Amélioration des services vidéo fournis à travers les réseaux radio mobiles

Khaled Bouchireb

École Doctorale
d'Informatique,
Télécommunications
et Électronique de Paris

Thèse

présentée pour obtenir le grade de docteur

de l'Ecole Nationale Supérieure des Télécommunications

Spécialité : Informatique et Réseaux

Khaled BOUCHIREB

Amélioration des services vidéo fournis à travers les réseaux radio mobiles

Soutenue le 25 Juin 2010 devant le jury composé de

Philippe Godlewski	Président
Jérôme Lacan	Rapporteur
Pierre Siohan	Rapporteur
Véronique Vèque	Examinatrice
Pierre Duhamel	Directeur de thèse
Michel Kieffer	Directeur de thèse

Acknowledgements

First, I would like to thank my supervisors Pierre Duhamel and Michel Kieffer for the guidance and all the instructions and advice they provided throughout this work. I greatly appreciated their support.

I would also like to thank Alcatel-Lucent Bell Labs for the technical and financial support of this work. Special thanks go to the team of Sylvaine Kerboeuf for the valuable feedbacks and numerous insights.

I would also like to thank Dr. Cédric Marin, H264 expert and former PhD student at Alcatel-Lucent Bell Labs and Laboratoire des Signaux et Systèmes, without whom H264 video-based simulations would not have been possible.

Thank you to all the members of Laboratoire des Signaux et Systèmes and Ecole Nationale Supérieure des Télécommunications for their (direct or indirect) contribution to this work.

Thank you to my friend, Dr. Antonio Cipriano, for his moral help and support.

Thank you to Pr. Philippe Godlewski (Ecole Nationale Supérieure des Télécommunications), Dr. Pierre Siohan (Orange/France Télécom R&D), Dr. Jérôme Lacan (Institut Supérieur de l'Aéronotique et de l'Espace) and Pr. Véronique Vèque (Université Paris Sud) for accepting to form the jury of this PhD dissertation defense.

Last but not least, a big thank you to my parents and my brother for their full and unconditional support.

Abstract

In this thesis, video communication systems are studied for application to video services provided over wireless mobile networks. This work emphasizes on point-to-multipoint communications and proposes many enhancements to the current systems:

First, a scheme combining robust decoding with retransmissions is defined so that the number of retransmissions is reduced and the quality of the received video can be controlled. As opposed to current retransmissionless and retransmission-based schemes, this scheme also offers the possibility to trade throughput for quality and vice versa. A parameter allows to choose the throughput-quality trade-off.

Then, the transmission of a two-level scalable video sequence towards several clients is considered. Schemes using the basic Go-back-N (GBN) and Selective Repeat (SR) Automatic Repeat reQuest (ARQ) techniques are studied. A new scheme is also proposed and studied. The new scheme reduces the buffering requirement at the receiver end while keeping the performance optimal (in terms of the amount of data successfully transmitted within a given period of time). The different schemes were shown to be applicable to 2G, 3G and WiMAX systems.

Finally, we prove that retransmissions can be used in point-to-multipoint communications up to a given limit on the number of receivers (contrary to the current wireless systems where ARQ is only used in point-to-point communications). If retransmissions are introduced in the current Multicast/Broadcast services (supported by the 3GPP and mobile WiMAX), the system will guarantee a certain amount of receivers to have the nominal quality whereas the current Multicast/Broadcast services do not garantee any receiver of the nominal quality.

Résumé étendu en Français

Il y a récemment eu une explosion des services Multimedia fournis par les réseaux sans fil, grâce au développement et la mise en place de systèmes et autres infrastructures favorisant la fourniture de ce genre de service (GPRS/EDGE, UMTS/HSDPA, WiMAX, DVB-H).

Parmi toutes les applications Multimedia, la vidéo est incontestablement l'application la plus exigeante en termes de bande passante. Pour lutter contre le manque de bande passante au niveau du spectre radio alloué au système, la méthode la plus efficace consiste à regrouper tous les récepteurs intéressés par un même contenu vidéo dans un même groupe pour lequel les mêmes ressources radio sont utilisées pour la transmission (système point-à-multipoint).

Aussi, comme pour tous les services numériques fournis à travers des réseaux cellulaires, le signal transmis subit de nombreuses dégradations (atténuations, distortions, pollution par du bruit ...) et les informations numériques véhiculées par le signal sont corrompues par des erreurs (certains bits du flux binaire changent de valeur), affectant ainsi la qualité de la vidéo reçue. Pour lutter contre ces erreurs de transmission, plusieurs méthodes peuvent être utilisées (codage canal au niveau physique, décodage robuste au niveau applicatif ou encore retransmissions au niveau MAC, RLC ou TCP) mais cela se fait au détriment du débit, d'où l'intérêt de s'intéresser au compromis débit/qualité pour ce type d'application.

Cette thèse s'intéresse principalement aux systèmes de communications vidéo Point-à-Multipoint et au compromis débit/qualité dans les systèmes de communications vidéo.

Nous résumons ici en Français les éléments importants rapportés dans les chapitres de la thèse (en Anglais):

Chapitre 1 : Communication Networks

Ce chapitre commence par décrire les réseaux de communications numériques d'une façon générale. L'architecture physique et l'architecture logique y sont décrites. Par la suite, les réseaux cellulaires (qui peuvent être vus comme une extension sans fil du réseau Internet) sont décrits plus en détails étant donné que les chapitres suivants traitent de la fourniture des services vidéo à travers ce type de réseau. L'accent est mis sur les réseaux WiMAX IEEE 802.16-2004 car ce type de réseau a pour principal objectif la fourniture d'une connexion Internet sans fil et aussi la fourniture de services multimédia (dans les zones urbaines). A noter que la couche application n'est pas décrite dans ce chapitre mais plutôt dans le chapitre 2.

Chapitre 2 : Video Compression

Ce chapitre porte sur la compression vidéo. Il est basé sur la norme H264/AVC. Dans un premier temps, les outils de compression sont décrits brièvement. Par la suite, des notions générales propre à la vidéo (telles que la capture d'une vidéo et la qualité vidéo) sont décrites avant de détailler la norme vidéo H264/AVC. Cette norme est la dernière en date en matière de vidéo et a été développée conjointement par le Moving Pictures Experts Group (MPEG) et le Video Coding Experts Group (VCEG). Les performances de la compression selon la norme H264/AVC en terme de compromis débit/distortion sont ensuite présentées. Les résultats montrent que cette norme dépasse clairement les normes précédentes (MPEG-4 et H.263), offrant ainsi une compression plus efficace des séquences vidéo.

Dans cette thèse, cette norme est utilisée comme norme de référence. Le codec H264/AVC n'est nul autre que la couche application dans le cas de la transmission d'une vidéo compressée selon cette norme. Dans ce cas, la couche application se compose de deux sous-couches : la Video Coding (sub)Layer (VCL) et la Network Abstraction (sub)Layer (NAL). La sous-couche VCL convertit la séquence d'images en un flux binaire compressé. La sous-couche NAL quant à elle, convertit le flux binaire compressé en un flux de paquets appelés NAL Units (NALUs) transmis à la couche inférieure (RTP/UDP ou TCP) pour être transporté à travers le(s) réseau(x) jusqu'à l'utilisateur final.

Chapitre 3 : Improved Retransmission Scheme for Video Communications

Traditionnellement, on avait le choix entre un système basé sur le décodage robuste qui laisse passer beaucoup d'erreurs mais qui possède un débit maximum et un système qui ne laisse passer aucune erreur mais dont le débit est nettement inférieur au premier système à cause des retransmissions qu'il utilise pour rendre la transmission fiable.

Par ailleurs, le système à décodage robuste est incompatible avec l'ARQ.

Dans ce chapitre, on essaie de définir un système qui utilise aussi bien les retransmissions que la correction d'erreur (via le décodage robuste) de telle façon à ce que

- La qualité et le débit puissent être controlés.
- Le décodage robuste ne soit plus incompatible avec l'ARQ.
- Les retransmissions inutiles de paquets contenant des erreurs qui peuvent être corrigées (par un décodage robuste) puissent être évitées (améliorant ainsi le débit).

En effet, un paquet contenant des erreurs est systématiquement retransmis dans un système ARQ. Sachant que le décodage robuste peut corriger des erreurs, il est possible d'éviter la retransmission de certains paquets si le décodage robuste de ses données est considéré suffisamment fiable. Il est donc nécessaire de définir une mesure de la fiabilité du décodage robuste et un paramètre définissant le niveau minimum de fiabilité.

Ce système de décision a est modélisé comme un test d'hypothèses avec les hypothèses H_1 et H_0 définies comme suit:

H_1 : La séquence décodée $\underline{\hat{s}}$ est la séquence qui a été émise.

H_0 : La séquence décodée $\underline{\hat{s}}$ n'est pas la séquence qui a été émise.

On suppose dans un premier temps qu'un paquet vidéo contient une seule séquence vidéo sur laquelle le décodage robuste peut être effectué.

Si la décision après le test est en faveur de H_1, le paquet est accepté et son contenu est utilisé pour la reconstruction de la séquence vidéo encodée et transmise.
Si la decision après le test est en faveur de H_0, le paquet est rejeté par le récepteur et retransmis par l'émetteur.
La séquence décodée $\underline{\hat{s}}$ est celle qui a la plus grande probabilité a posteriori (décodage au maximum a posteriori)

$$\underline{\hat{s}} = \arg \max_{\underline{s}_j, j\in\{1,\ldots,K\}} P(\underline{s}_j|\underline{r}) = \arg \max_{\underline{s}_j, j\in\{1,\ldots,K\}} p(\underline{r}|\underline{s}_j)$$

Où

$\underline{\hat{s}}$ est la séquence émise (dans le paquet).

$\underline{r}$ est le vecteur des observations (résultat de la modulation et transmission de la séquence $\underline{s}$).

$\underline{s}_j$ est la $j^{\text{ème}}$ séquence valide (au sens de la syntaxe de l'encodeur).

K est le nombre total de séquences valides.

Le test d'hypothèses est caractérisé par, entre autres, la probabilité de fausse alarme P_F et la probabilité de détection P_D

$P_F = P(\text{Choisir } H_1|H_0 \text{ est vraie})$.

$P_D = P(\text{Choisir } H_1|H_1 \text{ est vraie})$.

La probabilité de fausse alarme est la probabilité d'accepter un paquet erroné. Elle représente donc une mesure de la qualité (quand P_F diminue la qualité est améliorée).

La probabilité de détection est la probabilité d'accepter un paquet non erroné, ou encore la probabilité de ne pas faire une retransmission inutile. Elle représente donc une mesure indirecte du débit (quand P_D augmente, le nombre de retransmissions inutiles diminue et par conséquent le débit augmente).

Diminuer le nombre de retransmissions (et donc améliorer le débit) en maintenant le même niveau de qualité revient à diminuer la probabilité de détection pour une probabilité de fausse alarme donnée, ce qui est connu sous le nom de Critère de Neyman-Pearson qui se traduit par le test suivant

$$\Lambda(\underline{r}) = \frac{p(\underline{r}|H_1)}{p(\underline{r}|H_0)} \underset{H_0}{\overset{H_1}{\gtrless}} \lambda$$

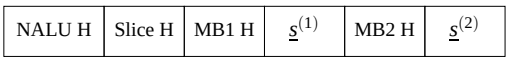

Figure 1: Format d'un paquet video H264 (NALU) quand le Data Partitioning n'est pas utilisé. Example : 2 Macro Blocs transportés dans le paquet.

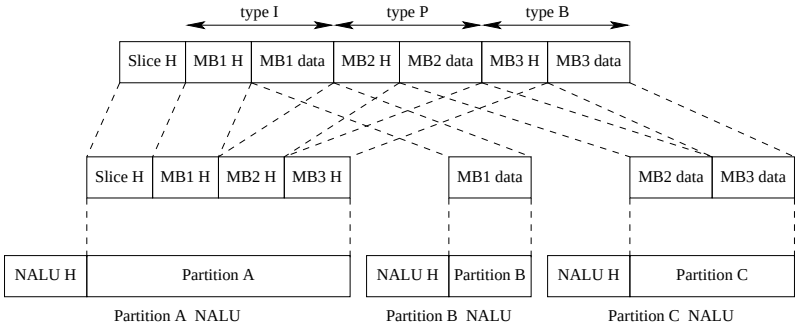

Figure 2: Format d'un paquet video H264 (NALU) quand le Data Partitioning est utilisé. Example : 3 Macro Blocs (1 de type I, un de type P et un de type P) transportés dans le paquet.

Le rapport de vraisemblance $\Lambda(\underline{r}) = \frac{p(\underline{r}|H_1)}{p(\underline{r}|H_0)}$ représente donc la mesure de fiabilité du décodage (citée ci-dessus) alors que le seuil du test λ représente le niveau de fiabilité du décodage exigé pour accepter un paquet dont le CRC n'est pas valide (les paquets dont le CRC est valide sont systématiquement acceptés).

On démontre que dans notre cas le rapport de vraisemblance s'écrit

$$\Lambda(\underline{r}) = (1 - P(\underline{s} = \underline{b}_1)) \times \frac{p(\underline{r}|\underline{s} = \underline{b}_1)}{\sum_{j=2}^{K} p(\underline{r}|\underline{s} = \underline{b}_j) P(\underline{s} = \underline{b}_j)}$$

Le rapport de vraisemblance s'exprime donc en fonction des informations à priori et des informations à posteriori sur les séquences valides. Les métriques à posteriori sont données par le décodeur robuste séquentiel alors que les probabilités à priori sont calculées par des formules théoriques (indépendemment du vecteur des observations $\underline{r}$).

L'application du système de retransmission à une transmission vidéo H264 se fait sur les séquences CAVLC. En effet, dans H264, un Macro Bloc 16×16 pixels est encodé sous forme de séquences CAVLC (jusqu'à 16 séquences CAVLC). Les en-têtes de la couche application et les en-têtes vidéo sont également présents dans le paquet vidéo qui transporte un slice (une partie ou la totalité d'une image). Le format du paquet vidéo H264 est représenté par la figure 1 pour le cas où le Data Partitioning n'est pas utilisé et par la figure 2 pour le cas où le Data Partitioning est utilisé.

Un paquet vidéo H264 contenant d'une façon générale plusieurs séquences sur lesquelles le décodage robuste peut s'appliquer, une généralisation est nécessaire. Ainsi, on considère que le décodage robuste du paquet est fiable si le décodage robuste de chacune de ses séquences l'est, i.e. le test s'écrit

Table 1: $PSNR$ et nombre moyen de transmissions $\overline{N}_{SARQ}$ des 3 premières images (IPP) de *Forman.cif* pour une taille de paquet de 500 bits à un SNR de 9 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	27.6	29	32.3	35.6	37.8	39.8	40.4	40.62	40.64
$\overline{N}_{SARQ}$	1	1.16	1.52	1.92	2.33	2.91	3.15	3.34	3.42

Table 2: $PSNR$ et nombre moyen de transmissions $\overline{N}_{SARQ}$ des 3 premières images (IPP) de *Forman.cif* pour une taille de paquet de 1500 bits à un SNR de 9.5 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	30.8	31.8	34.7	37.2	38.8	40.3	40.6	40.62	40.64
$\overline{N}_{SARQ}$	1	1.31	2.25	3.57	5.35	7.43	8.11	8.43	8.5

$$\min\{\Lambda(\underline{r}_i), \forall i \in \{1,\ldots,N\}\} \underset{H_0}{\overset{H_1}{\gtrless}} \lambda$$

avec

$$\Lambda(\underline{r}_i) = (1 - P(\underline{s}_i = \underline{b}_{i,1})) \times \frac{p(\underline{r}_i|\underline{s}_i = \underline{b}_{i,1})}{\sum_{j=2}^{K} p(\underline{r}_i|\underline{s}_i = \underline{b}_{i,j}) P(\underline{s}_i = \underline{b}_{i,j})}$$

où $\underline{b}_{i,j}$ est la séquence classée au $j^{\text{ème}}$ rang après le décodage robuste de la séquence $\underline{s}_i$ basé sur l'observation $\underline{r}_i$.

Les performances du système de retransmission défini ont été evaluées par simulation. Dans les simulations, les en-têtes ont été considérés correctement reçus alors que les données ont été modulées en BPSK et transmises sur un canal AWGN. Trois valeurs différentes de la taille du paquet et du SNR ont été utilisées.

Les résultats des simulations de la transmission des 3 premières images de la séquence Forman au format CIF sont illustrés par les tables 1, 2 et 3 et les figures 3, 4 and 5).

Comme on peut le constater, en faisant varier le seuil du test entre 1 et l'infini on peut balayer un nombre infini de compromis débit/qualité se trouvant entre les chiffres du système robuste (sans ARQ) et du système ARQ. Il est par ailleurs particulièrement intéressant de remarquer que la courbe devient quasi-plate quand on s'approche du point ARQ, ce qui représente un gain en débit à qualité quasi-nominale (puisque le nombre de retransmissions est réduit pour quasiment la même qualité). Ainsi, le gain en débit à 0.4 dB en dessous du PSNR nominal pour le cas de la transmission de la première image (I) de la séquence Forman est de 6%, 13% et 19% pour les 3 cas (SNR et taille de paquet) considérés, respectivement. Le gain en débit à 0.3 dB en dessous du PSNR nominal pour le cas

Table 3: $PSNR$ et nombre moyen de transmissions $\overline{N}_{SARQ}$ des 3 premières images (IPP) de *Forman.cif* pour une taille de paquet de 5000 bits à un SNR de 10.5 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	36.9	37.2	38	39.4	40	40.45	40.62	40.63	40.64
$\overline{N}_{SARQ}$	1	1.23	1.96	3	4.18	5.96	6.65	6.75	6.98

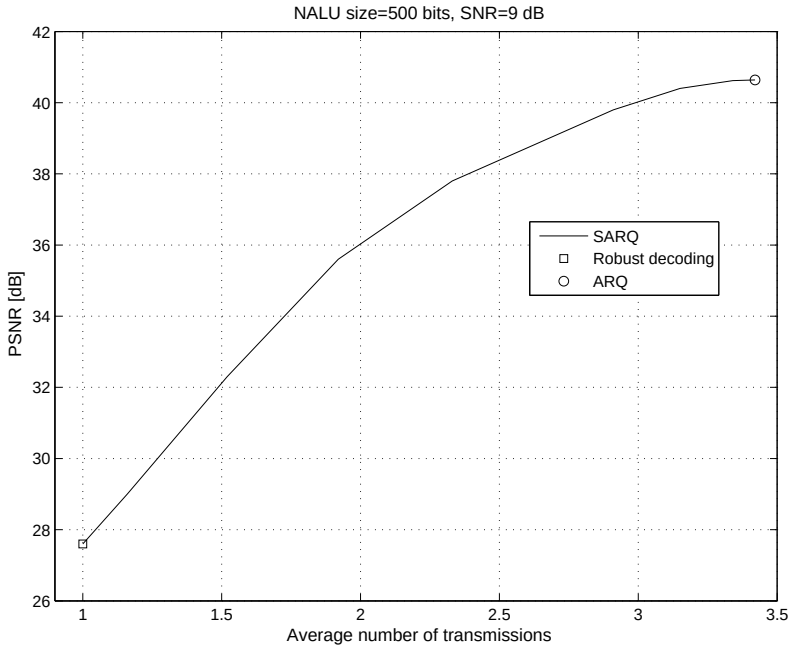

Figure 3: PSNR moyen des 3 premières images (IPP) de *Forman.cif* en fonction du nombre moyen de transmissions pour une taille de paquet de 500 bits à un SNR de 9 dB.

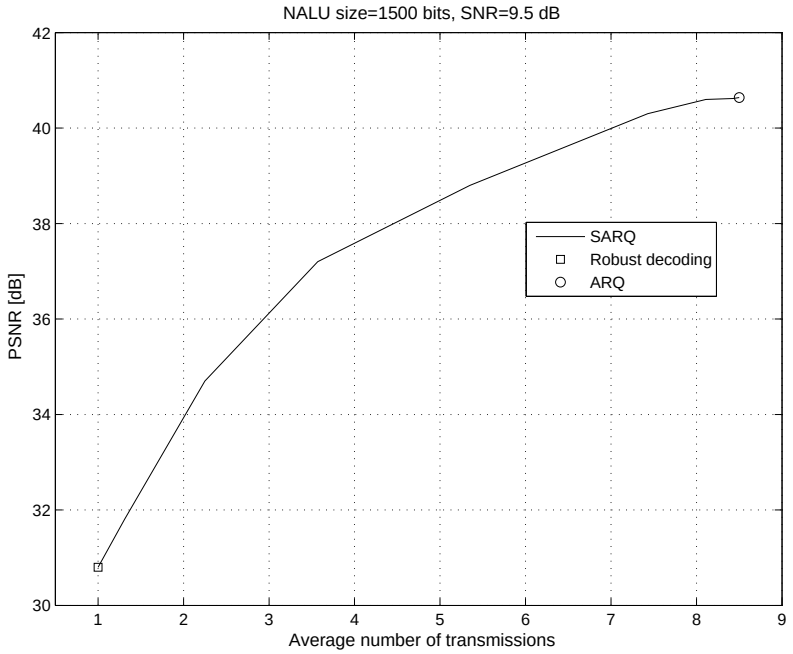

Figure 4: PSNR moyen des 3 premières images (IPP) de *Forman.cif* en fonction du nombre moyen de transmissions pour une taille de paquet de 1500 bits à un SNR de 9.5 dB.

de la transmission des 3 premières images (IPP) de la séquence Forman est de 8.5%, 14% et 17% pour les 3 cas (SNR et taille de paquet) considérés, respectivement. Si on accepte

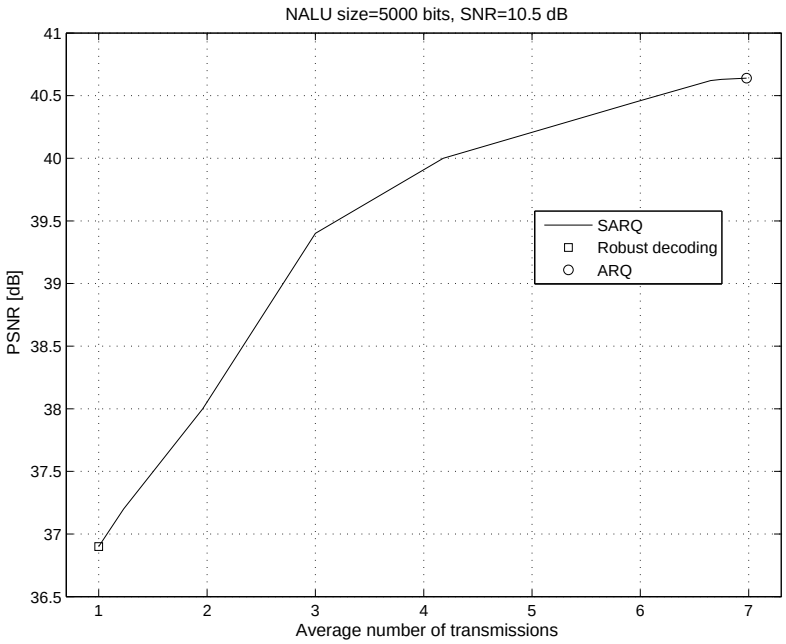

Figure 5: PSNR moyen des 3 premières images (IPP) de *Forman.cif* en fonction du nombre moyen de transmissions pour une taille de paquet de 5000 bits à un SNR de 10.5 dB.

des baisses de débit plus importantes, par exemple à 2-3 dB en dessous du PSNR nominal (baisses perceptibles à l'œil humain), le gain en débit est beaucoup plus important: pour le cas de la transmission de la première image (I) de la séquence Forman, le gain en débit est de 43%, 57% et 274% pour les 3 cas (SNR et taille de paquet) considérés, respectivement, et pour le cas de la transmission des 3 premières images (IPP) de la séquence Forman, le gain en débit est de 46%, 58% et 256% pour les 3 cas (SNR et taille de paquet) considérés, respectivement.

Par la suite, la question de l'implémentation du système proposé dans les systèmes pratiques (étroitement liée à celle du décodage robuste) est abordée. Cette dernière est basée sur la présence d'un décodeur canal de type SISO (Soft Input Soft Output) au niveau de la couche physique, suivi d'un processus permettant de faire remonter les informations soft (paquets d'APPs au lieu de paquets de bits) jusqu'à la couche application, siège du décodeur vidéo. L'exemple utilisé étant celui d'une interface radio de type IEEE 802.16-2004 WiMAX.

Il est aussi montré que l'implémentation du système proposé nécessite la mise en place d'un mécanisme intercouche permettant à la couche application de commander les retransmissions MAC.

En conclusion, dans ce chapitre une nouvelle technique de retransmission est définie pour application dans les systèmes de communications vidéo. Elle combine décodage robuste et retransmissions. La décision de retransmettre un paquet est modélisée par un test d'hypothèses. On montre que le critère de Neyman-Pearson doit être utilisé pour réduire le nombre de retransmissions tout en maintenant le même niveau de qualité. Les résultats

de simulations basées sur des données encodées selon la norme H264 confirment que le seuil du test permet de régler le compromis débit/qualité. La qualité nominale est atteinte pour une valeur infinie du seuil et un bas débit, mais les résultats montrent qu'une qualité quasi-nominale peut être atteinte pour une valeur finie du seuil et un débit plus élevé. Le gain en termes de débit est alors fonction de l'efficacité du décodage robuste, qui elle, dépend de la plage de SNR et de la taille des paquets.

Chapitre 4 : Transmission Schemes for Scalable Video Streaming in Point-to-Multipoint Communications

Dans ce chapitre, on s'intéresse à la problématique de la transmission d'une vidéo scalable dans un système point-à-multipoint. Une vidéo scalable est une vidéo constituée non pas d'un mais de plusieurs flux de données d'importances (et donc de priorités) différentes. Le premier flux est le flux de base qui contient la vidéo. Sans ce flux, la vidéo (séquence d'images) ne peut être reconstruite. Ce flux contient une vidéo de qualité de base et est indispensable. Les autres flux sont des flux d'amélioration, i.e. la disponibilité d'un de ces flux permet d'améliorer la qualité de la vidéo en la raffinant (ajout d'une précision sur les valeurs des pixels) comparé au cas où seuls les flux de niveau inférieur sont disponibles. A noter qu'un flux n'est utile que dans son intégralité et que si tous les flux d'importance hiérarchique plus faible sont également disponibles. Les flux d'amélioration sont optionnels ou accéssoires.

Les paquets du flux de base sont notés "paquets I" alors que les paquets des flux d'amélioration sont notés "paquets A". L'approche la plus directe pour la transmission de ce type de vidéo consiste à utiliser une protection inégale d'erreurs, i.e. à transmettre les paquets du flux de base avec plus de protection que les paquets des flux d'amélioration. L'inconvénient de cette technique est qu'elle ne garantit pas la réception du flux de base. Pour y remédier on peut imaginer un système qui transmettrait tous les flux en mode acquitté (i.e. en utilisant l'ARQ) mais un tel système ne respecte pas la hiérarchie des flux (accorde la même importance à tous les flux et les traite tous comme des flux de base).
Ainsi, pour garantir la réception du flux de base tout en respectant l'importance des flux, on propose d'utiliser une transmission en mode acquitté du flux de base et une transmission en Best Effort (mode non acquitté) des flux d'amélioration. Dans une telle transmission on peut imaginer deux phases de transmission: Une première phase ou le flux de base est transmis en utilisant une technique ARQ du type Go-Back-N (GBN) ou Selective Repeat (SR) suivie d'une deuxième phase de transmission cyclique en mode non acquitté des paquets des flux d'amélioration.

Un système Point-à-Multipoint est un système avec un seul émetteur et un ou plusieurs récepteurs, le nombre de récepteurs K pouvant varier.

Entre l'émetteur et un récepteur donné il y a un canal unicast. Ce canal est supposé binaire symétrique (BSC) et est caractérisé par un taux d'erreurs binaires donné. On suppose par ailleurs que les canaux unicast sont sans mémoire (les erreurs ont lieu de façon tout à fait indépendante) et indépendants les uns des autres (les erreurs ayant lieu sur un canal sont indépendantes des erreurs ayant lieu sur les autres canaux).

On suppose également que l'émetteur mémorise les acquittements (ou non acquitte-

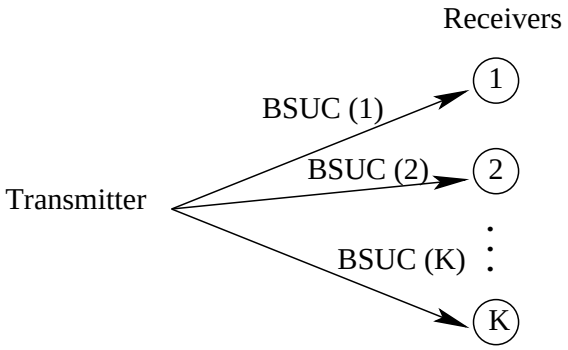

Figure 6: Système Point-à-Multipoint.

ments) des paquets de façon à ce que seuls les acquittements (ou non acquittements) des utilisateurs n'ayant pas encore acquitté un paquet donné soient considérés un moment donné. Ceci permet au nombre de transmissions d'augmenter de façon logarithmique en fonction du nombre d'utilisateurs plutôt que de façon exponentielle.

On suppose enfin que tous les récepteurs se trouvent dans les mêmes conditions radio (i.e. que les différents canaux unicast sont caractérisés par le même taux d'erreurs binaires ε)

Dans le GBN (cf. figure 7), quand le paquet n'est pas correctement reçu, il est retransmis un temps d'aller-retour plus tard (correspondant à la transmission de (N-1) paquets) ainsi que tous les (N-1) paquets qui ont été transmis entre temps, quelque soit le résultat de leur transmission. Ainsi, dans l'exemple de la figure 7, les paquets 4, 5, 6 et 7 sont retransmis inutilement (étant donné qu'ils ont été correctement reçu lors de la première tentative). Ces retransmissions inutiles se traduisent par un faible débit. L'avantage est que dans cette technique, les paquets sont acceptés en séquence, ce qui fait que cette technique ne nécessite pas de bufferisation au niveau du récepteur.

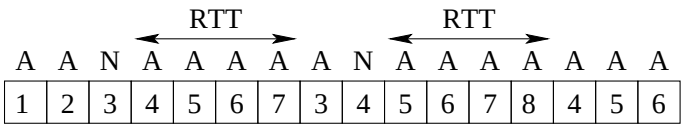

Figure 7: Go-Back-N ARQ avec $N = 4$.

Dans le SR (cf. figure 8), seuls les paquets non acquittés sont retransmis. Il n'y a donc pas de retransmission inutile et le débit est le maximum que l'on peut obtenir dans un système ARQ. L'inconvenient est que les paquets n'étant pas forcément acquittés en séquence, une bufferisation des paquets correctement reçus est nécessaire si des paquets avec un numéro de séquence plus faible n'ont pas encore été reçu. Dans le pire cas, on a besoin de mémoriser autant de paquets que l'on a transmettre.

Dans notre cas, on peut profiter de l'existence des flux d'amélioration pour essayer de combiner les avantages des deux techniques (GBN et SR). En effet, si le paquet I3 n'est

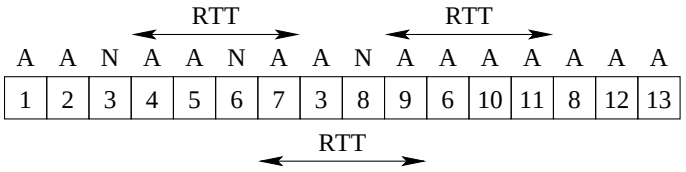

Figure 8: SR ARQ avec $N = 4$.

pas correctement reçu (cf. figure 9), ce dernier est retransmis un temps d'aller-retour plus tard mais si les paquets transmis entre temps sont correctement reçus, on propose au lieu de les retransmettre inutilement de transmettre des paquets A. D'une façon plus générale, on peut utiliser un mécanisme avec une fenêtre d'observation et une fenêtre de retransmission (cf. figure 10). Le contenu de la fenêtre de retransmission est déduit des résultats des transmissions qui ont eu lieu dans la fenêtre d'observation: les paquets I non acquittés sont retransmis, les paquets A sont remplacés par d'autres paquets A (transmission cyclique) et les paquets I acquittés sont remplacés par des paquets A.

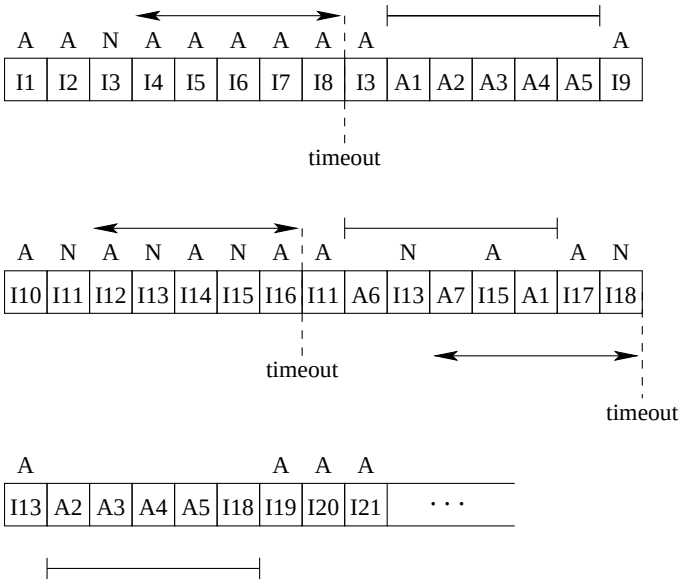

Figure 9: Système proposé pour la transmission de vidéos scalables.

Dans ce cas, on n'a besoin de mémoriser que $(N-1)$ paquets (cf. figure 12) au lieu de $(L_1 - 1)$ paquets (cf. figure 11), où L_1 est le nombre total de paquets du premier flux.

Pour des raisons de simplicité, les simulations ont été effectuées avec deux flux de 20 paquets de 50 bits chacun. La mesure de performance utilisée (notée x) est le pourcentage du nombre moyen de récepteurs ayant reçu les deux flux à la fin du temps que l'on s'est donnée pour la transmission.

Les résultats illustrés sur la figure 13 (obtenu avec 10 récepteurs, un taux d'erreurs binaires de 10^{-3} et un N égal à 10) montrent que le GBN est nettement moins performant

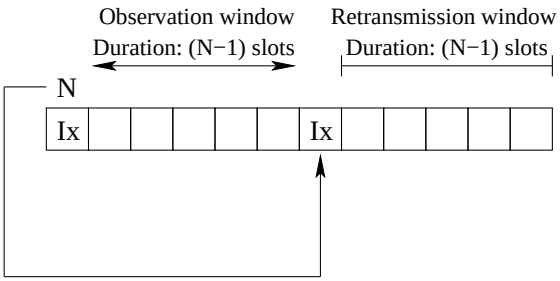

Figure 10: Fenêtre d'observation et fenêtre de retransmission.

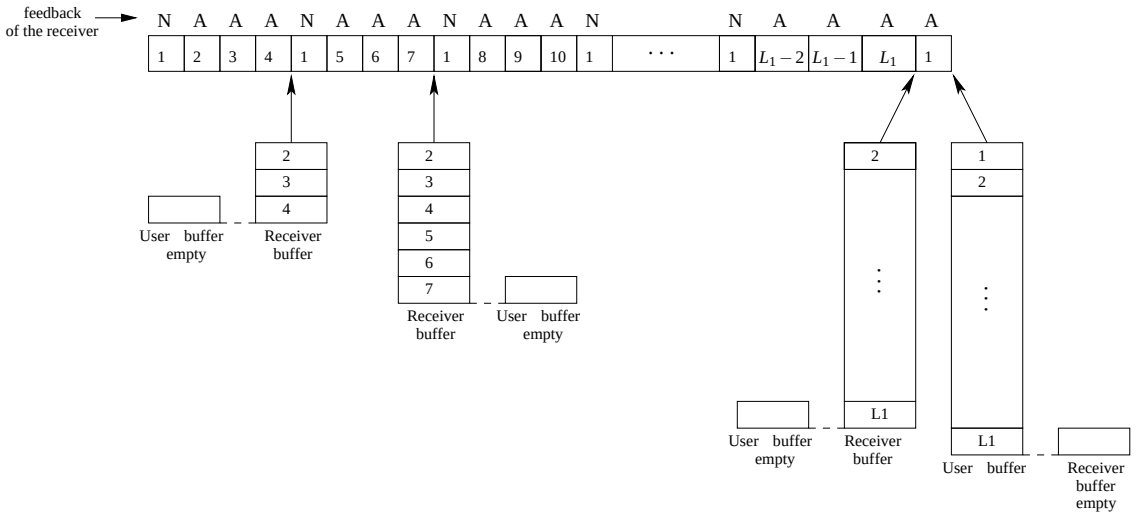

Figure 11: Evolutions de la mémoire du récepteur et de celle de l'application (avec $L_1 - 1$ multiple de $N - 1 = 3$) dans la technique SR.

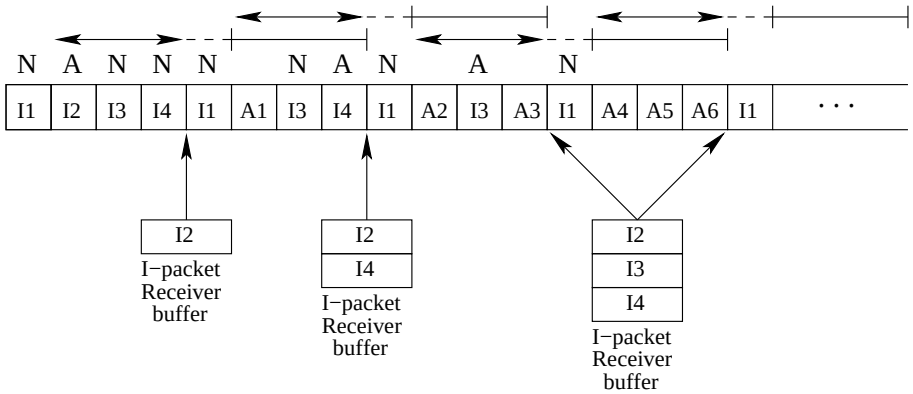

Figure 12: Evolutions de la mémoire du récepteur et de celle de l'application (avec $N - 1 = 3$) dans la techique proposée.

que les deux autres techniques et que à capacité de mémorisation égale (N paquets) la technique proposée est plus performante que le SR (avec une capacité de mémorisation

	Taille de la mémoire nécessaire (en nombre de paquets)
GBN	0
SR	$L_1 - 1$
NS	$N - 1$

Table 4: Mémoire nécessaire pour maintenir des performances optimales

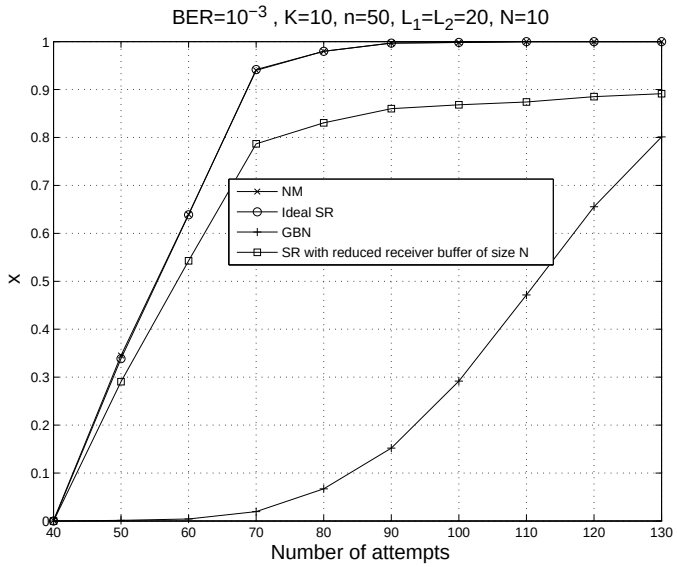

Figure 13: Performance du système proposé (NM) comparé aux systèmes GBN, SR (taille de mémoire=N) et Ideal SR (taille de mémoire $=L_1$).

de L_1 paquets, le SR est aussi performant que le système proposé). En d'autre termes, le système proposé présente un compromis performances/besoin de mémorisation meilleur que celui du Selective Repeat.

Un certain nombre de mécanismes de transmission en mode acquitté d'un seul flux et vers un seul récepteur (cf. figure 16) existe déjà dans les systèmes cellulaires. Au niveau transport, des retransmissions du type GBN sont effectuées par le protocole TCP. Au niveau du réseau d'accès, un à deux mécanismes de type SR ARQ sont disponibles par système (technologie) au niveau de la couche 2. Pour l'application des systèmes de transmission de vidéos scalables, il est nécessaire de généraliser au cas de plusieurs de récepteurs. Au niveau transport, le protocole NACK-Oriented Reliable Multicast (NORM) est l'équivalent de TCP dans le mode point-à-multipoint. Pour les mécanismes de niveau 2, il est nécessaire de prendre en compte les acquittements (ou non acquittements) de tous les récepteurs.

En conclusion, dans ce chapitre les techniques de transmission d'une vidéo scalable à 2 niveaux vers plusieurs récepteurs sont étudiées. Le nombre de paquets à transmettre est fixe et connu. Des extensions des systèmes Go-back-N (GBN) et Selective Repeat (SR) sont étudiées. Un nouveau système est aussi proposé et étudié. Les performances de ces systèmes sont évaluées par simulations. On montre que le système proposé réduit le besoin en

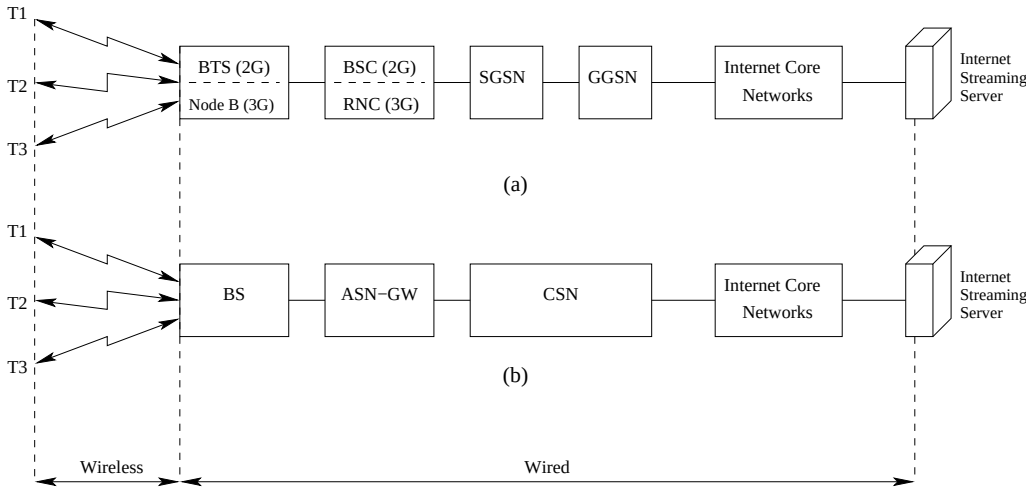

Figure 14: Architecture d'une connexion serveur-mobile de bout en bout en point-à-multipoint (a) Cas de systèmes 2G et 3G. (b) Cas d'un système 802.16 WiMAX.

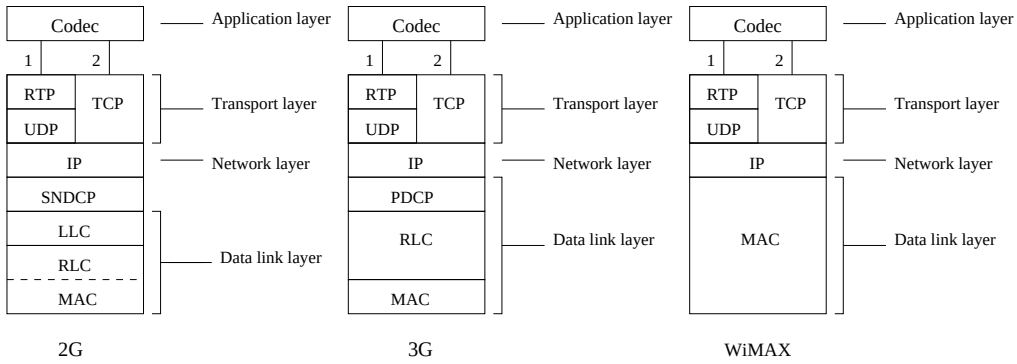

Figure 15: Pile protocolaire pour le transport de données vidéo sur des systèmes 2G, 3G ou IEEE 802.16 WiMAX.

termes de bufferisation tout en présentant des performances optimales. Les résultats montrent également que l'augmentation du nombre de récepteurs n'affecte pas énormément les performances du système alors qu'elle améliore considérablement l'efficacité d'utilisation de la bande passante. On présente également une étude analytique du système proposé. Enfin, on considère le problème de l'application des systèmes étudiés aux systèmes pratiques 2G, 3G et WiMAX en identifiant les quelques adaptations nécessaires à cet effet.

