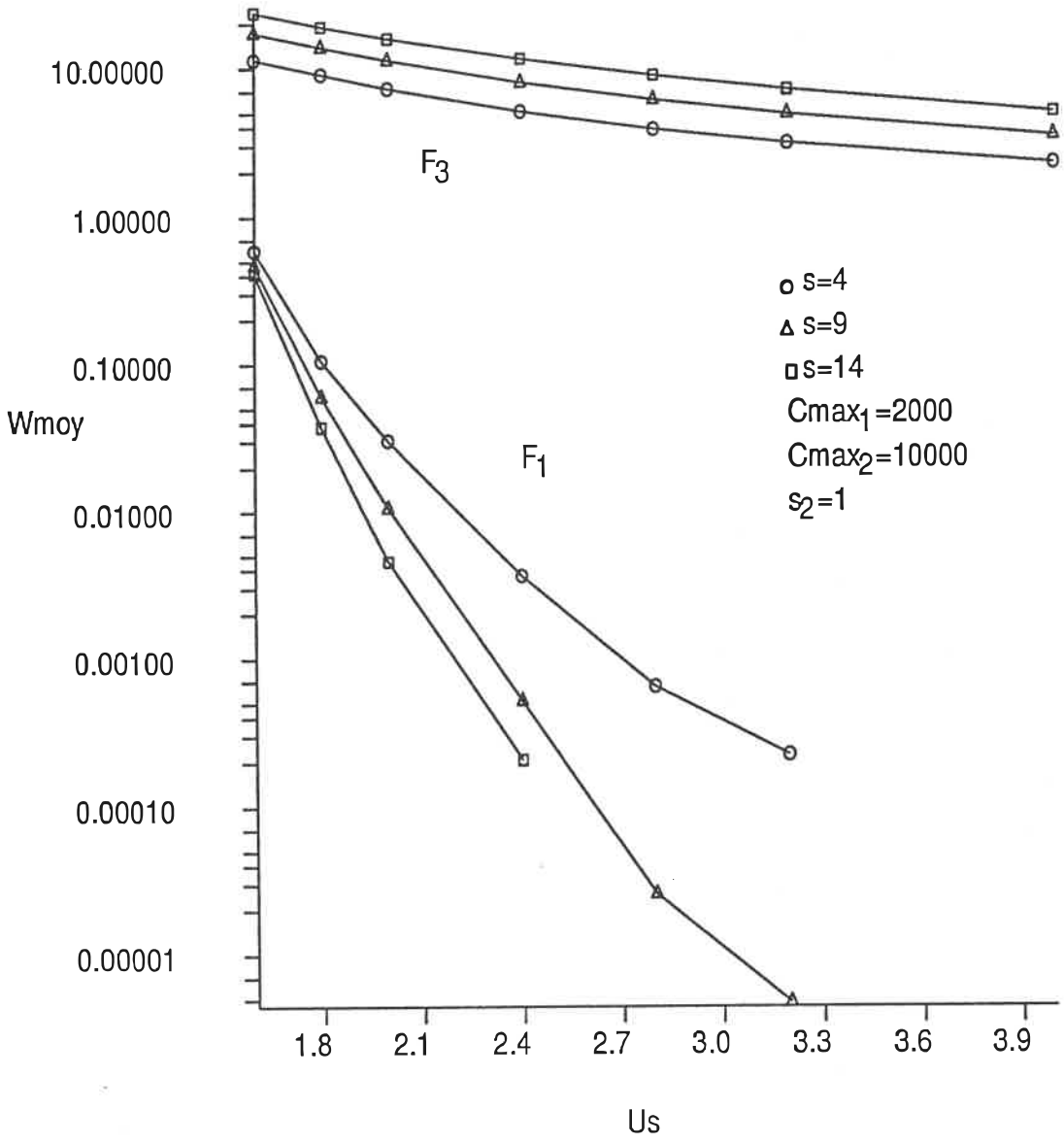


Figure 6.1 Longueur moyenne des files d'attente F_1 et F_3 en fonction de Us pour L=519 et Eb/No=2.7 dB

fraction du gain total U_1s_1 allouée au décodeur de réserve. De même, F_2 est fortement fonction de cette fraction. La file d'attente moyenne F_2 n'est pas représentée ici car elle n'est pas affectée par l'augmentation du nombre de processeurs s_1. En effet, puisque U_2s_2 est constant pour un produit U_1s_1 fixe, la file d'attente F_2 est constante pour tout s_1 étant donné un produit U_1s_1. De plus, une augmentation du gain de vitesse U_1 pour s_1 donné diminue les files d'attente moyennes à l'entrée et à la sortie. Il faut cependant constater que la diminution de la file d'attente à la sortie est très minime et qu'il n'y a donc pas avantage à augmenter U_1 pour diminuer F_3 moyen.

6.5 Effet de la variation de $Cmax_1$ pour s_1 et s_2 fixes

L'effet de la variation de $Cmax_1$ a été analysé pour $Cmax_1$=1000, 1500, 2000 et $Cmax_2$=10000. Les figures 6.2 à 6.4 donnent les résultats pour s_2=1 et s_1=4, 9, 14 respectivement. Ces figures peuvent être comparées car le produit U_1s_1 est constant. Comme il a été vu au chapitre 5, l'augmentation de $Cmax_1$ augmente la file d'attente moyenne à l'entrée. Par contre, la file d'attente moyenne devant le décodeur de réserve diminue. Ce dernier phénomène s'explique parce que l'augmentation de $Cmax_1$ implique une diminution de la probabilité de débordement des blocs. Il y a donc moins de blocs qui se rendent au décodeur de réserve pour un gain de