Chapitre 5 : On the use of Automatic Repeat Request in Multicast/Broadcast services

Dans ce chapitre, on s'intéresse aux performances de l'ARQ dans les systèmes Point-à-Multipoint avec pour but de déterminer les limites d'utilisation de l'ARQ en termes de nombre d'utilisateurs sous une contrainte de débit. Les services Multicast/Broadcast supportés par les systèmes de 3ème génération et 802.16 WiMAX n'utilisent pas l'ARQ étant

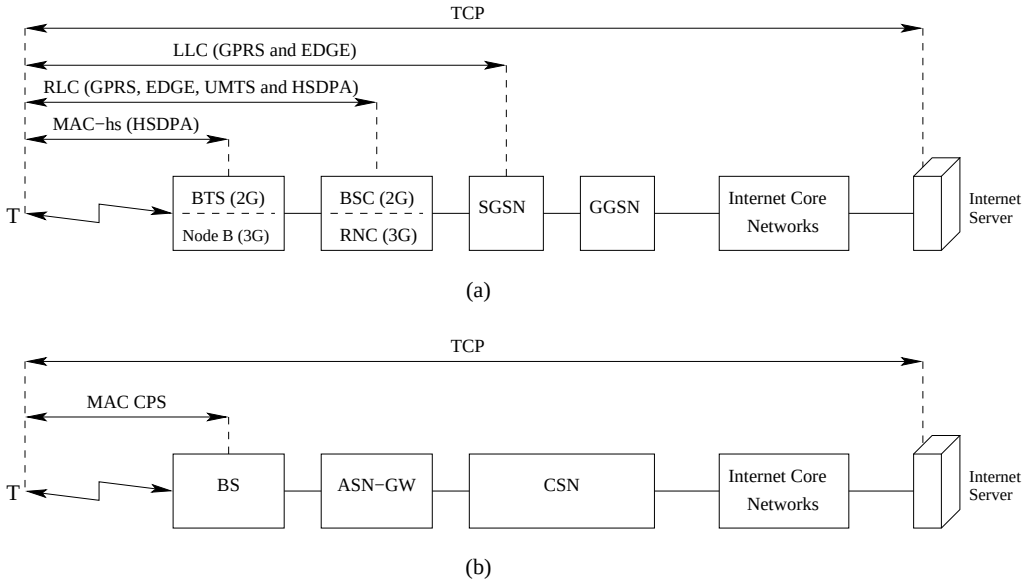

Figure 16: Niveaux de retransmissions dans une connexion TCP (a) Dans le cas de réseaux 2G et 3G. (b) Dans le cas de systèmes 802.16 WiMAX.

donné que le nombre d'utilisateurs pouvant être très grand la diminution du débit due à l'utilisation de l'ARQ risque d'être considérable.

Dans ce chapitre, on essaie de définir les limites de l'ARQ dans les systèmes Point-à-Multipoint avec pour but de déterminer les limites d'utilisation de l'ARQ en termes de nombre d'utilisateurs sous une contrainte de débit. En dessous de cette limite, les utilisateurs servi en mode ARQ bénéficient d'une qualité nettement meilleure que celle dont ils auraient bénéfécié sans ARQ avec des gains de PSNR pouvant dépasser les 10 dB.

Les différents canaux unicast sont supposés binaires symétriques (BSC) et sont caractérisés par un taux d'erreurs binaires donné chacun. On suppose par ailleurs que les canaux unicast sont sans mémoire (les erreurs ont lieu de façon tout à fait indépendante) et indépendants les uns des autres (les erreurs ayant lieu sur un canal sont indépendantes des erreurs ayant lieu sur les autres canaux).

On suppose également que l'émetteur mémorise les acquittements (ou non acquittements) des paquets de façon à ce que seuls les acquittements (ou non acquittements) des utilisateurs n'ayant pas encore acquitté un paquet donné soient considérés à un moment donné. Ceci permet au nombre de transmissions d'augmenter de façon logarithmique en fonction du nombre d'utilisateurs plutôt que de façon exponentielle.

On suppose enfin que les K récepteurs peuvent être divisés en G groupes d'utilisateurs. Chaque groupe d'utlisateurs étant caractérisé par le même taux d'erreurs binaires sur les canaux unicast des différent récepteurs appartenant au groupe en question. Ainsi, le $g^{\text{ème}}$ groupe est composé de K_g utilisateurs et est caractérisé par un taux d'erreurs binaires ε_g.

Il est alors évident que

$$K_1 + K_2 + \cdots + K_G = 1$$

Dans ce cas, on démontre que le nombre moyen de transmissions (i.e. la moyenne du

nombre de tentatives nécessaires pour qu'un paquet soit acquitté par tous les récepteurs) s'écrit:

$$\begin{aligned}\overline{M} &= \sum_{m=1}^{M_{max}-1} m \left[\prod_{g=1}^{G}(1-{P_{u,g}}^{m})^{K_g} - \prod_{g=1}^{G}(1-{P_{u,g}}^{m-1})^{K_g}\right] \\ &\quad + M_{max} \times \left[1 - \prod_{g=1}^{G}(1-{P_{u,g}}^{M_{max}-1})^{K_g}\right]\end{aligned}$$

avec

M_{max} : Nombre maximum de tentatives par paquet.

$P_{u,g} = 1-(1-\varepsilon_g)^n$: Probabilité d'une transmission unicast non acquittée au sein du groupe "g".

L'efficacité débit η s'écrit

$$\eta = \frac{1}{\overline{M}}\frac{n-n_h}{n} = \frac{1}{\overline{M}}\eta_{max}$$

$$\eta_{max} = \frac{n-n_h}{n}$$

avec

η_{max}: Efficacité débit lorsque les retransmissions ne sont pas utilisées.

n_h : Taille de l'en-tête (en bits).

n : Taille du paquet (en bits).

Les paramètres utilisés sont $M_{max} = 10$, $n = 1024$, $n_h = 48$ (un numéro de séquence de 16 bits et un CRC de 32 bits).

En pratique, toute contrainte sur le débit moyen se traduit (en fonction du système) en contrainte sur l'efficacité débit, et donc en contrainte sur le nombre moyen de transmissions: Si la contrainte sur le débit est telle que $\eta = \eta_{max}$, i.e. $\overline{M} = 1$ alors l'ARQ ne peut pas être utlisé. Si par contre la contrainte de débit est telle que $\eta < \eta_{max}$, i.e. $\overline{M} > 1$ alors l'ARQ peut éventuellement être utilisé (en fonction des paramètres du systèmes).

Si on considère dans un premier temps que tous les récepteurs se trouvent dans les mêmes conditions radio (i.e. $G = 1$, $K_g = K$ et $\varepsilon_g = \varepsilon$), dans ce cas là une contrainte sur l'efficacité débit $\eta \geq \eta_0 = \alpha\eta_{max}$ avec $\alpha < 1$ se traduit par une limitation sur le nombre de récepteurs $K \leq K_{max} = f(\eta_0)$.

La figure 17 montre le résultat pour des taux d'erreurs binaires $\varepsilon = 10^{-4}$, $\varepsilon = 10^{-5}$ et $\varepsilon = 10^{-6}$. Comme on peut le voir, K_{max} est multiplié par un facteur 10 lorsque le taux d'erreurs binaires est divisé par 10 (dans le cas considéré). Aussi, si la contrainte débit impose une baisse de 15% par rapport au débit maximal, le nombre maximum d'utilisateurs est de 20 à un taux d'erreurs binaires de 10^{-5} et de 200 à un taux d'erreurs binaires de 10^{-6}.

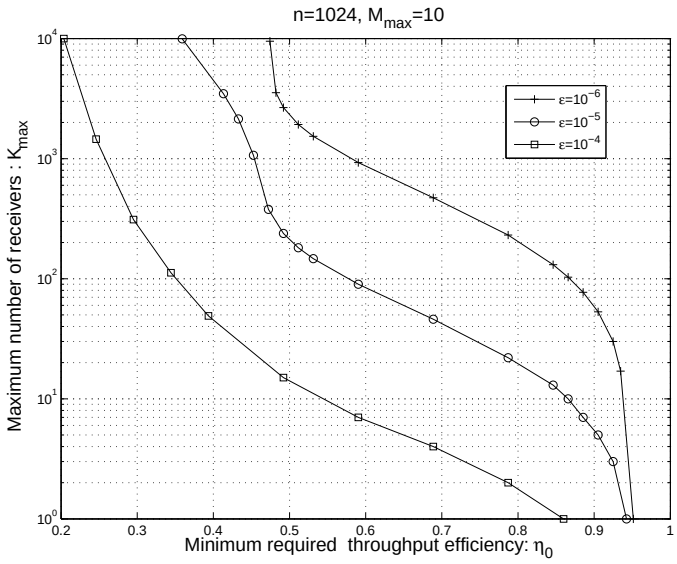

Figure 17: Nombre maximum d'utilisateurs K_{max} en fonction de l'efficacité débit minimale dans des condtions radio homogènes.

Si à présent on considère que les récepteurs sont dans des conditions radio différentes, un algorithme est nécessaire pour établir une liste de récepteurs pouvant être servi en mode acquitté (i.e. avec ARQ). Pour maximiser le nombre d'utilisateurs acceptés servi en mode acquitté, il est nécessaire de considérer les utilisateurs se trouvant de bonne conditions radio en priorité. L'algorithme consiste alors à rajouter les utilisateurs un par un et estimer l'efficacité débit qui en résulte. Dès que la contrainte n'est plus vérifiée, le dernier utilisateur ajouté est exclu de la liste et le résultat représente la liste définitive (puisque à ce moment là tout ajout de récepteur fait que la contrainte n'est plus vérifée).

Le résultat est illustré sur la figure 19 où le plus faible taux d'erreur binaire est de 10^{-5} et avec un progression géométrique de raision r pour le reste des canaux.

On note que K_{max} diminue de plus en plus avec la dégradation globale des conditions radio (augmentation de r). Ceci dit la diminution est relativement faible pour des contraintes débit élevées.

En conclusion, dans ce chapitre on étudie les systèmes de communications Point-à-Multipoint (PMP) pour application à la fourniture de vidéos non scalables dans un environnement PMP. On commence par exprimer analytiquement l'efficacité débit d'un tel système avec des récepteurs se trouvant dans différentes conditions radio. On suppose que le système SR ARQ avec la stratégie Dynamic Retransmission Group Reduction (DRGR) est utilisé. On utilise alors le résultat de ce calcul analytique pour définir la notion de capacité PMP d'un canal fréquentiel dans le mode acquitté (Acknowledged Mode : AM), i.e. le mode qui utilise les retransmissions ARQ, par opposition au mode non acquitté (unacknowledged mode) qui n'utilise pas les retransmissions ARQ.

Ensuite, on définit le Acknowledged Mode (AM) service system et le Best-Effort Ac-

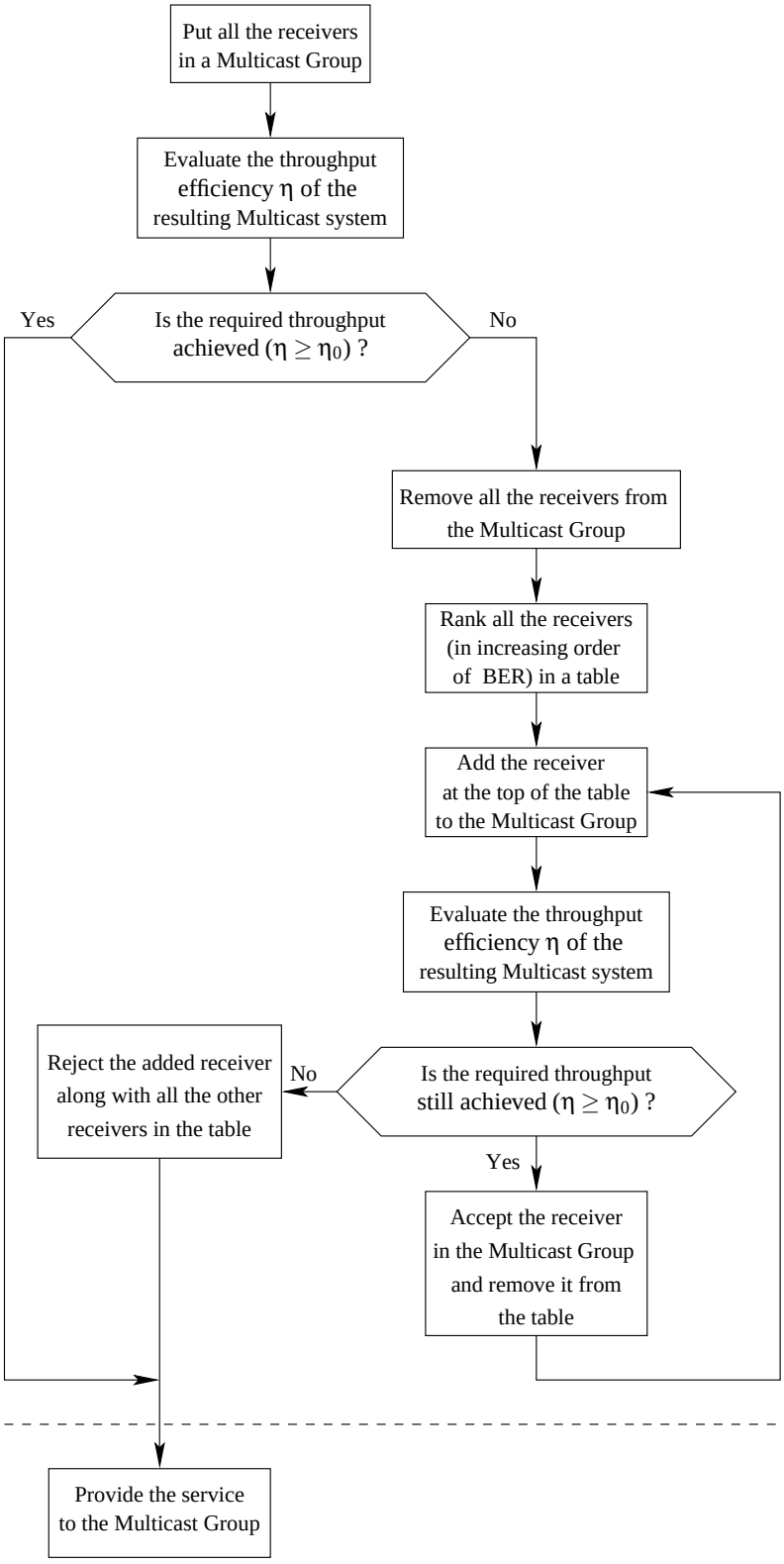

Figure 18: Algorithme de sélection des récepteurs à servir en mode acquitté.

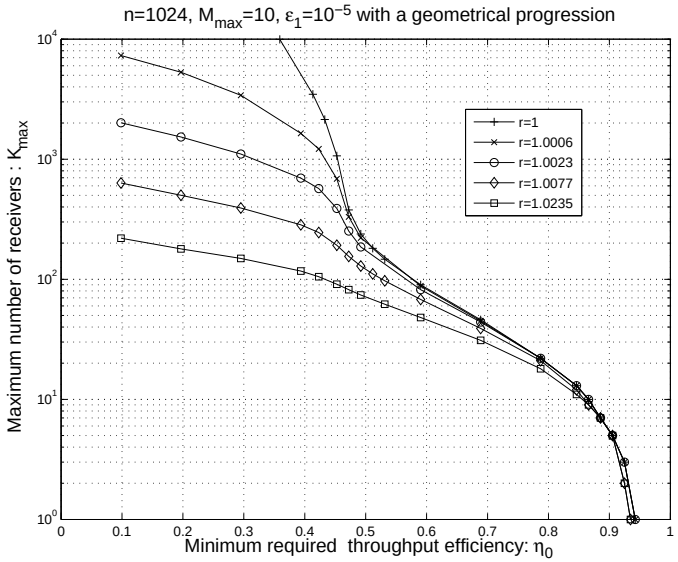

Figure 19: Nombre maximum d'utilisateurs K_{max} en fonction de l'efficacité débit minimale avec une progression géométrique des taux d'erreurs binaires.

knowledged Mode (BEAM) service system. Le système AM utilise uniquement le mode acquitté et n'accepte qu'un nombre limité d'utilisateurs. Le système BEAM, quant à lui, accepte tous les utilisateurs. Par défaut, il utilise le mode acquitté. Si la capacité PMP en mode acquitté est dépassée, il bascule sur le système non acquitté. On définit également les algorithmes associés dans le cas ou plusieurs canaux fréquentiels sont définis.

Enfin, on montre, grâce à l'étude précédente, que les retransmissions ARQ peuvent être utilisées jusqu'à une certaine limite en termes de nombre d'utilisateurs, permettant ainsi d'améliorer la qualité de la vidéo fournie.

Conclusions

Dans cette thèse, un système combinant le test d'hypothèses aux méchanismes de décodage robuste et de retransmissions ARQ a été proposé et étudié. Ce système est désigné par le nom de Soft ARQ (SARQ). A également été proposé et étudié un nouveau système de transmission d'une vidéo scalable à 2 niveaux dans un environnement Point-à-Multipoint. Une optimisation des systèmes de transmission d'une vidéo non scalable dans un environnement Point-à-Multipoint a aussi été effectuée.

L'étude du système SARQ a montré que

- Contrairement aux systèmes à décodage robuste (sans retransmissions) et aux systèmes à retransmissions basées sur le CRC, le système SARQ offre la possibilité de choisir le compromis débit/qualité grâce au seuil de test.

- Le gain en débit de ce système, comparé à l'ARQ classique (qui garantit une qualité nominale) augmente avec la capacité de correction du décodage robuste.

- Un mécanisme intercouches est nécessaire pour implémenter le décodage robuste et/ou le SARQ sur des systèmes pratiques.

L'étude sur la transmission d'une vidéo scalable à deux niveaux a montré que :

- Le nouveau système proposé est optimal en termes de performances tout en réduisant les besoins de bufferisation au niveau du terminal récepteur.
- L'augmentation du nombre d'utilisateurs ne détériore que légèrement les performances du système alors qu'elle améliore considérablement l'efficacité d'utilisation de la bande passante.

Enfin, l'étude sur la transmission d'une vidéo non scalable dans un environnement Point-à-Multipoint a montré que contrairement à ce qui est fait dans les systèmes Multicast/Broadcast actuels, on peut utiliser les retransmissions ARQ jusqu'à une certaine limite sur le nombre d'utilisateurs, permettant ainsi d'améliorer le niveau global de qualité vidéo apporté par ces services.

Contents

Acronyms

AM	Acknowledged Mode
APL	Application Layer
APP	A Posteriori Probability
ARQ	Automatic Repeat ReQuest
ASN	Access Service Network
ASN-GW	ASN Gateway
AWGN	Additive White Gaussian Noise
BCS	Block Control Sequence
BER	Bit Error Rate
BICM	Bit Interleaved Coded Modulation
BS	Base Station
BSC	Base Station Controller
BSN	Block Sequence Number
BSUC	Binary Symmetric Unicast Channel
BTS	Base Transceiver Station
CABAC	Context-based Adaptive Binary Arithmetic Coding
CAVLC	Context-based Adaptive Variable Length Coding
CN	Core Network
CPS	Common Part Sublayer
CSN	Connectivity Service
DRGR	Dynamic Retransmission Group Reduction
EDGE	Enhanced Data rates for Global Evolution
FCH	Frame Control Header
FCS	Frame Control Sequence
FRG	Fixed Retransmission Group
GGSN	Gateway GPRS Support Node
GPRS	General Packet Radio Service
HSDPA	High-Speed Downlink Packet Access
IP	Internet Protocol
LR	Likelihood Ratio
LRT	Likelihood Ratio Test
MRG	MultiReceiver Group
MS	Mobile Station
MSE	Mean Squared Error
NAL	Network Abstraction Layer
NALU	Network Abstraction Layer Unit
OFDM	Orthogonal Frequency Division Muliplexing
OFDMA	Orthogonal Frequency Division Multiple Access
OSI	Open Systems Interconnection

PDCP	Packet Data Convergence Protocol
PDU	Protocol Data Unit
PSNR	Peak Signal to Noise Ratio
RAN	Radio Access Network
RNC	Radio Network Controller
RTP	Real-time Transport Protocol
RTT	Round-Trip Time
SAP	Service Access Point
SDU	Service Data Unit
SGSN	Service GPRS Support Node
SN	Sequence Number s
SNDCP	Sub Network Dependent Convergence Protocol
SR	Selective Repeat
SS	Subscriber Station
SW	Stop and Wait
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UE	User Equipment
UM	Unacknowledged Mode
UMTS	Universal Mobile Telecommunications System
VLC	Variable Length Code
VCL	Video Coding Layer
WiMAX	Worldwide Interoperability for Microwave Access

List of Figures

Introduction

Multimedia services provided by wireless networks emerged as an important technology and attracted much attention recently, with the development of new telecommunication infrastructures and standards (GPRS/EDGE, UMTS/HSDPA, WiMAX, DVB-H). Unlike voice services, video services are characterized by large bandwidth requirements, which can be hundreds of times higher than the bandwidth required by voice services, and when these services are provided through wireless networks, one faces the problem of scarce bandwidth resources.

In multimedia applications, the data to be transmitted are compressed by a source encoder at the application layer. Therefore, one way of reducing the bandwidth used is to increase the compression rate in order to reduce the amount of data to be transmitted and hence the bandwidth necessary to transmit them. The mechanism for video compression utilizes a hybrid of temporal and spatial prediction, transform coding and variable length coding. The combination of these methods provides high compression gain, but at the same time makes the encoded video very sensitive to channel errors. A single transmission error can lead to the loss of large parts of the video at the receiver. The recovery of data loss within a video communication system can be solved by correcting errors using the redundancies inherent to the video stream, which is known as robustness or by retransmitting the erroneous packets, known as Automatic Repeat reQuest (ARQ). The two schemes do not allow to trade throughput for quality. Robustness-based schemes, by not using retransmissions, have maximum throughput but poor quality whereas ARQ schemes, by making use of retransmissions, provide the best possible quality at the expense of a low throughput. In this thesis, we propose a scheme which not only aims at improving the throughput-quality compromise but also provides the possibility to trade throughput for quality and vice versa, by tuning a parameter characterizing the system.

Another efficient way to reduce the necessary bandwidth would be by gathering all the customers asking for the same multimedia content in one group of receivers to which the data are conveyed using the same channel. The bandwidth is then reduced by a factor equal to the total number of receivers. This approach is valid in case multiple customers per cell are interested in the same content. It is particularly useful for multimedia applications like video file transfer or video streaming applications because many receivers may be interested in the same video content. The current Multicast/Broadcast services do not use retransmissions. However, retransmission of erroneous packets is very suitable to recover from missing data and improve the provided quality. In this thesis, we prove that retransmissions can be used in point-to-multipoint communications up to a given limit on the number of receivers.

On the other hand, scalable video codecs offer the possibility to have several qualities of the same encoded video, by providing at its output two streams (or more). If the video decoder is provided with the first encoded stream, the video obtained after the decoding operation is of basic quality. The other streams are quality enhancement streams, i.e. each time the decoder is provided with an additional stream, the displayed video quality is upgraded. In other words, the basic stream is indispensable if the end user wants to watch the video sequence while the other streams are only optional, i.e. it would be preferable but not indispensable to have them. In this thesis, schemes for the transmission of a two-level scalable video sequence towards several clients were studied.

Note that the focus is on the recent H.264/AVC standard but the proposed methods apply to other video coding standards too.

Chapter 1

Service provision over cellular networks

1.1 Introduction

This chapter provides a general description of the end-to-end link over which multimedia data are transmitted between an Internet server and an end user's mobile terminal, through a cellular network. We consider 2G, 3G and IEEE 802.16-2004 WiMAX systems.

Section 1.2 describes the physical architecture whereas section 1.3 describes the logical architecture of the networks considered.

1.2 End-to-end architecture

Most of the time, the terminal uses the uplink to request a service (data the terminal wants to access) that is provided by the network through the downlink. The requested data are generally stored and available at an Internet server, which represents the transmitter end, while the terminal represents the receiver end of the communication. The architecture of the network in question lies between these two ends (the Internet server and the terminal) as illustrated in Fig. 1.1.

The end-to-end link is composed of two parts : A wired part between the Internet server and the Base Station and a wireless part between the Base Station and the terminals. The wired part consists of the Radio Access Network (RAN), the Core Network (CN) and the Internet part (succession of Internet Routers the data packets are routed through up to the CN). Radio networks can then be seen as an extension of the Internet.

In the wired part, packets may be lost due to congestion (router buffer overflow) or misrouting. In the wireless part, packets may be lost due to transmission errors (erroneous packets are discarded at the receiver). When data are not available at the terminal, the corresponding packets may be retransmitted by the server or some other intermediate element of the radio network. Packet retransmission is known as Automatic Repeat ReQuest (ARQ).

In case the data missing at the receiver are not retransmitted, the receiver will have to do with some unavailable data, which affects the overall quality. In case of video for example, Error Concealment techniques may be used so that the decoder does not crash, but the received video will not have nominal quality.

In case the data missing at the receiver are retransmitted, the integrity of the received

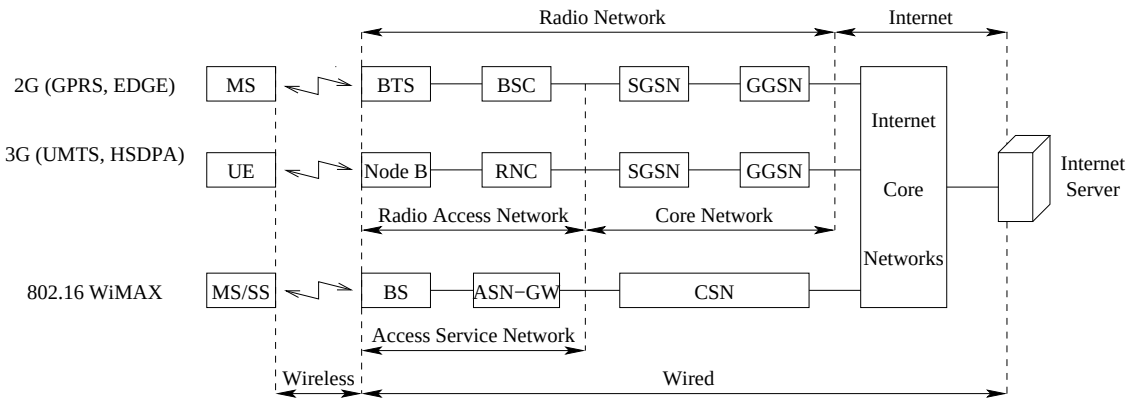

Figure 1.1: Architecture of a server-terminal end-to-end connection over 2G, 3G and WiMAX systems.

data is ensured. In case of video for instance, the received video will have nominal quality. When a packet is correctly received, the receiver sends an *acknowledgement* to the transmitter (server or intermediate network element), otherwise it either sends a *negative acknowledgement* or doesn't send anything (after a given time-out period, the packet is retransmitted). This requires the peer entities to be able to identify the different packets. In order to do so, the transmitter assigns a *sequence number* to each packet. If the sequence number is represented over n_s bits, the sequence range is $0, \ldots, 2^{n_s} - 1$. This sequence range is used cyclically. A sliding window of a size less than 2^{n_s} is used to ensure that the cyclically reusable sequence numbering works properly.

1.3 Cellular Networks protocol stacks

Two protocols are widely used for the transport of data over networks that use the Internet Protocol : the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). The Real-time Transport Protocol (RTP) is used in conjunction with UDP in case of real-time applications.

TCP ensures end-to-end reliable transfer through the use of retransmissions of lost packets. UDP provides a Best Effort transport, i.e. if packets are lost, they are not retransmitted.

Fig. 1.2 shows the whole protocol stack that may be used to transport data over a 2G(GPRS, EDGE), 3G(UMTS/HSDPA) or IEEE 802.16 WiMAX network. Note that packets follow either path 1 or path 2, i.e. TCP and the RTP/UDP combination cannot be used at the same time.

Considering the donwnlink, the downward direction represents the data flow on the network side (from the server to the base station) whereas the upward direction represents the data flow at the (receiving) terminal. A simplified representation of how the protocols are implemented on the different network elements is shown in Fig. 1.3. Note that other protocols are used at lower levels of intermediate elements.

Below, we briefly discuss the TCP, RTP, UDP and IP packet formats, as well as the

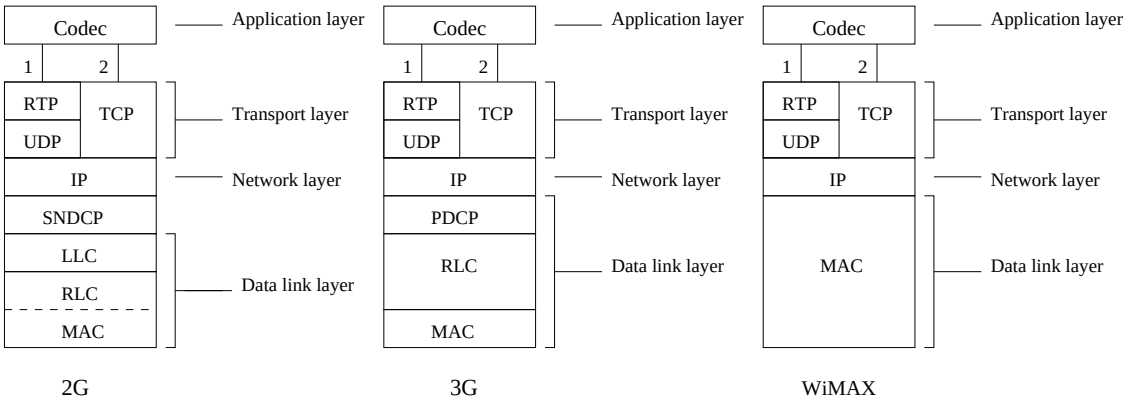

Figure 1.2: Protocol stack for the transport of real-time applications data over the 2G, 3G and WiMAX systems.

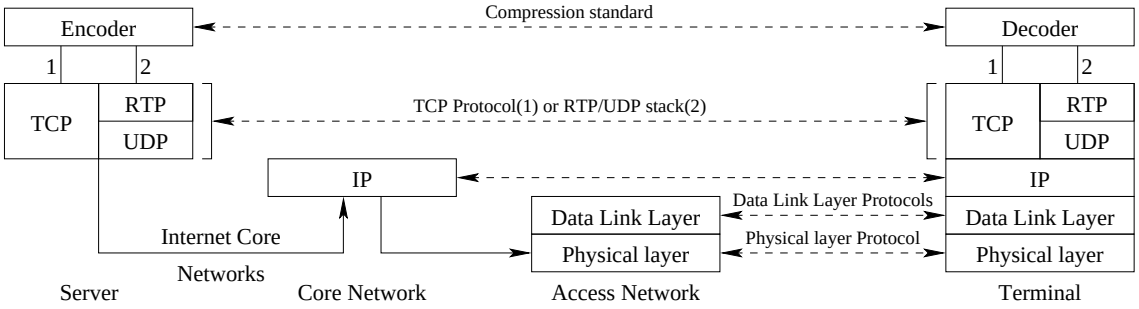

Figure 1.3: Protocol stack in an end-to-end link through a radio network.

data link layers of 2G and 3G systems. The MAC layer of IEEE 802.16-2004 WiMAX is discussed in more detail.

1.3.1 TCP packet format

When TCP is used (instead of UDP or RTP/UDP), it contains the application layer data, which are encapsulated in the *payload* field of the packet, and to which is appended a header. The header contains, among other fields:

- The *Source port* field (16 bits) identifies the sending port.
- The *Destination port* field (16 bits) identifies the receiving port.
- The *Sequence number* field (32 bits) is the sequence number of the first data octet in the packet.
- The *Data offset* field (4 bits) specifies the size of the TCP header in 32-bit words.
- The *Window* field (16 bits) specifies the size of the receive window, i.e the number of bytes (beyond the sequence number in the acknowledgement field) that the receiver is currently willing to receive.

1.3.2 RTP packet format

In real-time applications, the Real-time Transport Protocol (RTP) (in conjunction with UDP) is generally preferred to TCP. The RTP packet contains the application layer data, which are encapsulated in the *payload* field of the packet, and to which is appended a header. A *fixed header* is always present. This header may be extended. The fixed header contains, among other fields:

- The *PT (Payload Type* field (7 bits) identifies the content of the RTP payload and determines its interpretation by the application. For H264/AVC data, this value is equal to 105.
- The *Sequence number* field (16 bits) increments by 1 for each RTP packet.

The *extension header* allows individual implementations to experiment with functions that require additional information to be carried in the RTP header.

1.3.3 UDP packet format

When UDP is used in conjunction with RTP, it contains the RTP data (when used alone, it contains the application data), which are encapsulated in the *payload* field of the packet, and to which is appended a 64-bit header. The fields of the header are described as follows

- The *Source Port* field (16 bits) identifies the sending port.
- The *Destination Port* field (16 bits) identifies the receiving port.
- The *Length* field (16 bits) specifies the length in bytes of the entire datagram (packet) : header and data
- The *Checksum* (16 bits) is used for error-checking of the header and data.

1.3.4 IP packet format

There are two widely deployed versions of the IP protocol : IPv4 and IPv6. We only consider IPv4 since it is the dominant version. IP packets contain the transport layer data, which are encapsulated in the *payload* field of the packet, and to which is appended a header. The header contains, among other fields:

- The *Version* field (4 bits) specifies the IP protocol version. For IPv4, this has a value of 4.
- The *Protocol* field (8 bits) indicates the next level protocol used in the data portion of the IP packet. TCP corresponds to value 6 and UDP to value 17.
- The *Header checksum* (16 bits) is used for error-checking of the header.
- The *Source address* (32 bits) contains IP address of the sender.
- The *Destination address* (32 bits) contains the IP address of the receiver.
- The *Options* field (variable size) contains additional header fields that may follow the destination address field.

1.3.5 Data Link Layer

In UMTS and HSDPA (see Fig. 1.4), The IP packet is encapsulated into a Packet Data Convergence Protocol (PDCP) PDU. The PDCP layer performs TCP/IP and UDP/IP header compression and decompression. The PDCP PDU is then delivered to the RLC sublayer and fits into an RLC SDU. The RLC sublayer segments the RLC SDU and adds a header to each segment to form the RLC PDU. The MAC layer adds its header to the RLC PDU to form the MAC PDU. A CRC is appended to the MAC PDU data prior to its delivery to the physical layer.

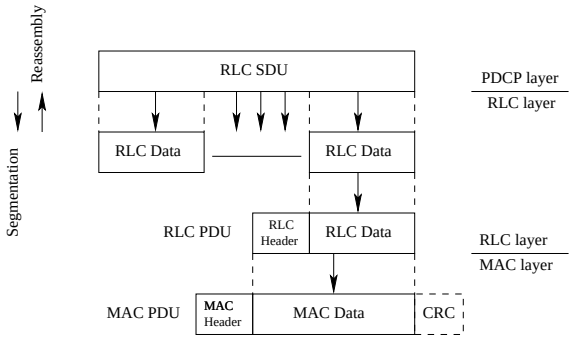

Figure 1.4: Data flow at the link layer in 3G systems.

In GPRS and EDGE (see Fig. 1.5). The RLC and MAC layers are implemented in one layer. After segmentation, an RLC header is added to each segment, then a CRC called the Burst Control Sequence (BCS) protecting the RLC header and data is appended, and finally the MAC header is added to form the RLC/MAC radioblock which is delivered to the physical layer.

The LLC frame segmented by the RLC/MAC layer contains a Frame Header (FH) and a Frame Control Sequence (FCS). The FCS is a 24-bit CRC that protects both the header and the data. The LLC frame data represent the Sub Network Dependent Convergence Protocol (SNDCP) layer data. The SNDCP layer encapsulates the IP packets into sub-network formats (called SNPDUs) and performs header compression to make for efficient data transmission.

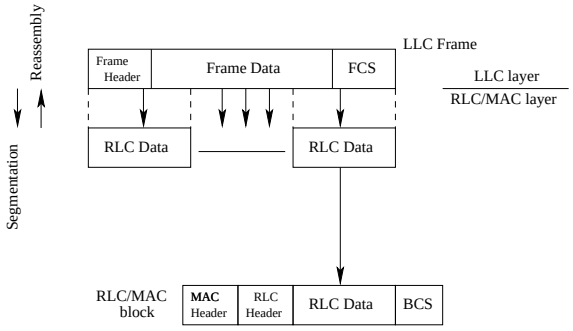

Figure 1.5: Data flow at the link layer in 2G systems.

In IEEE 802.16-2004 WiMAX, the MAC layer comprises three sublayers (see Fig. 1.6) :

- The service-specific Convergence Sublayer (CS).
- The MAC Common Part Sublayer (CPS).
- The security sublayer.

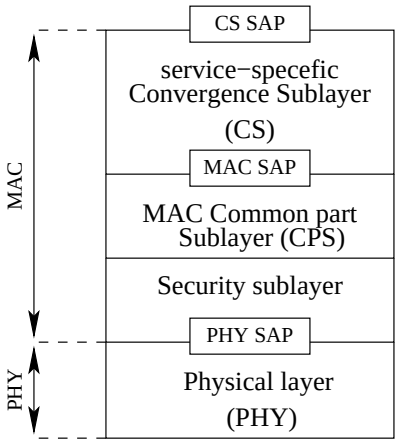

Figure 1.6: IEEE 802.16 protocol layers.

There are two types of PDUs at the MAC level (see Fig. 1.6) :

- The CS PDU exchanged between CS and the MAC CPS.
- The MAC CPS PDU exchanged between the security sublayer and the PHY layer.

There is no PDU exchanged between MAC CPS and the security sublayer because the security sublayer doesn't add any header, it just encrypts the MAC PDU (except its header) and passes it to the PHY layer. That is why the PDU exchanged between the security sublayer and the PHY layer is the MAC CPS PDU, which could be encrypted in case of a secure connection. The MAC CPS PDU also represents the PDU of the whole MAC layer, that is why it is called the MAC PDU, bearing in mind that it is a possibly encrypted MAC CPS PDU. Similarly, the CS SDU represents the SDU of the whole MAC layer, that is why it is called the MAC SDU.

1.3.6 The Convergence Sublayer

The CS resides on top of the MAC CPS. It is used for the transport of all packet-based protocols such as Internet Protocol (IP), Point-to-Point Protocol (PPP), and the IEEE 802.3 (Ethernet). It optionally suppresses a repetitive portion of the header at the transmitter and restores it at the receiver.

In the sequel we will only consider the case of the IP protocol since we are interested in the case of the transmission of video data originating from an internet server using the RTP/UDP/IP or the TCP/IP protocol stacks under the application layer.

1.3.7 The Common Part Sublayer

The MAC CPS provides the core MAC functionality of system access, bandwidth allocation, connection establishment, and connection maintenance. It receives the data from the CS through the MAC SAP.

The MAC (CPS) PDU format There are three types of MAC PDUs:

- MAC management PDUs (control messages).
- User data PDUs.
- Bandwidth request PDUs.

MAC PDUs shall be of the form illustrated in Fig. 1.7. Each PDU shall begin with a 6-byte header and may contain a 32-bit CRC.

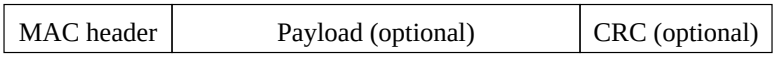

Figure 1.7: MAC PDU general format.

Bandwidth request PDUs contain a *bandwidth request header* and no payload. MAC management PDUs and user data PDUs contain a *generic header* and a payload. The payload consists of subheaders and higher layer data.

Five types of subheaders may be present. Only three of them are relevant for the downlink:

1. The Fragmentation subheader, used only when packing is not used regardless of whether fragmentation is used or not.
2. The FAST-FEEDBACK allocation subheader, used only with the OFDMA PHY (2 types of physical layer are possible: the OFDM PHY and the OFDMA PHY, both use the OFDM transmission technique).
3. The packing subheader, used only when packing is used.

The packing and fragmentation subheaders are mutually exclusive (they cannot both be present within the same MAC PDU).

The packing subheader is said to be a *per SDU subheader*. The other subheaders are called *per PDU* subheaders. The per PDU subheaders appear only once in the PDU. The Packing SubHeader (PSH) appears as many times as there are SDUs or fragments of SDUs packed in the PDU in question. When packing is not used, the payload contains one per PDU subheader or several per PDU subheaders appearing once followed by an SDU (or fragment of SDU) as shown in Fig. 1.8.

When packing is used, the payload contains one or several per PDU subheaders appearing once, followed by a series of SDUs (or fragments of SDU) each preceded by the corresponding PSH (see Fig. 1.9). There are as many packing subheaders as packed SDUs (or fragments of SDUs) because the PSH contains the Sequence Number of the packed SDU (or fragment of SDU).

The generic header contains, among other fields:

Generic MAC Header	per PDU subheaders	SDU (or fragment of an SDU)	CRC (optional)

Figure 1.8: MAC PDU format when packing is not used.

Generic MAC Header	per PDU subheaders	PSH 1	SDU (or fragment of an SDU) 1	PSH 2	SDU (or fragment of an SDU) 2	CRC (optional)

Figure 1.9: MAC PDU format when packing of two SDUs (or fragments of SDUs) is used.

- The *CI (CRC Indicator)* field (1 bit) indicates whether a CRC is appended to the PDU (payload and header) or not.
- The *EC (Encryption Control)* field (1 bit) indicates whether the payload is encrypted or not.
- The *LEN (Length)* field (11 bits) indicates the length in bytes of the MAC PDU including the MAC header and the CRC if present.
- The *Type* field (6 bits) indicates the subheaders and the special payloads present in the message payload, notably:
  - The second bit indicates whether a packing subheader is present or not.
  - The third bit indicates whether a fragmentation subheader is present or not.
  - The fourth bit (Extended Type) indicates whether the present fragmentation subheader or packing subheaders are extended or not (see below).

The Fragmentation Subheader (FSH) contains the following fields:

1. Frame Control (FC) field (2 bits) : Indicates the fragmentation state of the payload

   00 = no fragmentation

   01 = last fragment

   10 = first fragment

   11 = continuing (middle) fragment

2. If ARQ is enabled, a Block Sequence Number (BSN) field (11 bits) : Sequence number of the first block in the current SDU (fragment). Else, a Fragment Sequence Number (FSN) of either 3 bits or 11 bits (depending on the value of the Extended Type bit).
3. *Reserved* field (3 bits) set to zero.

The Packing Subheader (PSH) contains the following fields:

1. Frame Control (FC) field (2 bits) : Indicates the fragmentation state of the payload

   00 = no fragmentation

   01 = last fragment

10 = first fragment

11 = continuing (middle) fragment

2. If ARQ is enabled, a Block Sequence Number (BSN) field (11 bits), sequence number of the first block in the current SDU or fragment of SDU (see below). Else, a Fragment Sequence Number (FSN) of either 3 bits or 11 bits (depending on the value of the Extended Type bit).

3. *Length* field (11 bits) : Length of the SDU (fragment), including the packing sub-header.

SDU reconstruction and delivery when ARQ is not enabled When ARQ is not enabled, SDUs or fragments of SDUs are numbered using the Fragment Sequence Number (FSN).

In case fragmentation is used, the receiver uses the FC field of the FSH to reassemble the fragmented SDUs. When the FC field contains the binary word 00, the unfragmented SDU is delivered directly to the upper layers. When the FC field contains the code 10 or the code 11, the fragment of the SDU is buffered until the last fragment (FC=01) is received. As soon as the last fragment is received, the SDU is reconstructed and delivered to the upper layers (see Fig. 1.10).

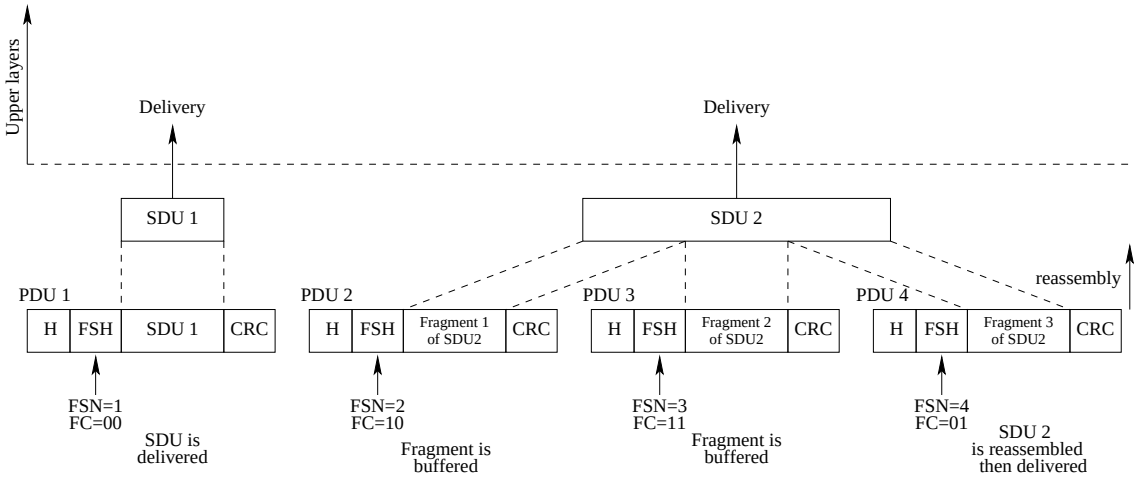

Figure 1.10: SDU reconstruction when packing is not used.

When packing is used (possibly with fragmentation), the length fields of the PSHs are used to determine the SDUs (or fragments of SDUs) that have been packed in the PDU. Then the FC fields of the PSHs are used for the (possible) reassembly and the delivery of the SDUs to the upper layers (see Fig. 1.11).

SDU reconstruction and delivery when ARQ is enabled The ARQ mechanism is part of the MAC.