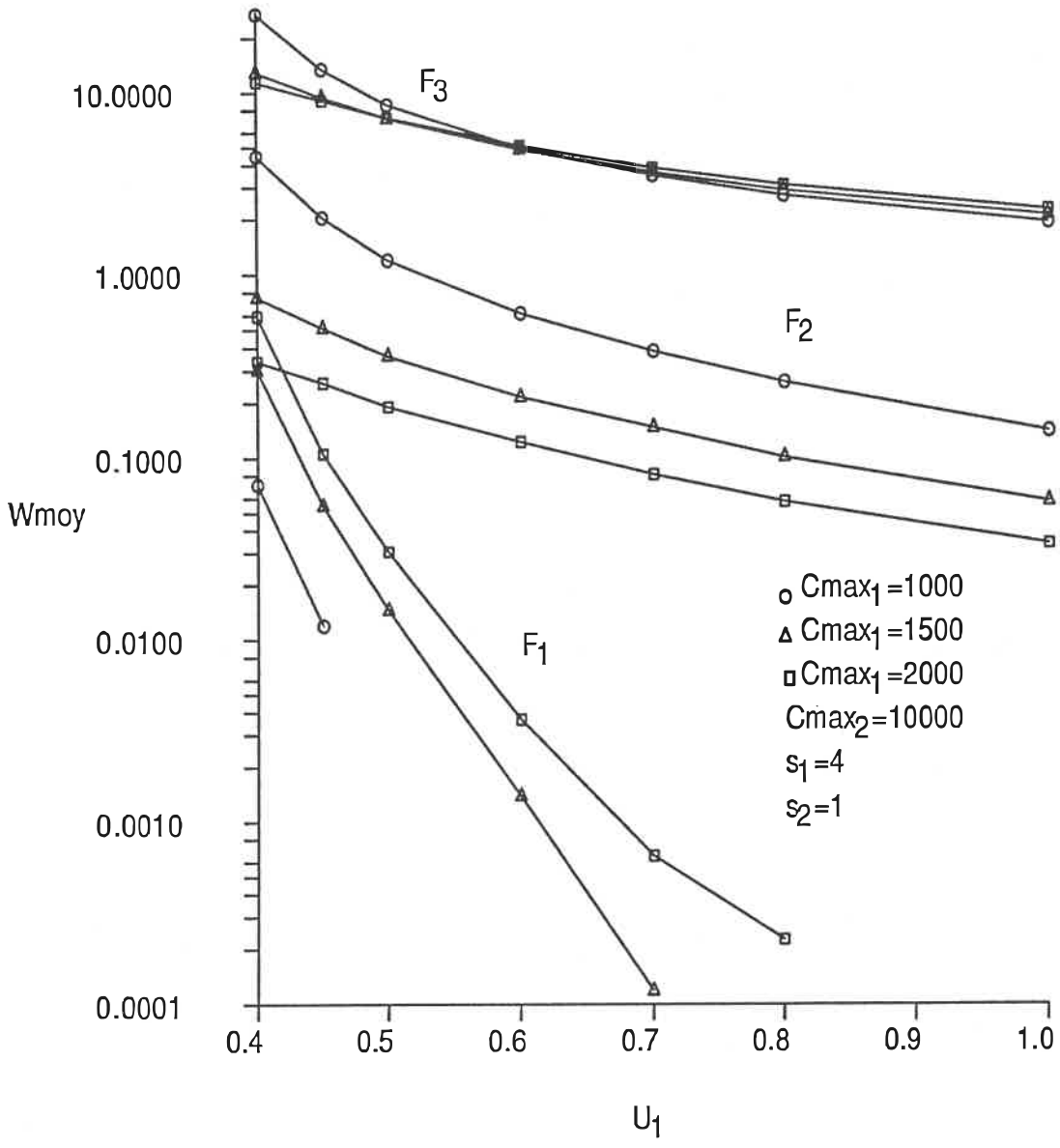


Figure 6.2 Longueur moyenne des files d'attente F_1 et F_3 en fonction de U_1 pour s_1=4, L=519 et Eb/No=2.7 dB

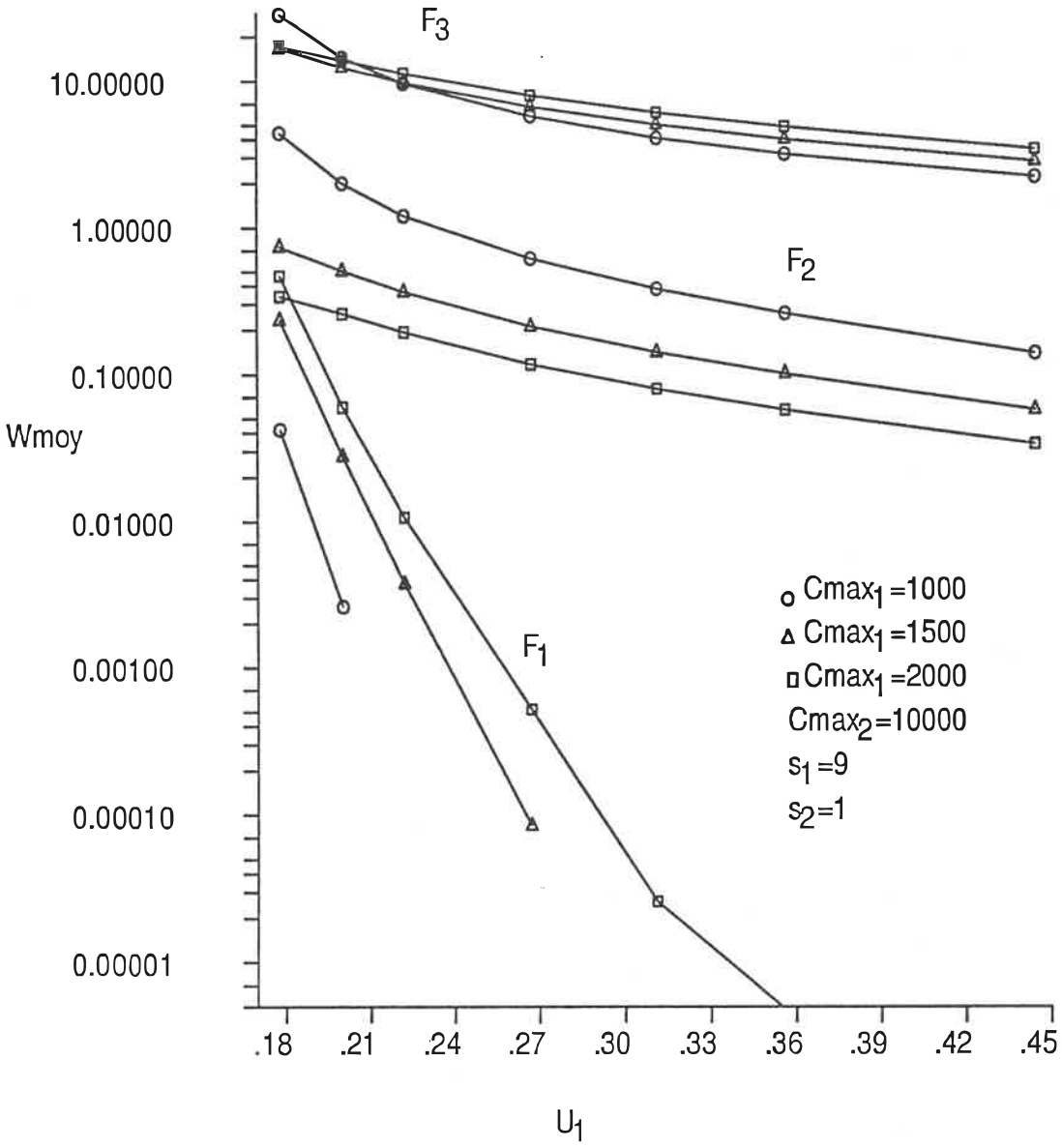


Figure 6.3 Longueur moyenne des files d'attente F_1 et F_3 en fonction de U_1 pour s_1=9, L=519 et Eb/No=2.7 dB

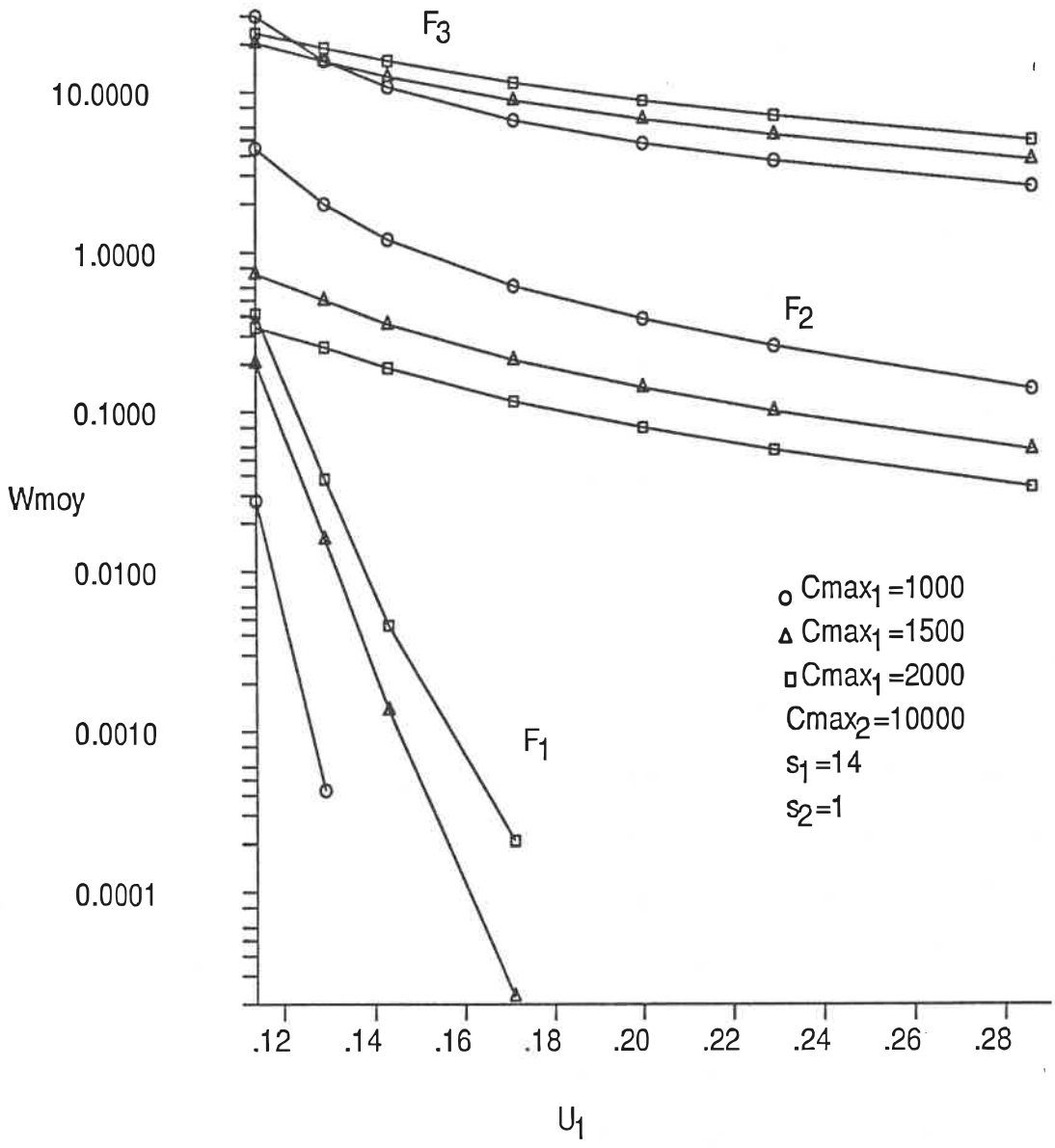


Figure 6.4 Longueur moyenne des files d'attente F_1 et F_3 en fonction de U_1 pour s_1=14, L=519 et Eb/No=2.7 dB

vitesse U_2 donné. Ici encore U_1s_1 varie de 1.6 à 4 et $U_2s_2=U_2$ varie de .81 à 2.012 avec $s_2=1$. Le comportement de la file d'attente à la sortie est quant à lui plus complexe. En effet, lorsque U_1s_1 est faible, l'augmentation de $Cmax_1$ diminue la file d'attente à la sortie. Par contre, lorsque U_1s_1 est élevé, l'augmentation de $Cmax_1$ augmente la file d'attente à la sortie. Cette augmentation n'est cependant pas très grande. Il apparaît donc qu'il est inutile d'essayer de tirer avantage de ce minimum.

L'effet de l'augmentation de $Cmax_1$ étant si diversifié, il est avantageux d'observer non pas les files d'attente séparément mais plutôt de tracer les sommes des files d'attente moyennes $F_1+F_2+F_3$ ou F_1+F_2 selon que la mise en ordre des blocs est effectuée ou non avant la livraison à l'usager. La figure 6.5 donne les résultats pour $s_1=4$, $s_2=1$ et $Cmax_2=10000$. Il s'avère que globalement, l'augmentation de $Cmax_1$ diminue la somme des files d'attente moyennes $F_1 + F_2$. L'effet global sur la somme des trois files d'attente est le même que celui observé précédemment sur F_3 seul. Ce phénomène était prévisible puisque si $Cmax_1$ augmente, il y a moins de blocs qui débordent et par conséquent moins de blocs se rendent au décodeur de réserve.

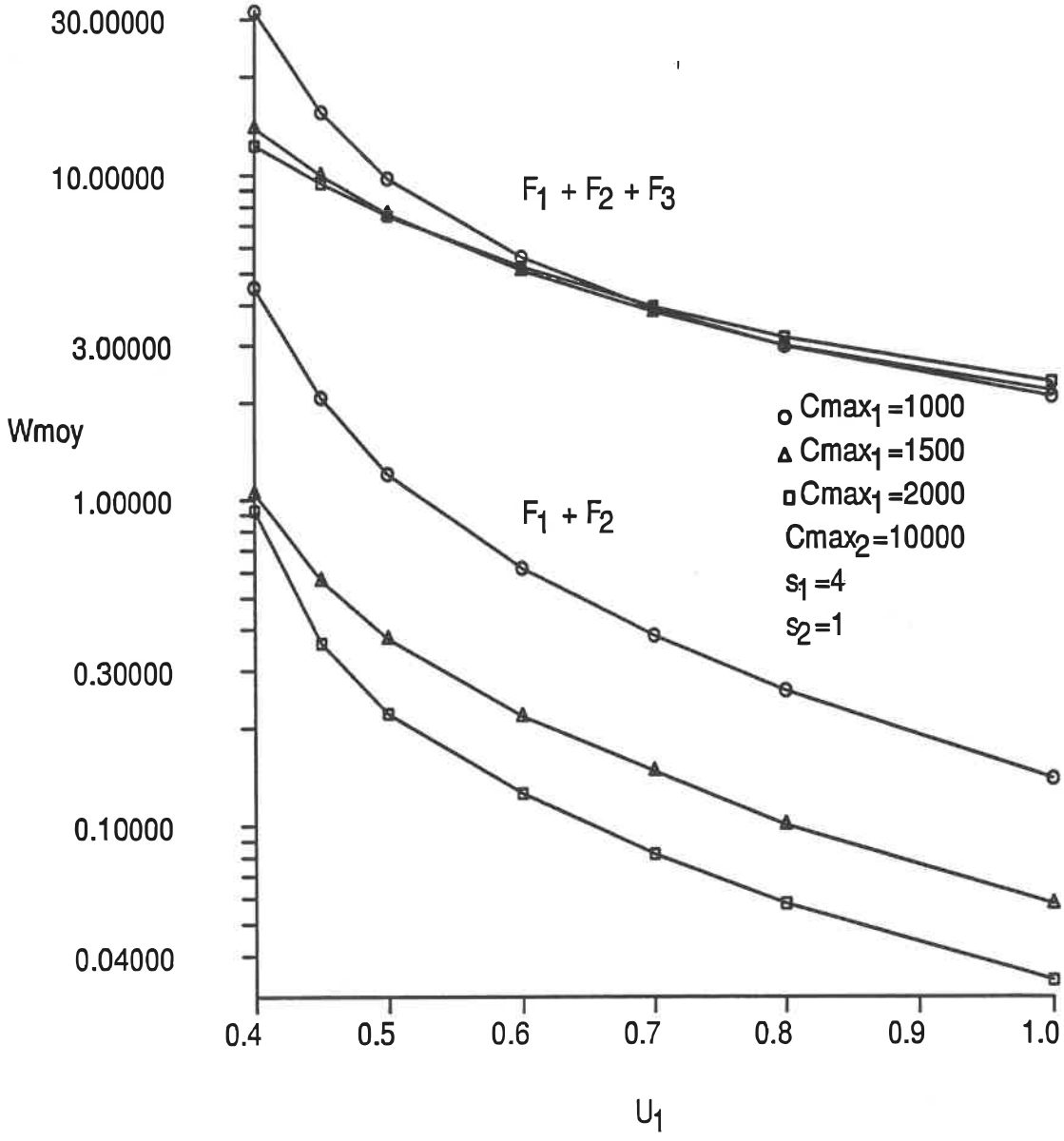


Figure 6.5 Somme des longueurs moyennes des files d'attente en fonction de U_1 pour s_1=4, L=519 et Eb/No=2.7 dB