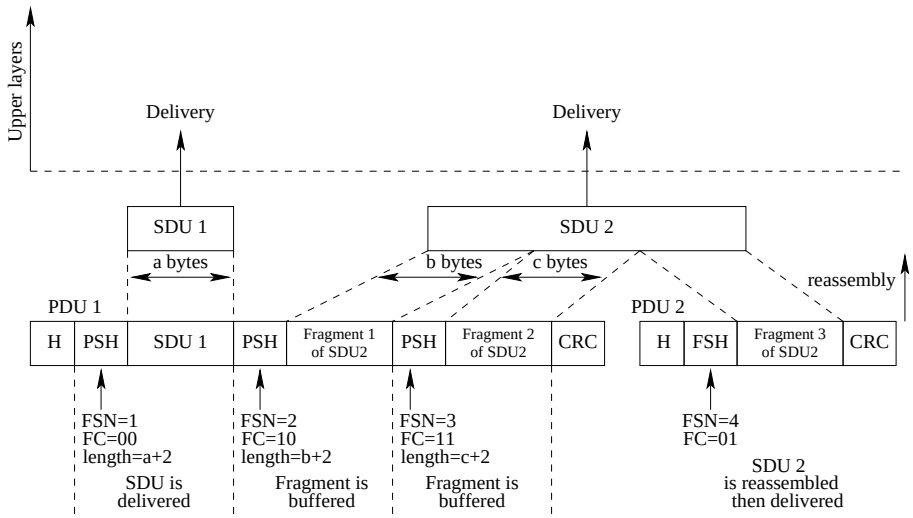

Figure 1.11: SDU reconstruction when packing is used (unextended headers are assumed).

A MAC SDU is *logically* partitioned into blocks whose length is specified by a parameter called ARQ-BLOCK-SIZE. When the length of the SDU is not an integer multiple of the block size, the final block of the SDU is formed using the SDU bytes remaining after the final full block has been determined. Fragmentation occurs only on ARQ block boundaries (see Figs. 1.12 and 1.13).

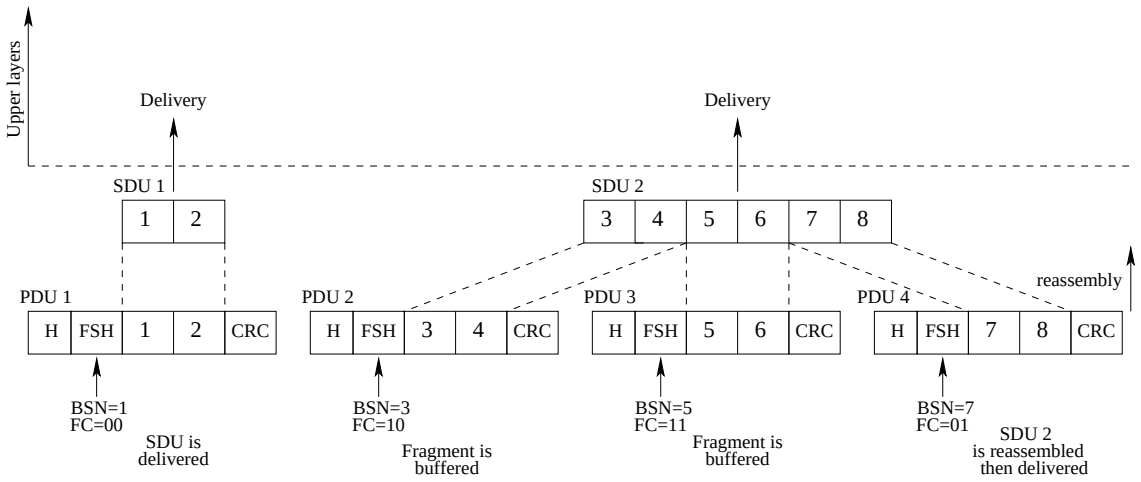

Figure 1.12: Block partitioning and SDU reconstruction when packing is not used and ARQ is enabled.

Fragmentation and Packing subheaders contain a BSN (Block Sequence Number), which is the sequence number of the first ARQ block in the data following the subheader (see Figs. 1.12 and 1.13).

The ARQ mechanism is discussed in detail in appendix B.

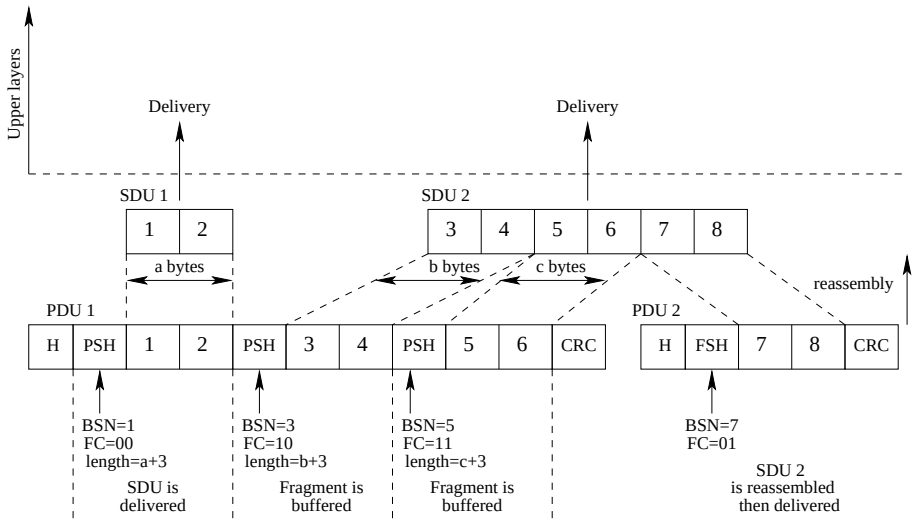

Figure 1.13: Block partitioning and SDU reconstruction when packing is used and ARQ is enabled.

1.3.8 The security sublayer

Definition (Security Association) The set of security information a BS and one or more SSs share in order to support secure communications.

When encryption is used, the MAC PDU payload is encrypted at the transmitter and decrypted at the receiver. The generic MAC header shall not be encrypted (see Fig. 1.14).

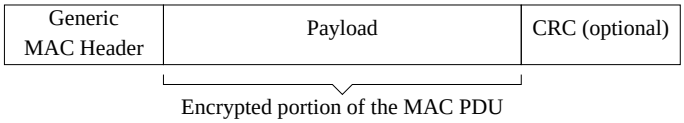

Figure 1.14: MAC PDU encryption.

In our study, we do not consider data encryption, that is to say, no Security Association is mapped to our connections.

1.3.9 Concatenation

Concatenation is the process by which multiple MAC PDUs (MPDUs) are combined into a single PHY SDU or burst. MAC management MPDUs, user data and bandwidth request MPDUs may be combined in the same burst.

Figure 1.15 shows a MAC burst in which n MAC PDUs have been concatenated. Note that the physical layer is OFDM-based and padding is generally used so that the size of the burst corresponds to an integer number of OFDM symbols at the PHY level, i.e the PHY burst contains an integer number of OFDM symbols.

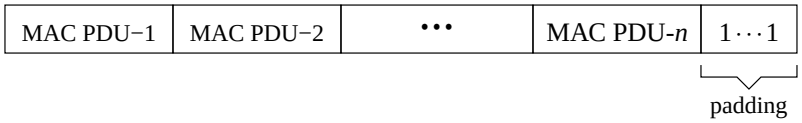

Figure 1.15: Concatenation of several MPDUs into a single transmission (burst).

1.4 Conclusion

In this chapter, a general description of 2G, 3G and IEEE 802.16-2004 WiMAX cellular networks was given. The physical architecture and the protocol stack of the end-to-end link over which multimedia services are provided to mobile end users were both addressed. The stress was put more on the IEEE 802.16-2004 fixed WiMAX for IEEE 802.16-2004 was an established standard (when this work was underway contrary to the 802.16e mobile WiMAX standard which was still in progress) meant for multimedia service provision, in urban areas (in rural areas, it is meant for provision of Internet access to customers who do not have access to DSL). Chapter 2 discusses video compression and the application layer in the case of an H264 video transmission.

Chapter 2

Video Compression

2.1 Introduction

This chapter is devoted to video compression. General digital video-related notions are briefly discussed in sections 2.2 and 2.3. An overview of the H264/AVC video standard is given afterwards in section 2.4.

Note that most of the material in this chapter was copied from the thesis of Cédric Marin (103).

2.2 Video Capture

A digital video is first captured by a camera and compressed by a video encoder. Then, it is either stored or transmitted.

Digital video is the representation of a sampled video scene in digital form. Each spatio-temporal sample (picture element or *pixel*) is represented as a number or set of numbers that describes the brightness and colour of the sample.

In the RGB colour system, a colour image sample is represented with three numbers that indicate the relative proportions of Red, Green and Blue (the three additive primary colours of light). Any colour can be created by combining red, green and blue in varying proportions.

The YUV colour system is another way of efficiently representing colour images. Y is the luminance (luma) component whereas U and V are the color (chrominance or chroma) components. Components Y, U and V are calculated as a weighted average of R, G and B.

2.3 Video Quality

In order to specify, evaluate and compare video communication systems it is necessary to determine the quality of the video images displayed to the viewer. Measuring visual quality is difficult and often imprecise because there are so many factors that can affect the results. Visual quality is inherently *subjective* and is influenced by many factors that make it difficult to obtain a completely accurate measure of quality. For example, a viewer's opinion of visual quality can depend very much on the task at hand, such as passively watching a DVD movie, actively participating in a videoconference or trying

to identify a person in a surveillance video scene. Several test procedures for subjective quality evaluation are defined in ITU-R Recommendation BT.500-11 (38).

The complexity and cost of subjective quality measurement make it attractive to be able to measure quality automatically using an algorithm. Developers of video compression and video processing systems rely heavily on so-called objective (algorithmic) quality measures. The most widely used measure is Peak Signal to Noise Ratio (PSNR), which depends on the Mean Squared Error (MSE) between an original image I_o and a reconstructed image I_r, relative to $(2^n-1)^2$ (the square of the highest-possible signal value in the image, where n is the number of bits per image sample).

Considering images of size $L \times H$ pixels, the MSE is defined by

$$MSE = \frac{1}{L \cdot H} \sum_{i=1}^{H} \sum_{j=1}^{L} [I_o(i,j) - I_r(i,j)]^2 \tag{2.1}$$

and the PSNR is given by

$$PSNR = 10 \log_{10} \left(\frac{(2^n-1)^2}{MSE} \right) \tag{2.2}$$

Note that, generally, reconstructed images have an acceptable quality starting from 30 dB of PSNR and that image samples are represented over 8 bits ($n = 8$) most of the time.

2.4 The H264/AVC standard

The H264/AVC standard is used as our video reference in this thesis.

In H264/AVC, the basic processing element is the *Macroblock*. A macroblock is an image area of 16×16 luma samples and associated chroma samples (8×8 samples with the 4:2:0 sampling pattern). In order to encode the whole image, the decoder encodes successively all its macroblocks.

Further, some encoding operations require smaller elements for their corresponding processing. The basic element of a macroblock is the *block*. A block corresponds to an image area of 4×4 pixels, i.e. a luminance macroblock consists of 16 blocks and a chrominance macroblock consists of 4 blocks. Figure 2.1 shows the subdivision of a luminance image into macroblocks and blocks.

The H264/AVC encoder consists of a Video Coding Layer (VCL), which converts the captured video sequence into a compressed binary stream and a Network Abstraction Layer (NAL) which formats the binary stream into a flow of video packets (see Fig. FigH264Layers).

2.4.1 The Video Coding Layer (VCL)

General structure The block diagram of an H264 encoder is illustrated in Fig. 2.3. The encoder includes two dataflow paths, an *encoding path* and a *decoding path*.

An input frame is processed in units of a macroblock. Each macroblock is encoded in intra or inter mode and, for each block in the macroblock, a prediction is formed based on reconstructed picture samples. In Intra mode, the prediction is formed from samples in the current frame that have previously been encoded, decoded and reconstructed. In

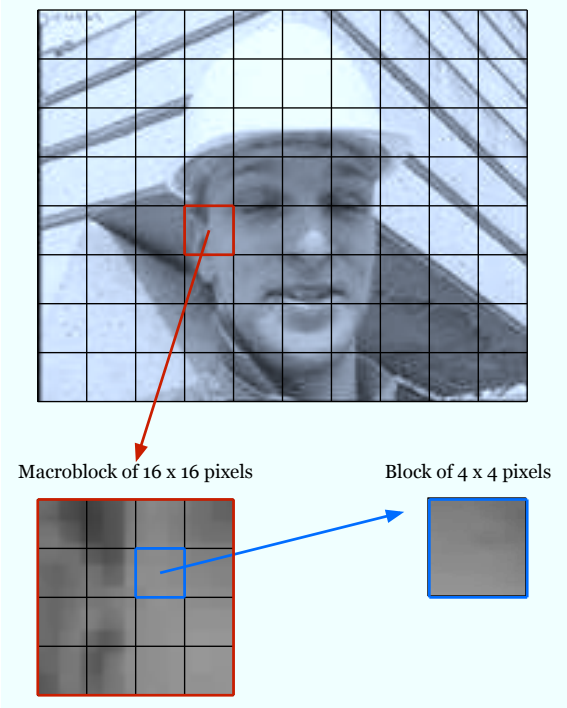

Figure 2.1: Subdivision of a luminance image into macroblocks and blocks.

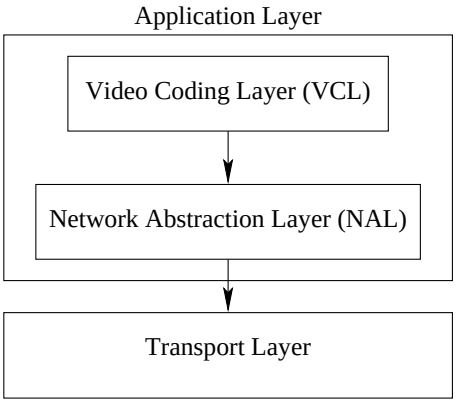

Figure 2.2: H264/AVC sublayers.

Inter mode, the prediction is formed by motion-compensated prediction from a reference picture. The prediction reference picture for each macroblock partition may be chosen from a selection of past or future pictures (in display order) that have already been encoded, reconstructed and filtered. The prediction is subtracted from the current block to produce a residual (difference) block that is transformed (using a block transform) and quantized to give a set of quantized transform coefficients which are reordered and entropy encoded. The entropy-encoded coefficients, together with the motion vectors and side information required to decode each block within the macroblock (prediction modes, quantizer parameter, etc.) form the compressed bitstream which is passed to a Network Abstraction Layer

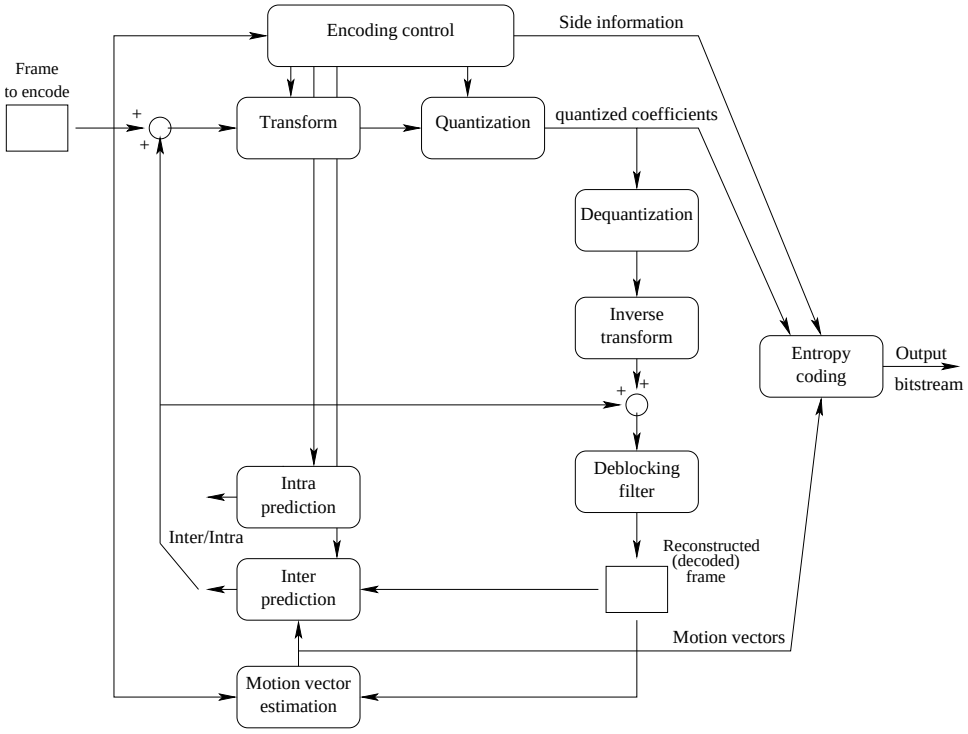

Figure 2.3: H264 encoder.

(NAL) for transmission or storage.

As well as encoding and transmitting each block in a macroblock, the encoder decodes (reconstructs) it to provide a reference for further predictions. The quantized transform coefficients are dequantized and inverse transformed to produce a difference block. The prediction block is added to prediction block to create a reconstructed block (a decoded version of the original block). A filter is applied to reduce the effects of blocking distortion and the reconstructed reference picture is created from a series of reconstructed blocks.

The structure of the decoder is much simpler than that of the encoder. The decoder receives a compressed bitstream from the NAL and entropy decodes the data elements to produce a set of quantized transform coefficients. The quantized transform coefficients are then dequantized and inverse transformed to produce a difference block. The prediction block is added to create a reconstructed block (a decoded version of the original block). A filter is applied to reduce the effects of blocking distortion and the reconstructed reference picture is created from a series of reconstructed blocks.

Five types of images are supported by the H.264/AVC standard: I, P, B, SI and SP. SI and SP images are used to switch between different video streams and are not used in the presence of only one video stream. An I image may contain only I macroblock types, a P image may contain P and I macroblock types and a B image may contain B and I macroblock types. I macroblocks are predicted using intra prediction from decoded samples in the current slice. P macroblocks are predicted using inter prediction, each from one reference picture. B macroblocks are predicted using inter prediction from 2 reference

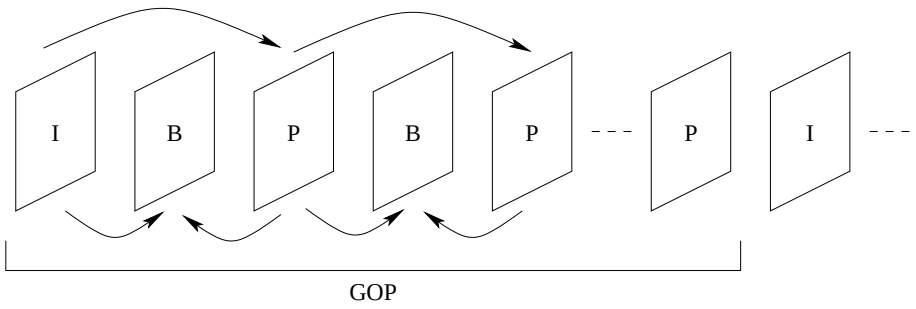

Figure 2.4: Structure of a GOP and dependancies between frames.

pictures.

Ideally, an I image should appear with a change of scene, i.e. when the temporal redundancies are negligeable. However, detecting a change of scene is a difficult task (algorithmically) and therefore encoders include an I image at regular intervals, separating thus different Groups Of Pictures (GOPs). A GOP starts with an I image and is followed by a sequence of P and B images. The standard structure of a GOP is illustrated in Fig. 2.4. The first image of a GOP resets the reference image memory (*Instantaneous Decoding Refresh* or IDR). This technique allows the decoder to resynchronize with the stream in case of lossy transmission.

Below, we describe the operation of the different tools discussed above in more details.

Intra prediction In intra mode a prediction block is formed based on previously encoded and reconstructed blocks and is subtracted from the current block prior to encoding. For the luma samples, a prediction block is formed for each 4×4 block or for a 16×16 macroblock. There are a total of nine optional prediction modes for each 4×4 luma block, four modes for a 16×16 luma block and four modes for the chroma components. The encoder typically selects the prediction mode for each block that minimizes the difference between the prediction block and the block to be encoded.

Inter prediction Inter prediction creates a prediction model from one or more previously encoded video frames using block-based motion compensation. The luminance component of each macroblock (16×16 samples) may be split up in four ways (see Fig. 2.5) and motion compensated either as one 16×16 macroblock partition, two 16×8 partitions, two 8×16 partitions or four 8×8 partitions. If the 8×8 mode is chosen, each of the four 8×8 sub-macroblocks within the macroblock may be split in a further 4 ways (see Fig. 2.6), either as one 8×8 sub-macroblock partition, two 8×4 sub-macroblock partitions, two 4×8 sub-macroblock partitions or four 4×4 sub-macroblock partitions.

Each partition or sub-macroblock partition in an inter-coded macroblock is predicted from an area of the same size in a reference picture. A motion vector is associated to each partition of a macroblock. This vector represents the offset between the two areas (the partition to be predicted and the area matching it in the reference picture).

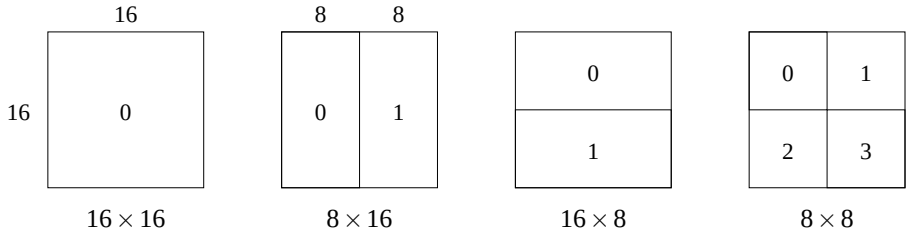

Figure 2.5: Macroblock partitions: 16×16, 8×16, 16×8, 8×8.

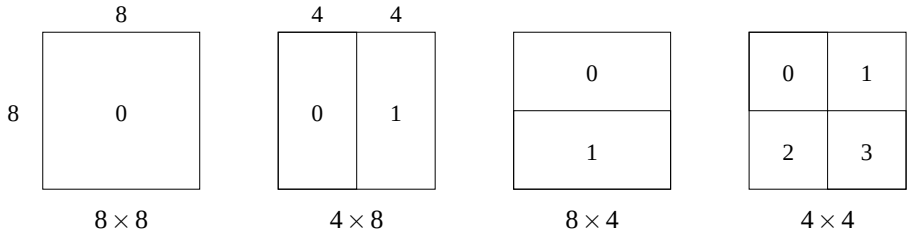

Figure 2.6: Sub-macroblock partitions: 8×8, 4×8, 8×4, 4×4.

Each chroma component in a macroblock (U and V) has half the horizontal and vertical resolution of the luminance (luma) component. Each chroma block is partitioned in the same way as the luma component, except that the partition sizes have exactly half the horizontal and vertical resolution (an 8×16 partition in luma corresponds to a 4×8 partition in chroma; an 8×4 partition in luma corresponds to 4×2 in chroma and so on). The horizontal and vertical components of each motion vector (one per partition) are halved when applied to the chroma blocks.

Transform and quantization H.264 uses three transforms depending on the type of residual data that is to be coded: a core (DCT-based) transform for all 4×4 luma and chroma blocks in the residual data, a Hadamard transform for the 4×4 array of luma DC coefficients in intra macroblocks predicted in 16×16 mode and a Hadamard transform for the 2×2 array of chroma DC coefficients.

The DCT based transform is applied to all 4×4 residual luma and chroma blocks and is based on the matrix H_1 given by

$$H_1 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & 2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \tag{2.3}$$

If the macroblock is encoded in 16×16 Intra prediction mode, the DC coefficients of each 4×4 block (transformed using the core transform described above) are transformed again using the 4×4 Hadamard transform with the following matrix:

Positive value	Signed value	Codeword
0	0	1
1	+1	010
2	−1	011
3	+2	00100
4	−2	00101
5	+3	00110
6	−3	00111
7	+4	0001000
8	−4	0001001
9	+5	0001010
10	−5	0001011
...	...	...

Table 2.1: Exp-Golomb coding mapping table.

$$H_2 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \tag{2.4}$$

Also, the DC coefficients of each 4×4 block of chroma coefficients are grouped in a 2×2 block and are further transformed using the 2×2 Hadamard transform with the following matrix

$$H_3 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{2.5}$$

A matrix X is transformed into matrix Y according to the equation

$$Y = H_i \cdot X \cdot H_i^T \tag{2.6}$$

where H_i may represent H_1, H_2 or H_3.

Entropy Coding In H264, elements are encoded using either the Context-based Adaptive Based Varibale Length Codes (CAVLC) (41) or Context-based Adaptive Binary Arithmetic Coding (CABAC) (42).

In this thesis, we have used only CAVLC. In CAVLC, two compression techniques are used. The first technique is based on Exponential-Golomb (Exp-Golomb) coding (43) and encodes all elements (MB type, quantizer step, motion vectors ...) except for residuals. The second technique encodes the residual transform coefficients.

Exp-Golomb coding In this type of coding, every element is first represented by an integer value, and then mapped onto a VLC according to table 2.1.

Encoding of residual transform coefficients After prediction, transformation and quantization, blocks are typically sparse (contain mostly zeros) and the highest non zero coefficients are often equal to ± 1. These coefficients will be denoted T1s.

As shown in Fig. 2.7, each 4×4 block of quantized transform coefficients is mapped to a 16-element array in a zig-zag order. The coefficients are thus sorted in increasing order of frequency (the higher the index of the coefficient in the array, the higher its frequency) and their levels will statistically tend to decrease.

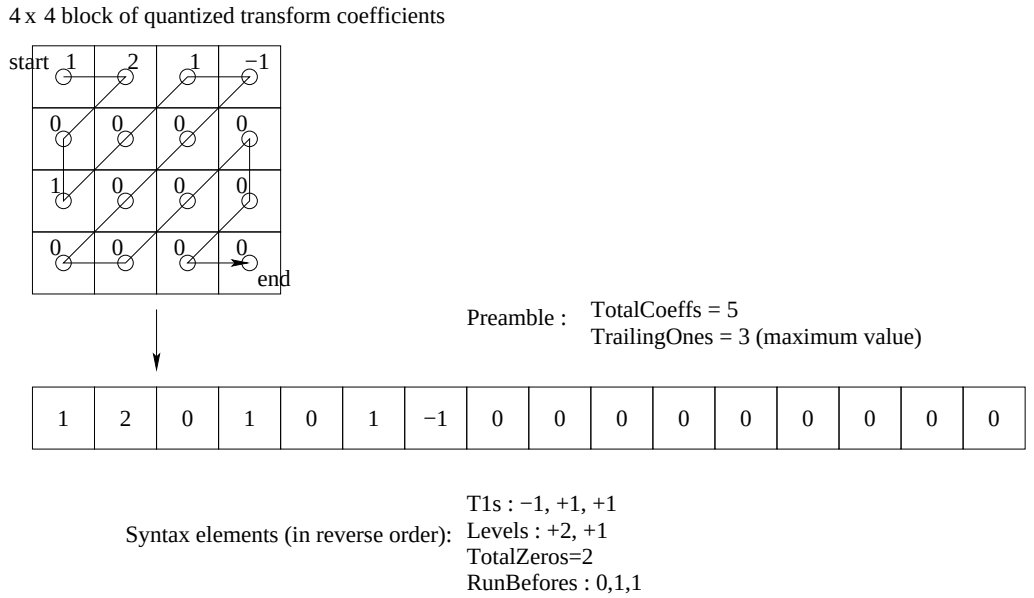

Figure 2.7: Encoding of residual transform coefficients of a 4×4 block using CAVLC.

The number of non-zero coefficients (*TotalCoeffs*) and the number of T1s (*TrailingOnes*) are encoded first. These two parameters are combined in the same codeword (*CoeffToken*), obtained using one of the VLC tables defined in Figs 1 and 2 of appendix C. The selection of the VLC table depends on the number of non-zero coefficients contained in the neighbouring blocks.

At the second step, the sign and magnitude of each non-zero coefficient are encoded *in reverse order*, starting with the highest frequency and working back towards the DC coefficient. For the encoding of T1s, only the sign is encoded with a single bit. For the encoding of the other non-zero coefficients (*Levels*), the procedure is more complex. This technique assumes that the magnitude of coefficients increases as the frequency decreases. In a simplified way, the VLC table used to encode the sign and magnitude of a coefficient is based on the previous coefficient.

The number of zeros preceding the highest frequency nonzero coefficient (*TotalZeros*) is encoded at the third step using the VLC tables illustrated in Fig. 5 of appendix C. The VLC table is chosen as a function of *TotalCoeffs*.

Finally, the number of zeros preceding each non-zero coefficient (*RunBefore* is encoded

in reverse order using one of the tables illustrated in Fig. 7 of appendix C. The choice of the table depends on the number of zeros that have not yet been encoded (*ZerosLeft*).

2.4.2 The Network Abstraction Layer (NAL)

A video picture is coded as one or more slices, each containing an integral number of macroblocks from 1 (1 MB per slice) to the total number of macroblocks in a picture (1 slice per picture). A coded picture may be composed of different types of slices. Fig. 2.8 shows a simplified illustration of the syntax of a coded slice. The slice header defines (among other things) the slice type and the coded picture that the slice belongs to. The slice data consists of a series of coded macroblocks. Each MB consists of a MB header and MB data. The MB header contains parameters like the MB type, the prediction mode and the sub-macroblock partitioning. The MB data contains the coded residual data (CAVLC sequences). Note that the MB header contains a 4-bit parameter called *coded-block-pattern*, each bit of which corresponds to an 8×8 block of the MB. When the 8×8 block is not coded (all zero), the corresponding bit is set to 0. On the other hand, when the 8×8 is coded (into 4 CAVLC sequences), the bit is set to 1. The number of CAVLC sequences in the MB data field is $4 \times W_H(coded - block - pattern - luma)$ where W_H denotes the Hamming weight.

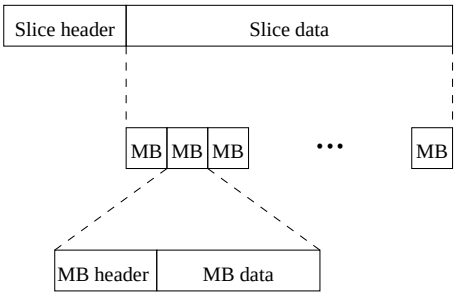

Figure 2.8: Slice syntax.

The Network Abstraction Layer encapsulates the coded slice into a Network Abstraction Layer Unit (NALU) by appending to it a NALU header (the coded slice being the NALU data).

2.5 Conclusion

In this chapter, video compression was discussed based on the H.264/AVC standard. First, general matters specific to video sequences such as video capture and video quality were discussed. Then, the H.264/AVC standard was described briefly. This standard is the latest in video compression and is used as our reference for video compression.

In H.264/AVC video transmission applications, the H.264/AVC codec corresponds to the application layer at the top of the overall protocol stack discussed in chapter 1. The lower layers (RTP/UDP/IP/DLL/PHY or TCP/IP/DLL/PHY) were detailed in chapter 1 whereas the application layer was detailed in this chapter and consists of two sublayers: the Video Coding (sub)Layer (VCL) and the Network Abstraction (sub)Layer (NAL).

The VCL sublayer converts an image sequence into a coded bitstream. The NAL sublayer formats the coded bitstream into a sequence of video packets called NAL Units (NALUs) which are delivered to the lower layer (RTP/UDP or TCP) for transport over the network.

The next chapters discuss the enhancement of video transmission based applications and services.

Chapter 3

Improved Retransmission Scheme for Video Communications

3.1 Introduction

In wireless communications, the received signal may be heavily corrupted and results in many errors. Besides, highly compressed streams like video streams are very sensitive to transmission errors. A single transmission error can lead to the loss of large parts of the video at the receiver. The recovery of data loss within a digital communication system has classically been solved via two methods : Automatic Repeat reQuest (ARQ) and Forward Error Correction (FEC). ARQ involves receiver loss notification to the source (through a return channel) and subsequent retransmission of lost data. In contrast, FEC involves the repairing of lost data over the transmission channel through the use of redundancies added by a channel encoder at the transmitter. The two schemes do not allow to trade throughput for quality. FEC schemes, by not using retransmissions, have maximum throughput but poor quality whereas ARQ schemes, by making use of retransmissions, provide the best possible quality at the expense of a low throughput. In this chapter, we propose a scheme which not only aims at improving the throughput-quality trade-off but also provides the possibility to trade throughput for quality or vice versa, by tuning a parameter characterising the system.

Usually, both methods (ARQ and FEC) are jointly used since even in the presence of FEC, there may be some residual errors. In this chapter, we concentrate on ARQ techniques, and for easing the notations, the FEC is not explicitly taken into account in the notations.

The rest of this chapter is organized as follows : Section 3.2 reviews the conventional multimedia transmission systems. In section 3.3, a new scheme is proposed. Section 3.4 explains how this scheme is applied to the transmission of H264 video data. Section 3.5 presents simulation results. Section 3.6 addresses the question of the practical implementation of the scheme and finally, section 3.7 concludes the chapter.

3.2 Conventional compressed data transmission systems

Consider the general uncoded multimedia transmission system of Fig. 3.1. The source samples are coded into a binary stream $\underline{s}$ by the source encoder. The binary stream $\underline{s}$ is

then mapped onto a symbol stream $\underline{x}$ (according to a given modulation) that is transmitted over a discrete physical channel. At the receiver, an estimation $\underline{\tilde{s}}$ of the transmitted binary stream $\underline{s}$ is obtained based on the received symbol stream $\underline{r}$ by first taking a decision on the transmitted symbols, and then performing the inverse mapping. This stream is then used by the source decoder to reconstruct the source samples. Since there are errors in the estimated sequence $\underline{\tilde{s}}$, the reconstructed source is different from the original one. The Mean Square Error (MSE) is generally used to measure the distortion between the original sequence and the reconstructed one. In the case of video communication systems, the Peak Signal to Noise Ratio (PSNR) is used instead of the MSE.

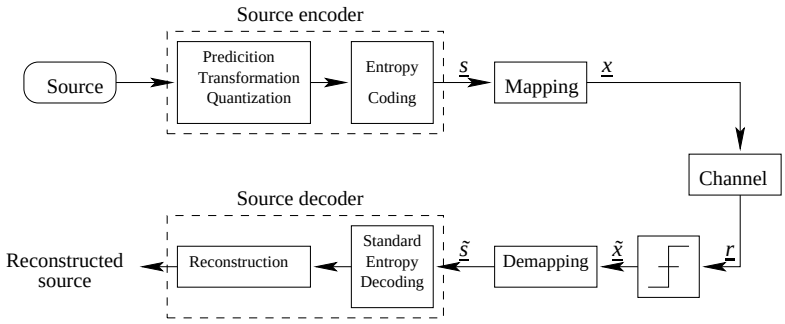

Figure 3.1: General block diagram of a basic multimedia transmission system.

In the sequel, $\underline{s}$ is no longer the whole stream but only a part of it. Precisely, we consider that $\underline{s}$ is an n-bit binary sequence consisting of an integer number of codewords. Also, assume that n is an integer multiple of the number of bits per constellation symbol $B = \log_2(M_c)$ where M_c is the size of the constellation. In this case, $\underline{s}$ is mapped onto a sequence of n/B constellation symbols :

$$\begin{aligned} \underline{s} &= (s_1, s_2, \ldots, s_n) \in GF(2)^n \\ \underline{x} &= (x_1, x_2, \ldots, x_{n/B}) \in \mathcal{A}^{n/B} \end{aligned} \tag{3.1}$$

where $\mathcal{A}$ is the set of the symbols forming the constellation.
Similarly,

$$\begin{aligned} \underline{\tilde{s}} &= (\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_n) \in GF(2)^n \\ \underline{\tilde{x}} &= (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_{n/B}) \in \mathcal{A}^{n/B} \end{aligned} \tag{3.2}$$

For simplicity, we assume that the channel corrupts the symbol sequence with a complex Additive White Gaussian Noise of variance σ^2

$$\underline{r} = \underline{x} + \underline{\nu} \tag{3.3}$$

with

$$\underline{\nu} = (\nu_1, \nu_2, \ldots, \nu_{n/B}) \in \mathbb{C}^{n/B}$$
$$\underline{r} = (r_1, r_2, \ldots, r_{n/B}) \in \mathbb{C}^{n/B}$$

Equation (3.3) is equivalent to

$$r_i = x_i + \nu_i, \qquad \forall i \in \{1, \ldots, n/B\} \tag{3.4}$$

with

$$\nu_i \sim \mathcal{N}\left(0, \begin{bmatrix} \sigma^2/2 & 0 \\ 0 & \sigma^2/2 \end{bmatrix}\right), \qquad i \in \{1, \ldots, n/B\} \tag{3.5}$$

When the source encoding is not perfect (which is often the case in practice for multimedia systems), there are (small) redundancies left by the encoder. Also, in the case of video, the binary stream is structured in a very specific way, which introduces additional redundancies. This structure is characterized by a set of constraints that the output stream meets. In this case, every n-bit binary sequence $\underline{s}$ meets a set of constraints.

Also, in practical situations, the stream output by the source encoder has to be transmitted as packets formed by appending a Cyclic Redundancy Check (CRC) to a binary sequence $\underline{s}$ (see Fig. 3.2). The CRC is used for error detection at the receiver and consists of a set of n_{CRC} parity or redundancy bits computed based on the bits $s_1, s_2, \ldots, s_n$. Below, we assume that n_{CRC} is also an integer multiple of the number of bits per constellation symbol B.

There are two ways to improve the quality of the above communication system : The first is by exploiting (at the decoder) the redundancies left by the source encoder and by the packetization. This is known as *Robust Decoding*. The second is by making use of retransmissions of erroneous packets, which is known as *Automatic Repeat reQuest (ARQ)* (109)(110).

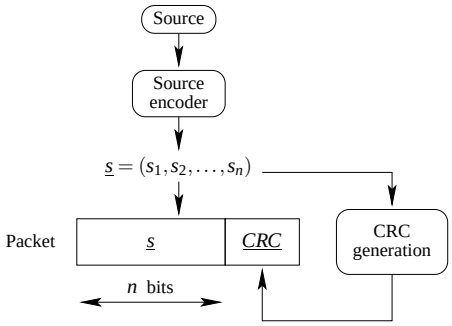

Figure 3.2: Packetization of the output bitstream of the source encoder.

3.2.1 Robust decoding-based (forward) schemes

These schemes use the redundancies left by the encoder to correct transmission errors in a channel decoding manner. Due to the structure of the bitstream output by the encoder,

for a size of n bits, there are $K << 2^n$ valid binary sequences. The robust source decoder chooses one of these valid sequences as the decoded sequence $\underline{\hat{s}}$ (see Fig. 3.3) (66)(70). Obviously, the actual sequence $\underline{s}$ is part of this set (of valid sequences).

In the sequel, $APP(\cdot)$ is used to denote A Posteriori Probabilities while $\pi(\cdot)$ is used to denote a priori probabilities. For example, the a priori probability of random event A is

$$P(A) = \pi(A)$$

and the A Posteriori Probability of event A is

$$P(A|\underline{r}) = APP(A)$$

The optimal robust decoder uses a decision rule such that the probability of a correct decision is maximized, hence the probability of error is minimized. This decision rule is based on the posterior probabilities defined as

$$P(\text{sequence } \underline{s}_j \text{ was transmitted}|\underline{r}) = P(\underline{s} = \underline{s}_j|\underline{r}) = APP(\underline{s} = \underline{s}_j), \quad j = \{1, 2, \ldots, K\}$$

The optimum decoder decides in favor of the sequence corresponding to the maximum of the set of posterior probabilities $\{APP(\underline{s} = \underline{s}_j), \forall j \in \{1, \ldots, K\}$, that is the Maximum A posteriori Probability (MAP) criterion.

The posterior probabilities may be expressed as

$$\begin{aligned} APP(\underline{s} = \underline{s}_j) &= P(\underline{s} = \underline{s}_j|\underline{r}) \\ &= \frac{p(\underline{s} = \underline{s}_j, \underline{r})}{p(\underline{r})} \\ &= \frac{p(\underline{r}|\underline{s} = \underline{s}_j)P(\underline{s} = \underline{s}_j)}{p(\underline{r})} \end{aligned}$$

$P(\underline{s} = \underline{s}_j)$ is the a priori probability that sequence $\underline{s}_j$ is sent and will be denoted by $\pi(\underline{s} = \underline{s}_j)$. $p(\underline{r}|\underline{s} = \underline{s}_j)$ is the likelihood of sequence $\underline{s}_j$, which can be expressed as

$$p(\underline{r}|\underline{s} = \underline{s}_j) = \frac{1}{(\pi\sigma^2)^{n/2}} e^{-\frac{||\underline{r} - \underline{x}_j||^2}{\sigma^2}} \tag{3.6}$$

where $\underline{x}_j$ is the symbol vector sequence $\underline{s}_j$ is mapped onto.

Since we have no a priori on the valid sequences ($\pi(\underline{s} = \underline{s}_j) = 1/K$ for $j \in \{1, \ldots, K\}$), we have

$$\begin{aligned} APP(\underline{s} = \underline{s}_j) &= \frac{1}{Kp(\underline{r})} p(\underline{r}|\underline{s} = \underline{s}_j) \\ &\propto p(\underline{r}|\underline{s} = \underline{s}_j) \end{aligned} \tag{3.7}$$

The decision rule based on finding the sequence that maximizes $APP(\underline{s} = \underline{s}_j)$ is equivalent to finding the sequence that maximizes $p(\underline{r}|\underline{s} = \underline{s}_j)$.

$$\underline{\hat{s}} = \arg \max_{\underline{s}_j, j \in \{1, \ldots, K\}} p(\underline{r}|\underline{s} = \underline{s}_j) \tag{3.8}$$

Given that the search is performed among all sequences compatible with the encoder syntax has two useful effects : (i) all estimated sequences (even those in error) can be understood by the source decoder, which will not "crash", (ii) some "error correction effect" is introduced, since $\hat{\underline{s}}$ is more often equal to $\underline{s}$ than $\tilde{\underline{s}}$ is (63)(65)(66)

$$P(\hat{\underline{s}} = \underline{s}) > P(\tilde{\underline{s}} = \underline{s}). \tag{3.9}$$

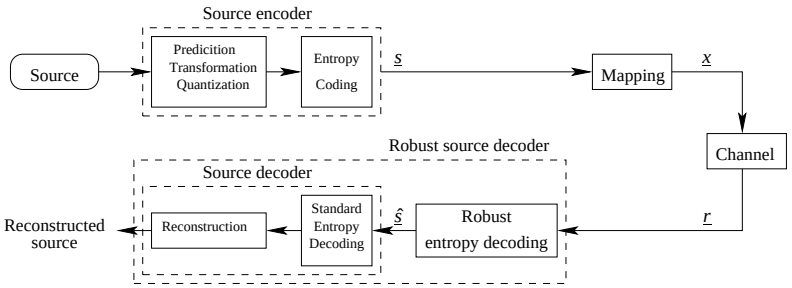

Figure 3.3: Block diagram of a multimedia transmission system using robust source decoding.

3.2.2 ARQ schemes

ARQ schemes are feedback systems based on the retransmission of erroneous packets. When a packet is received, the CRC is checked and if errors are detected, the receiver discards the packet and asks for its retransmission (by sending a Negative Acknowledgement through the feedback channel), otherwise, the packet is accepted and the receiver sends an Acknowledgement to the transmitter. As a result, the communication is made reliable at the cost of an increase in the number of packets which are transmitted.

Fig. 3.4 shows the block diagram of a system using ARQ. Note that at the receiver, the CRC is removed only if no errors are detected after it is checked, otherwise the whole packet is dropped and the source decoder is not provided with data).

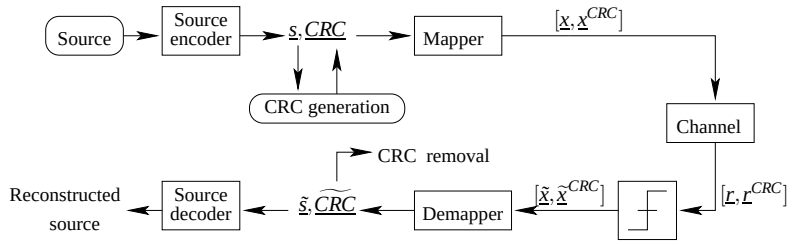

Figure 3.4: Block diagram of a multimedia transmission system using ARQ.

3.3 Proposed retransmission scheme

As explained in section 3.2, robust decoding systems are based on the error correction tool while ARQ systems are based on the retransmission tool. The proposed system uses

both : When a packet is received, the CRC is checked, if it is detected in error, instead of discarding it as in the case of ARQ systems, the receiver first tries to correct the errors by performing a robust decoding on the received symbols. Then, the receiver has to decide between two cases (i) it accepts the packet, in which case the likeliest sequence is used for reconstructing the original image or (ii) it discards the packet and asks for its retransmission. The whole process relies on the decision whether the sequence estimated by the robust decoder is reliable enough or not. This method is denoted below as "Soft ARQ"-SARQ-.

For the purpose of precisely formalizing this process, define the following hypotheses, associated to the two possible decisions :

H_1 : $\underline{\hat{s}}$ is the sequence that was actually sent.

H_0 : $\underline{\hat{s}}$ is not the sequence that was actually sent.

The receiver thus has to decide in favor of one of these two hypotheses. This is formally a hypothesis test that can be characterized by the probability of False alarm P_F and the probability of Detection P_D :

$$P_F = P(\text{Choose } H_1 | H_0 \text{ is true})$$
$$P_D = P(\text{Choose } H_1 | H_1 \text{ is true})$$

P_F represents the probability to accept a packet containing some wrong sequence. P_D represents the probability to accept a packet which contains the right sequence. Conversely, $(1 - P_D)$ represents the probability that a packet containing the right sequence is rejected.

P_F is a measure of errors (hence of quality). $(1 - P_D)$ is a measure of useless retransmissions (hence P_D is a measure of throughput).

By minimizing P_F, there will be fewer errors in the accepted packets, which enhances the quality of the received video. On the other hand, maximizing P_D enhances the throughput. Our aim is to try to enhance the throughput while trying to keep the same level of quality. In order to do that, we have to maximize P_D for a given P_F. This criterion is known as the Neyman-Pearson criterion, which results in the *Likelihood Ratio Test* (LRT) defined by :

$$\Lambda(\underline{r}) \underset{H_0}{\overset{H_1}{\gtrless}} \lambda \tag{3.10}$$

Where

$$\Lambda(\underline{r}) = \frac{p(\underline{r}|H_1)}{p(\underline{r}|H_0)} \tag{3.11}$$

is the Likelihood Ratio (LR), with

$p(\underline{r}|H_1)$: Likelihood of H_1 (pdf of $\underline{r}$ when H_1 is true).