6.6 Comparaison avec le multidécodeur simple

Le reste du chapitre est consacré à la comparaison du multidécodeur avec décodeur de réserve (système 2) avec le multidécodeur simple du chapitre 5 (système 1). Il s'avère difficile de comparer les performances de ces deux configurations car elles n'ont pas le même nombre de files d'attente (3 au lieu de 2). Il est donc important d'établir de bons critères de comparaison.

6.6.1 Critères de comparaison

La meilleure façon de comparer les deux systèmes est de considérer que le système 2 fonctionne en régime permanent et que par conséquent les décodeurs de réserve reçoivent tous leurs blocs des premiers décodeurs et aucun du canal. Les critères de comparaison sont les suivants: les deux systèmes doivent avoir le même débit efficace, le même nombre de processeurs en parallèle et le produit Us doit être le même pour les deux systèmes.

Le multidécodeur avec décodeur de réserve peut être vu de deux façons différentes. Le système 2 peut soit être analysé en tenant compte de tous les décodeurs incluant le décodeur de réserve, soit en ne tenant compte que des

décodeurs simples du premier étage. Quelque soit le choix, il est clair que le comparaison n'est pas facile.

En considérant tous les décodeurs incluant le décodeur de réserve, le système 2 est comparé à un système 1 dont $s=s_1+s_2$. Dans ce cas, le système 2 est désavantagé du point de vue de la file d'attente à l'entrée car il dispose de s_2 décodeurs de moins que le système 1. De plus, pour une comparaison équitable, il faut que $Us=U_1s_1+U_2s_2$. Cette condition implique qu'une partie de la puissance de calcul des décodeurs simples est perdue au profit des décodeurs de réserve.

D'autre part, le fait de considérer uniquement les décodeurs simples du système 2 versus le système 1 implique que le nombre de processeurs du système 1 doit être égal au nombre de décodeurs du premier étage du système 2. De cette façon, le décodeur de réserve ne recevant pas de blocs du canal est ignoré. Il ne sert qu'à améliorer le débit efficace du système. Ce décodeur pourrait être choisi avec un gain suffisamment élevé pour que la deuxième file d'attente soit très petite. Quant à la normalisation du produit Us, il faut que le produit Us total du système 1 soit égal à celui du système 2. Puisque, dans l'analyse, le décodeur de réserve n'intervient que pour le débit efficace, il faut comparer le produit U_1s_1 du système 2 avec le produit Us du système 1.

Dans ce mémoire, c'est l'option où les décodeurs de réserve sont ignorés qui a été choisie. Ce choix est justifié tout simplement parce qu'il semblait plus facile à appliquer. La dernière chose à spécifier est le débit efficace des deux systèmes.

Dans le système 1, le débit efficace est déterminé par la taille des piles des décodeurs séquentiels mis en parallèle tandis que dans le système 2, c'est la taille de la pile du décodeur de réserve qui fixe le débit efficace du système. Par exemple, si le système 2 a un décodeur de réserve ayant une pile permettant $Cmax_2$=10000 bits, il faut le comparer avec un système 1 dont les s décodeurs séquentiels ont tous des piles permettant $Cmax_1$=10000 bits. La complexité totale de ces deux systèmes semble donc très différente. Le modèle de coût qui suit sert à évaluer le niveau de complexité de chacune des approches. En utilisant une architecture à file systolique prioritaire telle que décrite par Pierre Lavoie [17], Normand Bélanger [18] a développé un modèle de coût en transistors d'un tel système basé sur les puces réalisées à l'Ecole Polytechnique de Montréal. Ce modèle est le suivant:

$$\text{coût d'un décodeur} = C_1 + C_2 + C_3 + C_4 \qquad (6.1)$$

où C_1 est le coût de l'extenseur

C_2 est le coût de l'historique

C_3 est le coût du module bloc

C_4 est le coût de la file prioritaire

Le nombre de transistors nécessaires pour la réalisation de ces différents modules sont calculés à partir des valeurs obtenues pour les puces faites à l'Ecole Polytechnique. Ces valeurs sont les suivantes:

nombre de transistors par extenseur	7000
nombre de transistors par enregistrement de file (sur un circuit)	857
nombre d'enregistrements par circuit intégré de file	21
nombre de circuits pour avoir des enregistrements de file complets	5
nombre de tansistors par bit d'historique	6
nombre de transistors par module bloc	27000
nombre de bits par bloc de message	505

A l'aide de ces paramètres, un modèle de coût a été établi [18]:

$$C_1 = n_{ext} \times 2 \text{ extenseurs} \tag{6.2}$$

$$C_2 = n_{bit} \times [\,(1 + [\log_2 P_h])/8] \times 8 \times 1024 \times 2^{\min([\log_2(Ph/1024)],5)} \times [P_h/(1024 \times 2^{\min([\log_2(Ph/1024)],5)})] \tag{6.3}$$

$$C_3 = n_{bloc} \tag{6.4}$$

$$C_4 = n_{enr} \times [P_f/e_{ci}] \times e_{ci} \times c_{enr} \tag{6.5}$$

où

n_{ext} est le nombre de transistors par extenseur

n_{bit} est le nombre de transistors par bit d'historique

P_h est la profondeur de l'historique

n_{bloc} est le nombre de transistors par module bloc

n_{enr} est le nombre de transistors par enregistrement

P_f est la profondeur de la file prioritaire

e_{ci} est le nombre d'enregistrement par circuit de file

c_{enr} est nombre de circuits de file nécessaire pour avoir des enregistrements complets

On constate que l'équation du coût dépend de deux paramètres du système: la profondeur de l'historique et la longueur de la file prioritaire. Il a été demontré par P. Lavoie [19] et N. Bélanger [18] que si Cmax tend vers l'infini, les performances du décodeur séquentiel commencent à se dégrader

lorsque P_f = Cmax/10. C'est pourquoi, la profondeur de la file prioritaire a été choisie égale à Cmax/10.

D'après les équation 6.1 à 6.5, il s'ensuit que le coût d'un sytème 1 constitué de 10 décodeurs ayant $Cmax_1$ égal à 10000 est de 76.86 millions de transistors. De même, le coût d'un système 2 constitué de 10 décodeurs ayant $Cmax_1$=2000 et un décodeur de réserve ayant $Cmax_2$=10000 est de 32.83 millions de transistors. Quoique le système 2 ait un décodeur de plus que le système 1, il est constitué de la moitié moins de transistors. Cette constatation nous amènera, à la section 6.7, à reconsidérer la façon de comparer les deux systèmes. Tout d'abord, voici quelques résultats de simulation à la section 6.6.2.

6.6.2 Résultats de simulation

Cette section présente les résultats de la comparaison du système 1 au système 2. Cette comparaison se fait sur la base des critères énoncés à la section précédente.

La figure 6.6 donne les files d'attente moyennes F_1 et F_3 du système 1 pour $Cmax_1$=10K et s=4, 9 et 14. Il faut rappeler que le système 1 n'a pas de file d'attente F_2. Pour la comparaison des systèmes 1 et 2, seules les files d'attente F_1 et F_3 ont été considérées. En effet, la file

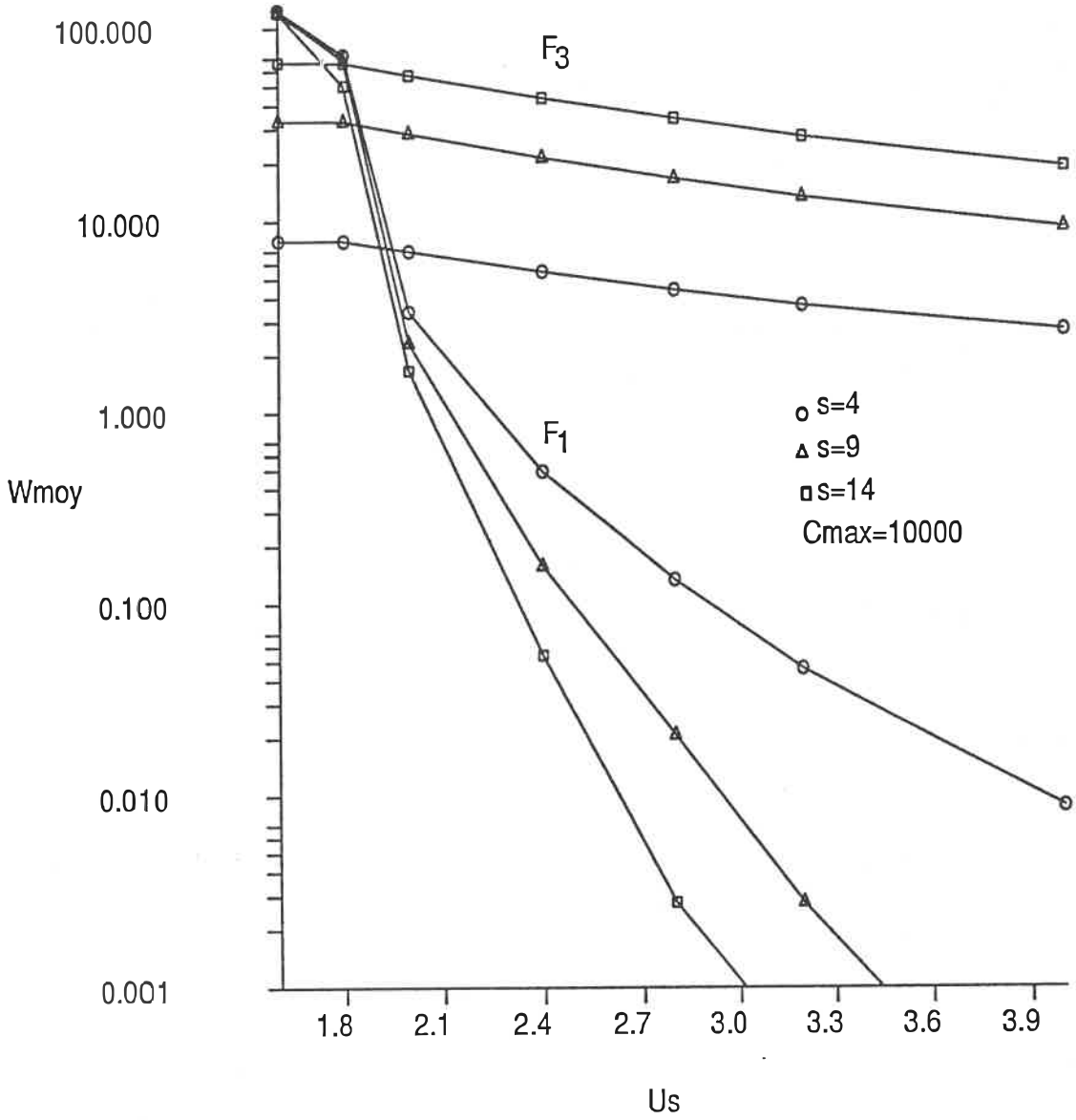


Figure 6.6 Longueur moyenne des files d'attente F_1 et f_3 en fonction de Us pour le système 1 avec L=519 et Eb/No=2.7 dB

d'attente F_2 peut être aussi petite que souhaité en choisissant un décodeur ayant un gain de vitesse élevé ou bien en augmentant s_2. De plus, il a été vu que pour un produit U_1s_1 fixe, la variation de s_1 n'affecte pas F_2. Les figures 6.7 et 6.8 donnent les files d'attente moyennes F_1 et F_3 respectivement en fonction du produit Us (U_1s_1 pour le système 1) pour $Cmax_1$=10K avec le système 1 et $Cmax_1$=2K et $Cmax_2$=10K pour le système 2. Il est clair que pour de faibles produits Us et pour un nombre de processeurs s donné, l'option avec décodeur de réserve diminue d'un facteur 100 la file d'attente à l'entrée. Quant à la file d'attente à la sortie, elle est aussi diminuée mais par un facteur 2 seulement. Il faut noter que le gain U_2 choisi pour le décodeur de réserve, est un gain très faible c.-à-d. proche de l'instabilité. Par conséquent, il est probable qu'avec un gain U_2 plus élevé, la file d'attente à la sortie serait plus petite. En effet, les blocs qui retardent le plus la remise en ordre sont ceux qui débordent. En les faisant circuler plus rapidement dans le système, la file d'attente F_3 peut être diminuée.