$p(\underline{r}|H_0)$: Likelihood of H_0 (pdf of $\underline{r}$ when H_0 is true).

Assume that the robust decoder is a sequential decoder that is modelled by a process which ranks the sequences $\{\underline{s}_1, \ldots, \underline{s}_K\}$ from the likeliest to the least likely. Let us denote by $\underline{a}_j$ the sequence ranked at the jth position. Obviously, $\underline{a}_1$ is $\underline{\hat{s}}$. The set of sequences $\{\underline{a}_1, \ldots, \underline{a}_K\}$ is equal to the set $\{\underline{s}_1, \ldots, \underline{s}_K\}$ but the the K-tuple $(\underline{a}_1, \ldots, \underline{a}_K)$ changes with $\underline{r}$ while the K-tuple $(\underline{s}_1, \ldots, \underline{s}_K)$ is fixed and independent of $\underline{r}$.

The likelihood of H_1 is the conditional pdf of $\underline{r}$ when $\underline{\hat{s}}$ is the sequence that was actually sent. Since $\underline{\hat{s}} = \underline{a}_1$, then, the likelihood of H_1 is that of sequence $\underline{a}_1$:

$$p(\underline{r}|H_1) = p(\underline{r}|\underline{s} = \underline{a}_1) = \frac{1}{(\pi\sigma^2)^{n/2}} e^{-\frac{||\underline{r}-\underline{y}_1||^2}{\sigma^2}} \tag{3.12}$$

where $\underline{y}_1$ is the symbol vector sequence $\underline{a}_1$ is mapped onto.
The likelihood of H_0 can be expressed as

$$p(\underline{r}|H_0) = \frac{p(\underline{r}, H_0)}{P(H_0)} \tag{3.13}$$

$p(\underline{r}, H_0)$ can be expressed as

$$\begin{aligned} p(\underline{r}, H_0) &= p(\underline{r}, \underline{s} \neq \underline{a}_1) \\ &= p(\underline{r}, \underline{s} \in \{\underline{a}_2, \ldots, \underline{a}_K\}) \\ &= \sum_{j=2}^{K} p(\underline{r}|\underline{s} = \underline{a}_j) P(\underline{s} = \underline{a}_j) \\ &= \sum_{j=2}^{K} p(\underline{r}|\underline{s} = \underline{a}_j) \pi(\underline{s} = \underline{a}_j) \end{aligned} \tag{3.14}$$

$P(H_0)$ is the a priori probability of H_0 and can be written as

$$\begin{aligned} P(H_0) &= P(\underline{s} \neq \underline{a}_1) \\ &= P(\underline{s} \in \{\underline{a}_2, \ldots, \underline{a}_K\}) \\ &= \sum_{j=2}^{K} P(\underline{s} = \underline{a}_j) \\ &= \sum_{j=2}^{K} \pi(\underline{s} = \underline{a}_j) \end{aligned} \tag{3.15}$$

where $\pi(\underline{s} = \underline{a}_j)$ is the *a priori* probability that $\underline{s}$ is equal to $\underline{a}_j$, i.e. the probability that $\underline{s}$ ends up at the jth position. These probabilities are discussed below, in the same section.

Hence, $p(\underline{r}|H_0)$ reads

$$p(\underline{r}|H_0) = \frac{\sum_{j=2}^{K} p(\underline{r}|\underline{s} = \underline{a}_j)\pi(\underline{s} = \underline{a}_j)}{\sum_{j=2}^{K} \pi(\underline{s} = \underline{a}_j)} \tag{3.16}$$

Substituting (3.12) and (3.16) into (3.11) we obtain the following expression of the Likelihood Ratio

$$\Lambda(\underline{r}) = \left(\sum_{j=2}^{K} \pi(\underline{s} = \underline{a}_j) \right) \times \frac{p(\underline{r}|\underline{s} = \underline{a}_1)}{\sum_{j=2}^{K} p(\underline{r}|\underline{s} = \underline{a}_j)\pi(\underline{s} = \underline{a}_j)} \tag{3.17}$$

Now consider the set of a priori probabilities $\{\pi(\underline{s} = \underline{a}_j),\ j \in \{1, \ldots, K\}\}$ but before trying to express these probabilities in closed form, it is necessary to explain the difference between $\pi(\underline{s} = \underline{s}_j)$ and $\pi(\underline{s} = \underline{a}_j)$ $(\forall j \in \{1, \ldots, K\})$. Recall that $\underline{s}_j$ is a binary sequence that is fixed and independent of $\underline{r}$ while sequence $\underline{a}_j$ is a sequence that changes with $\underline{r}$.

$$\underline{s}_1 = [10], \quad \underline{s}_2 = [01] \quad \text{and} \quad \underline{s}_3 = [11] \tag{3.18}$$

In this example, $n = 2$ and $K = 3$. $\underline{a}_j$ $(\forall\{1, 2, 3\})$ may be equal to $\underline{s}_1$ for a given realization of $\underline{r}$, to $\underline{s}_2$ for another realization of $\underline{r}$ and to $\underline{s}_3$ for yet another realization of $\underline{r}$.

Hence, $\pi(\underline{s} = \underline{s}_j)$ is the a priori probability that sequence $\underline{s}_j$ is sent whereas $\pi(\underline{s} = \underline{a}_j)$ is the a priori probability that $\underline{s}$ is equal to the sequence ranked j^{th}, i.e. the a priori probability that sequence $\underline{s}$ ends up at the j^{th} position after robust decoding of $\underline{s}$ is performed based on $\underline{r}$.

Now, combining (3.7) and (3.17) we can write $\Lambda(\underline{r})$ as

$$\Lambda(\underline{r}) = \left(\sum_{j=2}^{K} \pi(\underline{s} = \underline{a}_j)\right) \times \frac{APP(\underline{s} = \underline{a}_1)}{\sum_{j=2}^{K} APP(\underline{s} = \underline{a}_j)\pi(\underline{s} = \underline{a}_j)} \tag{3.19}$$

Hence, $\Lambda(\underline{r})$ combines the a priori information based on the modelling of the ranking mechanism and the a posteriori information provided by the observation $\underline{r}$ of the transmitted symbols.

From (3.19), we can see that the lowest possible value of $\Lambda(\underline{r})$ is 1 since in the worst case (total uncertainty) we have $APP(\underline{s} = \underline{a}_j) = 1/K, \forall j \in \{1, \ldots, K\}$ and in this case $\Lambda(\underline{r})$ is equal to 1. In the other situations, the $APP(\underline{s} = \underline{a}_j)$ is different from the APPs of the sequences ranked between the second and the last position, and $\Lambda(\underline{r})$ is greater than one. Actually, the larger the difference between $APP(\underline{s} = \underline{a}_1)$ and the APPs of the other sequences, the larger is $\Lambda(\underline{r})$ and the more reliable is the decoding. This confirms that the LR is indeed a measure of the reliability of the decoding.

For the sake of simplicity, let us now consider that there is no structure in $\underline{s}$, i.e. that $K = 2^n$ and that we use a BPSK modulation.

Some likely configuration of the ranking $(\underline{a}_1, \ldots, \underline{a}_K)$ is described as follows :

Sequence $\underline{a}_1$ is first followed by the sequences that differ in 1 position with it, then by the sequences that differ in 2 positions, then by the sequences that differ in 3 positions ...etc. There are $\binom{n}{1} = n$ sequences that differ in 1 position with $\underline{a}_1$. With this configuration, the probability that $\underline{s}$ ends up at one of the positions in the set $\{2, 3, \ldots, n+1\}$ is the same and equal to $\varepsilon(1-\varepsilon)^{n-1}$ where ε is the probability of a bit error given by

$$\varepsilon = Q\left(\sqrt{\frac{2E_b}{\sigma^2}}\right) \tag{3.20}$$

where E_b is the energy per transmitted bit (a "0" is mapped onto symbol $-\sqrt{E_b}$ and a "1" is mapped onto symbol $+\sqrt{E_b}$).

Similarly, the probability that $\underline{s}$ ends up at one of the positions in the set $\{n+2, n+3, \ldots, n+\binom{n}{2}+1\}$ is the same and equal to $\varepsilon^2(1-\varepsilon)^{n-2}$. This process may be continued up to the last position ($\underline{a}_K$ is the complement of sequence $\underline{a}_1$).

Thus, we obtain the following expressions of the a prioris $\pi(\underline{s} = \underline{b}_j)$

$$\pi(\underline{s} = \underline{a}_j) = \varepsilon^l (1-\varepsilon)^{n-l}, \qquad \forall l \in \{0, \ldots, n\}, \forall j \in \left\{ \sum_{m=0}^{l-1} \binom{n}{m} + 1, \ldots, \sum_{m=0}^{l} \binom{n}{m} \right\} \quad (3.21)$$

Note that the a priori probabilities hence obtained are not exact because they are based on the most likely configuration of the ranking, but it may be verified that they represent good approximations to the exact values. Also, to derive these quantities, we first assumed that $K = 2^n$ while in our case we have $K < 2^n$, which introduces an additional inaccuracy in the formulas above (in which we have to add a normalization factor so that the a priori probabilities of the valid sequences add up to 1).

Simulation results Preliminary simulations (with BPSK-modulated binary sequences) were run to check the behaviour of the test. Sequential M-algorithm-based decoding was used, meaning that a bit-by-bit decoding was performed (in n steps) and only the M likeliest sequences were kept (along with their corresponding likelihoods) at a given step of the decoding. The results are shown in figures 3.5, 3.6 and 3.7 for sequences of $n = 6$ bits, $n = 8$ bits and $n = 10$ bits, respectively. As can be seen, there is a steep fall of the probability of false alarm as the test's threshold increases, confirming thus the expected behaviour.

Note, however, that the fall of the probability of false alarm becomes less steep as the length of the sequence increases. This is due to the fact that the metric the hypothesis testing is based on uses the information of all the sequences (from the likeliest to the least likely) and not just the M likeliest sequences. On the other hand, the number of sequences increases exponentially with the length of the sequence, making thus the number of likelihoods available for the computation of the metric (at the end of the decoding) a very small percentage of the overall number of likelihoods necessary for the compuation of the metric, ideally. The fact that the sequences kept at each step are the *likeliest* does limit the negative impact of the absence of the sequences that are eliminated during the decoding process (the likelihoods of which are *more or less* negligible), but only partially.

Also note that when the SNR increases, higher thresholds are necessary to achieve the same type of behaviour, which makes perfect sense since when the SNR increases, the metric of the test becomes better (increases) and therefore, in order to get the same acceptance/rejection rate the threshold of the test needs to be adapted accordingly (i.e. increased in the same proportions).

3.4 Application to the transmission of H264 video data

So far, we have assumed that each packet is composed of a data part and a CRC. The data part consists of an integer number of codewords generated by the source encoder, while the CRC part consists of a set of parity bits that are computed based on the data part of the packet. At the receiver, the presence of errors in the received packet $[\underline{\tilde{x}}, \underline{\tilde{x}}^{CRC}]$ can be detected by checking the CRC.

Let us now consider the case of an H264/AVC video encoder and see how the SARQ (LRT-based ARQ) scheme can be applied to the transmission of H264 encoded data.

The H264/AVC encoder consists of a Video Coding Layer (VCL), which efficiently represents the video content and a Network Abstraction Layer (NAL) which formats the

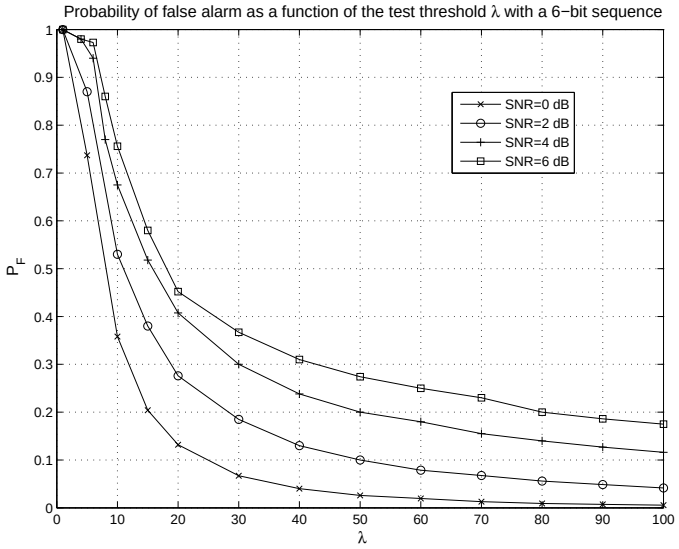

Figure 3.5: Probability of false alarm as a function of test threshold λ for $n = 6$ and $M = 64$.

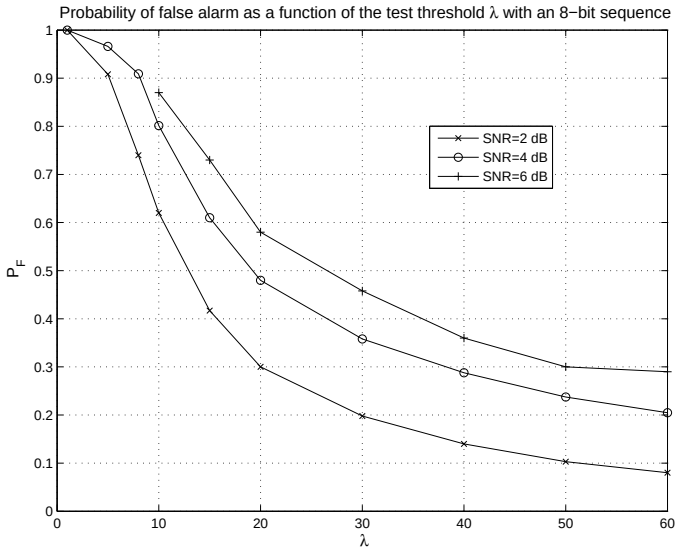

Figure 3.6: Probability of false alarm as a function of test threshold λ for $n = 8$ and $M = 64$.

VCL representation and provides header information. Prior to encoding, each image is partitioned into slices that contain an integer number of 16 pixels×16 pixels Macro Blocks (MB).

The Video Coding Layer The samples of each MB are first predicted, yielding (prediction) residuals. These residuals are then transformed and quantized. In H264 AVC, the (16 pixels × 16 pixels) MB contains 4 8 pixels × 8 pixels blocks. In case the quantized

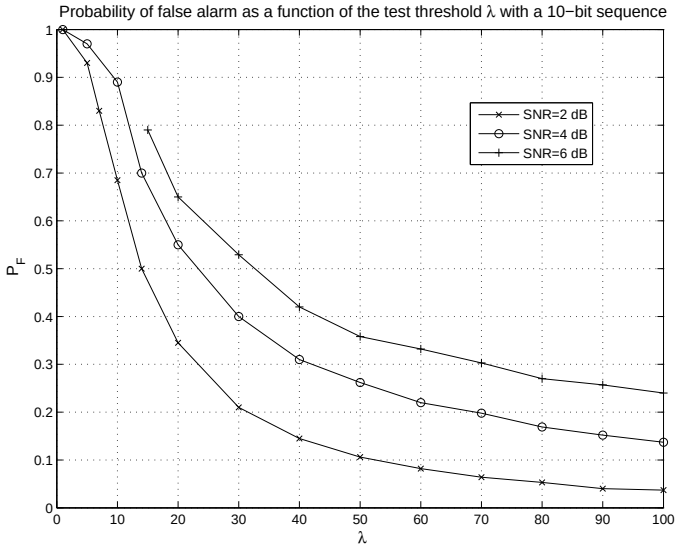

Figure 3.7: Probability of false alarm as a function of test threshold λ for $n = 10$ and $M = 32$.

transform coefficients of an 8 pixels $\times$ 8 pixels block are all zero, the (8 pixels $\times$ 8 pixels) block is not coded. Otherwise, the 8 pixels $\times$ 8 pixels block is further decomposed into 4 4 pixels $\times$ 4 pixels blocks, each 4 pixels $\times$ 4 pixels block is mapped onto a binary sequence consisting of an integer number of codewords. A codeword is selected from a Variable Length Code (VLC) table and the sequence the 4×4 block was mapped onto is called a CAVLC sequence.

Hence, a MB is mapped onto 0, 4, 8 12, or 16 CAVLC sequences depending on the number of all zero 8 $\times$ 8 (quantized transform coefficients) blocks.

Each CAVLC sequence satisfies a set of properties known as the semantic and syntaxic properties of the code which reduce the number of valid sequences (see (66)(69)(70) for a more precise description). If we consider $\underline{s}$ as a CAVLC sequence, we can perform robust decoding on the corresponding received symbol vector $\underline{r}$.

The Network Abstraction Layer This layer encapsulates the coded slice data in a video packet. Fig. 3.8 illustrates how a 2-MB coded slice is packetized in a NAL Unit (NALU).

MB_1 is coded into sequence $\underline{s}^{(1)}$ which represents MB1 data. Similarly, MB_2 is coded into sequence $\underline{s}^{(2)}$ which represents MB_2 data. The MB header contains parameters like the MB type and the prediction mode (for Intra prediction) or the motion vectors (for Inter prediction). The slice header contains parameters like the slice type and the coded picture the slice belongs to. Finally, the main information contained in the NALU header is the NALU type.

Note that the MB header contains a 4-bit parameter called *coded-block-pattern-luma*, each bit of which corresponds to an 8×8 block of the MB. When the 8×8 block is not coded (all zero), the corresponding bit is set to 0. On the other hand, when the 8×8 is coded (into 4 CAVLC sequences), the bit is set to 1. The number of CAVLC sequences in the MB data field is $4 \times W_H(\textit{coded-block-pattern-luma})$ where W_H denotes the Hamming

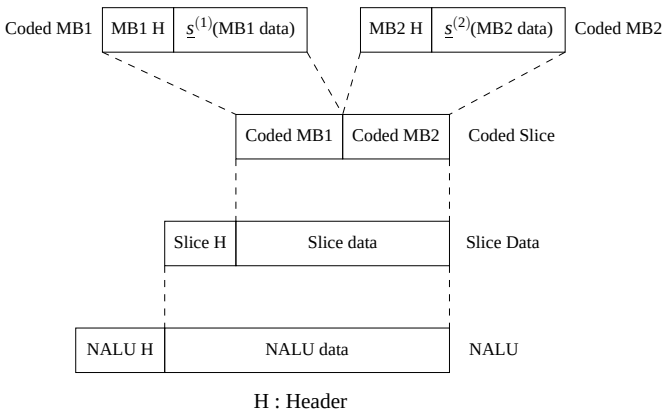

Figure 3.8: Encapsulation of a coded slice data into a video packet (NALU).

weight.

Fig. 3.9 shows the different fields contained in a NALU. More generally, let N_{MB} be the number of coded MBs that are encapsulated in the packet and L_m the number of CAVLC sequences MB_m is mapped onto (i.e. the number of CAVLC sequences in MB_m data field). Note that $L_m = 4 \times W_H(\textit{coded-block-pattern-luma}(m))$ where $\textit{coded-block-pattern-luma}(m)$ is the the *coded-block-pattern-luma* parameter of MB_m.

If MB_m is coded onto the binary sequence $\underline{s}^{(m)}$, then

$$\underline{s}^{(m)} = \left[\underline{s}_1^{(m)}, \underline{s}_2^{(m)}, \ldots, \underline{s}_{L_m}^{(m)}\right] \tag{3.22}$$

$\underline{s}_l^{(m)}$ is the l^{th} CAVLC sequence of the m^{th} MB encapsulated in the NALU. $\underline{s}_l^{(m)}$ is then CAVLC sequence number $l + \sum_{k=1}^{m-1} L_k$ in the packet.

Now, let

$$\underline{s}_l^{(m)} = \underline{s}_i \qquad \text{with} \qquad i = l + \sum_{k=1}^{m-1} L_k$$

The NALU contains $N = \sum_{m=1}^{N_{MB}} L_m$ CAVLC sequences $\underline{s}_1, \underline{s}_2, \ldots, \underline{s}_N$ on which a robust decoding may be performed and a LR may be computed.

In order to apply the SARQ scheme, we have to generalize the test defined in section 3.3 for 1 sequence to the case of several sequences. The most straightforward approach is to perform the robust decoding of each CAVLC sequence and make as many tests as there are sequences in the NALU. If all the tests are positive (the LR is greater than the fixed threshold), the decision is made in favor of H_1.

The test is thus defined by

$$\min\left\{\Lambda(\underline{r}_i), \forall i \in \{1, \ldots, N\}\right\} \underset{H_0}{\overset{H_1}{\gtrless}} \lambda \tag{3.23}$$

where $\underline{r}_i$ is the field of received symbols corresponding to the field of sequence $\underline{s}_i$ and $\Lambda(\underline{r}_i)$ is the LR that may be computed after robust decoding of sequence $\underline{s}_i$ using $\underline{r}_i$.

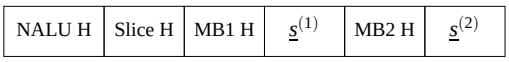

Figure 3.9: Fields of the NALU.

3.5 H264 video-based simulation results

In its extended profile of H264/AVC (34), an error resilience mode is provided, which classifies the slice data according to their impact on the video quality. Three partitions are defined

- Partition A contains the slice header and the MB headers.
- Partition B consists of texture coefficients of the INTRA coded blocks.
- Partition C regroups the texture coefficients of the INTER coded blocks.

After encoding, the coded slice is partitioned and each partition of the coded slice is encapsulated in a NALU (Fig. 3.10).

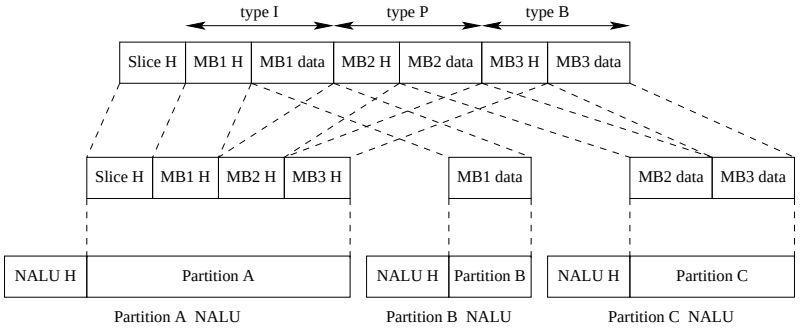

Figure 3.10: Encapsulation of data partitioned slices in video packets.

Simulations were run using data partitioned slices of *Foreman.cif* with a one slice/image configuration. In order to keep the simulations times within reasonable limits, only simulations with the first image (I) and the first 3 images (IPP) of *Foreman.cif* were performed. A 4-byte CRC was appended to each NALU prior to modulation. Packets were BPSK-modulated prior to transmission. Partition A NALUs were considered heavily protected and correctly received. On the other hand, the fixed-size packets associated to the B and C partitions were transmitted on a noisy channel and corrupted by transmission errors. In this scenario, noise and errors affect only the data part while the headers are received without errors. This choice was motivated by the fact that errors in the headers have dramatic effects on the decoding of the video, and the fact that robust decoding has an error-correcting effect only on the data.

Table 3.1: $PSNR$ and average number of transmissions $\overline{N}_{SARQ}$ of the SARQ-transmitted first image (I) of *Foreman.cif* scheme for a packet size of 500 bits at an SNR of 9 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	27.5	29	32.2	35.5	38.5	41.1	41.7	41.95	42
$\overline{N}_{SARQ}$	1	1.15	1.53	1.93	2.37	2.93	3.21	3.39	3.4

Table 3.2: $PSNR$ and average number of transmissions $\overline{N}_{SARQ}$ of the SARQ-transmitted first image (I) of *Foreman.cif* for a packet size of 1500 bits at an SNR of 9.5 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	31.4	32.4	34.8	37.3	39.5	41.6	41.9	42	42
$\overline{N}_{SARQ}$	1	1.28	2.29	3.67	5.4	7.5	8.08	8.45	8.5

Special thanks to Cédric Marin who performed the simulations and provided us with the results that are presented below.

The results of the simulations based on the transmission of the first image (I) of *Foreman.cif* (with a nominal quality of 42 dB) are summarized in tables 3.1, 3.2 and 3.3 for packet sizes of 500 bits, 1500 bits and 5000 bits and respective SNRs of 9, 9.5 and 10.5 dB.

Since $\Lambda(\underline{r})$ is a random variable the minimum value of which is 1, the LR test is always in favor of H_1 when $\lambda = 1$ and all packets are accepted at the first attempt. Hence $\lambda = 1$ corresponds to the case of the forward robust decoding scheme, which is characterized by a $PSNR$ that is more or less low (depending on the SNR and the packet size): 27.5 dB for a packet size of 500 bits at an SNR of 9 dB, 31.4 dB for a packet size of 1500 bits at an SNR of 9.5 dB and 37.4 dB for a packet size of 5000 bits at an SNR of 10.5 dB. This $PSNR$ is denoted below as the starting $PSNR$ and corresponds to the $PSNR$ obtained with the (forward) robust decoding scheme. When the test threshold λ is increased, the quality is enhanced at the cost of an increased average number of transmissions. For large values of λ, the PSNR gets close to the nominal quality (42 dB in our case) obtained with the (CRC-based) ARQ scheme. The nominal quality is reached for $\lambda = \infty$. In this case, the LR test is always in favor of H_0 and all packets with an invalid CRC are discarded, meaning that a packet is (re)transmistted until it is received correctly. Hence $\lambda = \infty$ corresponds to the case of the (CRC-based) ARQ scheme which is characterized by a nominal quality and a high average number of retransmissions, thus a low throughput. By varying the threshold λ from 1 to infinity one can go from the robust decoding scheme up to the CRC-based scheme (see Figs. 3.11, 3.12 and 3.13).

On the other hand, even though the nominal quality is only reached with $\lambda = \infty$ the results show that quasi-nominal quality can be reached for a finite value of the threshold

Table 3.3: $PSNR$ and average number of transmissions $\overline{N}_{SARQ}$ of the SARQ-transmitted first image (I) of *Foreman.cif* for a packet size of 5000 bits at an SNR of 10.5 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	37.4	38.4	39.4	40.2	40.9	41.7	41.92	41.98	42
$\overline{N}_{SARQ}$	1	1.25	2.03	3.09	4.41	6.42	7.2	7.5	7.63

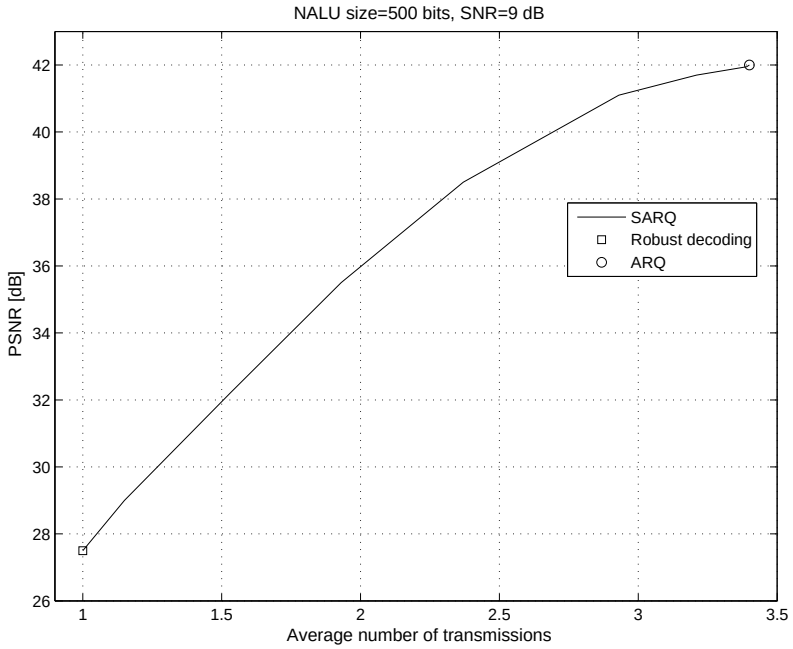

Figure 3.11: PSNR of the SARQ-transmitted first image (I) of *Foreman.cif* as a function of the average number of transmissions for a packet size of 500 bits at an SNR of 9 dB.

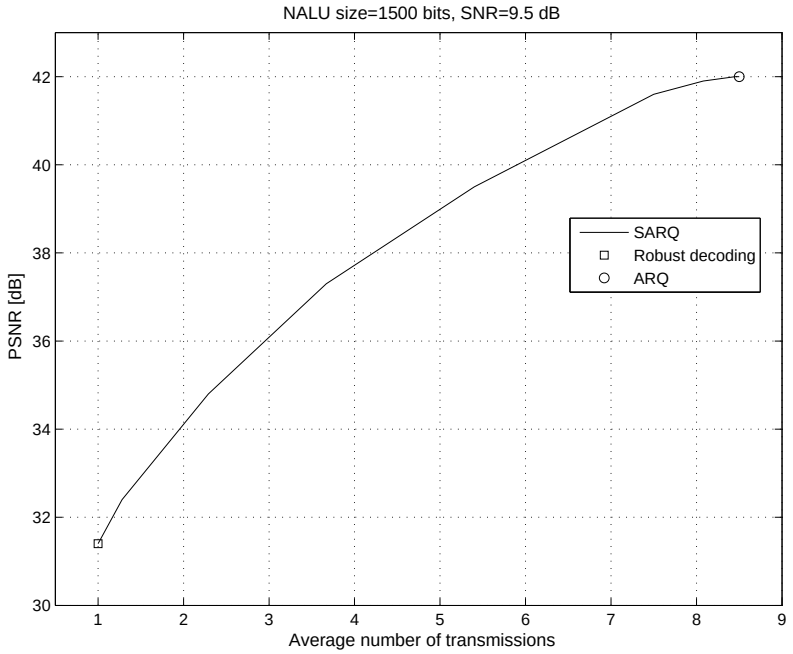

Figure 3.12: PSNR of the SARQ-transmitted first image (I) of *Foreman.cif* as a function of the average number of transmissions for a packet size of 1500 bits at an SNR of 9.5 dB.

λ, that is for a λ on the order of 500 when the starting $PSNR$ is lower than 30 dB and for a λ on the order of 250 when the starting $PSNR$ is greater than 30 dB. At quasi-nominal

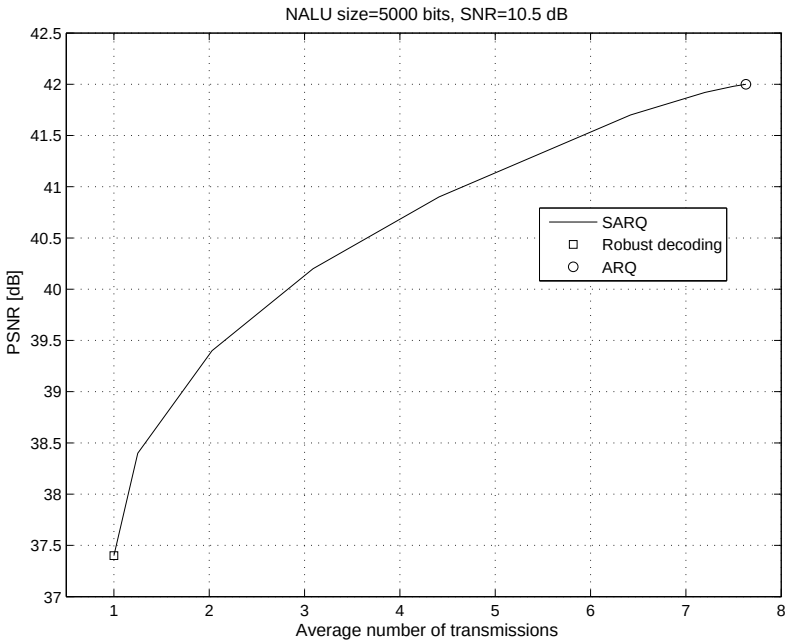

Figure 3.13: PSNR of the SARQ-transmitted first image (I) of *Foreman.cif* as a function of the average number of transmissions for a packet size of 5000 bits at an SNR of 10.5 dB.

quality, the average number of transmissions of a packet is reduced from 3.4 to 3.21 for a starting $PSNR$ of 27.5 dB, from 8.5 to 7.5 for a starting $PSNR$ of 31.4 dB and from 7.63 to 6.42 for a starting $PSNR$ of 37.4 dB, corresponding to respective reduction rates of approximately 6%, 12% and 16%, and respective throughput gains of 6%, 13% and 19%. The differences in terms of throughput gain are due to the fact that the SARQ expected gain is based on the error correction capacity of robust decoding which is higher in the third case than in the second case and higher in the second case than in the first case.

Regardless of the error capacity of the robust decoding, the throughput gain at quasi-nominal quality shows that SARQ offers a better throughput/quality trade-off than the CRC-based ARQ. As a matter of fact, the LRT aims at minimizing the average number of transmissions for a given quality and therefore aims at providing the best throughput/quality trade-off, which is confirmed by these results. Thus, the (CRC-based) ARQ (nominal quality but low throughput) and the (retransmissionless) robust decoding scheme (highest throughput but poor quality) are extreme cases with no possible throughput/quality trade-off. The SARQ scheme, on the other hand, combines hypothesis testing with robust decoding and ARQ processing to offer the possibility to trade throughput for quality (and vice versa) with the *best possible* throughput/quality trade-off.

Also, the throughput can be enhanced further if we consider a subjective approach of quality in the sense that in practice, most of the time, the humain eye notices a difference between two videos starting from a 2-3 dB PSNR difference, which would result in a throughput gain of approximately 275% for a packet size of 5000 bits at an SNR of 10.5 dB, a throughput gain of approximately 57% for a packet size of 1500 bits at an SNR of 9.5 dB and a a throughput gain of at least 43% for a packet size of 500 bits at an SNR of 9 dB.

Table 3.4: $PSNR$ and average number of transmissions $\overline{N}_{SARQ}$ of the SARQ-transmitted first 3 images (IPP) of *Foreman.cif* for a packet size of 500 bits at an SNR of 9 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	27.6	29	32.3	35.6	37.8	39.8	40.4	40.62	40.64
$\overline{N}_{SARQ}$	1	1.16	1.52	1.92	2.33	2.91	3.15	3.34	3.42

Table 3.5: $PSNR$ and average number of transmissions $\overline{N}_{SARQ}$ of the SARQ-transmitted first 3 images (IPP) of *Foreman.cif* for a packet size of 1500 bits at an SNR of 9.5 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	30.8	31.8	34.7	37.2	38.8	40.3	40.6	40.62	40.64
$\overline{N}_{SARQ}$	1	1.31	2.25	3.57	5.35	7.43	8.11	8.43	8.5

Similar results are obtained with the simulations based on the transmission of the first 3 images (IPP) of *Foreman.cif* (see tables 3.4, 3.5 and 3.6 and corresponding figures 3.14, 3.15 and 3.16): Quasi-nominal quality is reached for a λ on the order of 500 when the starting $PSNR$ is lower than 30 dB and for a λ on the order of 250 when the starting $PSNR$ is greater than 30 dB. The approximate average number of transmissions reduction rates at quasi-nominal quality are 8%, 13% and 15%, corresponding to respective throughput gains of 8.5%, 14% and 17% for packet sizes of 500 bits, 1500 bits and 5000 bits at SNRs of 9 dB, 9.5 dB and 10.5 dB, respectively.

Throughput gains of 256%, 58% and 46% can be achieved at 2-3 dB lower than nominal quality (degradation noticeable to the human eye) for packet sizes of 5000 bits, 1500 bits and 500 bits at SNRs of 10.5 dB, 9.5 dB and 9 dB, respectively.

3.6 Practical implementation of robust decoding and SARQ

Equations (3.7) and (3.19) show that robust decoding and SARQ are perfectly applicable in presence of channel coding, provided that the receiver uses a Soft Input Soft Output (SISO) decoder. The SISO decoder computes the APP of each bit of the sequence $APP(s_i = 1) = P(s_i = 1|[\underline{r}, \underline{r}^{CRC}]), \forall i \in \{1, \ldots, n\})$. The APP of sequence $\underline{a}_j$ $APP(\underline{s} = \underline{a}_j)$ can then be computed as the product of the associated bit APPs

$$APP(\underline{s} = \underline{a}_j) = \prod_{i=1}^{n} APP(s_i = a_{j,i}) \tag{3.24}$$

where $a_{j,i}$ is the i^{th} bit of sequence $\underline{a}_j$.

The standard receiver performs hard output channel decoding, which delivers estimated binary packets. The binary packets are then delivered through all the layers of the protocol stack up to the application layer where the source decoder resides.

Table 3.6: $PSNR$ and average number of transmissions $\overline{N}_{SARQ}$ of the SARQ-transmitted first 3 images (IPP) of *Foreman.cif* for a packet size of 5000 bits at an SNR of 10.5 dB.

λ	1	10	25	50	100	250	500	1000	∞
$PSNR$(dB)	36.9	37.2	38	39.4	40	40.45	40.62	40.63	40.64
$\overline{N}_{SARQ}$	1	1.23	1.96	3	4.18	5.96	6.65	6.75	6.98

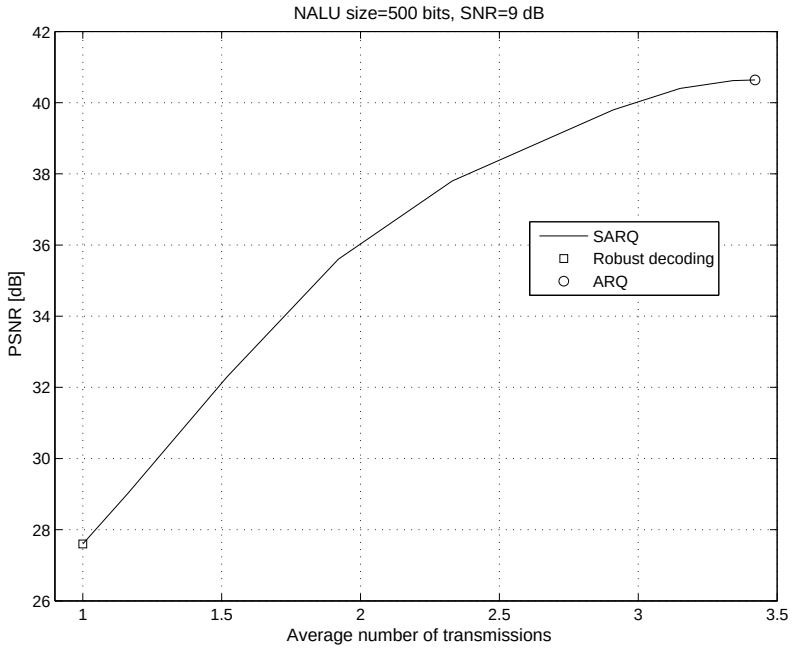

Figure 3.14: PSNR of the SARQ-transmitted first 3 images (IPP) of *Foreman.cif* as a function of the average number of transmissions for a packet size of 500 bits at an SNR of 9 dB.

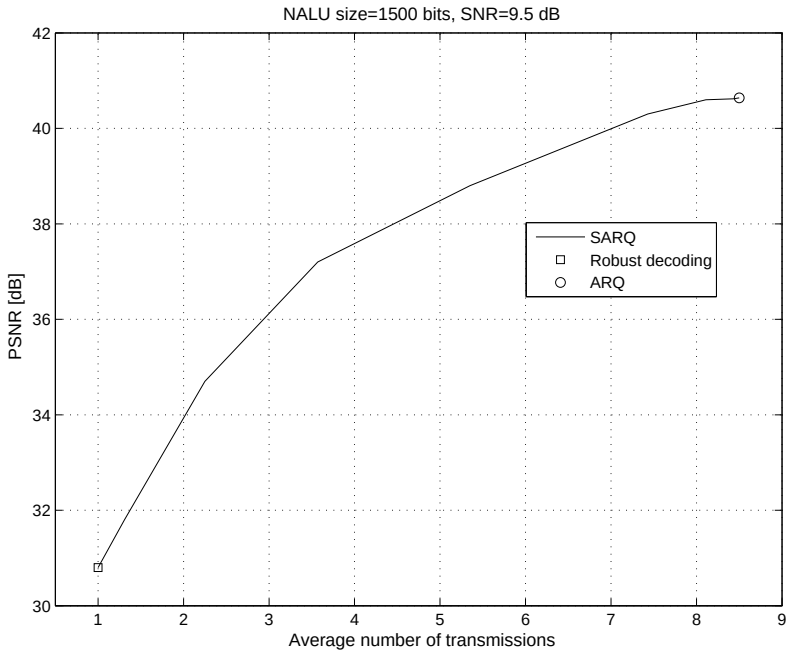

Figure 3.15: PSNR of the SARQ-transmitted first 3 images (IPP) of *Foreman.cif* as a function of the average number of transmissions for a packet size of 1500 bits at an SNR of 9.5 dB.

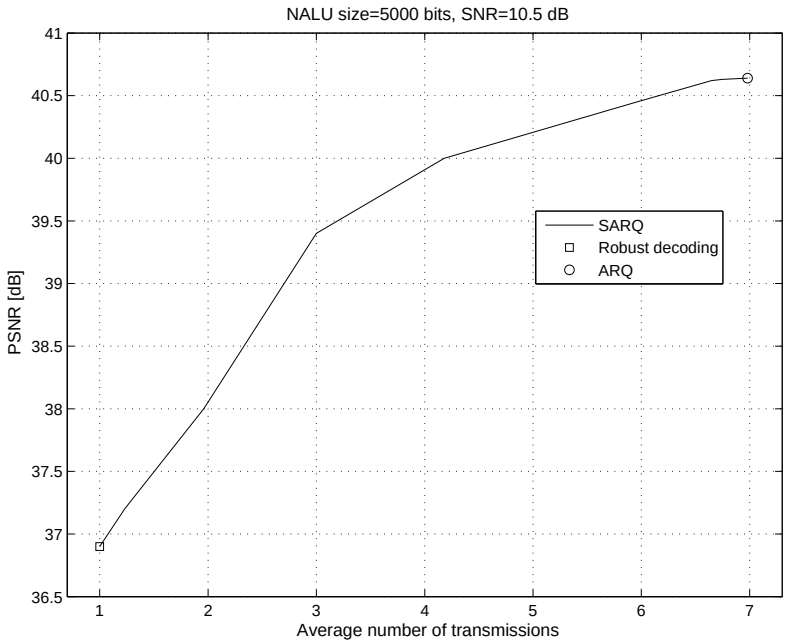

Figure 3.16: PSNR of the SARQ-transmitted first 3 images (IPP) of *Foreman.cif* as a function of the average number of transmissions for a packet size of 5000 bits at an SNR of 10.5 dB.

In order to implement Robust Decoding and SARQ, we need to bring 3 changes to the current standard processing at the receiver :

1. Perform SISO channel decoding instead of hard output decoding.
2. Deliver APPs of video sequences bits to the application layer so that Robust source Decoding can be performed.
3. Perform Robust Decoding (using the APPs) instead of standard source decoding.
4. Deactivate the CRC-based retransmission mechanism.

Assuming that SISO decoding is performed at the physical layer, we discuss below the delivery of the resulting APPs based on the example of 802.16-2004 WiMAX.

Note that in case of SARQ, an hypothesis testing should be performed resulting in a decision as to whether the video packet should be accepted or not, the decision will then be transmitted to the MAC layer where the ARQ mechanism is implemented.

Mathematically, a packet is a vector of bits $\underline{b} = (b_1, \ldots, b_{L(\underline{b})}) \in GF(2)^{L(\underline{b})}$ where $L(\underline{b})$ is the length of vector $\underline{b}$, and an APP packet is a vector of APPs (reals) $APP(\underline{b}) = (APP(b_1), \ldots, APP(b_{L(\underline{b})})) \in \mathbb{R}^{L(\underline{b})}$. APP Packets contain real values that can be represented using the floating point representation on 16 bits for instance. Obviously, the delivery of APPs requires a lot more memory at the receiver than the delivery of bits does. For instance, if a 16 bit-floating-point representation is used, the required memory is 16 times larger. This is the price to pay for the advantages of Robust Decoding and Soft ARQ.

Before considering the delivery of APPs from the physical layer to the application layer at the receiver, we first detail the corresponding operations at the transmitter, as well as the packet format at each layer or sublayer of the protocol stack. For this purpose, we consider H264 video data and the 802.16-2004 WiMAX radio interface. Note that for convenience, the word "packet" is used for all layers, even though this word is more used for the IP layer. For instance, the UDP layer packet is called UDP packet rather than UDP datagram. Also, we assume that the packets are fragmented only at the MAC layer. In case a packet is fragmented at a higher layer, the same method can be applied. We also consider the RTP/UDP/IP stack instead of the TCP/IP stack.

3.6.1 Packet delivery at the transmitter

The application layer Fig. 3.17 and 3.18 show the APL operation and the video packet (NALU) format when Data Partitioning (DP) is not used and when DP is used respectively.

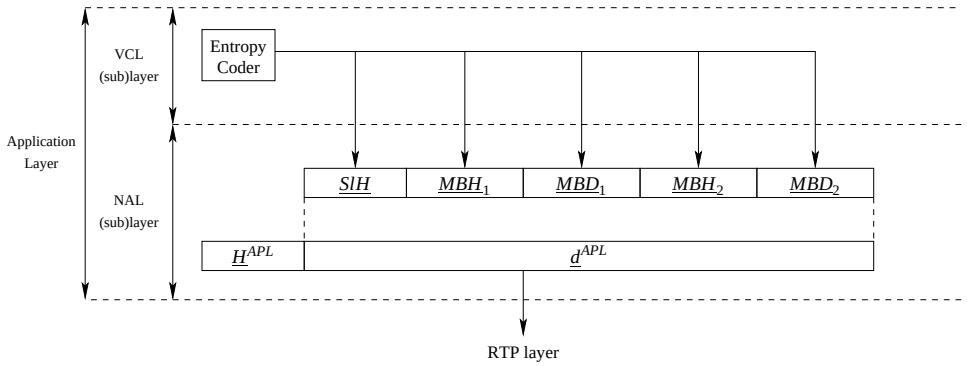

Figure 3.17: Application layer operation (at the transmitter) when Data Partitioning is not used.

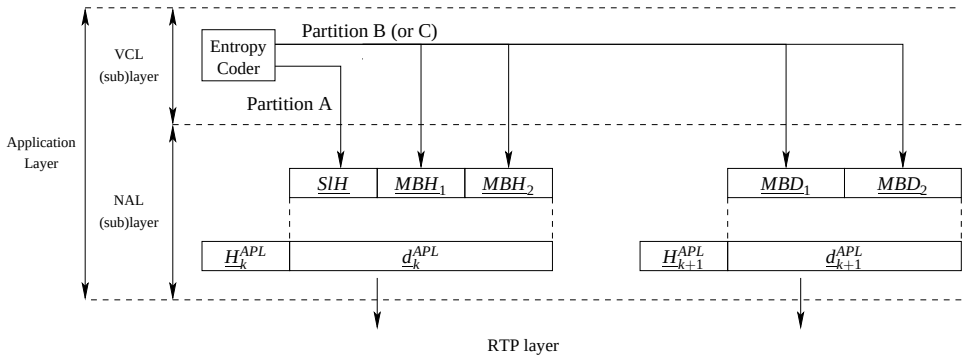

Figure 3.18: Application layer operation (at the transmitter) when Data Partitioning is used.

The following fields are defined :

- $\underline{SlH}$: Slice Header.
- $\underline{MBH}_i$: Header of the "i^{th}" coded MB.

- $\underline{MBD}_i$: Data (residuals) of the i^{th} coded MB.
- $\underline{d}^{APL}$: NALU (application layer packet) data.
- $\underline{H}^{APL}$: NALU (application layer packet) Header.

The RTP layer Fig. 3.19 shows the RTP layer operation and the RTP packet format.

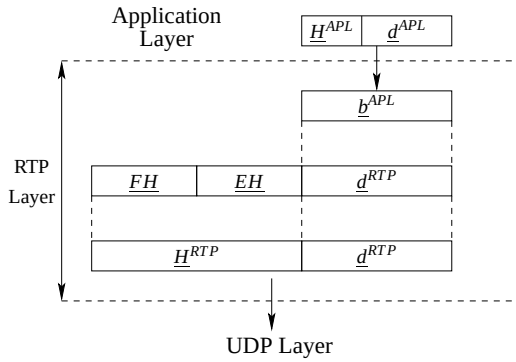

Figure 3.19: RTP layer operation at the transmitter.

Note that the RTP Layer sees the application layer packet as a vector of bits $\underline{b}^{APL}$.
The following fields are defined :

- $\underline{H}^{RTP}$: RTP Header.
- $\underline{d}^{RTP}$: RTP data.

Note that the RTP Header consists of a Fixed Header $\underline{FH}$ and an Extension Header $\underline{EH}$.

The UDP layer Fig. 3.20 shows the UDP layer operation and the UDP packet format.

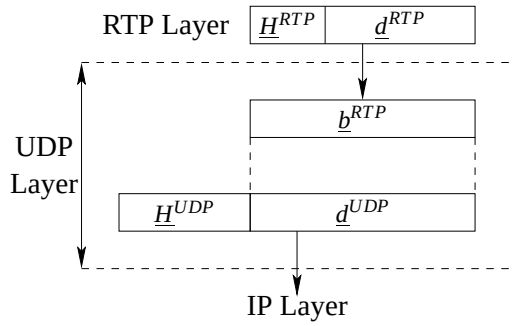

Figure 3.20: UDP layer operation at the transmitter.

Note that the UDP Layer sees the RTP packet as a vector of bits $\underline{b}^{RTP}$.
The following fields are defined

- $\underline{H}^{UDP}$: UDP Header.
- $\underline{d}^{UDP}$: UDP data.

The IP layer Fig. 3.21 shows the IP layer operation and the IP layer packet format.

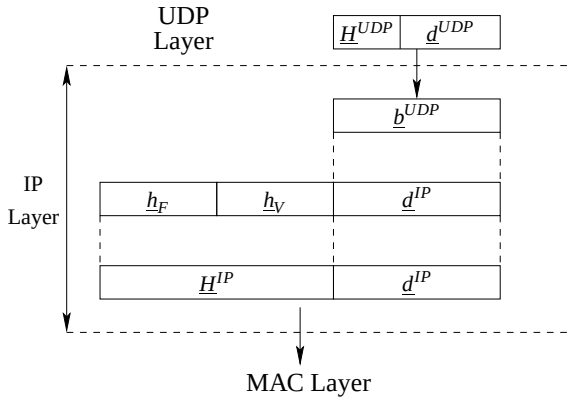

Figure 3.21: IP layer operation at the transmitter.

Note that the IP Layer sees the UDP packet as a vector of bits $\underline{b}^{UDP}$.
The following fields are defined

- $\underline{H}^{IP}$: IP Header.
- $\underline{d}^{IP}$: IP data.

Note that the IP header consists of a Fixed 20-byte field $\underline{h}_F$ and a variable "options" field $\underline{h}_V$.

The MAC layer The MAC Layer sees the IP packet as a vector of bits $\underline{b}^{IP}$.

Fig. 3.22 shows the MAC layer operation when packing and fragmentation are not used.

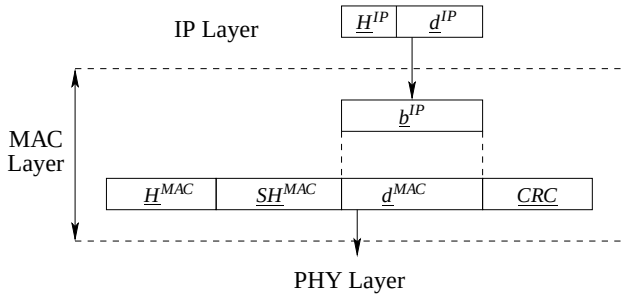

Figure 3.22: MAC layer operation (at the transmitter) when packing and fragmentation are not used.

The following fields are defined :

- $\underline{H}^{MAC}$: (Generic) MAC Header.
- $\underline{SH}^{MAC}$: (Per PDU) SubHeaders.

- $\underline{CRC}$: MAC Layer Cyclic Redundancy Check protecting the fields $\underline{H}^{MAC}$, $\underline{SH}^{MAC}$ and $\underline{d}^{MAC}$.
- $\underline{d}^{MAC}$: MAC data.

In this case : $\underline{d}^{MAC} = \underline{b}^{IP}$.

Fig. 3.23 shows the MAC layer operation when packing is not used and fragmentation is used, considering the example of an IP packet $\underline{b}^{IP}$ is fragmented into 2 segments.

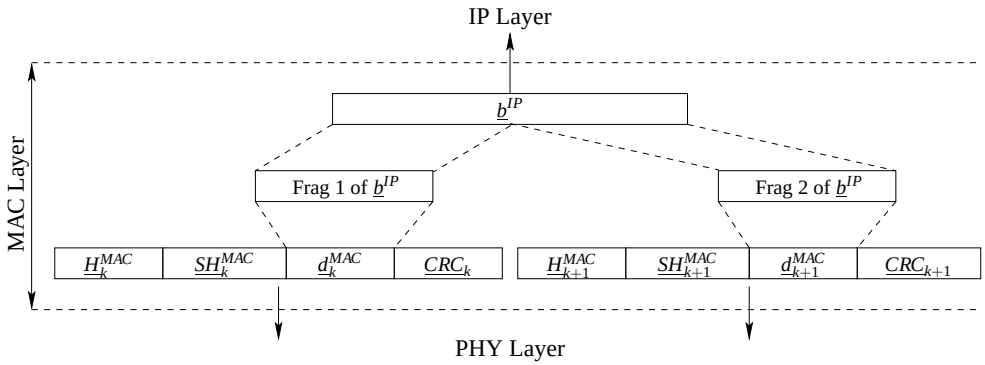

Figure 3.23: MAC layer operation (at the transmitter) when packing is not used and fragmentation is used.

The following fields are defined :

- $\underline{H}_k^{MAC}$: (Generic) MAC Header of the "k^{th}" MAC packet.
- $\underline{SH}_k^{MAC}$: (Per PDU) SubHeaders of the "k^{th}" MAC packet.
- $\underline{d}_k^{MAC}$: Data of the "k^{th}" MAC packet.
- $\underline{CRC}_k^{MAC}$: CRC of the "k^{th}" MAC packet.

$\underline{H}_{k+1}^{MAC}$, $\underline{SH}_{k+1}^{MAC}$, $\underline{d}_{k+1}^{MAC}$ and $\underline{CRC}_{k+1}^{MAC}$ are defined in the same way.
In this case, $\underline{d}_k^{MAC}$ is the first fragment of $\underline{b}^{IP}$ and $\underline{d}_{k+1}^{MAC}$ is the second fragment of $\underline{b}_k^{IP}$.

Fig. 3.24 shows the MAC layer operation when packing is used, considering the example of 2 IP packets packed in the same MAC packet. The first is fragmented into 2 segments.
The following fields are defined :

- $\underline{H}^{MAC}$: (Generic) MAC Header.
- $\underline{PPSH}$: Per PDU Subheaders.
- $\underline{d}_i^{MAC}$: i^{th} data field of the MAC packet, $\forall i \in \{1, 2, 3\}$
- $\underline{PSH}_i$: Packing SubHeader of the i^{th} data field of the MAC packet, $\forall i \in \{1, 2, 3\}$.

In this case, $\underline{d}_1^{MAC}$ is the first fragment of the IP packet $\underline{b}_k^{IP}$, $\underline{d}_2^{MAC}$ is the second fragment of the IP packet $\underline{b}_k^{IP}$ and $\underline{d}_3^{MAC}$ is the IP packet $\underline{b}_{k+1}^{IP}$.

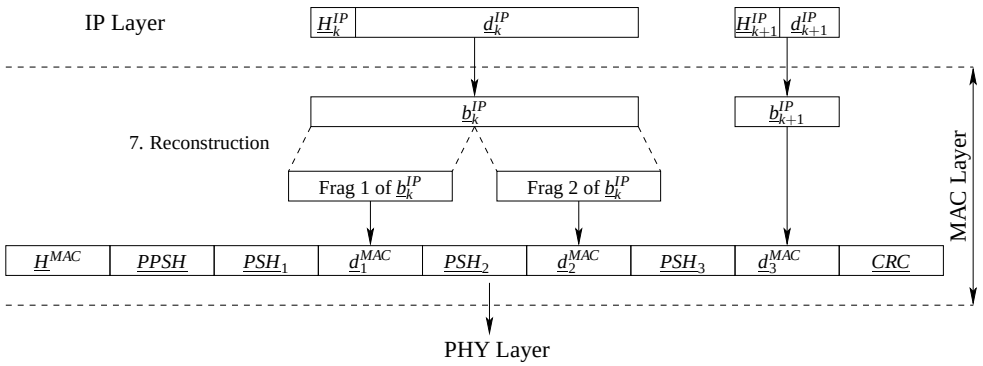

Figure 3.24: MAC layer operation (at the transmitter) when packing is used.

The physical layer Fig. 3.25 shows the operation of the physical layer prior to modulation and coding. $\underline{b}_k^{MAC}$ is a binary vector containing MAC layer packets (3 in the example of Fig. 3.25)

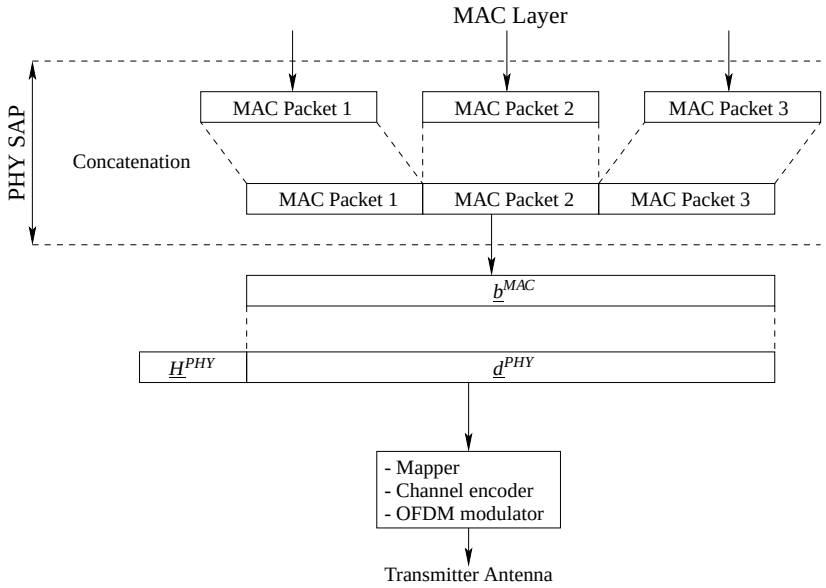

Figure 3.25: Physical layer operation at the transmitter.

The following fields are defined

- $\underline{H}^{PHY}$: Frame Control Header (FCH) containing the DLFP.
- $\underline{d}^{PHY}$: PHY data.

Note that the PHY data is a concatenation of several MAC packets. Also note the PHY layer is OFDM-based.

3.6.2 APP Packet delivery at the receiver

At the receiver, the SISO decoding provides PHY APP packets. These packets contain, of course, the APPs of video sequence bits, but also the APPs of the video headers, as well as the headers of all layers (from PHY to APL). The implementation of Robust Decoding or SARQ requires only the APPs of video sequence bits, more precisely of MB data bits. All the other fields (the headers) can only be exploited as hard data (bits). For this purpose, the APPs of headers need to be converted into binary (estimated) values. The most straight forward way to do that is to take a hard decision on each APP to get the corresponding binary value (estimate of the bit). For instance, the i^{th} bit H_i of header $\underline{H}$ is estimated as follows :

$$\hat{H}_i = \begin{cases} 1, & APP(H_i) > 1/2 \\ 0, & APP(H_i) \leq 1/2 \end{cases} \tag{3.25}$$

On the other hand, in (101)(102), it was shown that a substantial improvement is obtained when Header Recovery (HR) techniques are used. Header recovery techniques exploit interlayer and intralayer redundancies (redundancies in the header itself and redundancies between successive packets of the same layer). Just as Robust Decoding does for video sequences, the interlayer and intralayer redundancies, when present, reduce the set of possible headers from $GF(2)^{L(\underline{H})}$ of all possible binary combinations to a smaller set $\Omega_{\underline{H}} \subset GF(2)^{L(\underline{H})}$ of "valid" headers where $L(\underline{H})$ is the length of the header $\underline{H}$, the MAP criterion is used and the estimated header is given by

$$\underline{\hat{H}} = \arg \max_{\underline{h} \in \Omega_{\underline{H}}} APP(\underline{h}) = \arg \max_{\underline{h} \in \Omega_{\underline{H}}} \prod_{i=1}^{L(\underline{H})} APP(H_i = h_i) \tag{3.26}$$

Obviously, Header Recovery techniques require the length of the header field $L(\underline{H})$ to be known *a priori*. When the header length is not known *a priori*, as is the case of the Extension Header of RTP packets, this parameter is included in the header itself. Consequently, the header must be decoded bit by bit (by thresholding the APPs) starting from the left. the decoding of the header must stop at the last bit of the header, which is unknown a priori. When the last bit of the *Length* field of the header is decoded, the Length of the header is obtained.

Also, when there are little or no redundancies at all, bit per bit decoding and HR decoding are equivalent in terms of performance, bit per bit decoding should then be used since it is simpler.

In addition to header extraction and decoding, if concatenation or fragmentation and/or packing were used at the transmitter, the reverse operations (deconcatenation, defragmentation, unpacking) must be performed at the receiver. The necessary information to conduct these operations is contained in the headers which are decoded prior to these operations as will be explained in detail (case by case for each layer) below.

The physical layer Considering the example of Fig. 3.25 (PHY PDU of 3 MAC packets), the operation of the physical layer should proceed as illustrated in Fig. 3.26, according to the following steps

1. Extract and decode the Frame Control Header $\underline{H}^{PHY}$.

2. Deliver the PHY data APPs $APP(\underline{d}^{PHY})$ to the PHY SAP.
3. Deconcatenate the 3 MAC packets.
4. Deliver the first MAC APP packet.
5. Deliver the second MAC APP packet.
6. Deliver the third MAC APP packet.

The MAC layer Recall that the 6-bit type field of the (Generic) Header indicates which subheaders are present and which are not. Also, the type field of the Header indicates whether the fragmentation subheader (or the packing subheader) are extended or not. Consequently, the (Generic) Header must be decoded before the subheaders.

When packing and fragmentation are not used, the operation of the MAC layer should proceed as illustrated in Fig. 3.27, according to the following steps :

1. Extract and decode the (Generic) Header $\underline{H}^{MAC}$.
2. Extract and decode the (per PDU) SubHeaders $\underline{SH}^{MAC}$.
3. Extract the MAC data APPs $APP(\underline{d}^{MAC})$.
4. Deliver the APP packet to the IP layer.

When packing is not used and fragmentation is used, the operation of the MAC layer should proceed as illustrated in Fig. 3.28, according to the following steps :

1. Extract and decode the (Generic) Header $\underline{H}_k^{MAC}$ of the k^{th} MAC packet.
2. Extract and decode the (per PDU) SubHeaders $\underline{SH}_k^{MAC}$ of the k^{th} MAC packet.
3. Extract the first IP APP fragment.
4. Extract and decode the (Generic) Header $\underline{H}_{k+1}^{MAC}$ of the $(k+1)^{\text{st}}$ MAC packet.
5. Extract and decode the (per PDU) SubHeaders $\underline{SH}_{k+1}^{MAC}$ of the $(k+1)^{\text{st}}$ MAC packet.
6. Extract the second IP APP fragment.
7. Reconstruct the IP packet in terms of APPs.
8. Deliver the IP APP packet.

When packing is used, considering the example of Fig. 3.29, the operation of the MAC layer should proceed as illustrated in Fig. 3.29, according to the following steps :

1. Extract and decode the (Generic) Header $\underline{H}^{MAC}$.
2. Extract and decode the (per PDU) SubHeaders $\underline{PPSH}^{MAC}$.

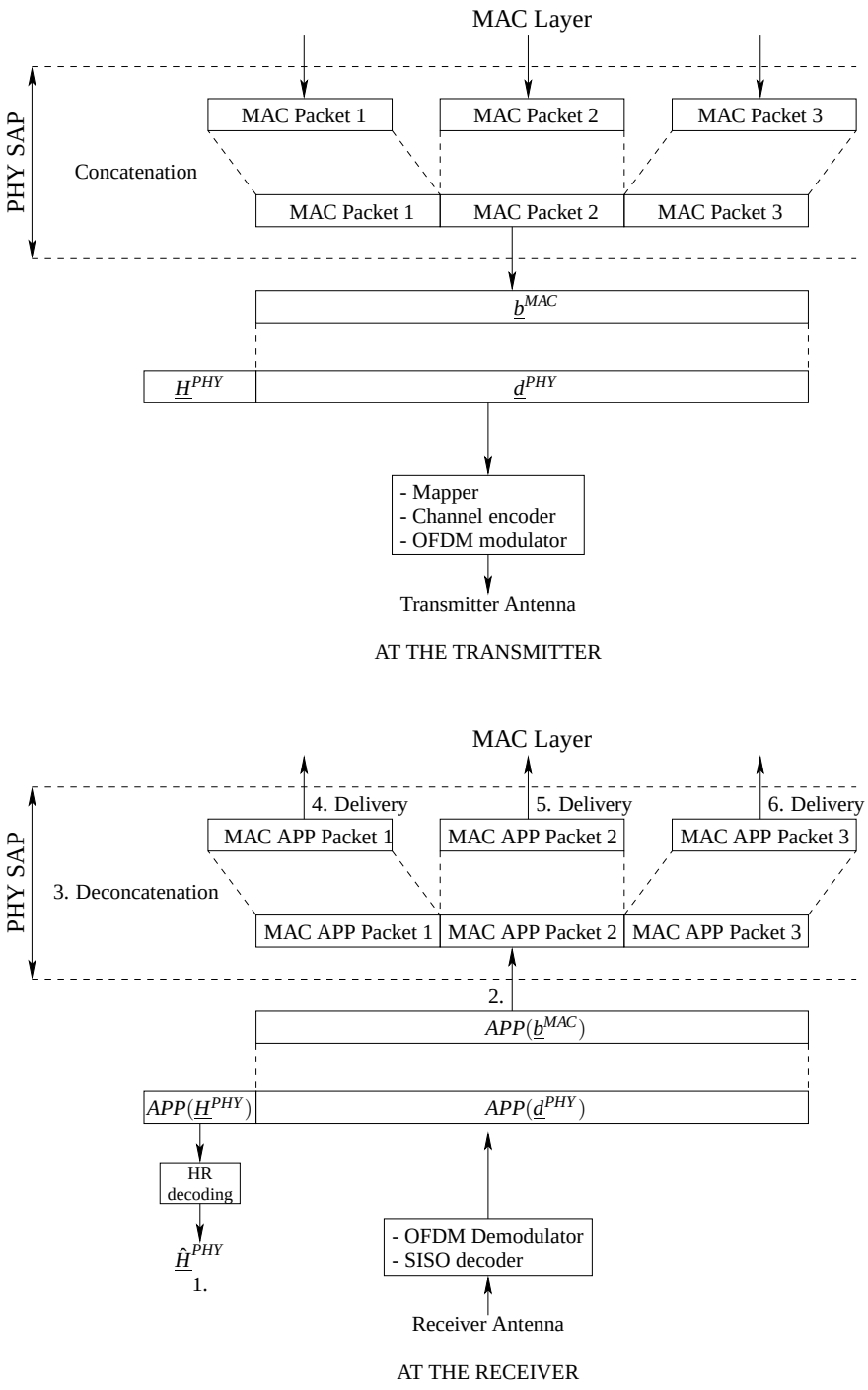

Figure 3.26: Physical layer operation.

3. Extract and decode the Packing SubHeader $\underline{PSH}_1$ of the first packed fragment.

4. Extract the first IP APP fragment.

5. Extract and decode the Packing SubHeader $\underline{PSH}_2$ of the second packed fragment.

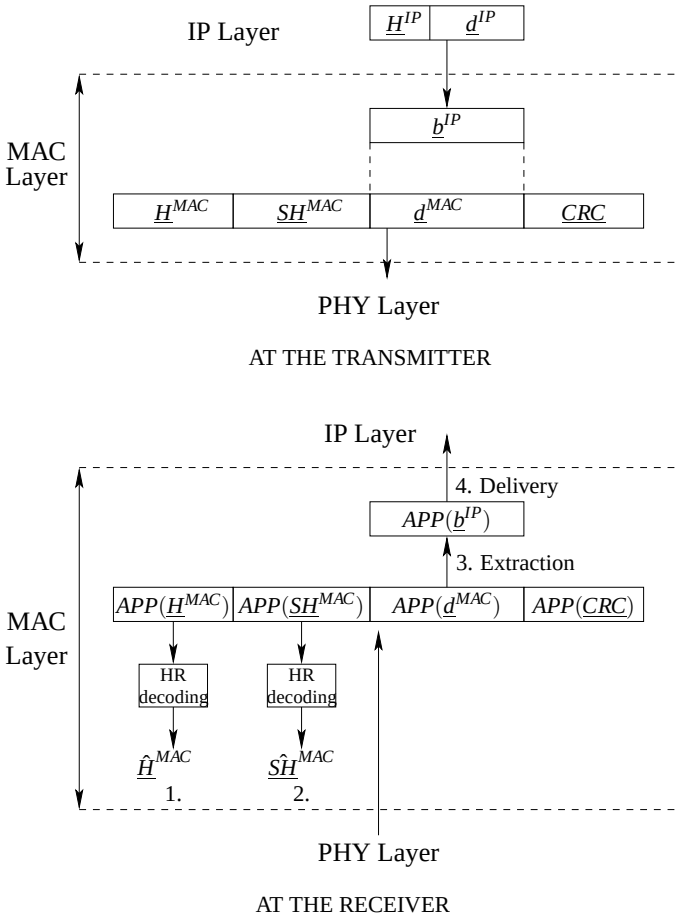

Figure 3.27: MAC layer operation when fragmentation and packing are not used.

6. Extract the second IP APP fragment.
7. Reconstruct the k^{th} IP packet in terms of APPs.
8. Deliver the k^{th} IP APP packet.
9. Extract and decode the Packing SubHeader $\underline{PSH}_3$ of the $(k+1)^{\text{st}}$ packed IP packet.
10. Extract the $(k+1)^{\text{st}}$ IP APP packet.
11. Deliver the $(k+1)^{\text{st}}$ IP APP packet.

Note that in all cases, a bit by bit hard decision should be taken on the data bits and if the CRC of the resulting binary packet is good, bits must be delivered instead of APPs. For this purpose, all MAC packets the CRC of which is good should be acknowledged by the receiver MAC layer and delivered as 0's and 1's (but in real values so that the compatibility with APP packets is preserved).

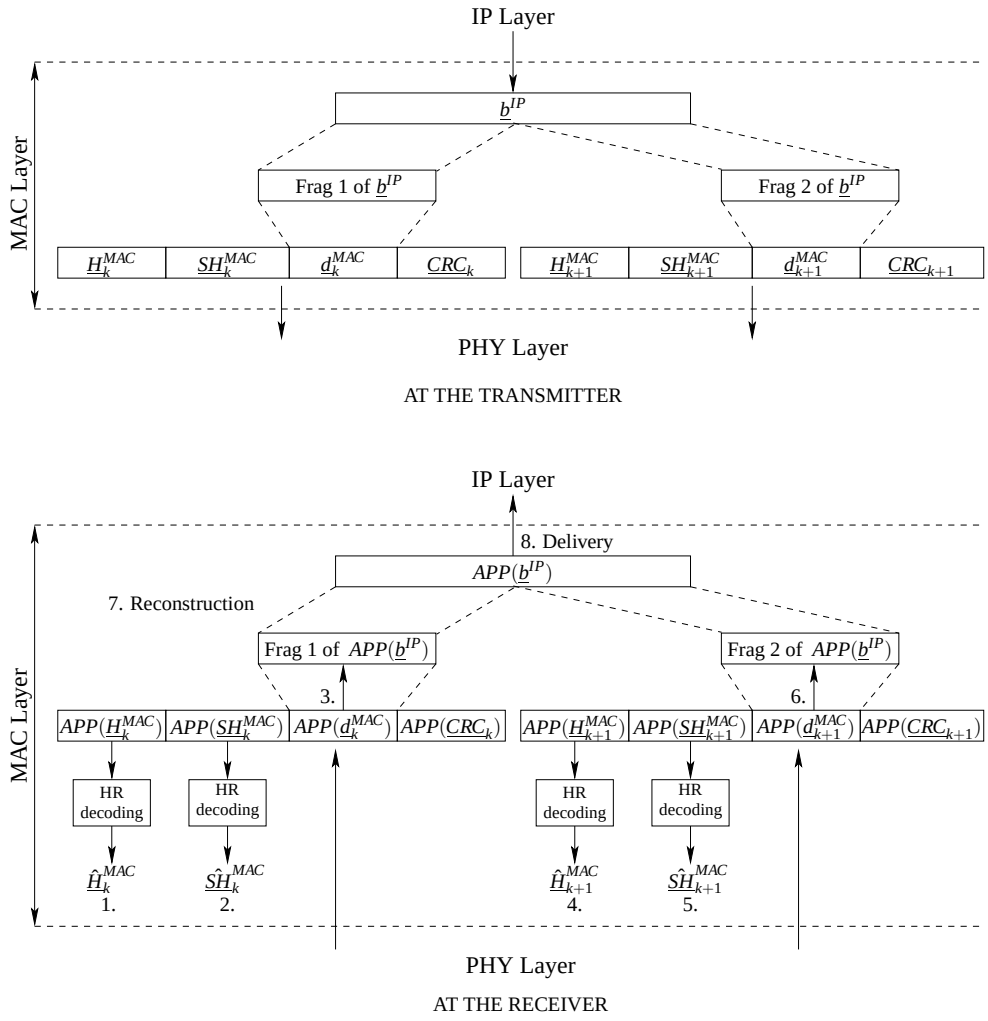

Figure 3.28: MAC layer operation when packing is not used and fragmentation is used.

The IP layer The length of the optional part of the header $\underline{h}_V$ is contained in the fixed part $\underline{h}_F$. Consequently, the Header cannot be decoded as a whole, it should be decoded in two stages starting by the decoding of the Fixed part. The operation of the IP layer should proceed as illustrated in Fig. 3.30, according to the following steps

1. Extract and decode the fixed part of the Header $\underline{h}_F$.
2. Extract and decode the variable part of the Header $\underline{h}_V$.
3. Deliver the IP data APPs $APP(\underline{d}^{IP})$ (UDP APP packet).

The UDP layer The operation of the UDP layer should proceed as illustrated in Fig. 3.31, according to the following steps

1. Extract and decode the UDP Header $\underline{H}^{UDP}$.

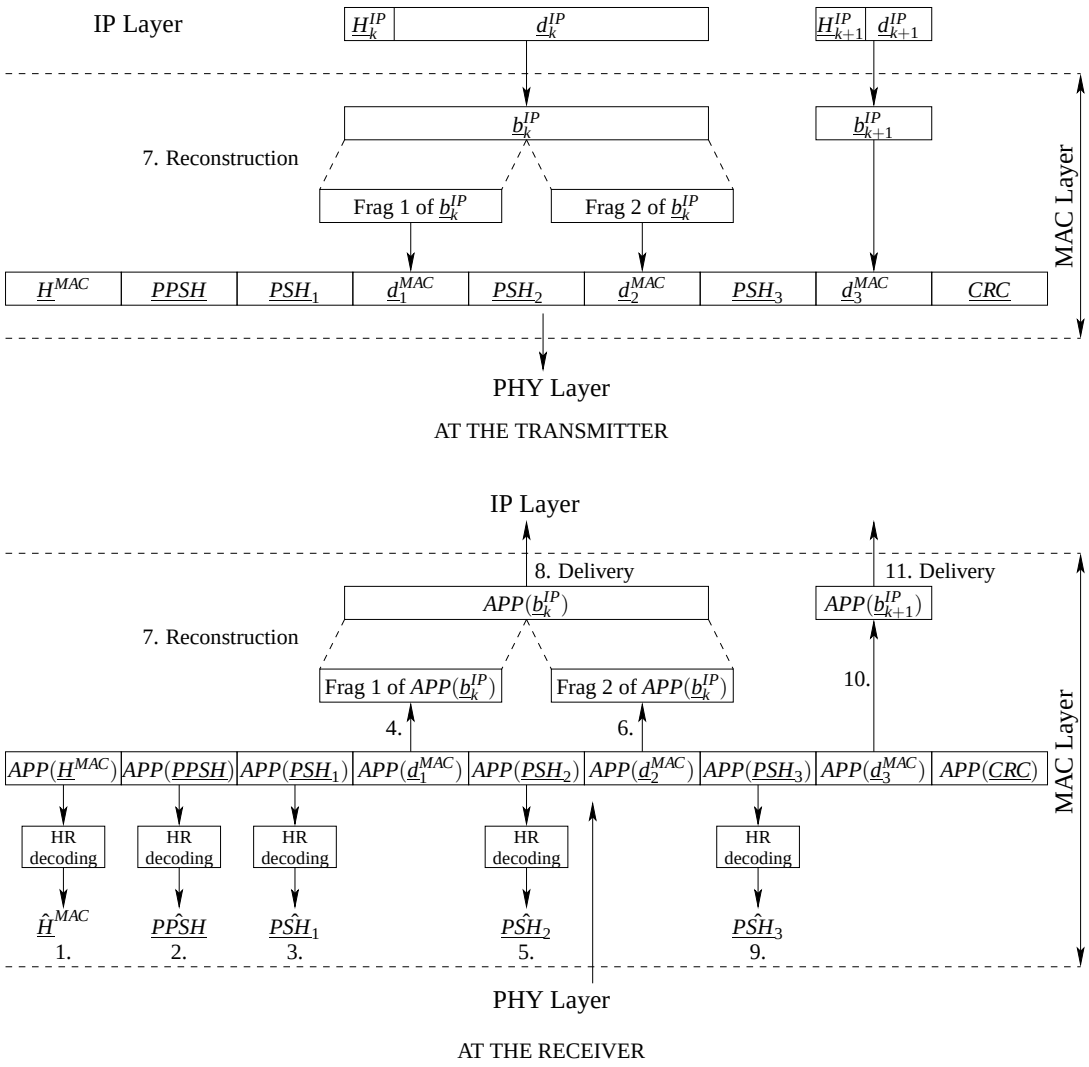

Figure 3.29: MAC layer operation when packing is used.

2. Deliver the UDP data APPs $APP(\underline{d}^{UDP})$ (RTP APP packet).

The RTP layer Recall that the RTP Header consists of a Fixed Header and an Extension Header. As mentioned earlier, the length of the Extension Header is not known a priori, this parameter (the Extension Header Length or EHL) is contained in the Extension Header itself. On the other hand, the header of the Fixed Header is known.

Consequently, the Fixed Header should be decoded first using HR techniques, then the Extension Header should be decoded bit by bit starting from the left as explained earlier. The operation of the RTP layer should proceed as illustrated in Fig. 3.32, according to the following steps

1. Extract and decode the Fixed Header $\underline{FH}$ using HR decoding.

2. Extract and decode the Extension Header $\underline{EH}$ bit by bit.

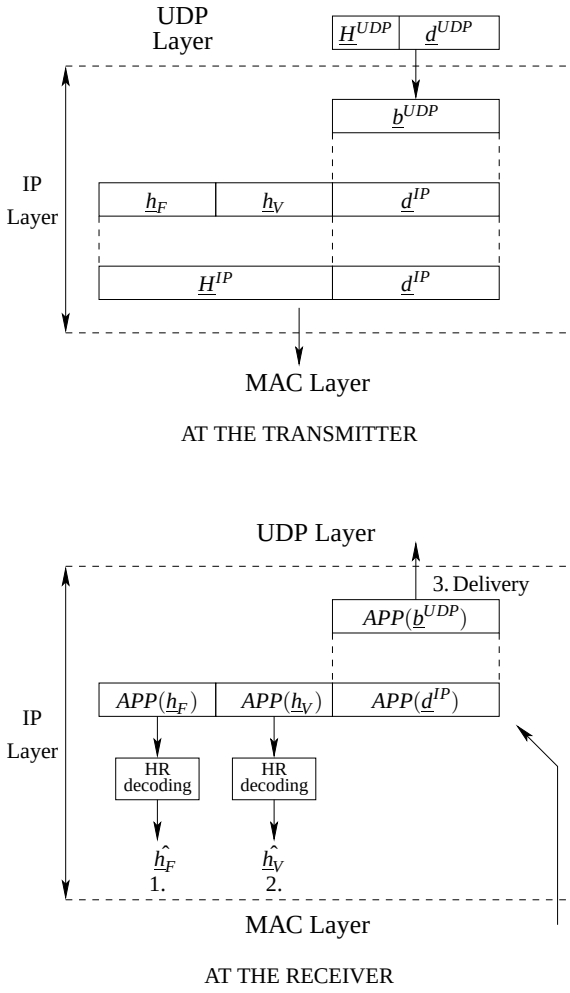

Figure 3.30: IP layer operation.

3. Deliver the RTP data APPs $APP(\underline{d}^{RTP})$ (APL APP packet).

The Application layer Considering the example of a 2 MB-slice Fig. 3.33 shows the operation of the application layer when Data Partitioning is not used while Fig. 3.34 shows the operation of the application layer when Data Partitioning is used. Robust decoding requires only the APPs on video sequence bits for their decoding. As a result, the APPs contained in the MB data fields are fed to the robust (entropy) decoder. The NALU Header and the video Headers (Slice Header, MB Headers) are decoded bit by bit and fed to the other blocks of the H264 decoder.

Note that in case of SARQ, the Likelihood Ratio Test is performed after Robust decoding, which results in a decision as to whether to accept and decode or reject the video packet. This decision is then transmitted to the MAC layer where the retransmission mechanism is implemented. All the MAC packets containing a part of a discarded NALU

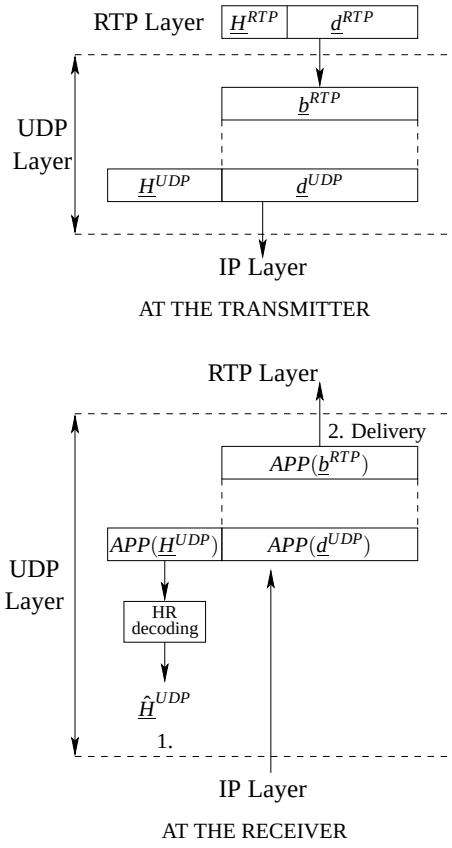

Figure 3.31: UDP layer operation.

and the CRC of which was not good should be discarded by the receiver MAC layer and retransmitted by the transmitter.

3.7 Conclusion

In this chapter, a new retransmission scheme has been defined for the transmission of video data. This scheme combines robust decoding with retransmissions. The decision to ask for a retransmission was modelled as an hypothesis testing problem. The analysis showed that the Neyman-Pearson criterion should be used to reduce the number of retransmissions while maintaining the same level of quality. Simulation results using H264 encoded data confirmed that the threshold of the test can be tuned to choose the throughput-quality trade-off. The nominal quality is reached with a threshold set to infinity and resulting in a low throughput, but results also show that a quasi-nominal quality can be reached with a finite threshold and a higher throughput. The throughput gain increases with the error correction capacity of the robust decoder. Results showed that *in the best case* (at maximum robust decoding error correction capacity), a throughput gain of at least 17% can be reached at quasi-nominal quality and a throughput gain of at least 250% can be reached at 2-3 dB lower. Better gains (on the order of 20% at quasi-nominal quality and 275% at 2-3 dB lower) can be reached when a single (intra-coded) image is transmitted.

The SARQ scheme, combines hypothesis testing with robust decoding and ARQ pro-

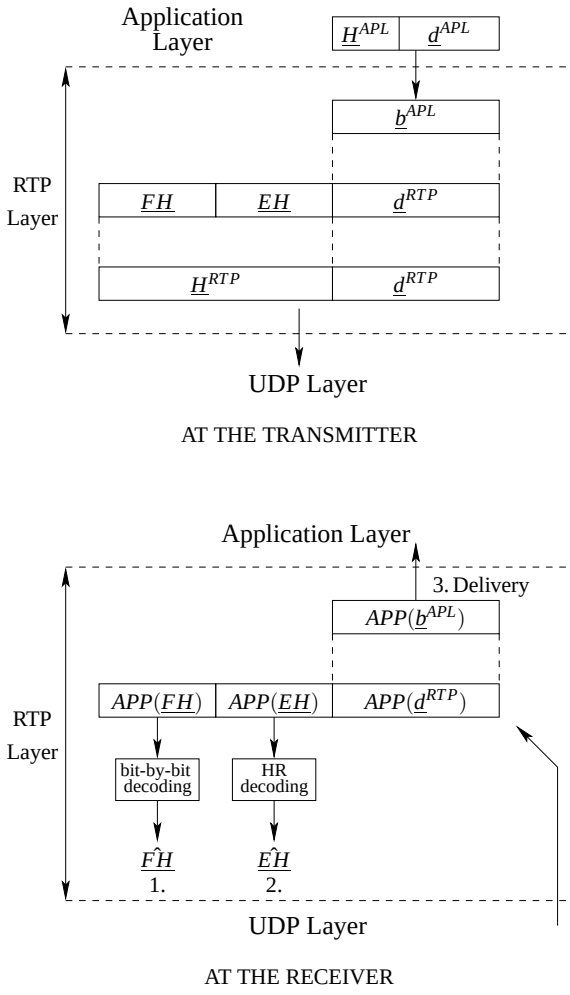

Figure 3.32: RTP layer operation.

cessing to offer the possibility to trade throughput for quality (and vice versa) with the *best possible* throughput/quality trade-off.

This chapter also addressed the question of the implementation of Robust Decoding and Soft ARQ on practical systems. Both schemes require a SISO channel decoding at the receiver and the delivery of soft data (APPs) up to the application layer. The delivery of APPs up to the application layers was discussed in detail assuming the existence of SISO decoder at the physical layer. The protocol stack used was based on a WiMAX air interface. At the application layer, H264 video packet format was assumed.

Further, it has been shown that the practical implementation of SARQ requires in addition a crosslayer mechanism where MAC level ARQ retransmissions are driven by the application layer. The decision to ask for a retransmission is taken after APP packets have been delivered to the application layer, and Robust Decoding as well as a Likelihood Ratio Test have been performed.

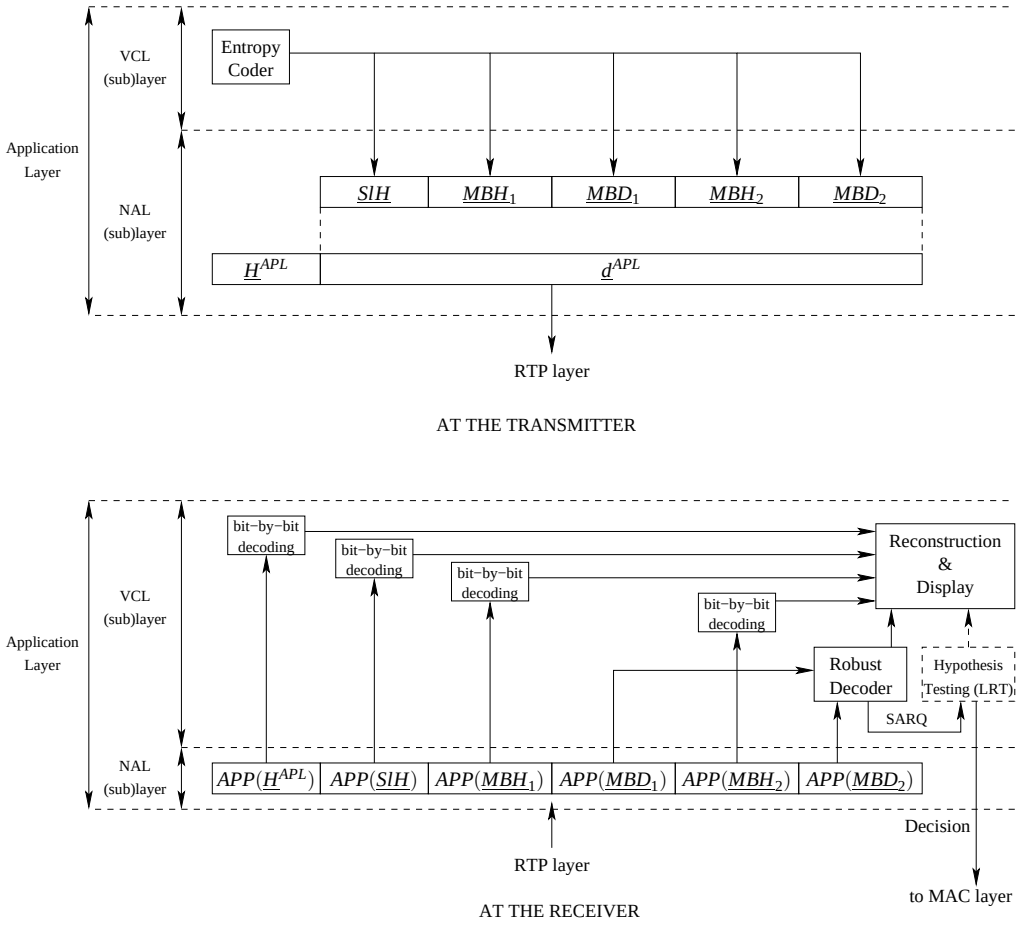

Figure 3.33: Operation of the Application layer when data partitioning is not used.