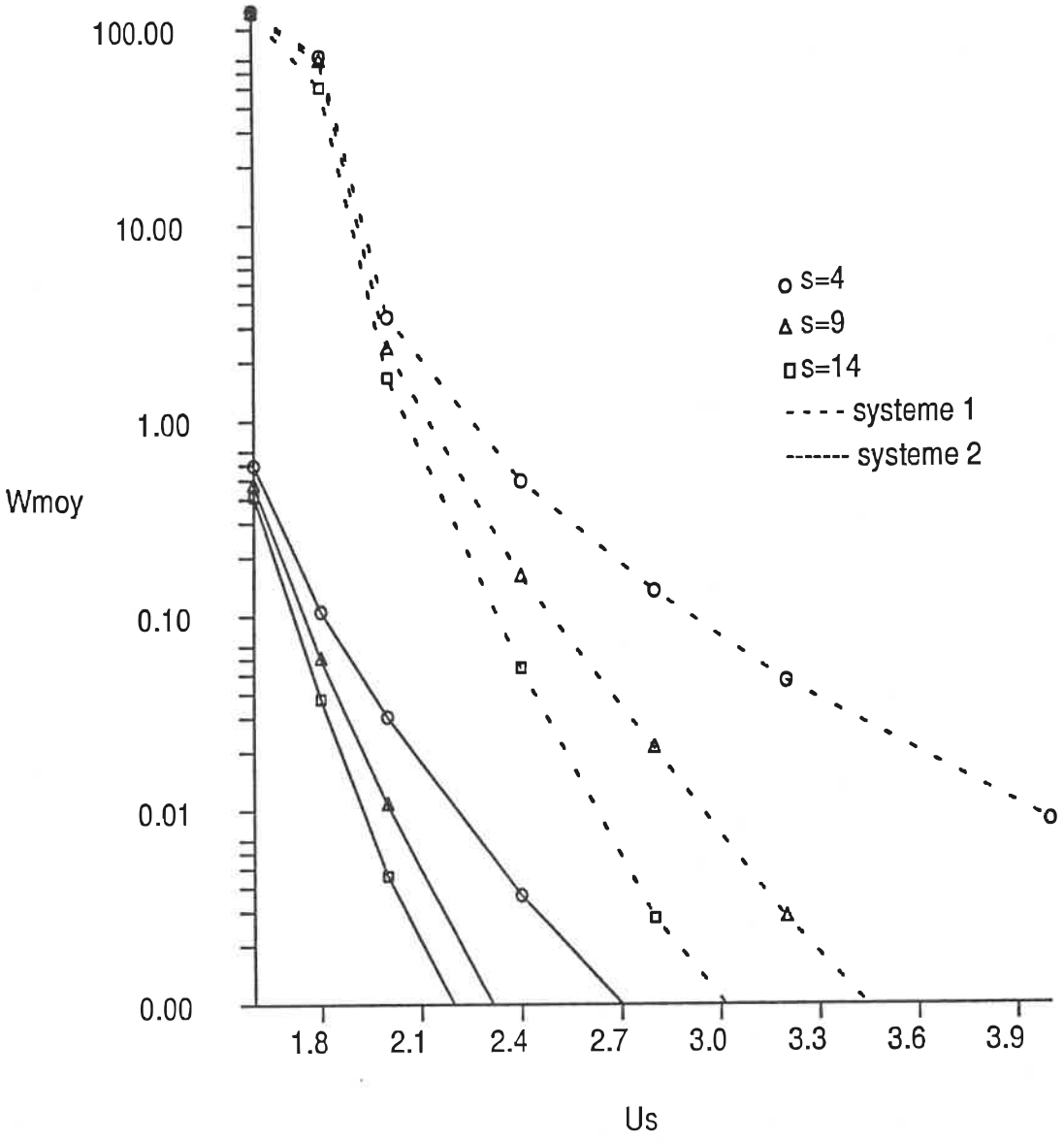


Figure 6.7 Longueur moyenne de F_1 en fonction de Us pour les systèmes 1 et 2 avec L=519 et Eb/No=2.7 dB

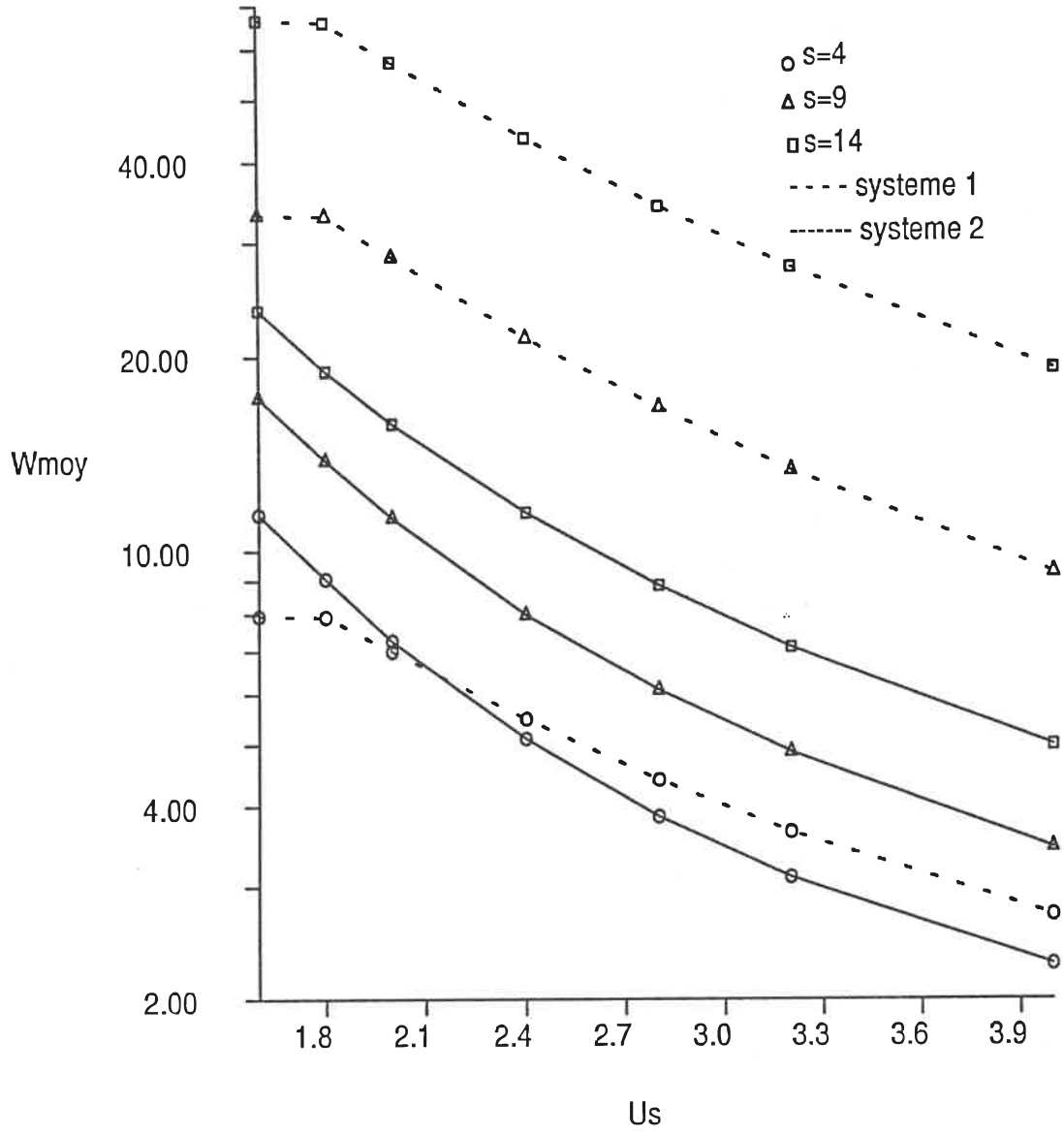


Figure 6.8 Longueur moyenne de F_3 en fonction de Us pour les systèmes 1 et 2 avec L=519 et Eb/No=2.7 dB

6.7 Discussion

L'analyse de coût faite à la section 6.6.1 et les résultats de la section 6.6.2 sont très révélateurs. En effet, l'analyse du coût a démontré que le coût en transistors d'un multidécodeur se chiffre autour de 80 Mtrs pour 10 processeurs ayant un historique de 20000 tandis que la somme des files d'attente moyennes est d'environ 200 blocs. Si des DRAM sont employés pour le tamponnage, les tampons coûtent 2 sym/bit x 519 bits x 3 bits /sym x 1 trs/bit = 3114.00 trs/bloc. Avec 200 blocs, les tampons coûtent donc .623 Mtrs. Il semble donc inapproprié de mettre beaucoup d'efforts pour minimiser ces files d'attente. Il y a plutôt avantage à concentrer notre énergie pour trouver une architecture de multidécodeur moins coûteuse. Dans cette optique, le multidécodeur avec décodeur de réserve s'avère une option très intéressante puisque son coût en transistors est la moitié de celui d'un multidécodeur simple ayant le même débit efficace.