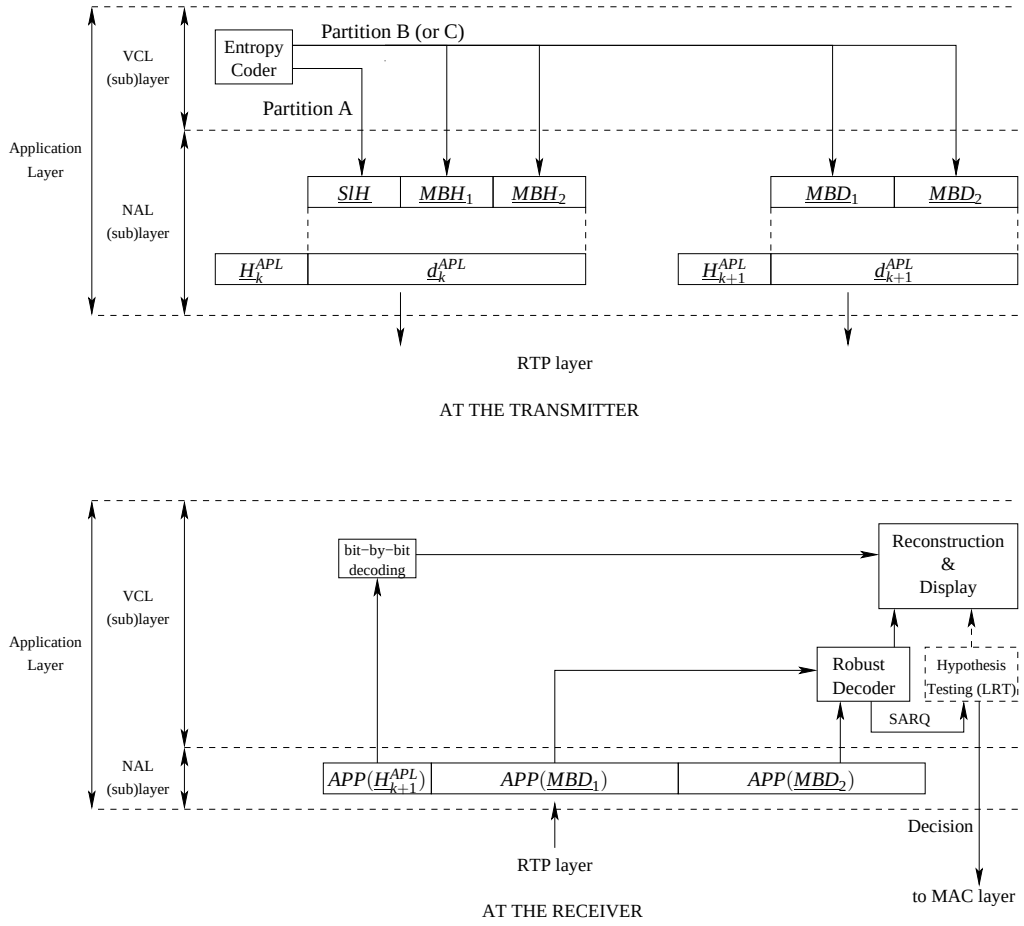

Figure 3.34: Operation of the Application layer when data partitioning is used.

Chapter 4

Transmission Schemes for Scalable Video Streaming in Point-to-Multipoint Communications

4.1 Introduction

Unlike voice services, video services are characterized by large bandwidth requirements, which can be hundreds of times higher than the bandwidth required by voice services. In order to reduce the bandwidth used, all the customers asking for the same video service could be gathered in one group of receivers to which the video content is conveyed using the same channel. The consumed bandwidth is then reduced by a factor equal to the total number of receivers. This approach is valid in case multiple customers per cell are interested in the same content. In case the same data are transmitted from a single source entity (e.g. a BS) to multiple endpoints, the communication is said to be Point-to-MultiPoint (PMP).

On the other hand, scalable video codecs offer the possibility to have several qualities of the same encoded video, by providing at its output two streams (or more). If the video decoder is provided with the first encoded stream, the video obtained after the decoding operation is of basic quality. The other streams are quality enhancement streams, i.e. each time we provide the decoder with an additional stream, the displayed video quality is upgraded. In other words, the basic stream is indispensable if the end user wants to watch the video sequence while the other streams are only optional, i.e. it would be preferable but not indispensable to have them.

A suitable scheme for transmitting such data would transmit the basic stream in a *reliable* way and the other streams in a *best effort* way. A reliable transmission is a transmission that ensures that the data are received correctly by making use of ARQ.

There are three basic ARQ schemes. The Stop-and-Wait scheme (SW), the Go-Back-N (GBN) scheme and the Selective Reject (SR) scheme. These techniques were first proposed and studied in the case of a point-to-point communication system (104)(105)(106)(107)(108)(109)(110). Then, several schemes based on the same principles were defined and studied for the case of a point-to-multipoint communication system (111)(112)(113)(114)(115)(116)(117)(118). All these works focused on the study of the reliable transmission of one flow of data, that is to say, all packets have the same importance and are part of the same stream. Besides,

it was assumed that there was always a new packet waiting to be transmitted at the transmitter and the throughput was calculated based on the transmission of an infinite number of packets.

In this chapter, we propose three schemes for transmitting a two-level scalable video to several receivers. The first two are extensions of the GBN and the SR basic schemes, while the third one is a new scheme allowing to reduce the size of the buffer necessary at the receiver end. In addition, the number of packets of each stream is finite.

The rest of the chapter is organized as follows. In section 4.2, the three schemes are presented. In section 4.3, numerical results of computer simulations are presented. In section 4.4, the throughput expression of the new scheme relative to that of the SR scheme is derived. In section 4.5, the application of the different schemes to practical systems is discussed. Finally, in section 4.6, conclusions are drawn.

4.2 The transmission schemes

Let us consider a multicast communication system where a single transmitter sends data to K receiving terminals in the form of packets (in practice, it could be the case of a base station transmitting to several terminals) and where the data are the output of a 2-level scalable video encoder. Given their importance, the packets containing video data of the first (basic) stream will be called *Indispensable* or "I" packets and the packets containing data of the second (quality enchancement) stream will be called *Accessory* or "A" packets.

We assume that all packets have the following information embedded in them

1) A bit indicating the nature of the packet (or the stream it belongs to).

2) A sequence number identifying the position of the packet in the stream.

3) A Cyclic Redundancy Check (CRC) which enables each receiver to detect transmission errors in a packet.

We also assume that the encoded video consists of several portions that are transmitted sequentially, each portion of the scalably encoded video consists of L_1 packets of type I and L_2 packets of type A.

An error-free transfer of "I" packets must be achieved so that we make sure each end user (customer) receives (i.e. can watch) at least the basic quality video. In order to make sure that all "I" packets are received, we must use a retransmission process based on the feedback of the receivers (ARQ technique). As for "A" packets, which are only optional, they should be transmitted in a *Best Effort* way, *cyclically* way as we propose, and without feedback from the receivers. Note that "I" packets should also have priority in the transmission scheme.

The receiving terminal, which could be a 2G Mobile Station (MS), a 3G User Equipment (UE) or a WiMAX terminal (SS/MS) is composed of two parts, the receiver part and the user part. The user consists of the video decoder and corresponds to the application layer of the terminal. It decodes the video stream(s) that are delivered to it by the receiver. The receiver, which corresponds to the lower layers of the receiving terminal, is responsible for the delivery of error-free packets in the right order. For that purpose, the receiver performs the CRC-checking and the packet reordering operations.

Reordering of the accepted packets is performed through buffering; When a packet is accepted, it is either stored in the receiver buffer or delivered to the user (where it is stored then decoded). Since packets are only delivered to the user in the right order (at least in streaming applications), an accepted packet is delivered only if all the packets with a smaller sequence number have been delivered. For example, suppose that when packet I2

is accepted packet I1 has not been accepted yet, packet I2 is then buffered until packet I1 is accepted, only then the two packets are delivered to the user (first packet I1, and then packet I2) and the memory used to buffer packet I2 is thus released. The same holds for the second stream. The receiver must then manage as many buffers as there are streams.

In the case of a one-stream video sequence. When the transmission starts, the user (decoder) buffer is empty, then as packets are delivered by the receiver, they are first stored in the user buffer and then decoded. The decoding (video playback) starts when the user buffer occupancy reaches a given level. This level is a parameter of the streaming application. A typical value corresponds to 5s of video playback, which means that after the video playback starts, the user has 5s to receive new packets, otherwise the image will freeze, this time constraint limits the number of possible retransmissions. This parameter should thus be chosen so as to minimize the probability of an image freeze (the larger the better) and to minimize the waiting time of the client (the smaller the better), a compromise must then be found. This parameter may somehow be managed by the client (at the user) that plays back the video.

In the case of a two-level scalable video sequence, the second stream is only useful in its entirety, and since the decoder must wait until the end of the allowed transmission time to know whether this stream is available or not, the decoding can only start when the whole second stream is received if it is the case and at the end of the allowed transmission time if it is not the case.

4.2.1 The basic schemes

As mentioned above, "I" packets should have priority and should be transmitted in a reliable way using an ARQ technique, and "A" packets should be transmitted cyclically in a best effort way.

Point-to-MultiPoint (PMP) ARQ schemes have been proposed in (111) (112) (113) (114). A PMP ARQ scheme can be seen as a point-to-point ARQ scheme if instead of considering that we have K different receivers, we consider that we have one group of K receivers. The MultiReceiver Group (MRG) acknowledges the packet if each receiver acknowledges it at least once, and it (the MRG) doesn't if at least one receiver doesn't acknowledge it. More precisely, the transmitter transmits a packet and expects the positive acknowledgements from all receivers. If it does not receive the acknowledgements from all receivers within a chosen time-out period, it retransmits the packet. Each receiver sends a positive acknowledgement when it receives an error-free packet.

In the SW scheme, the transmitter sends a packet and waits for acknowledgements from the receivers. In the meantime it doesn't transmit any other packet. If it receives the acknowledgements from all receivers within the fixed time-out period, it proceeds to the transmission of the next packet, otherwise it retransmits the same packet and waits again (without transmitting any other packet) for the acknowledgements.

The GBN and the SR schemes are of continuous type. The transmitter expects but does not wait for the acknowledgements of the packet after transmitting it, it actually immediately transmits the next packet. Concurrently, it receives and examines the stream of acknowledgements from the receivers. This feature makes good use of the bandwidth allocated. In the GBN scheme, the transmitter backs up to the unacknowledged packet and retransmits it along with all the following packets (N packets are retransmitted in total, $N-1$ representing the number of packets that can be transmitted during a time-out period). For the SR scheme, the transmitter retransmits only the unacknowledged packet.

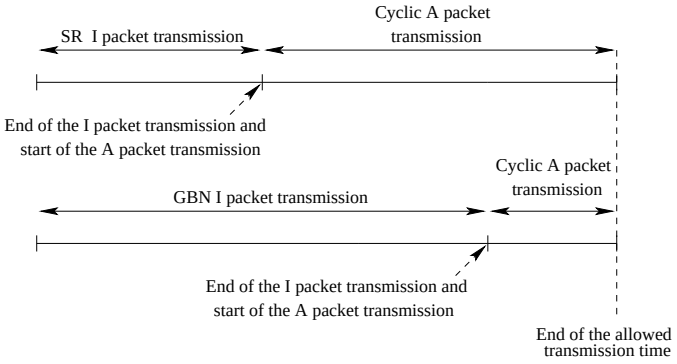

Figure 4.1: The basic SR and GBN schemes for transmitting a scalable video.

For all three schemes, two strategies can be used : the Dynamic Retransmission Group Reduction (DRGR) strategy and the Fixed Retransmission Group (FRG) strategy. In the DRGR strategy, after a packet transmission, the transmitter must receive positive acknowledgements from only those receivers which did not acknowledge successfully during earlier transmission attempts, before the present attempt is declared successful. The transmitter has then to memorize which receiver acknowledged which packet. In the FRG strategy, the transmitter ignores positive ackowledgements from receivers during the previous transmission attempts; after a packet transmission, the transmitter must receive positive acknowledgements from all receivers before it declares the present attempt successful. Actually, when using the DRGR strategy, the transmitter manages a memory of size $N \times K$ bits whereas in the FRG strategy, the transmitter manages a memory of N bits only.

A basic scheme for scalable video streaming would first transmit the "I" packets using an ARQ technique, and subsequently transmit the "A" packets in a cyclic best effort way (without feedback). As shown in Fig. 4.1, according to whether the ARQ technique used is the GBN or the SR, there are two basic-schemes which we will call the GBN scheme and the SR scheme. The conventional SW scheme is not considered because of its inadequacy with real time applications like video streaming. However, we will see in section 4.5 how parallalising the SW through several separate instantiation can be used to reach the performance of the SR ARQ scheme.

In the sequel, we will assume that all packets have the same length (n bits), that the time interval during which a packet is transmitted is called a *time slot* and that the time-out counter is set to expire after the transmission of exactly $(N-1)$ packets.

Let us now get a closer look at the end of phase 1, more precisely when the number of "I" packets which have not yet been ACKed becomes strictly less than N. During this phase, which turns out to be a middle phase (between the "I" packet transmission phase and the "A" packet transmission phase), there are idle times, as illustrated in Fig. 4.2 for the SR scheme where packets I3, I15 and I20 remain to be transmitted, the timeout is only shown for packet I3, but the same thing happens with I20. For the GBN scheme, instead of I3, I15 and I20, we will have I18, I19 and I20 (if $L_1 = 20$).

In order to optimize the schemes in terms of performance and channel use, "A" packets are transmitted during the idle time, as shown in Fig.4.3.

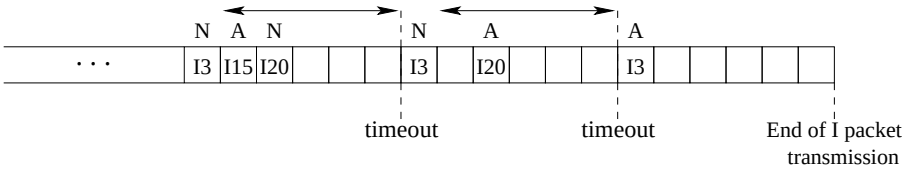

Figure 4.2: Middle phase of the SR scheme with idle times.

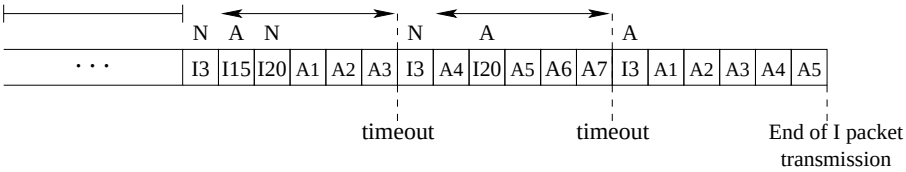

Figure 4.3: Middle phase of the SR scheme without idle times ($N-1=5$ and $L_2=7$).

4.2.2 The proposed scheme

In this paragraph, we propose a new scheme for transmitting the scalable video. Our technique is based on the concept of the *observation window*. When a packet is NACKed (packet I3 in Fig.4.4), it is retransmitted N slots later. The timeframe between the two transmissions represents the *observation window*. The $(N-1)$ packets transmitted after the retransmission represent the *retransmission window*, and are determined according to the following rules : An "I" packet transmitted and ACKed in the observation window is replaced by an "A" packet in the same position of the retransmission window. Similarly an "A" packet is replaced by another "A" packet. Finally, an "I" packet that is NACKed is simply retransmitted (e.g. packets I13 and I15 after the retransmission of I11 in the second line of Fig.4.4).

Note that the content of the retransmission window is directly deduced from the results of the transmissions that occured in the observation window, packet per packet. Also, note that a retransmission window can be itself an observation window if the retransmission is not successful. Outside the retransmission window, the transmitter uses the same transmission rule as the SR scheme in phase 1.

Also, in this scheme, though transmission of "A" packets can start well before phase 2, as explained above, there is also a middle phase when the number of "I" packets to transmit goes below N, the same transmission rule as in the two other schemes is used, i.e. when an "I" packet cannot be transmitted, an "A" packet is transmitted as shown in Fig. 4.6.

4.3 Numerical results

In this section, we compare the three schemes presented above in terms of performance and buffering requirement. For comparison, we will define a performance criterion.

The multicast communication system we consider consists of $(K+1)$ stations, one transmitter and K receivers. We call the path between the transmitter and a particular receiver a *(forward) unicast channel*, each Unicast channel is assumed to be a Binary Symmetric Channel (BSUC) (see Fig. 4.7). The unicast channels are assumed to produce independent noise processes and each noise process is assumed to be white, that is noise disturbs the bits transmitted on the BSUC at random. The BSUCs are also supposed

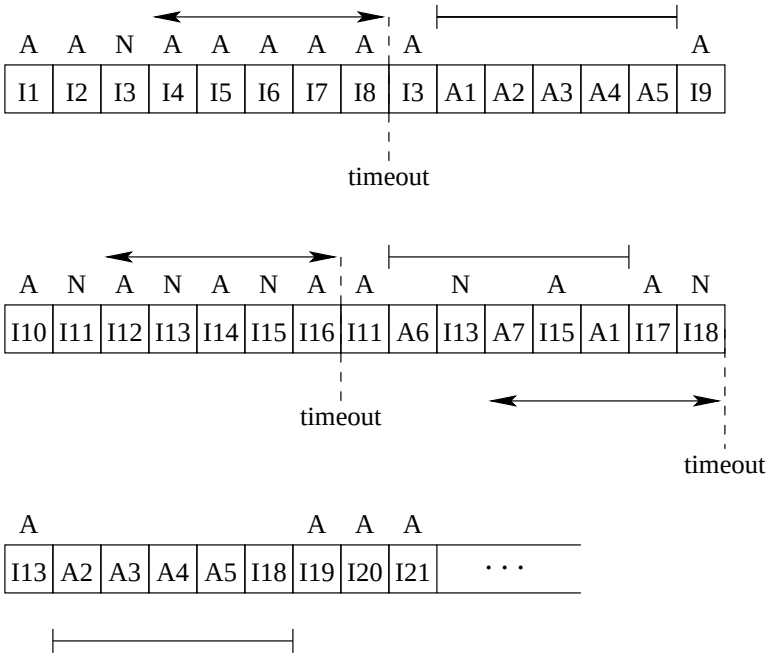

Figure 4.4: Phase 1 of the new scheme ($N-1=5$ and $L_2=7$).

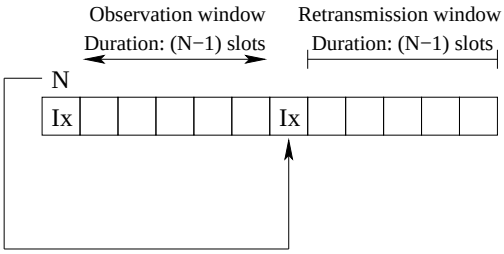

Figure 4.5: Observation window and retransmission window ($N-1=5$).

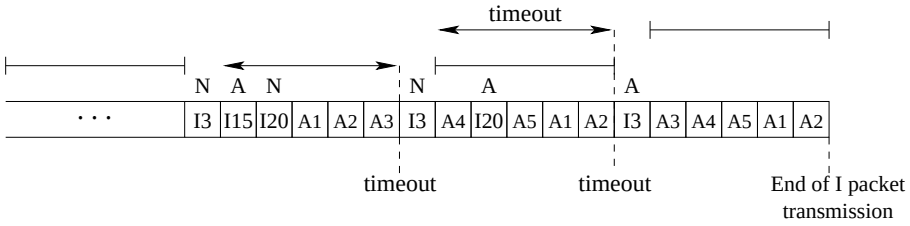

Figure 4.6: Middle phase in the new scheme ($N-1=5$ and $L_2=5$).

to have the same Bit Error Rate (BER) ε. As for the *reverse (feedback) channel*, it is assumed noiseless (i.e. error-free). Generally, the reverse feedback channels use a separate frequency band and the data are sent at a high transmit power and with a heavy physical layer protection (in terms of modulation and coding). Hence, the feedback channels will be assumed noiseless (error-free).

In the case of one flow (traditional ARQ techniques), the SR technique is optimal in terms of throughput and much more efficient than the GBN technique, particularly when

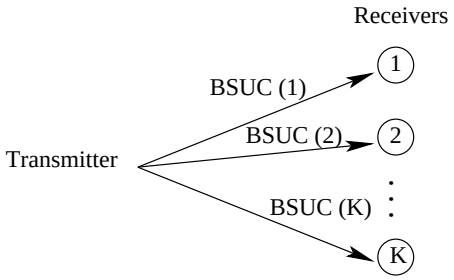

Figure 4.7: Channel model of the multicast environment.

the channel is highly error-prone. On the other hand, in the GBN technique, the receiver accepts and delivers the packets in their original sequence and therefore doesn't require buffering while in the SR technique, the receiver needs to buffer all the packets which cannot be delivered until one or several other packets are accepted and delivered. With a theoretically infinite stream, the receiver needs an infinite buffer, which is known as *Ideal Selective Repeat*.
In the sequel, we will define a performance criterion, and then compare the three schemes with respect to these two criteria (performance and memory), but first, let us recall the assumptions we made

1) All packets are of the same length (n bits).

2) The time-out counter is set to expire after the transmission of exactly $(N-1)$ packets.

3) The unicast channels are independent and memoryless.

4) The BER of each BSUC is ε.

5) The feedback channels are error-free.

So far, the main performance criterion was the throughput (109)(110)(111)(112)(113)(114)(115)(117), but this was for ARQ techniques used to transmit one very long (theoretically infinite) bit stream reliably. In our case, the three schemes are meant to transmit two finite bit streams, one in a reliable way (using feedback channels and retransmissions) and another one in a best effort way (without retransmissions).

Given that the first stream is finite and reliably transmitted (in order for all the receivers to have at least the basic quality video), the difference comes only from the second bitstream and its reception rate among the receivers. That's the reason why the performance criterion we use, which will be denoted by the variable x is defined as the average number of receivers having received the second bitstream (all "A" packets) at a given reference time. As a first approach, we could use the reference time to be t_0, that is, the time when the slowliest of the three schemes (the GBN scheme) finishes the transmission of the first flow, but the latter being a random variable whose statistics depend on the system parameters, it cannot be used as the reference. We choose as reference a fixed time like the time when the transmitter stops the transmission (see Fig. 4.8). This time is totally independent of t_0, it depends on the real-time application constraints.

4.3.1 Performance comparison

Let us first consider that all receivers are interested in the two streams of the video. The three schemes described in section 4.2 were implemented with $K = 10$ receivers, $\varepsilon = 0.0005$, $N = 10$ slots, $n = 50$ bits and $L_1 = L_2 = 20$ packets. These parameters are

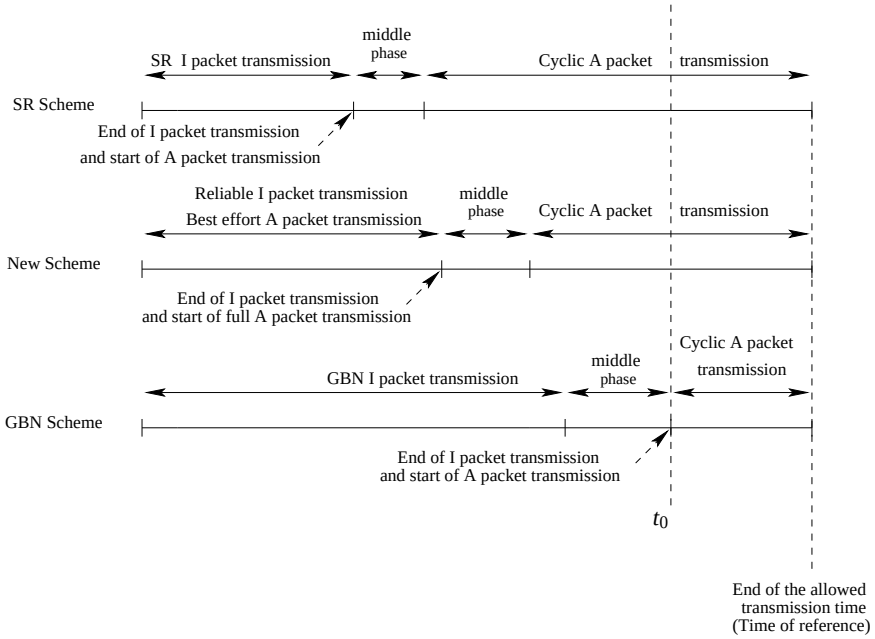

Figure 4.8: Schemes for transmitting a scalable video.

not very realistic (particularly the packet size) but they allowed us to keep the simulation times within very reasonable limits. The results of the first simulation are depicted in Fig. 4.9. The curves relative to the SR and the new scheme (NS) are superimposed, which means that the new scheme has got optimal performance just as the SR scheme. Though the BER is low and the packet size is small, we notice that the GBN is much less efficient than the two other schemes.

The performance of the new scheme is the same as that of the SR scheme because it performs retransmissions in a selective manner exactly as the SR scheme does, there are no useless retransmissions in this scheme also, the only difference is that in our scheme, "I" packets transmissions are interleaved with "A" packet transmissions which causes the new scheme to be slowlier than the SR scheme with respect to the transmission of "I" packets, but the total rate is the same.

The effect of the number of receivers and the BER is further investigated for the optimal schemes (SR and NS) in Figs. 4.10 and 4.11 respectively. A considerable increase in the number of receivers doesn't much affect the performance of the system since it results in a rather small additional transmission time to reach the same performance. This is explained by the fact that when using the DRGR strategy, the average number of transmissions grows logarithmically, i.e. slowly with the number of receivers (contrary to the FRG strategy, with which the average number of transmissions grows exponentially, i.e. very fast, with the number of receivers) as shown in (114). We can thus increase the number of receivers without seriously deteriorating the performance. Bearing in mind that the more receivers there are, the more efficient use is made of the bandwith, this result tells us that using multicast in such an application is a very good approach since it allows to make a very efficient use of the bandwith to the detriment of a small performance deterioration.

In contrast, the system performance is very sensitive to the bit error rate, for when the latter increases, the system performance degrades sharply, particularly for high values.

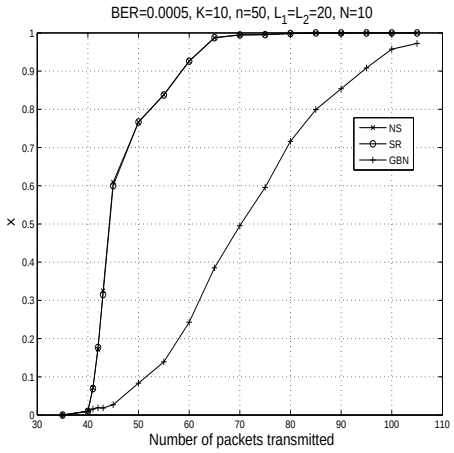

Figure 4.9: Performance of the three schemes for $K = 10$, $\varepsilon = 0.0005$, $N = 10$, $n = 50$, $L_1 = L_2 = 20$.

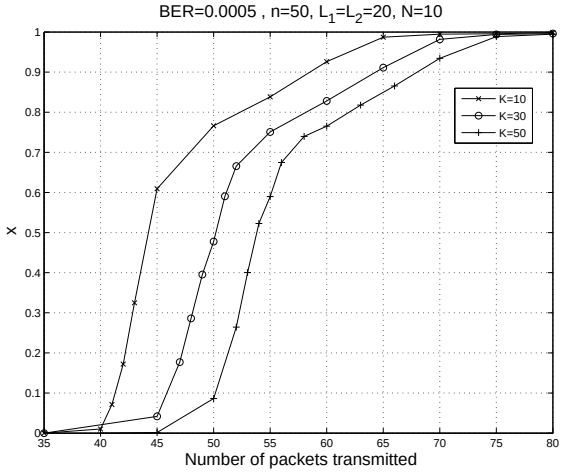

Figure 4.10: Performance of the optimal schemes with different group sizes.

Let us now consider that the receivers are split into two groups, a first group of K_1 receivers only interested in the first stream of the scalable video being transmitted and a second group (the $K_2 = K - K_1$ other receivers) interested in the two streams (full video). As explained in section 4.2, in the case of a user only interested in the first stream of the scalable video that is being transmitted, the decoding can start as soon as the user buffer is filled to a certain level. The higher this level is, the longer is the required delay but the less likely is an image freeze, a compromise between these characteristics of the provided service must determine the value of the buffer occupancy level required to start the decoding.

Fig. 4.12 plots the average delay required for a user (client) of the first group to be able to start the decoding (watching the video). The required user buffer occupancy level is varied between 5 packets and 30 packets (with $L_1 = 40$ and $K_1 = K_2 = 10$). The delay necessary with the NS is slightly higher than that of the SR scheme, which was expected since the NS performs the transmission of both streams more or less in parallel while the SR scheme is totally devoted to the transmission of the first stream in the first phase. This

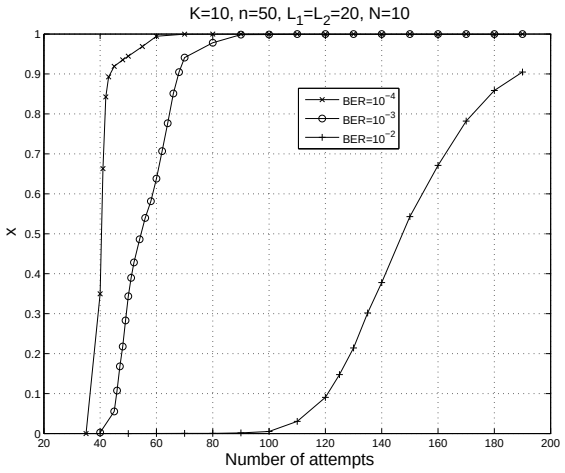

Figure 4.11: Performance of the optimal schemes with different bit error rates.

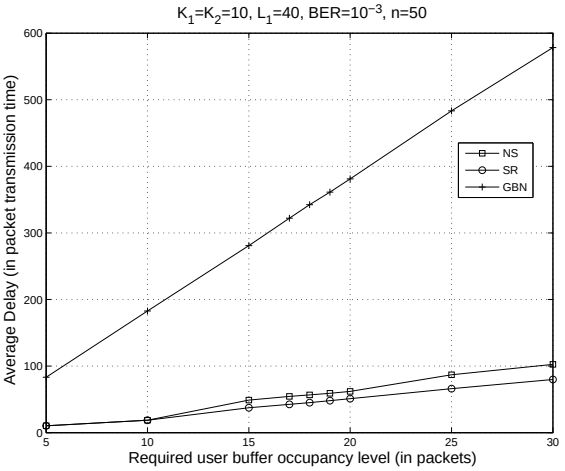

Figure 4.12: Average required delay for a user to reach the required buffer occupancy level.

represents the drawback of the NS, i.e. with this scheme, clients interested only in the basic quality video will wait a little longer than if SR were employed. As for the GBN scheme, it requires much more time due to its inefficiency.

4.3.2 Buffering requirement

Though the decoding of the two streams is performed in parallel, the reception must be managed by two independent processes, each dedicated to one stream, as explained earlier. It is obvious that buffering is necessary at the user because the decoder must buffer the data available for decoding as long as they have not been decoded, the user must be able to buffer as many packets as there are (L_1 "I" packets and L_2 "A" packets). These data are delivered by the receiver and since in the SR and NS schemes, "I" packets may be accepted in the wrong order (due to selective rejection/retransmission processes), buffering is also necessary at the receiver. Further, "A" packets are transmitted cyclically and due to the selective rejection (or acceptance) process, they may be accepted in the wrong order too. Buffering of "A" packets at the receiver must also be provided for all schemes.

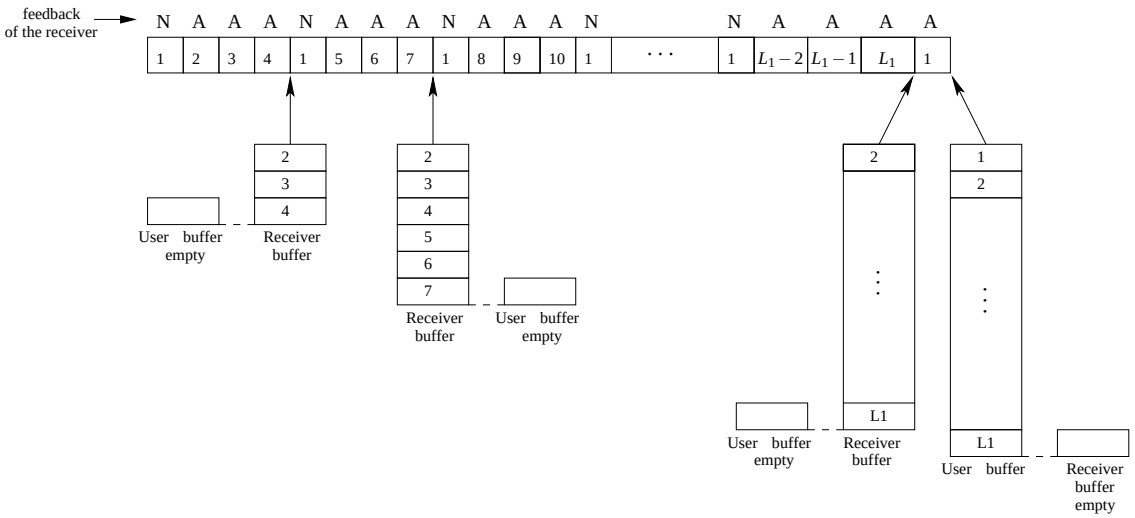

Figure 4.13: Progression of the receiver buffer and the user buffer contents (with $(L_1 - 1)$ multiple of $N - 1 = 3$) in the SR ARQ technique.

To better understand the buffering process and the buffer size necessary at the receiver, assume that the first packet transmitted in a one-stream L_1-packet SR scheme is NACKed and that all the packets with a higher sequence number were ACKed. The receiver cannot deliver these packets to the user until the first packet is correctly received. Until then, they are stored in the receiver buffer. Once the first packet is correctly received, all packets are delivered to the user and the receiver buffer is then released. This is the worst case scenario (illustrated in Fig. 4.13) which enables us to see that the receiver buffer size necessary to insure enough storing capacity in 100% of the cases is L_1. This is valid for a stream of L_1 packets transmitted reliably using a SR ARQ technique. If the same stream were transmitted using a GBN ARQ technique, no buffering would have been necessary at the receiver since packets are accepted and delivered in the right order which represents the main advantage of the GBN ARQ technique to the detriment of poor performance.

In the considered two-stream scalable video transmission, using GBN and SR schemes, no receiver buffer is required to store "I" packets in the case of the GBN scheme while the I-packet receiver buffer must be able to store L_1 packets in the case of the SR scheme.

In the new scheme, "A" packets replace "I" packets successfully received in retransmission windows so that no other "I" packet is accepted and buffered in the corresponding slot (see Fig. 4.14). The size of the buffer necessary at the receiver to store "I" packets is then given by a window size parameter, that is to say N packets. The buffer size necessary is then reduced from L_1 packets to N packets only, which is a physical limit imposed by the roundtrip propagation delay. It is particularly interesting when the number of "I" packets L_1 is large as compared to N.

Finally, since the first "A" packet may be the last to be accepted by a given receiver in the cyclic "A" packet transmission, the A-packet receiver buffer must be able to store L_2 packets in all three schemes. In the GBN and the SR schemes, one may think one buffer can be used instead of two since the two streams are not transmitted in parallel, but this is not quite true since during these two phases, there is a phase where both types of packets are transmitted.

Tables 4.1 and 4.2 recapitulate the buffer size necessary for "I" packets and "A" packets

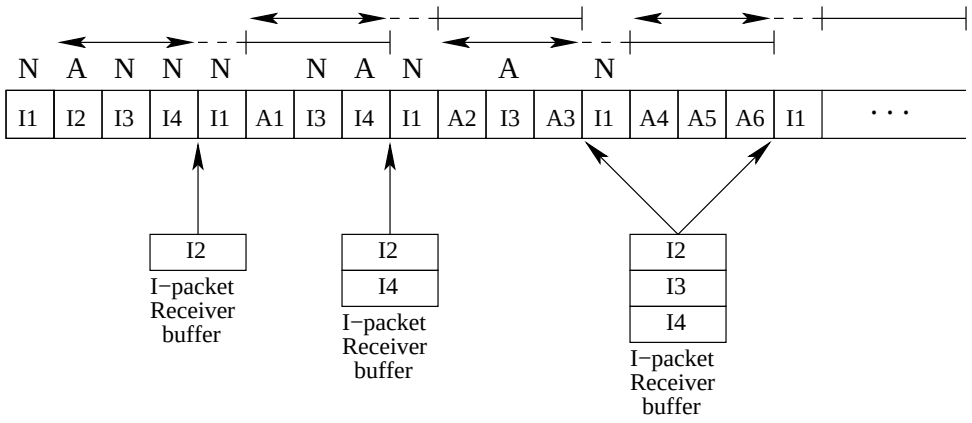

Figure 4.14: Progression of the I-packet receiver buffer's content ($N-1=3$) in the new scheme.

	At the receiver	At the user
GBN	0	L_1
SR	L_1	L_1
NS	N	L_1

Table 4.1: I-packet buffer size required to maintain optimal performance.

respectively. These buffer capacities are necessary to maintain optimal performance. Using receiver buffers with smaller sizes will cause a performance deterioration due to buffer overflows (for if a packet is correctly received but the corresponding buffer is full, it will be discarded by the receiver causing thus a retransmission that could have been avoided). At the user, if buffers with smaller sizes are used, data may be missing if the decoding (user buffer release) doesn't start early enough.

4.3.3 The buffering/performance compromise

Section 4.3.1 showed that the new scheme has optimal performance just as the SR scheme. On the other hand section 4.3.2 showed that the buffer size necessary to maintain this optimal performance is reduced for the new scheme, as compared to the SR scheme. Fig. 4.15 shows that when using the same buffer size (N packets), the new scheme clearly outperforms the SR scheme, which means that *the proposed scheme provides a better buffering/performance compromise, as compared to the SR scheme.*

	At the receiver	At the user
GBN	L_2	L_2
SR	L_2	L_2
NS	L_2	L_2

Table 4.2: A-packet buffer size required to maintain optimal performance.

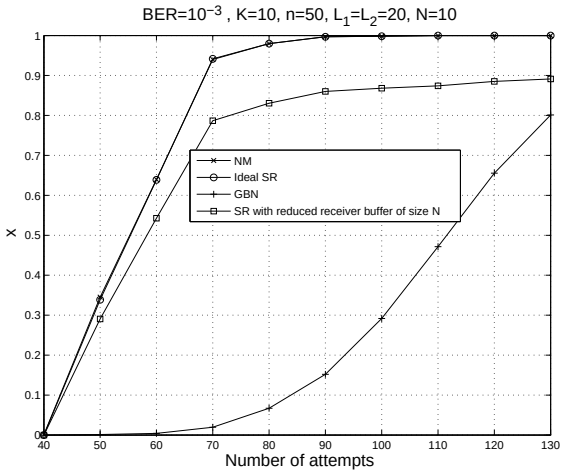

Figure 4.15: Performance comparison using the same buffer size.

4.4 Throughput analysis

In this section we describe analytically the operation of our scheme by calculating its throughput efficiency η. Recall that the throughput efficiency is defined as the ratio of the average number of information bits successfully accepted by the receiver per unit time to the total number of bits that could be transmitted per unit time. The throughput of continuous transmission ARQ techniques (GBN and SR) can be written as (109)(110)(115)

$$\eta = \frac{n_d}{n} \cdot \frac{1}{\overline{M}} \tag{4.1}$$

where n_d is the packet size without the CRC (i.e. the CRC consists of $(n - n_d)$ bits) and $\overline{M}$ is the average number of transmissions for a packet to be successfully transmitted (to the whole group). The throughput is only meaningful when the stream is very long, theoretically infinite. It expresses the rate loss or reduction due to the addition of redundancy bits (the $\frac{n_d}{n}$ factor) and to retransmissions (the $\frac{1}{\overline{M}}$ factor).

As we saw in the previous section, the total rate (I stream + A stream) is the same for the SR scheme and for the proposed scheme. On the other hand, if we consider only the "I" flow, then the throughput of the SR scheme is higher than that of the scheme we propose (in which their would be idle slots due to "A" packet transmissions). Apart from the idle slots, our scheme performs exactly in the same way as the SR scheme.

Let us define the following events, all relative to the new scheme

I : The transmitted packet is of type I.

A : The transmitted packet is of type A.

W : The packet is transmitted in an observation window.

$\bar{W}$: The packet is not transmitted in an observation window.

$ACKed$: After the transmission of the packet, it is acknowledged by all the receivers which have not aknowledged it before the current transmission.

$NACKed$: After the transmission of the packet, there will still be receivers (at least one) which have not yet acknowledged the packet.

and let $P(I)$, $P(A)$, $P(W)$, $P(\bar{W})$, $P(ACKed)$ and $P(NACKed)$ be the probabilities of theses events. Obviously, $P(I) + P(A) = 1$, $P(W) + P(\bar{W}) = 1$ and $P(ACKed) +$

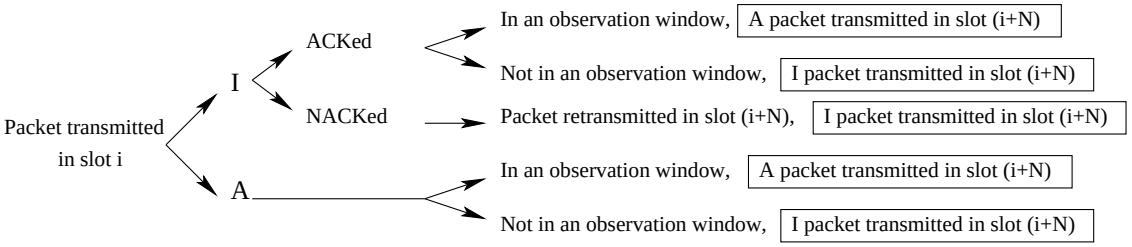

Figure 4.16: Link between (the nature of) the packet transmitted in slot i and (the nature of) the packet transmitted in slot $(i+N)$.

$P(NACKed) = 1$.

The new technique can be seen as a SR interleaved by "A" packet transmissions, hence its throughput can be written as

$$\eta_{NS} = \eta_{SR} \times P(I) \tag{4.2}$$

We must then derive an expression for $P(I)$.

The packet transmitted in a given slot depends entirely on the packet transmitted N slots earlier. In other words, the transmission of a packet in a given slot determines entirely the packet transmitted N slots later. Actually, according to whether the packet transmitted in slot i is of type A or I, to whether it is ACKed or NACKed and to whether it is transmitted in an observation window or not, there are 5 possible scenarios (Fig. 4.16)

1) If the packet is of type I, is ACKed and is transmitted in an observation window, the packet transmitted in slot $(i+N)$ is of type A.

2) If the packet is of type I, is ACKed and is not transmitted in an observation window, the packet transmitted in slot $(i+N)$ is of type I.

3) If the packet is of type I and is NACKed, the same packet is retransmitted in slot $(i+N)$ (whether in observation window or not).

4) If the packet is of type A and is transmitted in an observation window, the packet transmitted in slot $(i+N)$ is of type A.

5) If the packet is of type A and is not transmitted in an observation window, the packet transmitted in slot $(i+N)$ is of type I.

In order to calculate the throughput , we must assume that L_1 and L_2 are very large (theoretically infinite) and that their ratio is finite. In this case, the process is stationary, i.e. the probability that the packet transmitted in a given slot is an "I" packet is the same whatever the position of the slot.

The probability that the packet transmitted in a given slot is of type I is the sum of the probabilities of the paths that lead to the transmission of an "I" packet (see Fig. 4.16), i.e.

$$P(I) = P(I, ACKed, \bar{W}) + P(I, NACKed) + P(A, \bar{W}) \tag{4.3}$$

$P(I, ACKed, \bar{W})$ is given by

$$\begin{aligned} P(I, ACKed, \bar{W}) &= P(I)P(ACKed|I)P(\bar{W}|I, ACKed) \\ &= P(I)P(ACKed|I)(1 - P(W|I, ACKed)) \\ &= P(I)P(ACKed|I)P(1 - P(W|I)) \end{aligned} \tag{4.4}$$

for the fact that the packet is ACKed is independent of the fact that it is transmitted in an observation window, it depends on the channel quality, the packet size and the number of receivers.

Similarly,

$$\begin{aligned} P(I, NACKed) &= P(I)P(NACKed|I) \\ &= P(I)(1 - P(ACKed|I)) \end{aligned} \tag{4.5}$$

and

$$\begin{aligned} P(A, \bar{W}) &= P(A)P(\bar{W}|A) \\ &= (1 - P(I))(1 - P(W|A)) \end{aligned} \tag{4.6}$$

Under the assumption of the independence of events W and I(or A), we can write

$$P(W|I) = P(W)$$

$$P(W|A) = P(W)$$

This assumption tends to be satisfied for L_1 very large.
Thus, Equations (4.4) and (4.6) become

$$P(I, ACKed, \bar{W}) = P(I)P(ACKed|I)(1 - P(W)) \tag{4.7}$$

$$P(A, \bar{W}) = (1 - P(I))(1 - P(W)) \tag{4.8}$$

Substituting equations (4.5), (4.7) and (4.8) in (4.3), we obtain the following equation

$$P(I) = P(I)(1 - P(W))P(ACKed|I) + P(I)(1 - P(ACKed|I)) + (1 - P(I))(1 - P(W))$$

which can be put in the form

$$(1 - P(ACKed|I))P(I)P(W) - P(I) - P(W) + 1 = 0 \tag{4.9}$$

To express $P(I)$, we need to express $P(W)$ and $P(ACKed|I)$.
An "A" packet is transmitted in an observation window if and only if a packet transmitted less than $(N-1)$ slots earlier is of type I, is NACKed and is not transmitted in an observation window,

$$\begin{aligned} P(W) &= (N-1)P(I, NACKed, \bar{W}) \\ &= (N-1)P(I)P(NACKed|I)P(\bar{W}|I, NACKed) \\ &= (N-1)P(I)P(NACKed|I)P(\bar{W}|I) \\ &= (N-1)P(I)(1 - P(ACKed|I))(1 - P(W|I)) \end{aligned} \tag{4.10}$$

Still under the assumption of the independence of events W and I, equation (4.10) becomes

$$P(W) = (N-1)(1 - P(ACKed|I))P(I)(1 - P(W)) \tag{4.11}$$

which can be expressed in the form

$$P(W) = \frac{(N-1)(1-P(ACKed|I))P(I)}{1+(N-1)(1-P(ACKed|I))P(I)} \tag{4.12}$$

which gives the expression of $P(W)$ as a function of $P(I)$.

Substituting equation (4.12) in equation (4.9), we obtain a quadratic equation in the variable $P(I)$

$$\left[(N-1)P(ACKed|I)(1-P(ACKed|I))\right] P(I)^2 + P(I) - 1 = 0 \tag{4.13}$$

So, we must solve the following equation in the variable z

$$\left[(N-1)P(ACKed|I)(1-P(ACKed|I))\right] z^2 + z - 1 = 0 \tag{4.14}$$

the determinant of which is

$$\Delta = 1 + 4(N-1)P(ACKed|I)(1-P(ACKed|I))$$

Given that

$$\begin{aligned} (N-1) &> 0 \\ P(ACKed|I) &> 0 \\ 1-P(ACKed|I) &> 0 \end{aligned} \tag{4.15}$$

we have $\Delta > 1 > 0$. The equation has two real solutions z_1 and z_2 with

$$z_1 = \frac{-1-\sqrt{\Delta}}{2(N-1)P(ACKed|I)(1-P(ACKed|I))}$$

and

$$z_2 = \frac{-1+\sqrt{\Delta}}{2(N-1)P(ACKed|I)(1-P(ACKed|I))}$$

$\Delta > 1$, so $\sqrt{\Delta} > 1$ and therefore $z_1 < 0$ et $z_2 > 0$. $P(I)$ is a solution of equation (4.14) and satisfies $0 < P(I) < 1$. Given that z_2 is positive and z_1 is negative, z_1 is rejected. What is left to do is to check that z_2 is less than 1, which is equivalent to

$$-1+\sqrt{1+4(N-1)P(ACKed|I)(1-P(ACKed|I)} < 2(N-1)P(ACKed|I)(1-P(ACKed|I)$$

or to

$$0 < 4(N-1)^2 P(ACKed|I)^2 (1-P(ACKed|I))^2$$

which is satisfied.

It follows that the analytical expression of $P(I)$ established under the assumption of the independence of events W and I (or A) is

$$P(I) = \frac{-1+\sqrt{1+4(N-1)P(ACKed|I)(1-P(ACKed|I))}}{2(N-1)P(ACKed|I)(1-P(ACKed|I))} \tag{4.16}$$

We still have to derive an analytical expression for $P(ACKed|I)$ which is the probability that the packet transmitted *in a given slot* is ACKed by the whole group given that it's a packet of type I. It is the probability that the receivers which have not acknowledged the packet during the previous attempts do so during the current one. Knowing that it could be the 1^{st}, the 2^{nd}, ...the m^{th} attempt ..., $P(ACKed|I)$ can be written as

$$P(ACKed|I) = \sum_{m} P_0(m) \times P_1(m) \tag{4.17}$$

where $P_0(m)$ is the probability that the receivers who have not acknowledged the packet in the previous attempts do so at the current attempt given it's the m^{th} one, and $P_1(m)$ is the probability that the current attempt is the m^{th} one.

To derive the expression of $P(ACKed|I)$, we have to derive the expressions of $P_0(m)$ and $P_1(m)$.

We first define the following probabilities which will be useful for the derivation of $P_0(m)$ and $P_1(m)$ expressions

P_s : Probability of a successful (unicast) transmission.

P_u : Probability of an unsuccessful (unicast) transmission.

These two probabilities can be written as

$$P_s = (1-\varepsilon)^n \tag{4.18}$$

and

$$P_u = 1 - (1-\varepsilon)^n \tag{4.19}$$

Let us now get back to $P_0(m)$, which according to the definition above is the probability that the receivers which have not yet acknowledged the packet at the $(m-1)^{\text{st}}$ attempt do so at the m^{th} one. Let k be the number of the receivers which have not yet acknowledged the packet at the $(m-1)^{\text{st}}$ attempt.
The total number of receivers is K, so there are k receivers which have not received the packet correctly ($(m-1)$ times out of $(m-1)$ attempts) and $(K-k)$ which did receive it correctly (at least one time out of the $(m-1)$ attempts).

For a given k, the probability that the receivers which have not acknowledged the transmitted "I" packet up to the $(m-1)^{\text{st}}$ attempt do so at the m^{th} one is the probability that there are

1) k receivers which have not received the packet correctly $(m-1)$ times out of $(m-1)$ attempts, that is $(P_u^{m-1})^k$.

2) $(K-k)$ receivers which have received the packet correctly at least once in $(m-1)$ attempts, the probability of which is denoted by P'.

3) k successful unicast transmissions (corresponding to the k receivers left) at the m^{th} attempt, that is P_s^k.

Let us derive the expression of the second event, which can be written as

$$P' = P''^{(K-k)} \tag{4.20}$$

where P'' is the probability that a given receiver receives the packet correctly at least once in $(m-1)$ attempts.

Let P''' be the probability of the complementary event

$$P'' = 1 - P''' \tag{4.21}$$

P''' is the probability that a given receiver doesn't receive the packet correctly $(m-1)$ times in $(m-1)$ attempts, which we can write

$$P''' = P_u^{m-1} \tag{4.22}$$

It follows, from (4.20), (4.21) and (4.22) that P' is given by

$$P' = (1 - P_{UT}^{m-1})^{K-k}$$

Now, since k varies between 1 and K we have

$$\begin{aligned} P_0(m) &= \sum_{k=1}^{K} P_s^k \times (1 - P_u^{m-1})^{K-k} \times (P_u^{m-1})^k \\ &= \sum_{k=1}^{K} (P_u^{m-1}(1 - P_u))^k \times (1 - P_u^{m-1})^{K-k} \\ &= \sum_{k=1}^{K} a_1^k \times b_1^{K-k} \end{aligned}$$

where

$$\begin{cases} a_1 = P_u^{m-1}(1 - P_u) \\ b_1 = (1 - P_u^{m-1}) \end{cases}$$

Now

$$\begin{aligned} P_0(m) &= a_1 \sum_{k=1}^{K} a_1^{k-1} \times b_1^{K-k} \\ &= a_1 \sum_{k=0}^{K-1} a_1^k \times b_1^{K-1-k} \\ &= \frac{a_1}{a_1 - b_1}(a_1^K - b_1^K) \end{aligned}$$

for

$$\begin{aligned} a^K - b^K &= (a-b)(a^{K-1} + a^{K-2}b + \cdots + ab^{K-2} + b^{K-1}) \\ &= (a-b)\sum_{k=0}^{K-1} a^k b^{K-1-k} \end{aligned} \tag{4.23}$$

Replacing a_1 and b_1 by their expressions, we obtain the expression of $P_0(m)$

$$P_0(m) = \frac{P_u^{m-1}(1 - P_u)}{(2P_u^{m-1} - P_u^m - 1)} \left[(P_u^{m-1}(1 - P_u))^K - (1 - P_u^{m-1})^K \right] \tag{4.24}$$

We now derive the expression of $P_1(m)$ which is the probability that the current attempt is the m^{th} one. It's the probability of receiving less than K ACKs up to the $(m-1)^{\text{st}}$

attempt, more precisely the probability of receiving between 0 and $(K-1)$ ACKs in $(m-1)$ attempts. $P_1(m)$ is given by

$$\begin{aligned} P_1(m) &= \sum_{k=0}^{K-1} P(k \text{ ACKs in } (m-1) \text{ attempts}) \\ &= \sum_{k=0}^{K-1} P(\text{in } (m-1) \text{ attempts} : k \text{ receivers send ACKs and } (K-k) \text{ do not}) \\ &= \sum_{k=0}^{K-1} (1-P_u^{m-1})^k \times (P_u^{m-1})^{K-k} \\ &= \sum_{k=0}^{K-1} a_2^k \times b_2^{K-k} \end{aligned} \tag{4.25}$$

where

$$\begin{cases} a_2 = 1 - P_u^{m-1} \\ b_2 = P_u^{m-1} \end{cases}$$

Now

$$\begin{aligned} P_1(m) &= b_2 \sum_{k=0}^{K-1} a_2^k \times b_2^{K-1-k} \\ &= \frac{b_2}{a_2 - b_2}(a_2^K - b_2^K) \end{aligned} \tag{4.26}$$

Equation (4.26) is obtained using (4.23).
As a result, we have

$$P_1(m) = \frac{P_u^{m-1}}{(1-2P_u^{m-1})} \left[(1-P_u^{m-1})^K - (P_u^{m-1})^K\right] \tag{4.27}$$

Replacing $P_0(m)$ and $P_1(m)$ by their analytical expressions, we obtain the analytical expression of $P(ACKed|I)$ as a function of the different system parameters

$$\begin{aligned} P(ACKed|I) = \sum_m \frac{(1-P_u)P_u^{2(m-1)}}{(1-2P_u^{m-1})(2P_u^{m-1} - P_u^m - 1)} \times \\ \left[(P_u^{m-1}(1-P_u))^K - (1-P_u^{m-1})^K\right] \times \left[(1-P_u^{m-1})^K - (P_u^{m-1})^K\right] \end{aligned} \tag{4.28}$$

Fig. 4.17 shows the ratio of the new scheme throughput (for a very long "I" packet transmission) to that of the SR scheme as a function of a unicast channel bit error rate. For low bit error rates, there is a very good agreement between analytical results and simulation results. For higher bit error rates, a small difference appears first, and then it becomes more important for high bit error rates. The difference is due to the fact that the simulation is run with $L_1 = 1000$ ($L_1 = \infty$ ideally) and to the fact that the independence assumption is no longer quite satisfied.

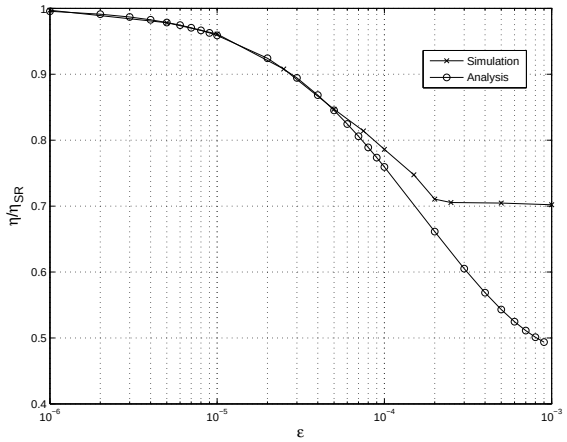

Figure 4.17: Throughput of the new scheme relative to the throughput of the SR.

The fact that the throughput of the proposed scheme is reduced with respect to that of the SR scheme is due to the fact that the new scheme transmits two streams at the same time while the SR ARQ technique is dedicated to the transmission of one stream only. For L_1 finite, the SR scheme finishes the transmission of the first stream faster than the new scheme, which finishes faster than the GBN scheme. This was already illustrated when the delay was studied in section 4.4.

4.5 Application to Wireless Systems

As explained in chapter 1, when streaming a scalable video through a cellular network towards one or several receivers, the video data originate from the Internet, more precisely, the scalably coded video data are stored at an Internet server. The receiving terminals represent the other end of the transmission system. The architecture of the network used lies between these two ends (see Fig.4.18).

Fig. 4.19 shows the whole protocol stack that may be used to transport video data over a 2G(GPRS/EDGE) or 3G(UMTS/HSDPA) network. Both TCP and UDP (in conjunction with RTP) can be used to transport video data over networks that use the Internet Protocol (IP). In case TCP is used, packets follow path 1. In case UDP is used (in conjunction with RTP), packets follow path 2.

Recall that TCP provides a reliable transmission mechanism between the receiving terminal and the server at the transport level. It is actually the highest operating reliable mechanism. Fig. 4.20 shows all the different possible retransmission levels for a unicast TCP connection over 2G, 3G and WiMAX cellular networks. Link level retransmissions are of SR type (see Appendix B) whereas TCP retransmissions are of GBN type.

All these retransmission mechanisms can be used to apply the schemes we studied in the previous sections, but they are meant for point-to-point communications (1 receiver) and applications of 1 stream. Adaptations to the case of point-to-multipoint communications and scalability are then necessary.

To adapt these mechanisms to the case of point-to-multipoint communications, TCP has been replaced by the NACK-Oriented Reliable Multicast (NORM) Protocol (18), a Reliable Multicast Transport (RMT) IETF standard for point-to-multipoint communications. As for the Link Level retransmissions, they require an Algorithmic (scheduling)

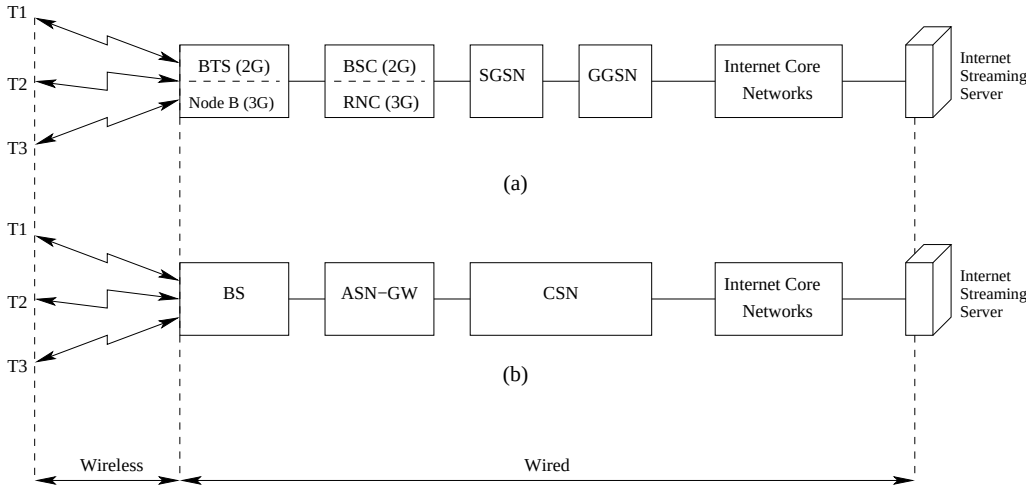

Figure 4.18: Architecture of a point-to-multipoint server-mobile terminals connection over (a) 2G and 3G networks. (b) WiMAX.

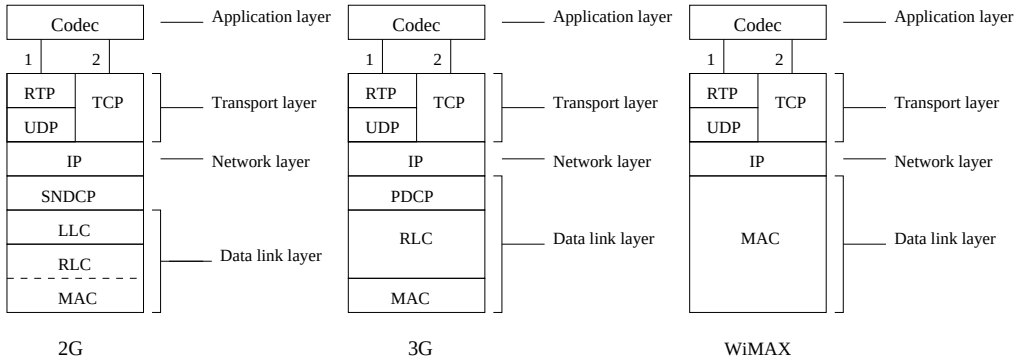

Figure 4.19: Protocol stack for the transport of video data over 2G, 3G and WiMAX cellular networks.

update since the transmitter must take into account the feedbacks from all receivers and must manage an ACK List (memorizing which receivers acknowledged the packets and which did not).

To adapt the mechanisms to the existence of 2 streams, the transmitter needs to manage two different buffers, one for the packets of each stream. The transmitter also needs to schedule the transmission of the packets according to the transmission rules of the scheme in question, which is also an Algorithmic update.

A transmission mechanism was proposed in (16) for streaming scalable video data over wired/wireless Internet. This scheme uses TCP for the transmission of "I" packets and UDP for the transmission of "A" packets so that the base-layer datagrams are transported reliably and the enhancement-layer datagrams are transported in a Best Effort way. "A" packet and "I" packets would then follow path 1 and path 2 of Fig. 4.19, respectively. This scheme is an application of the GBN scheme at the transport level since TCP retransmissions are of GBN type.

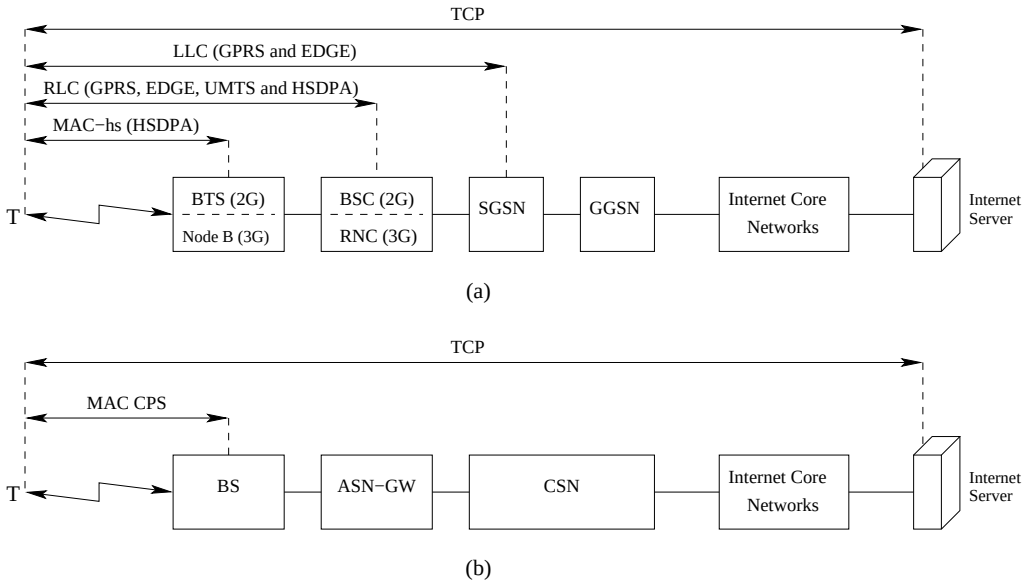

Figure 4.20: Possible retransmission levels in a TCP connection over (a) 2G and 3G networks. (b) WiMAX.

On the other hand, TCP retransmissions are basically used to recover from packet losses in the Internet due to congestion. These retransmissions also recover from losses due to transmission errors on the wireless link. Level 2 retransmissions deal exclusively with losses due transmission errors on the wireless link and are more efficient than TCP retransmissions from this point of view, due to their proximity to the terminal. For example, in 3G, The TCP RTT (server-Terminal RTT) is on the order of 400 ms, the RLC RTT (RNC-Terminal) on the order of 60 ms, and the MAC-hs RTT on the order of 11 ms. This comprises the signal duration, the processing time and the propagation time. If, for instance, in the case of a non-scalable unicast streaming application, 5s of buffering are necessary before the video playback starts, each TCP segment can be retransmitted at most 12 times, and RLC PDU 83 times and a MAC-hs PDU 454 times.

If the network congestion is not negligible, TCP must be used for the transmission of the basic stream, with or without level 2 retransmissions, but preferably with level 2 retransmissions since the latter improve the throughput. If, in contrast, the congestion is low, it is preferable to use the RTP/UDP combination with its lower delay and complexity and use link level retransmissions to deal with transmission errors on the wireless link.

Also, at the link level, both in-sequence and out-of-sequence delivery are generally possible. But when using TCP, studies of the interaction between TCP and link layer retransmissions (19)(20)(21) showed that in-sequence delivery of link level SDUs results in better performance at the TCP layer because receiving out-of-sequence packets (TCP segments) may cause a triple duplicate phenomenon (misinterpreted as congestion by TCP), which results in packet retransmissions and congestion window downsizing at the TCP layer.

4.6 Conclusion

In this chapter, we considered the transmission of a two-level scalable video sequence towards several clients. The number of packets to be transmitted was finite and known. Schemes using the basic GBN and SR ARQ techniques were studied. We also proposed and studied a new scheme. Computer simulations were used to evaluate the performance of the three schemes. The new scheme reduces the buffering requirement at the receiver end. Numerical results show that when all receivers are interested in the full quality video, the new scheme is optimal in data delivery speed. The results also show that the increase in the number of receivers doesn't much affect the system performance while it considerably improves the bandwith utilization efficiency. An analytical expression of the new scheme's throughput relative to that of the SR scheme was also derived, only the transmission of "I" packets was considered then. Analytical results were in good agreement with simulation results. Finally, the different schemes were shown to be applicable to 2G, 3G and WiMAX systems, with a few adaptations.

Chapter 5

On the use of Automatic Repeat Request in Multicast/Broadcast services

5.1 Introduction

In multimedia applications, the data to be transmitted are compressed by a source encoder at the application layer. Therefore, one way of reducing the bandwidth used is to increase the compression rate in order to reduce the amount of data to be transmitted and hence the bandwidth necessary to transmit them.

Another efficient way to reduce the necessary bandwidth would be by gathering all the customers asking for the same multimedia content in one group of receivers to which the data are conveyed using the same channel. The bandwidth is then reduced by a factor equal to the total number of receivers. This approach is valid in case multiple customers per cell are interested in the same content. It is particularly useful for multimedia applications like video file transfer or video streaming applications because many receivers may be interested in the same video content but also because video applications are characterized by large bandwidth requirements, which can be hundreds of times higher than the bandwidth required for voice services.

On the other hand, retransmission of erroneous packets is very suitable to recover from missing data and improve the provided quality.

Below, if a PMP communication uses ARQ retransmissions, it will be said to be in the Acknowledged Mode (AM), otherwise (if it does not use ARQ) it will be said to be in the Unacknowledged Mode (UM). Obviously, the Acknowledged Mode makes the communication reliable and provides a much better quality than the Unacknowledged Mode. However, under this mode, when the number of receivers increases, the throughput of the system decreases and may go below the limit required by the application.

Basic ARQ schemes were proposed and studied in (104)-(118). These works assumed that all receivers were in the same radio conditions (experienced the same BER). Besides, the focus was put on the evaluation of the performance of the studied scheme (generally expressed by the throughput) as a function of the channel quality (generally expressed by the BER) and the number of receivers. In this work, the more practical situation in which the different receivers may experience different BERs is considered. The threshold number above which the number of receivers is considered too large to use the Acknowledged Mode

is determined. This number represents the capacity of the system under some given application throughput constraint and for a given configuration of the radio conditions (the different BERs). This is realized at first in the presence of one frequency channel, and then is generalized to the case when several frequency channels are available.

The rest of this chapter is organized as follows. Section 5.2 describes the PMP system and analyses its throughput. Section 5.3 analyzes the behavior of point-to-multipoint communications when the different receivers are in different channel conditions. Section 5.4 focuses on the definition of the algorithms according to which PMP systems should perform under a throughput constraint and when one channel is available for the transmission. In section 5.5, these systems are generalized to the case when several channels are available. Section 5.6 discusses the application of these systems to Multicast/Broadcast services supported by the 3GPP and mobile WiMAX. Finally, Section 5.7 concludes the chapter.

5.2 Retransmission procedures and throughput analysis

As in the previous chapter, we consider the communication system consisiting of one transmitter and K receivers. Recall that the (multipath) propagation medium between the transmitter and a particular receiver is a *(forward) unicast channel*. Each Unicast Channel is modelled as a Binary Symmetric channel (BSUC). The unicast channels are assumed to produce independent noise processes and each noise process is assumed to be white, so that the bits transmitted on the BSUC are randomly erroneous.

Note that throughout this chapter, the term "(frequency) channel" denotes a frequency band (a network bandwidth resource), whereas the term "unicast channel (BSUC)" denotes the propagation medium between the transmitter and a particular receiver, and is characterized by the rate at which errors occur on it.

SW and GBN ARQ schemes being inefficient in terms of throughput, only SR is considered in this study. Further, it was shown in (114) that for the SR using the FRG strategy the average number of transmissions grows exponentially, i.e. very fast, with an increase in the number of receivers whereas for the SR using the DRGR strategy the average number of transmissions grows logarithmically, i.e. slowly, with the number of receivers. This makes the DRGR strategy very interesting for Multicast service provision, i.e. if we want to maximize the number of receivers that are provided with data in the Acknowledged Mode. For this reason, only the DRGR strategy is considered in this work. Below, the number of receivers is represented on a logarithmic scale, since the average number of retransmissions grows logarithmically (slowly) with the number of receivers.

Note that the DRGR strategy outperforms the FRG strategy at the expense of a K times larger memory required at the transmitter, which is not really an inconvenient since nowadays, it is no longer a problem to have a 1 Mega Byte memory as was the case two decades ago.

Assume that all packets consist of n_d data bits and an n_h-bit header (see Fig. 5.1). The header consists of an n_s-bit Sequence Number and an n_c-bit CRC. The CRC protects both the data field and the SN field. The size of the packet $n = n_d + n_h = n_d + n_s + n_c$ (in bits) will be assumed constant. This structure is similar to that of a MAC packet, the difference is that MAC packets contain, in addition to the SN and the CRC fields, other MAC header fields and the headers of the upper layers.

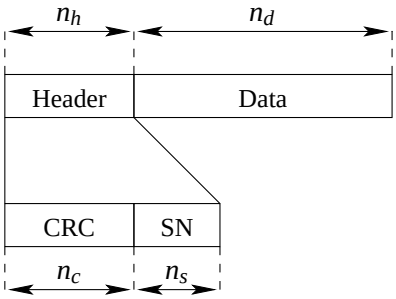

Figure 5.1: Structure of a packet.

If the source encoder at the application layer generates data at a rate R_0 [bits/sec], the necessary rate at the physical layer is $R_0 \times n/n_d > R_0$ [bits/sec] [1]. We assume that a frequency channel provides exactly the bandwidth necessary for transmitting binary data at the rate $R_0 \times n/n_d$ [bits/sec]. This ensures that, when retransmissions are not used, the throughput of the system is equal to the rate at which the source encoder generates data, i.e. R_0.

Let R be the throughput of the system at the application layer. The throughput efficiency of the system is given by

$$\eta = \frac{R}{R_0 \times \frac{n}{n_d}} = \frac{R}{R_0}\frac{n_d}{n} \tag{5.1}$$

If the size of the sequence range 2^{n_s} is large enough, the throughput efficiency can be written as (110)(111) :

$$\eta = \frac{n_d}{n}\frac{1}{\overline{M}} \tag{5.2}$$

where M is the number of transmissions a packet requires to be acknowledged by all receivers and $\overline{M}$ is the mean of M, i.e. the average number of attempts per packet.

In (5.2), the factor n_d/n accounts for the additional bandwidth required due to the overhead whereas the factor $1/\overline{M}$ accounts for the additional time this bandwidth is used due to retransmissions.

Now, combining (5.1) and (5.2) we obtain

$$\frac{R}{R_0} = \frac{1}{\overline{M}} \tag{5.3}$$

Under very strong real time constraints such as live TV, no delay is allowed and the required throughput R is equal to R_0, which does not allow for retransmissions. On the other hand, in streaming applications, the decoding starts after a given delay (which can be tuned at the receiver). This delay allows the application to run with a lower throughput ($\alpha R_0 \leq R \leq R_0$ with $\alpha < 1$). The higher the delay, the lower the required throughput. The condition $\alpha R_0 \leq R \leq R_0$ implies $1 \leq \overline{M} \leq 1/\alpha$. Hence the delay allows for a limited use of retransmissions. This limit will be used later in this chapter to find the limit on the number of receivers that can be accepted by the system in the Acknowledged Mode.

1. Note that the overhead due to the redundancy added by the channel encoder at the physical layer should also be taken into account

Before we proceed, let us point out that all the results presented and discussed below were obtained with a packet size of $n = 1024$ bits (128 bytes) including a 16-bit SN and a 32-bit CRC ($n_s = 16$ bits, $n_c = 32$ bits and hence $n_h = 48$ bits).

Let us first assume that the K BSUCs have the same BER ε, i.e. the different receivers are in the same radio conditions. This assumption may be considered true in satellite systems but not in terrestrial systems (cellular or not). However, it will be a first step towards the more general and more realistic case when the different receivers are in different radio conditions (depending on the distance to the transmitter, the fading, the shadowing ...).

In order for the packet to be correctly received, retransmissions are performed until the packet is correctly received by all receivers (at least once in all the attempts for each receiver). However, in practice, the number of attempts to successfully transmit a packet are always limited to a given number that generally ranges from 1 attempt (no retransmissions at all) to 20 attempts. In this case, retransmissions should go on until the packet is successfully transmitted by all receivers or until the maximum number of attempts is reached, i.e. if the maximum number of attempts is reached, the transmitter marks the packet as successfully transmitted whether the last transmission was successful or not. In this case, the receivers not having correctly received the packet will have to do with an erroneous packet.

For convenience, let us define the following parameters :

P_s : Probability of a successful *unicast* transmission (or attempt).

$P_u = 1 - P_s$: Probability of an unsuccessful *unicast* transmission (or attempt).

M : Number of transmissions of a packet for it to be marked as successfully transmitted.

M_{max} : Maximum number of attempts per packet (maximum number of times a packet can be transmitted).

Note that by unicast transmission we mean point-to-point transmission between the transmitter and a given receiver of the group. M is a random variable that takes its values in the set $\{1, 2, \ldots, M_{max}\}$.

Now, P_s is given by

$$P_s = (1 - \varepsilon)^n \tag{5.4}$$

and

$$P_u = 1 - P_s = 1 - (1 - \varepsilon)^n \tag{5.5}$$

To determine the throughput of the system, we need to determine the average number of transmissions for a packet to be marked as successfully transmitted, i.e. $\overline{M}$. For this purpose, we first need to determine the probability mass function of random variable M, as a function of P_s and P_u.

The probability that $M = m$ may be expressed as

$$P(M = m) = P(M \leq m) - P(M \leq m - 1) \tag{5.6}$$

Where $P(M \leq m)$ is the cumulative probability, it represents the probability that the packet is acknowledged by all K receivers within m attempts or fewer. It is then the probability that the packet is acknowledged at least once by each receiver in m attempts.

Obviously

$$P(M \leq 0) = 0 \tag{5.7}$$

and

$$P(M \leq M_{max}) = 1 \tag{5.8}$$

The last equality follows from the fact that after M_{max} attempts, the packet is marked as successfully received (by the transmitter) regardless of the result of the transmission.

For all m in $\{1, \ldots, M_{max} - 1\}$, $P(M \leq m)$ is the probability that the packet is successfully transmitted to each receiver at least once in m attempts.

The probability that a packet is successfully transmitted to one receiver at least once in m attempts is the complementary of the probability that a packet is not even once successfully transmitted to the receiver in m attempts. Given that the BSUCs are assumed independent, $P(M \leq m)$ can be expressed as follows

$$P(M \leq m) = (1 - {P_u}^m)^K, \quad \forall m \in \{1, \ldots, M_{max} - 1\} \tag{5.9}$$

Note that the above formula also holds for $m = 0$ (since $(1 - {P_u}^m)^K = 0$ when $m = 0$). Hence we write

$$P(M \leq m) = \begin{cases} (1 - {P_u}^m)^K & , \forall m \in \{0, 1, \ldots, M_{max} - 1\} \\ 1 & , m = M_{max} \end{cases} \tag{5.10}$$

Combining (5.6) and (5.10) we obtain the probability mass function of random variable M

$$P(M = m) = \begin{cases} (1 - {P_u}^m)^K - (1 - {P_u}^{m-1})^K & , \forall m \in \{1, \ldots, M_{max} - 1\} \\ 1 - (1 - {P_u}^{M_{max}-1})^K & , m = M_{max} \end{cases} \tag{5.11}$$

The average number of transmissions necessary for a packet to be marked as successfully transmitted is the expectation value of M

$$\begin{aligned} \overline{M} &= E[M] \\ &= \sum_{m=1}^{M_{max}} m P(M = m) \\ &= \sum_{m=1}^{M_{max}-1} m \left[(1 - {P_u}^m)^K - (1 - {P_u}^{m-1})^K\right] + M_{max} \times \left[1 - (1 - {P_u}^{M_{max}-1})^K\right] \end{aligned}$$

Fig. 5.2 plots the throughput efficiency of the system as a function of the maximum number of attempts per packet (M_{max}) and for various values of the BER. Clearly, the throughput decreases as M_{max} increases, which is normal since more retransmissions are allowed with larger values of M_{max}. Also, the throughput decreases when the channel quality degrades (ε increases), which makes perfect sense since more errors on the channel cause more retransmissions and therefore a lower throughput. The most important

thing to note is that beyond a threshold value M_0 of M_{max}, the curve becomes flat and the throughput does not change anymore. This is due to the fact that $P(M \leq m)$ is a decreasing function of m, the more retransmissions are performed the higher is the probability that the m^{th} transmission is successful and the lower the probability that more retransmissions are necessary. Mathematically, the probability that more retransmissions are necessary after M_{max} attempts goes to zero as M_{max} goes to infinity, but numerically, this probability goes to zero when M_{max} reaches the threshold value M_0 as shown in Fig. 5.2.

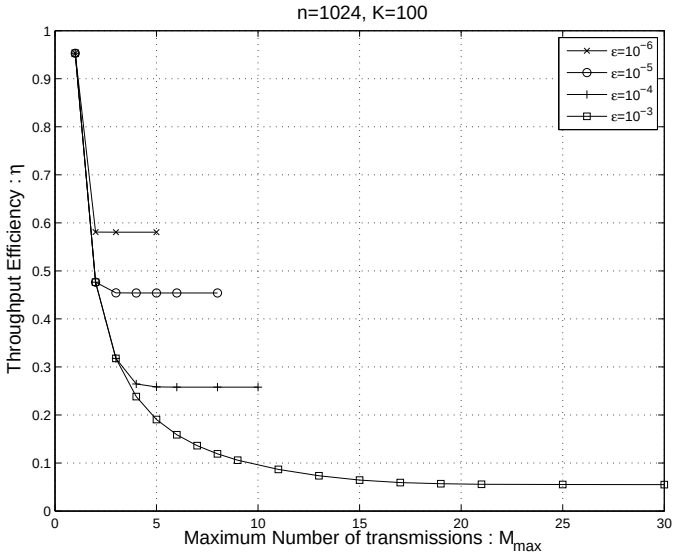

Figure 5.2: Effect of the maximum number of retransmissions on the throughput.

In order to achieve perfect reliable communication, there should be no limit on the number of retransmissions ($M_{max} = \infty$), a packet should be retransmitted as many times as necessary for it to be correctly received, but as shown above, the performance of the system converges when M_{max} is greater than or equal to M_0, since the probability that a packet is correctly received within M_0 attempts or lower goes to 1. Thus, we can achieve a *quasi-reliable* transmission if M_{max} is chosen to be greater than or equal to M_0. As Fig. 5.2 illustrates, the threshold value M_0 depends on the BER. The higher the BER, the higher M_0. For instance, $M_0 = 3$ when $\varepsilon = 10^{-5}$, $M_0 = 5$ when $\varepsilon = 10^{-4}$ and $M_0 = 20$ when $\varepsilon = 10^{-3}$. The last value is high due to the high and unusual corresponding BER. This type of BER is very rarely encountered in practice. For BERs more encountered in practice, $M_0 = 10$ is a high enough value to achieve quasi-reliable communication.

Consider now that the K BSUCs do not have the same BER. More precisely, let us assume that the MRG (MultiReceiver Group) of K receivers can be split into G smaller groups of receivers, in each group, the receivers experience the same BER, i.e. in group 1 the BSUCs have BER ε_1, in group 2 the BSUCs have BER ε_2, and so on. Let ε_g and K_g be the BER of and the number of receivers in group g. Obviously

$$\sum_{g=1}^{G} K_g = K \tag{5.12}$$

In the sequel, (K_g, ε_g) denotes a group of K_g receivers experiencing a BER ε_g.
As previously, we define $P_{s,g}$ and $P_{u,g}$ as follows

$P_{s,g}$: Probability of a successful *unicast* transmission in *group* g .

$P_{u,g}$: Probability of an unsuccessful *unicast* transmission in *group* g .

Note that by unicast transmission in group g we mean a point-to-point transmission between the transmitter and one receiver of group g.
Now, $P_{s,g}$ is given by

$$P_{s,g} = (1 - \varepsilon_g)^n \tag{5.13}$$

and

$$P_{u,g} = 1 - P_{s,g} = 1 - (1 - \varepsilon_g)^n \tag{5.14}$$

M still takes its values in the set $\{1, 2, \ldots, M_{max}\}$ but its statistics has changed as compared to the case when all unicast channels were similar.
Recall that $P(M \leq m)$ is the probability that the packet is marked as successfully transmitted within m attempts or fewer. In this case, and for all m in $\{0, 1, \ldots, M_{max}-1\}$, it is the probability that the packet is successfully transmitted to each receiver of group 1, and to each receiver of group 2, ..., and to each receiver of group G, at least once in m attempts.
The probability $P_{0,g}$ that a packet is successfully transmitted to *a given receiver of group* g at least once in m attempts is the complementary of the probability that a packet is not even once successfully transmitted to a given receiver of group g in m attempts

$$P_{0,g} = 1 - P_{u,g}^m, \qquad \forall m \in \{1, \ldots, M_{max} - 1\} \tag{5.15}$$

Now, given that the BSUCs are independent inside each group, the probability P_g that a packet is successfully transmitted to *each receiver of group* g at least once in m attempts is given by

$$P_g = {P_{0,g}}^{K_g} = \left(1 - P_{u,g}^m\right)^{K_g} \tag{5.16}$$

Now, given that the BSUCs are also independent between different groups, the probability that a packet is successfully transmitted to *each receiver of each group* at least once in m attempts is given by

$$\begin{aligned} P(M \leq m) &= \prod_{g=1}^{G} P_g \\ &= \prod_{g=1}^{G} \left(1 - P_{u,g}^m\right)^{K_g} \end{aligned}$$

The above equation also holds for $m = 0$ (since $\prod_{g=1}^{G}(1 - P_{u,g}^m)^{K_g} = 0$ when $m = 0$).
Hence we can write

$$P(M \leq m) = \begin{cases} \prod_{g=1}^{G}(1-{P_{u,g}}^{m})^{K_g} & ,\forall m \in \{0,1,\ldots,M_{max}-1\} \\ 1 & ,m = M_{max} \end{cases} \tag{5.17}$$

Combining (5.6) and (5.17) we obtain the probability mass function of M in this case

$$P(M = m) = \begin{cases} \prod_{g=1}^{G}(1-{P_{u,g}}^{m})^{K_g} - \prod_{g=1}^{G}(1-{P_{u,g}}^{m-1})^{K_g} & ,\forall m \in \{1,\ldots,M_{max}-1\} \\ 1-(1-{P_{u,g}}^{M_{max}-1})^{K_g} & ,m = M_{max} \end{cases} \tag{5.18}$$

The average number of transmissions for a packet to be marked as successfully transmitted is

$$\begin{aligned} \overline{M} &= E[M] \\ &= \sum_{m=1}^{M_{max}} mP(M=m) \\ &= \sum_{m=1}^{M_{max}-1} m\left[\prod_{g=1}^{G}(1-{P_{u,g}}^{m})^{K_g} - \prod_{g=1}^{G}(1-{P_{u,g}}^{m-1})^{K_g}\right] \\ &\qquad + M_{max} \times \left[1-\prod_{g=1}^{G}(1-{P_{u,g}}^{M_{max}-1})^{K_g}\right] \end{aligned} \tag{5.19}$$

5.3 Performance analysis

We first investigate the performance of a heterogenous system. The simplest possible heterogenous system consists of two groups ($G = 2$), the receivers of group g experiencing the same BER ε_g ($\forall g \in \{1,2\}$). The total number of receivers K is constant ($K = 20$), as well as the BER of the unicast channels in group 1 ($\varepsilon_1 = 10^{-6}$). The number of receivers in group 2 (K_2) is varied from 0 to $K = 20$ for different values of ε_2 (see Fig. 5.3). As can be seen, the performance of the system seriously deteriorates as the ratio $\varepsilon_2/\varepsilon_1$ increases. The performance also degrades when the number of receivers in bad conditions increases, but the throughput is less sensitive to the number of receivers in bad conditions (K_2) than to how bad these conditions are (ε_2). When a packet is transmitted, it is much more likely that it will be acknowledged by a receiver of group 1 than by a receiver of group 2. More generally, the receivers of group 1 tend to acknowledge a given packet faster than those of group 2. The latter keep provoking retransmissions (because of the high BER on their respective unicast channels) delaying thus the receivers of group 1 which are finished with the current packet and waiting to proceed with the next packet scheduled for transmission. In other words, as the ratio $\varepsilon_2/\varepsilon_1$ increases, the receivers in bad conditions tend to prevail and the transmitter tends to be driven mainly by the feedback from these receivers.

Since the receivers in bad conditions tend to spoil the overall performance, it could be fair, in the presence of two channels, to separate the two groups (map each group to a channel). This is shown in Fig. 5.4 where the performance of the three systems (the two groups separated and the two together) is depicted. This is of benefit to the receivers in good channel conditions (group 1) which are no longer handicapped by the receivers in the bad conditions (group 2).

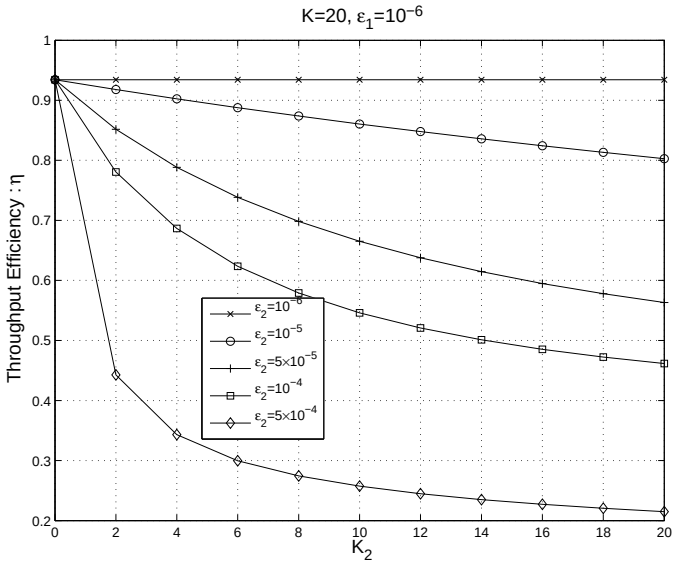

Figure 5.3: Performance of a heterogenous system with $G = 2$, $K = 20$, $\varepsilon_1 = 10^{-6}$ and different values of ε_2.

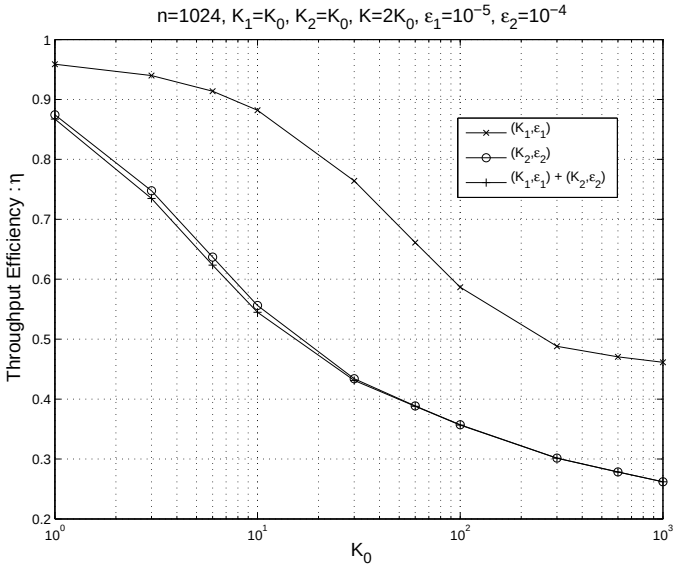

Figure 5.4: Performance of a heterogenous system with $G = 2$, $K_1/K_2 = 1$, $\varepsilon_1 = 5 \times 10^{-5}$ and $\varepsilon_2 = 10^{-4}$, as compared to when the two groups are separated.

Actually, this is also of benefit to the receivers of group 2, but it doesn't show clearly when $K_1 = K_2$ (as in Fig. 5.4). This shows when the ratio K_1/K_2 increases (as in Fig. 5.5 where we have $K_1 = 9K_2$).

This is due to the fact that when the ratio K_1/K_2 is a little higher than 1, it partially compensates for the high $\varepsilon_2/\varepsilon_1$ ratio. As a result, the receivers of group 2 do no longer necessarily prevail as illustrated in Figs. 5.6 and 5.7 (for $\varepsilon_2/\varepsilon_1$ lower than 20). In other

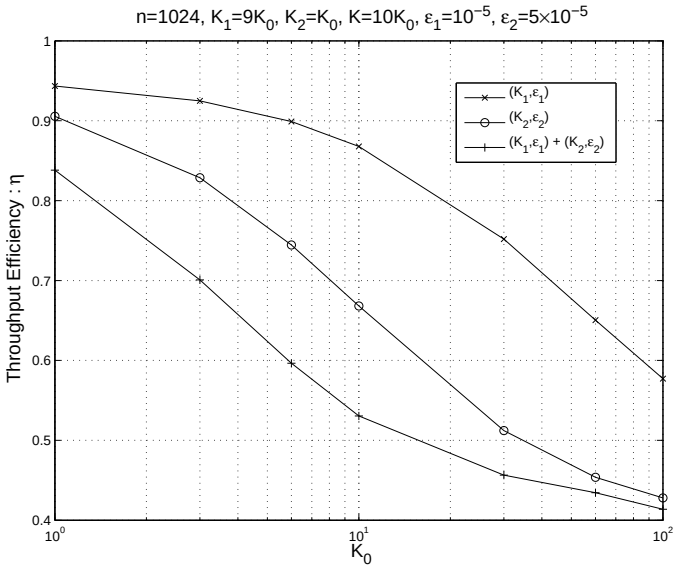

Figure 5.5: Performance of a heterogenous system with $G = 2$, $K_1/K_2 = 9$, $\varepsilon_1 = 10^{-5}$ and $\varepsilon_2 = 10^{-4}$, as compared to when the two groups are separated.

words, when there are more receivers in group 1, there will be more retransmissions provoked by one of the receivers of this group despite their lower BER, delaying sometimes the fewer receivers of group 2 which may have already acknowledged the packet.