Puisque le coût revêt une importance si grande, maintenir le coût constant serait une base de comparaison beaucoup plus prometteuse que de maintenir la somme des gains de vitesse constante. De cette façon, on dispose de décodeurs séquentiels dont le gain de vitesse est fixé par la technologie et dont la taille de l'historique, celle de la

file prioritaire ainsi que le nombre sont à déterminer. Le débit efficace est fixé par la taille de l'historique et le gain de vitesse du système peut être augmenté en augmentant le nombre de processeurs en parallèle. Par conséquent, tout devient une question de coût en transistors. Par exemple, on peut augmenter le gain de vitesse du système aux dépens du débit efficace ou de la probabilité de débordement de la file prioritaire. C'est une analyse très compliquée à faire mais en même temps très révélatrice.

6.8 Conclusion

Ce chapitre constitue une analyse très partielle du système 2. Certaines méthodes d'analyse ont été utilisées. Il faudrait maintenant en essayer d'autres pour bien cerner les performances du système 2.

Il reste à explorer l'option où le décodeur de réserve continue le décodage du bloc au lieu de le recommencer en entier. Comme dans le cas du système 1, un gain minimal permettant un fonctionnement optimal du système pour un nombre de processeurs donné peut être déterminé.

Un grand nombre de simulations restent à faire lorsqu'il y a plusieurs décodeurs de réserve. L'option à plusieurs décodeurs de réserve ne devrait pas influencer la

file d'attente à l'entrée mais celle à la sortie et aussi la file d'attente devant les décodeurs de réserve.

Il a aussi été montré que l'option avec décodeur de réserve a un coût en transistors beaucoup plus faible que le multidécodeur simple pour un débit efficace donné. De plus, le multidécodeur avec décodeur de réserve donne de meilleures performances en ce qui concerne les files d'attente à l'entrée et à la sortie. Il reste toujours le problème de la taille du tampon à la sortie. La discussion de la section précédente a mis en évidence le fait que le coût des tampons est négligeable par rapport au coût en transistors du système. Il s'avère donc que le multidécodeur avec décodeur de réserve est avantageux par rapport au multidécodeur simple pour des questions de coût et de performances et est une solution avantageuse lorsqu'on veut augmenter la vitesse de décodage d'un système.

CHAPITRE 7

CONCLUSIONS ET SUGGESTIONS POUR RECHERCHES FUTURES

Le but de ce mémoire était d'analyser les performances d'une nouvelle architecture: le multidécodeur. Cette architecture est intéressante parce qu'elle permet d'augmenter la vitesse de décodage. Il fallait en plus s'assurer que cette augmentation ne se faisait pas au détriment des autres performances du système (entre autres la file d'attente à l'entrée). De façon heuristique, il semblait que l'utilisation d'un multidécodeur ne pouvait que diminuer la file d'attente à l'entrée à cause d'un effet de moyennage et donc atténuer l'effet de la variabilité du nombre de calculs par bloc du décodeur séquentiel.

Le multidécodeur étant constitué de décodeurs séquentiels, il fallait connaître la loi de service de ceux-ci. Etant donné que seul le comportement asymptotique de la loi de service du décodeur séquentiel était connu, la première partie de ce mémoire a été consacrée à la caractérisation de cette loi en fonction du rapport signal-à-bruit et de la longueur des blocs d'information. Cette expression a permis le calcul des moyennes et des variances du nombre de calculs par bloc. De plus, elle a été très utile

à la réalisation du simulateur du multidécodeur simple et du multidécodeur avec décodeurs de réserve.

Les conditions limites et de stabilité gouvernant les files d'attente du multidécodeur simple et du multidécodeur avec décodeurs de réserve ont été établies à l'aide de la théorie des files d'attente.

Le multidécodeur simple a été simulé ce qui nous a permis de constater l'existence de deux compromis. Tout d'abord, comme Pau et Haccoun [3] l'ont décrit pour un seul serveur, l'augmentation de la taille de la pile du décodeur séquentiel augmente la file d'attente à l'entrée de même que le débit efficace du système. Il y donc un compromis entre la longueur de la file d'attente et le débit efficace pour un nombre de processeurs donné. De plus, s'il y a remise en ordre des blocs au récepteur, il faut faire un compromis entre la longueur de la file d'attente à l'entrée et celle de la file d'attente à la sortie lorsque le nombre de processeurs augmente.

La simulation du multidécodeur avec décodeur de réserve nous a permis de constater que celui-ci se comporte comme le multidécodeur simple concernant l'effet de l'augmentation du nombre de processeurs sur les files d'attente à l'entrée et à la sortie. Par contre, le compromis entre la file d'attente à l'entrée et le débit efficace ne tient plus. En effet, dans ce système, le débit efficace est

déterminé par la taille de la pile du décodeur de réserve tandis que la file d'attente à l'entrée est déterminée par la taille de la pile des décodeurs du premier étage. La comparaison avec le multidécodeur simple a permis d'établir que le multidécodeur avec décodeurs de réserve est une alternative plus performante et moins coûteuse que le multidécodeur simple.

Cette approche multidécodeur présente plusieurs avantages et quelques inconvénients par rapport à un système à un seul décodeur. Le premier avantage est qu'elle permet d'augmenter la vitesse de décodage d'un facteur s, s étant le nombre de décodeurs en parallèle. De plus, à cause de l'effet de moyennage, cette approche permet de diminuer la file d'attente moyenne à l'entrée. Il est aussi possible de déterminer un seuil minimal pour le produit Us permettant de limiter la file d'attente maximale à l'entrée à un bloc. Ce seuil diminue lorsque s augmente ce qui est très avantageux.

Le multidécodeur a trois désavantages. Premièrement, il existe un compromis entre la longueur de la file d'attente à l'entrée et le débit efficace du multidécodeur. Deuxièmement, si une remise en ordre est exigée au récepteur, la file d'attente à la sortie croît avec l'augmentation du nombre de processeurs. Finalement, le multidécodeur a un coût élevé en transistors.

Le premier inconvénient est éliminé par l'utilisation d'une architecture multidécodeur avec décodeurs de réserve. De plus, cette architecture coûte beaucoup moins cher en transistors qu'un multidécodeur simple ce qui permet d'attenuer le troisième inconvénient. Quant au deuxième inconvénient, il a été vu au chapitre 6 que le coût du tampon de sortie, s'il y a remise en ordre, est négligeable par rapport au coût du système.

Il faut donc conclure que le multidécodeur et plus particulièrement le multidécodeur avec décodeurs de réserve sont des approches très intéressantes pour augmenter la vitesse de décodage et améliorer le comportement Pareto de la file d'attente à l'entrée.

7.1 Recherches futures

Il reste encore beaucoup de recherches à faire pour analyser toutes les facettes de ce mémoire. Par exemple, puisque les paramètres de l'expression de la loi de service ont été obtenus par régression, les paramètres pour un autre rapport signal-à-bruit ou une autre longueur de bloc ne peuvent être prédits. Un travail complémentaire au chapitre 3 consisterait à réaliser un grand nombre de simulations et faire des graphiques des paramètres n_1, K_1 et K_2 en fonction de L et Eb/No. Il est peut-être possible aussi de déterminer

théoriquement à l'aide des processus de ramification (branching process) la loi de service du décodeur séquentiel.

Dans le cas du multidécodeur simple, il reste à déterminer l'effet sur le débit efficace et les files d'attente à l'entrée et à la sortie de la variation du rapport signal-à-bruit sur le canal et surtout de l'effet de la variation de la longueur des blocs. En effet, cette dernière alternative pourrait être intéressante si on constatait que la longueur des blocs n'influence pas le nombre de blocs dans la file d'attente à l'entrée. Dans ce cas, en maintenant L faible, il y aurait moins de bits dans le tampon d'entrée.

On pourrait aussi calculer la probabilité de débordement du tampon d'entrée à l'aide de simulations.

En ce qui concerne le seuil minimal de gain qui garantit que la file d'attente est d'une longueur maximale de 1 bloc, il pourrait être déterminé d'une autre façon. En effet, étant donné que la probabilité de débordement du tampon à l'entrée est non-nulle, il serait plus réaliste d'utiliser une valeur $Wmax^*$ définie comme étant la taille de la file d'attente pour laquelle la probabilité que le tampon soit supérieur à $Wmax^*$ est égale à une fraction (fixée) de la probabilité de débordement du tampon d'entrée. Avec cette nouvelle définition de Wmax, un seuil minimal pour le gain

peut être calculé. Et cette fois ce seuil peut être valide pour un nombre indéterminé de blocs simulés.

Tout ce mémoire est basé sur la technique FEC. Il reste que malgré tous les efforts faits, il y aura toujours des blocs qui feront déborder la pile. C'est pourquoi, il serait intéressant d'analyser l'approche hybride FEC/ARQ pour le système.

Dans le cas du multidécodeur simple, l'approche hybride FEC/ARQ résulterait dans un compromis débit efficace et longueur de la file d'attente moins clair. Il risque aussi d'y avoir des conséquences importantes si la remise en ordre des blocs est prise en compte.

Dans le cas du multidécodeur avec décodeurs de réserve, il y aura cette fois un compromis à faire entre la longueur de la file d'attente à l'entrée et le débit efficace car, la taille de la pile du décodeur de réserve influencera le nombre de retransmissions demandées.

BIBLIOGRAPHIE

[1] C. E. Shannon, "A Mathematical Theory of Communication", Bell Syst. Tech. J., Vol. 27, 1948, pp.379-423 (Part I), pp.623-656 (Part II).

[2] V. K. Bhargava, D. Haccoun, R. Matyas, P. Nuspl, "Digital Communication by Satellite", John Wiley, New York, 1981.

[3] P.-Y. Pau, D. Haccoun, "An analysis of sequential decoding with retransmission procedures", rapport technique EPM/RT-85-19, Editions de l'Ecole Polytechnique de Montréal, Montréal, 1985.

[4] D. Haccoun, article de conférence, Gretsi, 1985.

[5] R. G. Gallager, "Information Theory and Reliable Communication " Wiley, New York, 1968.

[6] R. M. Fano, "A Heuristic Discussion of Probabilistic Decoding", IEEE Trans. Inf. Theory, Vol IT-19, Apr. 1963, pp.64-73.

[7] Jelinek, F., "Fast Sequential Decoding Using a Stack", IBM Journal of Research and Development,Vol.13,pp.675-685,novembre 1969.

[8] I. M. Jacobs, E. R. Berlekamp, "A Lower Bound to the Distribution of Computation for Sequential Decoding", IEEE Trans. Inf. Theory, Vol. IT-13, Apr. 1967, pp.167-174.

[9] J.E. Savage,"The Computation Problem with Sequential Decoding",MIT Lincoln Lab. Tech. Rep. No.371, Feb.1965.

[10] F. Jelinek,"An Upper Bound on Moments of Sequential Decoding Effort", IEEE Trans. Inf. Theory, Vol. IT-15, Jan 1969, pp. 140-149.

[11] G. D. Forney, Jr., "Convolutional Codes III:Sequential Decoding",Inf. Control, Vol 25, July 1974,pp.267-297

[12] D. Falconer, "A Hybrid Sequential and Algebraic Decoding Scheme",Ph.D. dissertation, dept. of Elec. Eng., MIT, Cambridge, Mass., Feb. 1967.

[13] L. Kleinrock, "Queueing Systems volume I et II", John Wiley, New York, 1975.

[14] J. F. C. Kingman,"The Heavy Traffic Approximation in the Theory of Queues",in W. L. Smith and R. I. Wilkinson, eds., Proceedings of the Symposium on Congestion Theory, Univ. of North Carolina (Chapel Hill), 137-169 (1964).

[15] J. Köllerström,"Heavy Traffic Theory for Queues with Several Servers. I," Journal of Applied Probability, 11, 544-552 (1974).

[16] Johannesson, communication privée à David Haccoun.

[17] P. Lavoie, D. Haccoun, Y. Savaria, "Spécification d'un décodeur séquentiel rapide utilisant une queue prioritaire systolique", rapport technique EPM/RT-88-11, Editions de l'Ecole Polytechnique de Montréal, Montréal, 1988. syts

[18] N. Bélanger, "Architectures multiprocesseurs de décodeurs séquentiels à pile", Mémoire de maîtrise, Département de génie électrique, Ecole Polytechnique de Montréal, Montréal, Canada, août 1989.

[19] P. Lavoie, "Algorithme de décodage séquentiel à pile tronquée: analyse et implantation", Thèse de doctorat, Département de génie électrique, Ecole Polytechnique de Montréal, Montréal, Canada, à paraître en mai 1990.

ANNEXE

MODE D'UTILISATION DES SIMULATEURS

1.0 Simulateur du multidécodeur

Ce simulateur permet de simuler un multidécodeur dont les s processeurs peuvent avoir des tailles de pile différentes. Le gain de vitesse de chacun des décodeurs est identique.

La simulation peut se faire en utilisant deux façons de générer les temps de service: ils sont soit générés à l'aide de la loi de service du décodeur séquentiel ou bien si cette loi n'est pas connue, une deuxième option peut être envisagée qui consiste à lire dans un fichier des temps de service tirés de simulations. Ces deux options sont notées 0 et 1 respectivement. L'option 1 n'a pas été implantée dans le simulateur.

Une simulation est exécutée en tapant la commande :

multi in out stat

où multi est le fichier exécutable de multi.c
in est un fichier de données d'entrée
out et stat sont des fichiers de résultats

Le fichier "in" comprend les données suivantes:

option (choix entre 0 ou 1)

s (nombre de processeurs)

U (gain de vitesse)

Cmax[0] (taille de la pile du premier décodeur)

...

Cmax[s-1] (taille de la pile du dernier décodeur)

nbcons (nombre de blocs simulés)

longueur (longueur des blocs d'information)

alpha (exposant Pareto)

num (l'exposant n_1 de la fonction de densité)

xmax (x_m, abcisse du point maximal de la fonction de densité)

semence (semence du générateur de nombres aléatoires)

Le fichier "out" donne les données du fichier "in" en plus de la distribution du nombre de bloc dans les files d'attente à l'entrée et à la sortie. De plus, il donne les longueurs moyennes et maximales des files d'attente.

Le fichier "stat" donne aussi les données du fichier "in" avec les longueurs moyennes et maximales des files d'attente. C'est un fichier qui n'est pas détruit lorsqu'une nouvelle simulation est faite. On peut donc l'utiliser pour recueillir des statistiques sur toutes les simulations.

2.0 Simulateur du multidécodeur avec décodeurs de réserve

Ce simulateur permet de simuler un multidécodeur avec un décodeur de réserve. Les décodeurs simples ont des tailles de piles identiques mais peuvent avoir des gains de vitesse différents. Le fichier exécutable du simulateur est multidr. Une simulation est exécutée en tapant la commande:

multidr in out stat

Les fichiers "out" et "stat" donnent les mêmes résultats que précédemment. Le fichier "in" est un peu différent. Il contient les données suivantes:

option

s

s_2

gain[0]

...

gain[s-1]

$Cmax_1$

$Cmax_2$

nbcons

longueur

alpha

num

xmax

semence