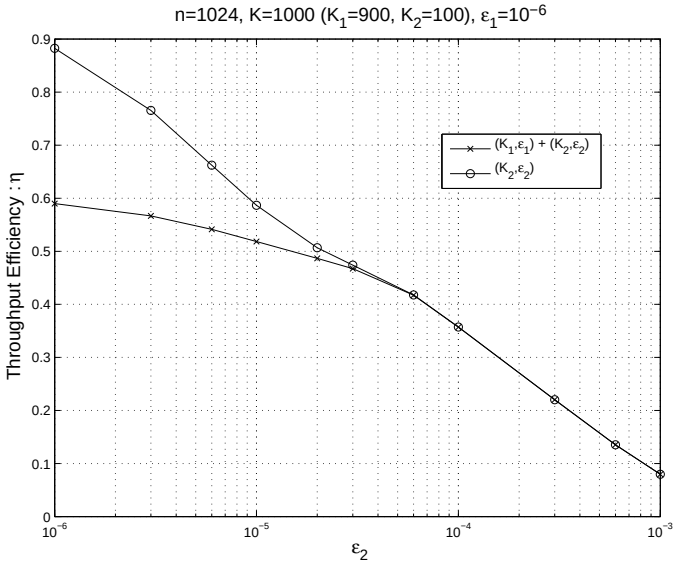

Figure 5.6: Performance of a heterogenous system with $G = 2$, $K_1/K_2 = 9$, $\varepsilon_1 = 10^{-6}$ compared to its group 2 homogenous system, as a function of the ratio $\varepsilon_2/\varepsilon_1$.

By still increasing the number of receivers in group 1 with respect to the number of receivers in group 2, group 1 is outperformed (at least for low values of K_1) by group 2 (see Fig. 5.8). This is due to the fact that when the ratio K_1/K_2 reaches a given threshold,

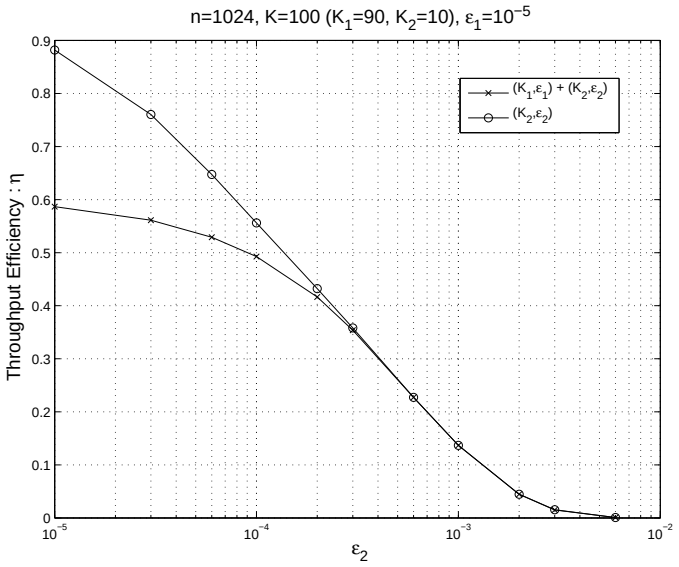

Figure 5.7: Performance of a heterogenous system with $G = 2$, $K_1/K_2 = 9$, $\varepsilon_1 = 10^{-5}$ compared to its group 2 homogenous system, as a function of the ratio $\varepsilon_2/\varepsilon_1$.

the difference in terms of numbers of receivers fully compensates for that of the BERs on the respective unicast channels. Above this threshold, the group of the receivers with the lower BER are outperformed by the group of the receivers with the higher BER, which does not make the approach (which consists of mapping a group of receivers in the same radio conditions on the same channel) fair and shows its limits.

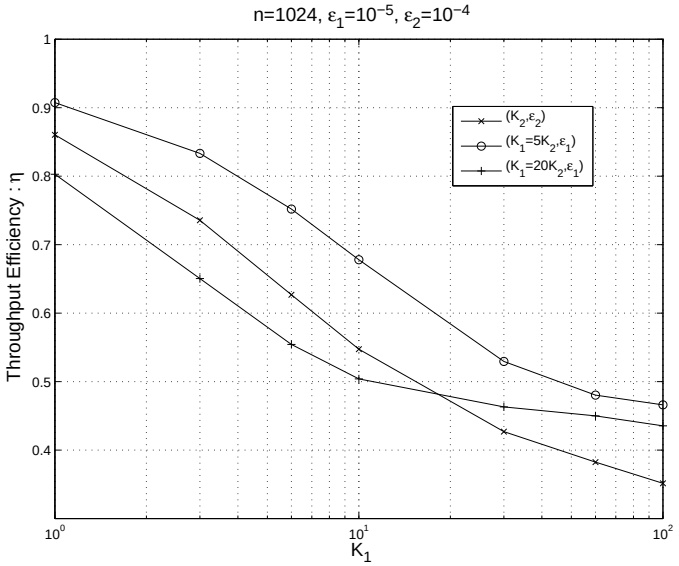

Figure 5.8: Performance of a heterogenous system with $G = 2$, $K_1/K_2 \in \{5, 20\}$, $\varepsilon_1 = 10^{-5}$ and $\varepsilon_2 = 10^{-4}$, as compared to when the two groups are separated.

This approach still remains valid when the number of receivers in the different groups

are more or less the same, but in practice, the different receivers experience different BERs and if enough precision is used, all receivers will have different BERs and it will be impossible to group them, i.e. there will be as many groups as there are receivers ($G = K$, each group consisting of one receiver). Still, we can define ranges of BERs and gather the receivers the BERs of which fall into a given predefined range in the same group, but even by doing so, the numbers of receivers in the different groups will not be the same and there may be groups with many more receivers than others.

In addition, this approach assumes that there are G channels available (i.e. there are as many channels available as there are groups), while in practice, there may be more (or less) channels available.

5.4 Point-to-Multipoint services when one channel is available

Assume now that a single channel is available. The receivers have then to be provided with the service (the data they ask for) through this channel. Regardless of the BER on their unicast channel, the performance of the PMP system degrades as the number of receivers increases. When this number becomes very large, there will be many retransmissions and the throughput will be very poor. When the throughput is too poor for the application, the system will no longer be able to accept receivers. The receiving terminals asking for the service will then be rejected by the system because it will not be able to ensure the required throughput. The channel has then a given capacity in terms of the number of receivers it can accept in the Acknowledged Mode. This capacity, which will be referred to as the *channel Point-to-MultiPoint Acknowledged Mode (PMP AM) capacity*, depends on the BERs of the different receivers, but is a direct consequence of the throughput constraint imposed by the application.

As an alternative to rejecting receivers from the system, the system can switch to the unacknowledged Mode (recall that in this mode, retransmissions are not used) achieving maximum throughput and accepting all receivers. Obviously, the PMP UM capacity is infinite.

In the sequel, a system which uses only the Acknowledged Mode (and hence rejects additional receivers when the channel PMP AM capacity is exceeded) will be referred to as an *AM service system* whereas a system which switches to the Unacknowledged Mode if it cannot accept all receivers (in the Acknowledged Mode mode) will be referred to as a *Best Effort Acknowledged Mode (BEAM) service system*, in the sense that the system uses the Acknowledged Mode as long as the PMP AM capacity is not exceeded, and switches to the Unacknowledged Mode only when this capacity is exceeded.

In the case of an AM service system, it is necessary to define an Algorithm which selects the receivers which will be provided with the service (and those which will not). This Algorithm should be defined in such a way that the system will accept as many receivers as possible, that is in such a way that a maximum of receivers will be provided with the service eventually. In other words, the Algorithm should maximize the channel PMP AM capacity.

Let $\eta_0 = (n_d/n) \cdot \alpha$ be the minimum throughput efficiency required by the application. The channel PMP AM capacity under the constraint $\eta \geq \eta_0$ depends on the Algorithm and on η_0. The higher η_0, the lower the capacity K_{max} (the function $K_{max} = f(\eta_0)$ is decreasing).

Fig. 5.9 shows the flowchart of the system operation. The system proceeds in 2 phases (separated by a dashed line). Phase 1 is a selection phase, the system decides which receivers to accept (and which to reject), these receivers form the MRG which will be provided with the service. Phase 2 is the transmission phase. In phase 1, the system adds and accepts receivers one by one until the throughput goes under the minimum throughput required by the application. The system manages two lists : A list of receivers asking for the service and a list of accepted receivers. When the throughput efficiency goes under the minimum required value η_0 after a receiver has been (momentarily) added, the added receiver is rejected. Note that a receiver is rejected once and for all, (if it is rejected, it will be deleted from the list of receivers asking for the service). This process goes on until there are no more candidate terminals left, the transmitter can then start to provide the selected receivers with the service they asked for (i.e. start the transmission phase).

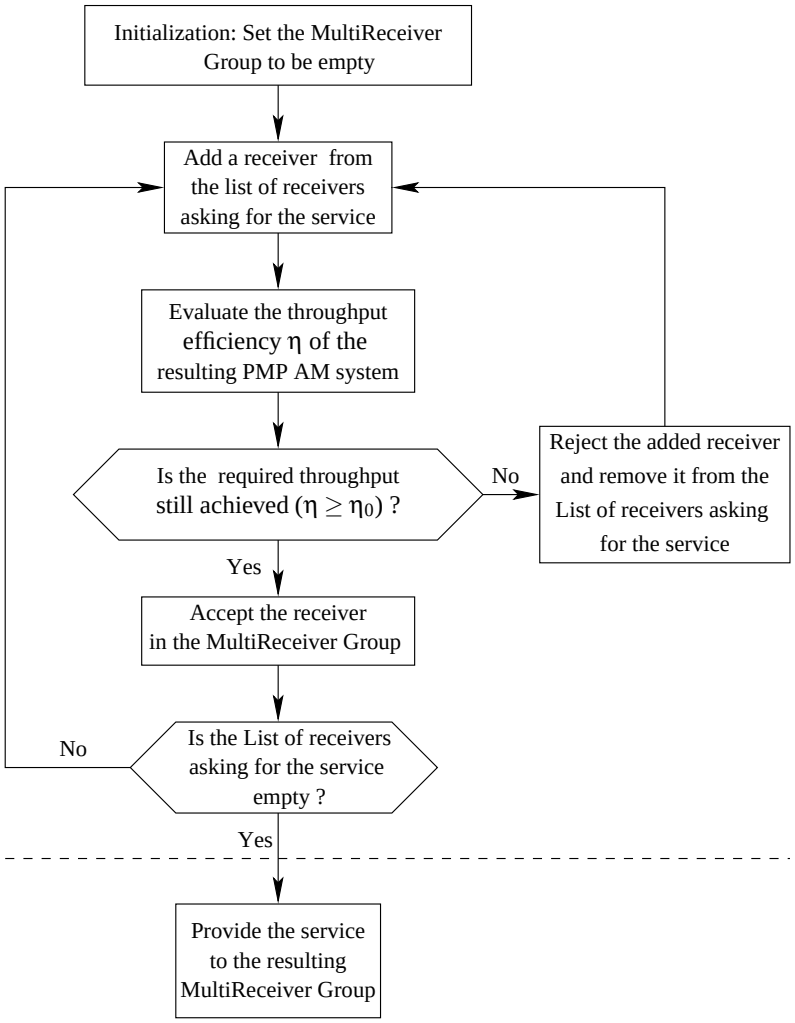

Figure 5.9: General flowchart of an AM service system.

Obviously, the channel PMP AM capacity is reached right before the throughput of the MultiReceiver Group formed (by the so far accepted receivers) goes under the minimum

required throughput when any of the remaining receivers is added.

In case of a BEAM service system, all the receivers asking for the service will be provided with it whether the channel PMP AM capacity is exceeded or not. The only question is whether to provide them with the service in the Acknowledged Mode or in the Unacknowledged Mode. Given that the Acknowledged Mode ensures a nominal quality most of the time (quasi-reliable transmission if enough retransmissions are allowed), this mode should be given priority, i.e. if the required throughput can be achieved in the Acknowledged Mode, this mode is chosen. The corresponding Algorithm is depicted in Fig. 5.10. In this case, phase 1 is not a receiver selection phase but a mode selection phase whereas phase 2 is still the transmission phase.

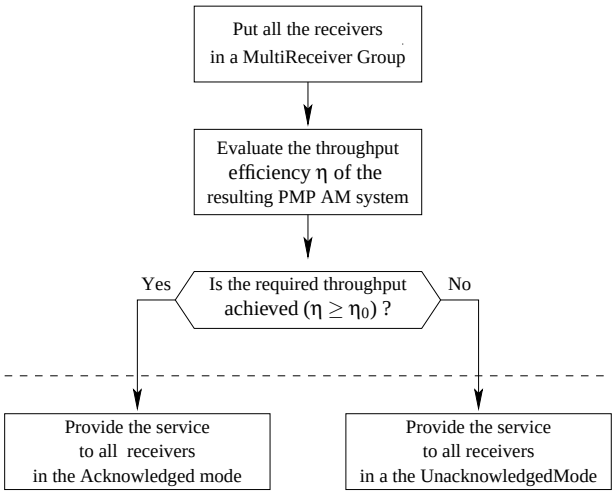

Figure 5.10: Flowchart of a BEAM service system.

Now, assume that all the receivers are in the same channel conditions and that the Algorithm adds the receivers one by one randomly. Fig. 5.11 depicts the maximum number of receivers that can be accepted in the Acknowledged Mode as a function of the minimum required throughput efficiency η_0 at a BER of 10^{-4} and for different values of the parameter M_{max}. As shown earlier, when $\varepsilon = 10^{-4}$, the transmission is quasi-reliable if $M_{max} \geq M_0 = 5$, below this value the system accepts more receivers (as shown in Fig. 5.11) but the communication is not quasi-reliable. The lower M_{max}, the higher the capacity, but the more error-prone the transmission.

In the sequel, we take $M_{max} = 10$, this value is high enough to achieve quasi-reliable communication (when the BER is not too high) and low enough to be possible in actual wireless systems (2G/3G, WiMAX).

Fig. 5.12 shows the channel PMP AM capacity for three different values of the BER on the unicast channels. We observe that with $\varepsilon = 10^{-6}$ the system accepts nearly 10 times as many receivers as with $\varepsilon = 10^{-5}$. Similarly, with $\varepsilon = 10^{-5}$ the system accepts nearly 10 times as many receivers as with $\varepsilon = 10^{-4}$. This result tells us that in the more general case encountered in practice when the different receivers are in different channel conditions, the receivers with lower BERs should be given priority (when forming the MRG) if we want to

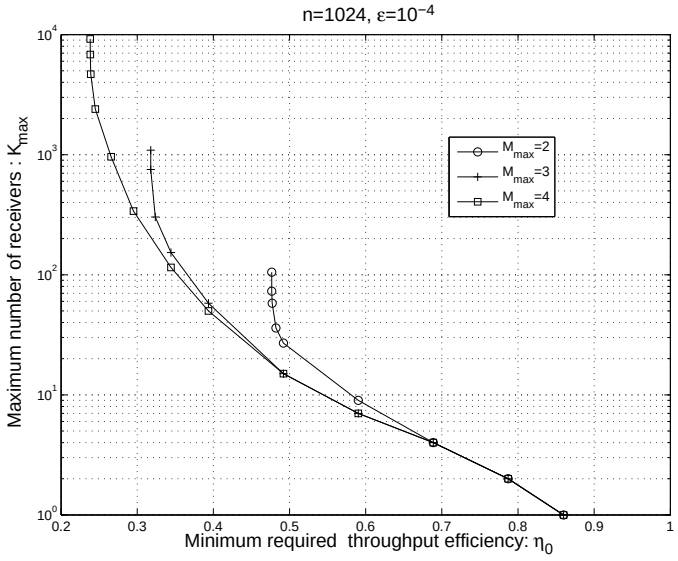

Figure 5.11: Effect of the maximum number of transmissions on the channel PMP AM capacity.

maximize the number of accepted receivers. In its operation, the Algorithm (in the case of an AM service system) must then start by ranking the receivers based on the BERs they observe on their respective channels. The receiver added at a given step will always be the one at the top of the table (the one with the lowest BER in the list of remaining receivers). Also, the first time that the evaluated throughput efficiency η goes under the threshold value η_0, the added receiver is rejected, along with all the other remaining receivers since they experience higher BERs. The flowchart of the complete version of the Algorithm is shown in Fig. 5.13. This Algorithm adds the receivers one by one in the order they come into after they are ranked (at the very beginning) in increasing order of the BER. This process goes on until there are no more receivers (all of them have been accepted) or until the channel PMP AM capacity is reached, in which case the system rejects all the remaining receivers.

On the other hand, the system may perform a test enabling it to know directly whether the channel PMP AM capacity is exceeded or not. If the capacity is not exceeded, it will not be necessary to go through the Algorithm of Fig. 5.13 because all receivers will eventually be accepted. Hence, this Algorithm can further be optimized by performing (just as in the case of a BEAM service system) a test to know whether the channel PMP AM capacity is exceeded or not. If the capacity is exceeded, the system performs the Algorithm of Fig. 5.13. If not, the system adds all receivers to the MRG and goes directly to step 2 (i.e. it starts the transmission phase). The resulting algorithm is depicted in Fig. 5.14.

In order to model the fact that in practice the BER changes from a receiver to another, we used a geometrical progression with common ratio r, i.e.

$$\varepsilon_{n+1} = \varepsilon_n \times r, \quad n \geq 1 \tag{5.20}$$

When $r > 1$, the BERs are increased by a factor r from the receiver ranked n^{th} to the

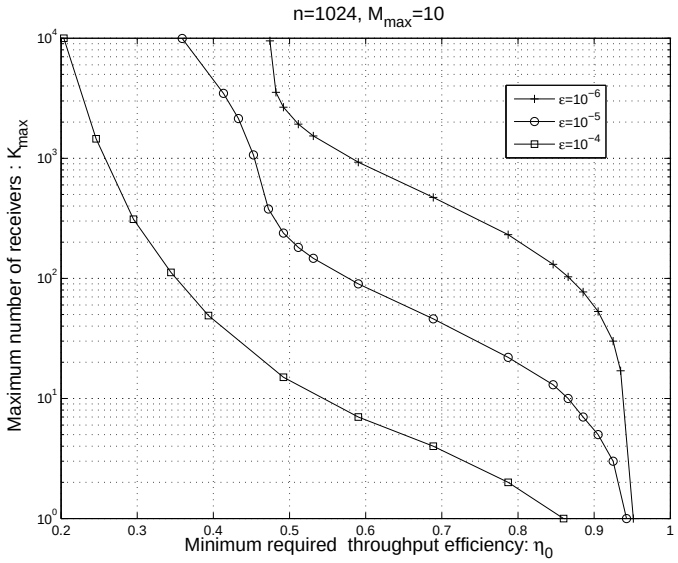

Figure 5.12: Channel PMP AM capacity K_{max} when all the receivers experience the same BER.

receiver ranked $(n+1)^{\text{st}}$. The larger is r, the larger is the gap between the different receivers (in terms of BER). For instance when $r = 1.0006$, the BER is multiplied by 10 at the 4000^{th} receiver. This number is reduced to 1000 when $r = 1.0023$, to 300 when $r = 1.0077$ and to 100 when $r = 1.0235$. When $r = 1$, the BER is constant.

The channel PMP AM capacity was plotted as a function of the minimum required throughput η_0 for the different values of r. The plots for the case when the lowest BER is $\varepsilon_1 = 10^{-6}$ are given in Fig. 5.15 and those for the case when $\varepsilon_1 = 10^{-5}$ are given in Fig. 5.16. The system accepts fewer receivers as the difference between the BERs increases. This is due to the fact that the performance of a point-to-point transmission degrades when the corresponding BER increases and so does the performance of the PMP system when the BERs of the different individual (unicast) channels tend to increase. The channel PMP AM capacity expresses, in a way, the performance a PMP system in the the Acknowledged Mode.

5.5 Point-to-Multipoint services when several channels are available

In this section, we assume that we have one or several channels available for the transmission. More precisely, let N_{ch} be the number of channels available. The service is then provided through these N_{ch} channels. In section 5.4, we considered the case when $N_{ch} = 1$. Now, we consider the more general case when $N_{ch} \geq 1$ and see how we can generalize the systems proposed in section 5.4.

In the case of an AM service system, the generalization is straightforward. The system needs to perform step 1 of the Algorithm in Fig. 5.14 as many times as there are channels. The system forms an MRG for each channel available for the service.

In the case of a BEAM service system, the system should perform the steps of phase

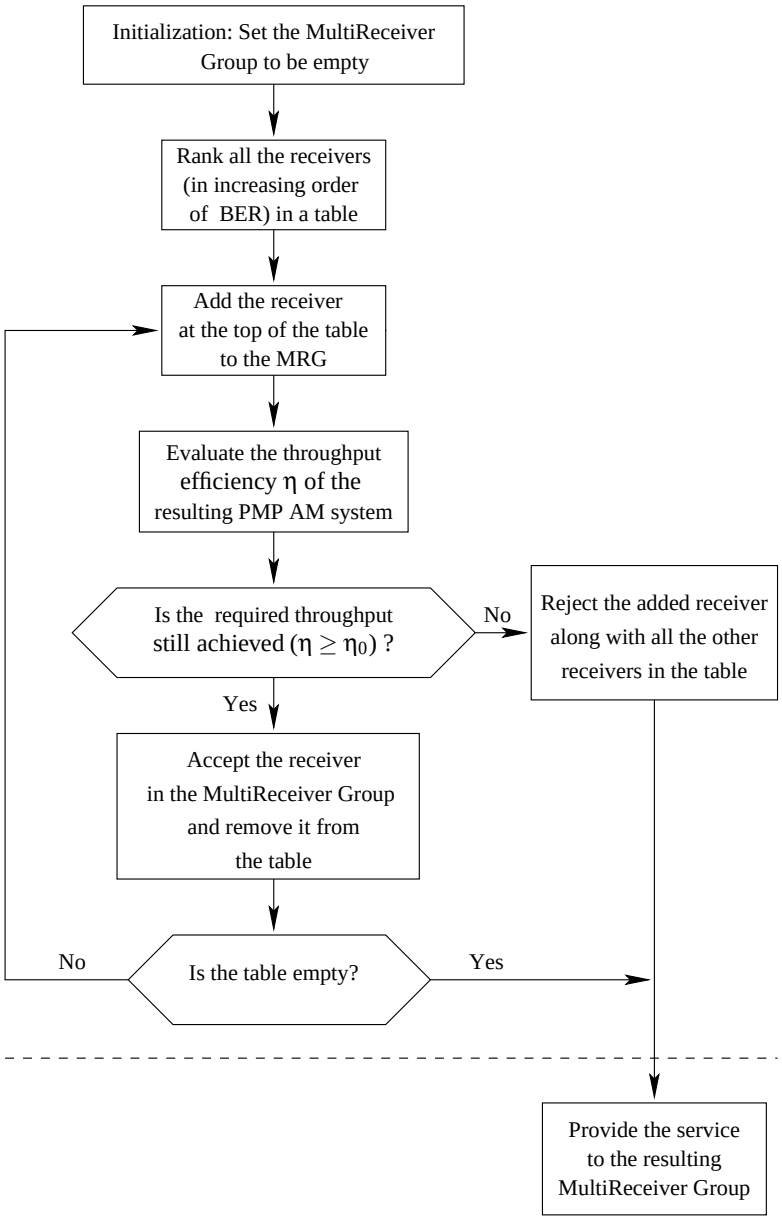

Figure 5.13: Flowchart of an AM system that maximizes the number of receivers accepted by the system.

1 of the Algorithm in Fig. 5.14 for the $(N_{ch} - 1)$ first channels, and then the Algorithm in Fig. 5.10 for the last channel. In other words, the system serves the receivers with the lower BERs in the Acknowledged Mode through the $(N_{ch} - 1)$ first channels and the remaining receivers in the Acknowledged or in the Unacknowledged Mode through the last channel (as it would do if only one channel were available). The choice of the mode for the last channel depends on whether the PMP AM capacity of the last channel is exceeded or not, as discussed in section 5.4.

Considering the case when all receivers experience the same BER, the maximum num-

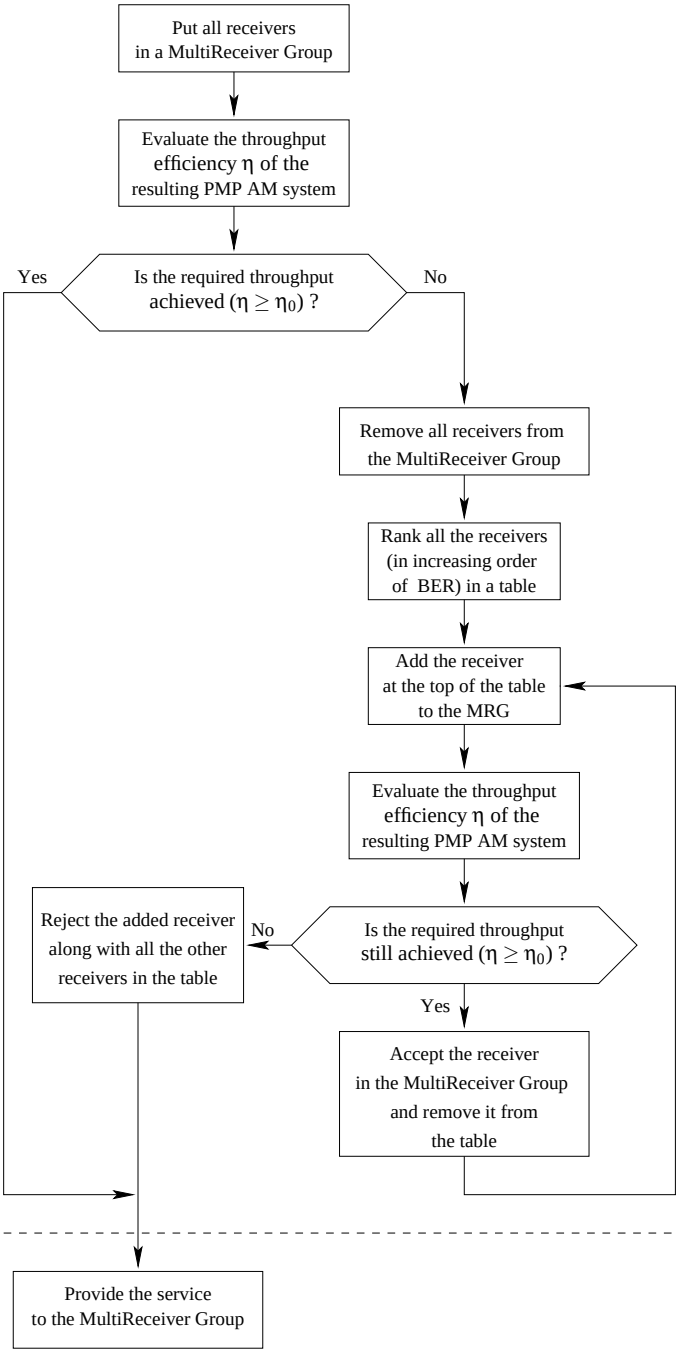

Figure 5.14: Flowchart of the optimized Algorithm of an AM service system that maximizes the number of receivers accepted by the system.

ber of receivers that can be accepted by the system is represented in Fig. 5.17 for different values of the number of channels available and for a BER of 10^{-6}. We logically notice that the number of receivers accepted by the system increases with the number of channels available. Further, we observe that this number is multiplied by the number of channels available. This is due to the fact that the BER is the same for all receivers, so the result

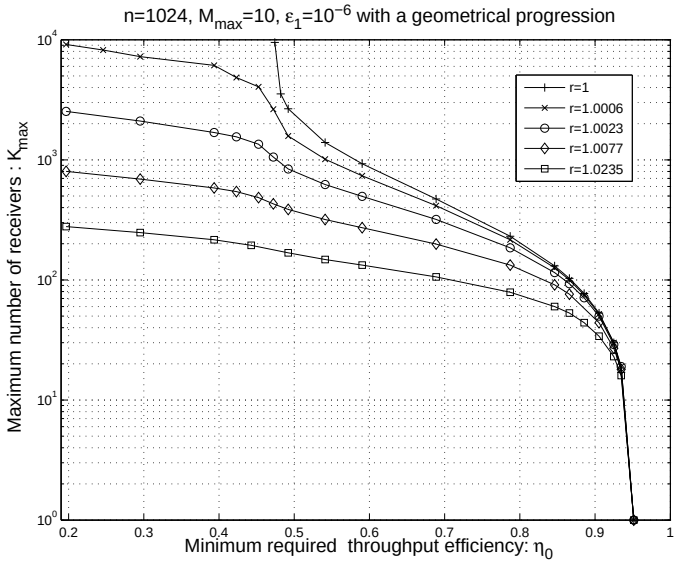

Figure 5.15: Channel PMP AM capacity K_{max} when the BERs follow a geometrical progression with $\varepsilon_1 = 10^{-6}$.

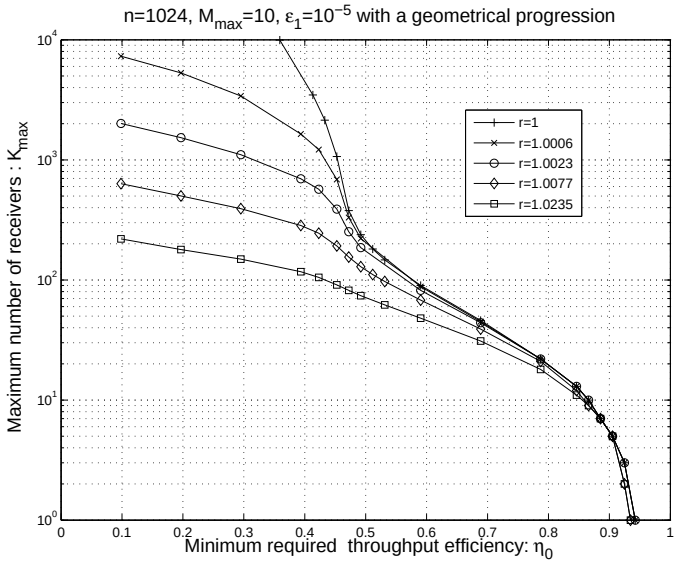

Figure 5.16: Channel PMP AM capacity K_{max} when the BERs follow a geometrical progression with $\varepsilon_1 = 10^{-5}$.

obtained is the same for each channel. As a result, the number of receivers accepted by the system is the number of receivers it would accept with one channel available, multiplied by the number of available channels.

We now consider the case when the receivers experience different BERs. We assume, as in section 5.4, that the BERs follow a geometrical progression of common ratio r.

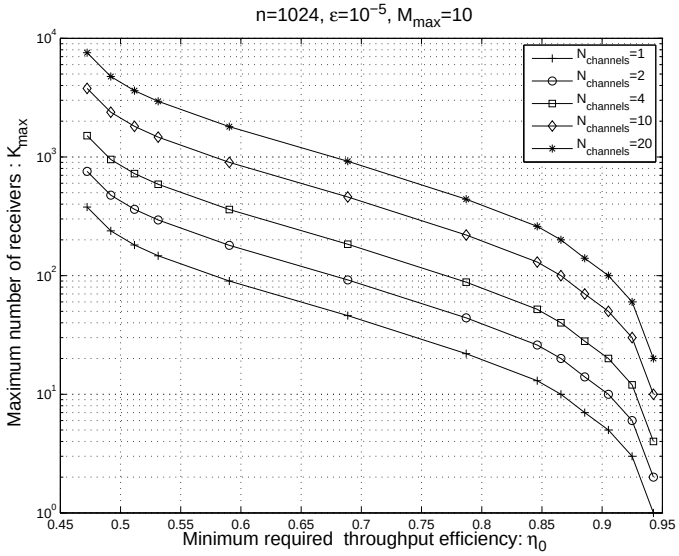

Figure 5.17: System capacity K_{max} when all the receivers experience the same BER.

The system capacity was plotted as a function of the minimum required throughput for different values of the number of channels N_{ch} and different values of the common ratio r. The lowest BER is $\varepsilon_1 = 10^{-5}$.

In Fig. 5.18, the effect of the common ratio on the system capacity was investigated with $N_{ch} = 10$. When r increases, the system capacity decreases. This is due to the fact that the performance of a point-to-point transmission degrades when the corresponding BER increases and so does the performance of a point-to-multipoint system when the BERs of the different unicast channels tend to increase. This effect was already obtained in section 5.4 when $N_{ch} = 1$ but also holds for the case when $N_{ch} > 1$.

In Fig. 5.19, the effect of the number of channels available on the system capacity was investigated for $r = 1.0023$. Again, we logically notice that the number of receivers accepted by the system increases with the number of channels available. Nevertheless, this time K_{max} does not increase linearly with N_{ch} as was the case in Fig. 5.17 for $r = 1$. Actually, the number of additional receivers that are accepted when an additional channel is made available decreases as compared to the previous channel. This is due to the fact that the receivers are ranked in increasing order of BER, so the receivers that are mapped onto channel 1 always have lower BERs than those mapped onto channel 2. Similarly, the receivers that are mapped onto channel 2 always have lower BERs than those mapped onto channel 3 ... etc. Thus, the number of receivers mapped onto channel 1 is larger than the number of receivers mapped onto channel 2, which is larger than the number of receivers mapped onto channel 3 ... etc.

This is further illustrated in Fig. 5.20 where for the case $N_{ch} = 10$, the number of receivers mapped on each of the 10 channels was represented for three different values of the common ratio r. The minimum required throughput was tuned so that the number of receivers accepted in the first channel is the same for the three cases (400 in Fig. 5.20), in order to compare the evolution of the number of additional receivers that are accepted by the system when more channels are put at the disposal of the service. Fig. 5.20 shows that when r increases, the difference in the number of receivers mapped onto two adjacent

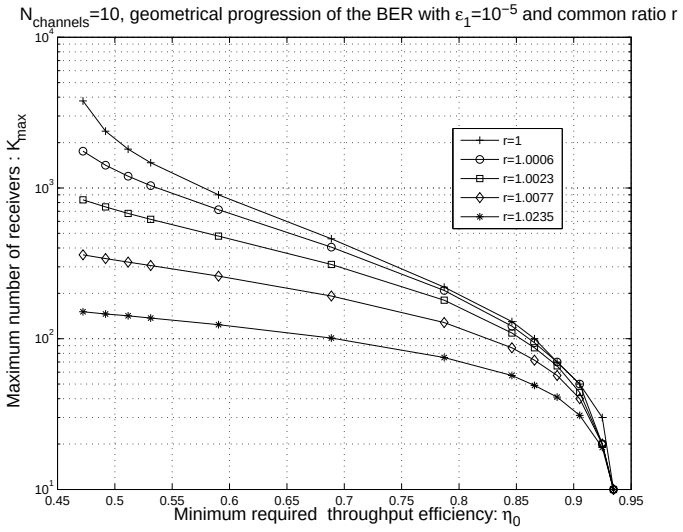

Figure 5.18: System capacity K_{max} when the BERs follow a geometrical progression with $\varepsilon_1 = 10^{-5}$, for different values of r.

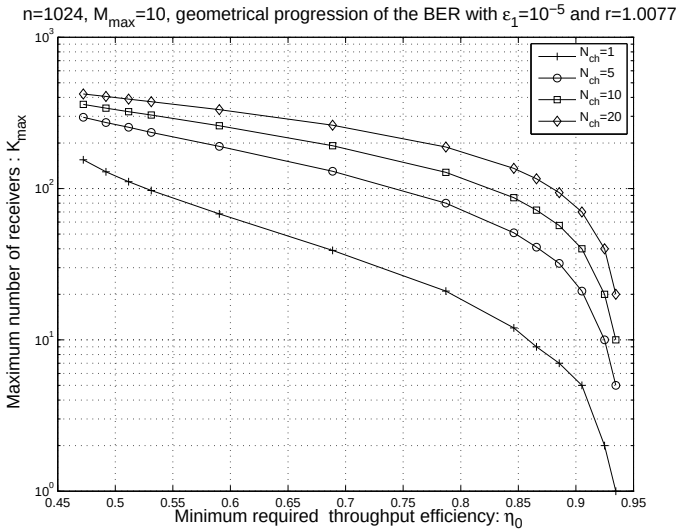

Figure 5.19: System capacity K_{max} when the BERs follow a geometrical progression with $\varepsilon_1 = 10^{-5}$, for different values of the number of available channels.

channels increases. When $r = 1$ (all receivers have the same BER), the number of receivers mapped onto the different channels is constant, which confirms the results obtained earlier.

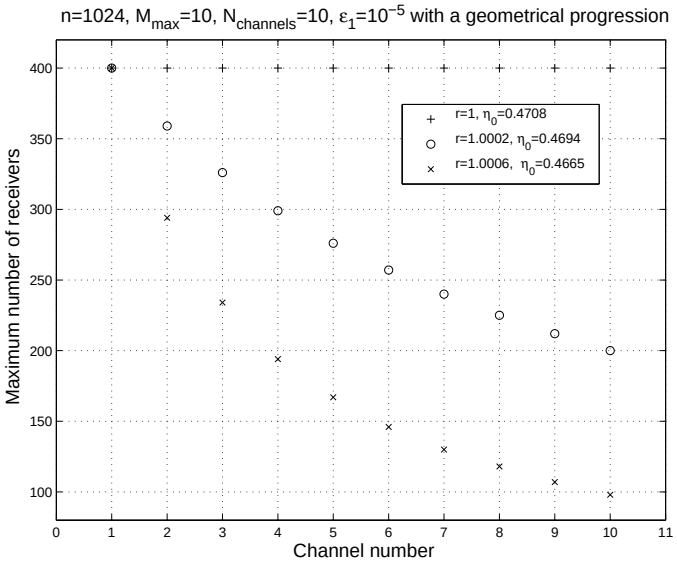

Figure 5.20: Distribution of the accepted receivers on the different channels.

5.6 Application to the Multicast/Broadcast (Multimedia) services in the 3GPP and mobile WiMAX

The 3GPP supports Multimedia Broadcast/Multicast Services (MBMS)(14)(15). Also, the mobile WiMAX system supports Multicast and Broadcast Services (MBS))(22)(23). MBS and MBMS are services that provide an efficient way to transmit multimedia streams to multiple users through a shared radio resource.

In MBS and MBMS, both Multicast and Broadcast use the Unacknowledged Mode. In other words, in MBS and MBMS, PMP communications do not use ARQ. The only difference between Multicast and Broadcast (in MBS and MBMS) is that Multicast services have membership-related processes such as joining and leaving processes, but Broadcast services do not. In other words, Multicast services are restricted to subscribers.

Theses services can be enhanced if we introduce the use of retransmissions based on the systems defined above. The table of the receivers asking for the service defined in the above Algorithms should be restricted to subscribers if need be (Multicast). In case no subscription is required (Broadcast), all the receivers asking for the service should be in the table. In MBS and MBMS, the aim is to provide the service to all the receivers in the table, therefore, if we want to enhance the quality provided to them by using ARQ, we should use the Algorithms of the BEAM system. By doing so, the network *guarantees* the receivers that are provided with the service in the Acknowledged Mode to have the nominal quality. The current MBS/MBMS services on the other hand, do not guarantee any receiver of having the nominal quality even if part of them can have it at times.

The percentage of the receivers guaranteed to have the nominal quality is represented in Fig. 5.21 for several values of the parameter N_{ch}. For $N_{ch} = 1$, this percentage switches from 100% to 0% when the channel PMP AM capacity is exceeded because the system switches from the Acknowledged Mode to the Unacknowledged Mode. The same goes when $N_{ch} > 1$ but only for the last channel, the $(N_{ch} - 1)$ first channels always use the Acknowledged Mode. As a result, the number of receivers guaranteed to have the

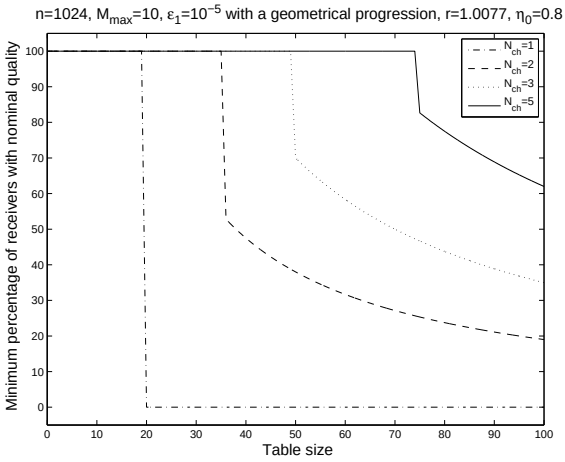

Figure 5.21: Percentage of receivers guaranteed to have the nominal quality as a function of the number of receivers in the list (which is restricted to subscribers in the case of Multicast).

nominal quality does not fall to zero but to the value $C(N_{ch}-1)$ (which represents the system PMP AM capacity with $(N_{ch}-1)$ channels) and their percentage is then given by $C(N_{ch}-1)/(Tablesize)$.

5.7 Conclusion

In this chapter, Point-to-Multipoint (PMP) communication systems were studied for application in service provision through current and future wireless systems. An analytical expression was derived for the throughput efficiency of systems using SR ARQ with the Dynamic Retransmission Group Reduction (DRGR) strategy when the receivers are in different channel (or radio) conditions. The throughput was then used to define the notion of a channel PMP capacity in the Acknowledged Mode, which is the mode that makes use of retransmissions, as opposed to the Unacknowledged Mode which does not use ARQ.

For service provision, the Acknowledged Mode (AM) service system and the Best-Effort Acknowledged Mode (BEAM) service system were proposed. The AM service system uses only the Acknowledged Mode and accepts a limited number of users. The BEAM service system accepts all users. It uses the Acknowledged Mode by default but switches to the Unacknowledged Mode if the system PMP Acknowledged Mode capacity is exceeded. The Algorithms governing the operation of these systems for a given number of available frequency channels were also defined in the chapter.

In current wireless systems, ARQ is used only in point-to-point communications. In this chapter, we proved that retransmissions can be used in point-to-multipoint communications up to a given limit on the number of users. If retransmissions are introduced in the current Multicast/Broadcast services (supported by the 3GPP and mobile WiMAX), the system guarantees a certain amount of end users to have a video of a nominal quality whereas the current Multicast/Broadcast services do not guarantee the nominal quality to any user.

Conclusions

In this thesis, we focused on the enhancement of video services provided through cellular networks. We provided a theoretical framework and a detailed analysis of the different schemes and solutions we have proposed, along with the associated numerical results.

The recovery of data loss within a video communication system is traditionally solved either by correcting errors using the redundancies inherent to the video stream, which is known as robustness or by retransmitting the erroneous packets, known as Automatic Repeat reQuest (ARQ). Robusteness-based schemes, by not using retransmissions, have maximum throughput but poor quality whereas ARQ schemes, by making use of retransmissions, provide the best possible quality at the expense of a low throughput. In this thesis, we proposed and studied an improved retransmission scheme that combines hypothesis testing to robust decoding and CRC-based ARQ operations. The decision to ask for a retransission of a packet was based on the processing of the received soft data. This scheme was denoted by Soft ARQ (SARQ).

Also, video services are characterized by large bandwidth requirements, which can be hundreds of times higher than the bandwidth required by voice services, and when these services are provided through wireless networks, one faces the problem of scarce bandwidth resources. An efficient way to reduce the necessary bandwidth would be by gathering all the customers asking for the same multimedia content in one group of receivers to which the data are conveyed using the same channel. The bandwidth is then reduced by a factor equal to the total number of receivers. On the other hand, scalable video codecs offer the possibility to have several qualities of the same encoded video, by providing at its output two streams (or more). If the video decoder is provided with the first encoded stream, the video obtained after the decoding operation is of basic quality. The other streams are quality enhancement streams, i.e. each time we provide the decoder with an additional stream, the displayed video quality is upgraded. In other words, the basic stream is indispensable if the end user wants to watch the video sequence while the other streams are only optional, i.e. it would be preferable but not indispensable to have them.

In this thesis, both scalable and non-scalable Point-to-Multipoint video communications were studied. In the non-scalable case, the study aimed at introducing the use of retransmissions in a Point-to-Multipoint scenario while in the scalable case, schemes using extensions of the basic GBN and SR ARQ techniques as well as a proposed new scheme were studied.

The study of the SARQ scheme showed that

- As opposed to (forward) robust decoding systems and CRC-based retransmission systems, it offers the possibility to trade throughput for quality (and vice versa) with the best possible "high throughput/quality" trade-off, meaning that it provides the best throughput for a given quality.

- The throughput gain as compared to the CRC-based ARQ (which garantees nominal quality) increases with the error correction effect of the robust decoder. *In the best case* (at maximum robust decoding error correction capability), a throughput gain of at least 17% can be achieved at quasi-nominal quality and a throughput gain of at least 250% can be achieved at 2-3 dB lower (corresponding to a quality change noticeable to the humain eye). Minimum throughput gains on the order of 20% (at quasi-nominal quality) and 275% can be achieved if a single (intra-coded) image is transmitted.

- A cross-layer mechanism is necessary to implement robust decoding and/or SARQ on practical systems.

The study on the transmission of a 2-layer scalable video in a Point-to-Multipoint environment showed that:

- The proposed new scheme for the transmission of a two-level scalable video is optimal in terms of performance (i.e. amount of data successfully transmitted within a given period of time) and reduces the buffering requirement at the receiver end.

- The increase in the number of receivers does not affect the system performance much while it considerably improves the bandwidth utilization efficiency.

Finally, the study on the transmission of a non-scalable video in a Point-to-Multipoint environment showed that even though ARQ is currently used only in point-to-point communications, ARQ can be used in Point-to-Multipoint communications up to a given limit on the number of receivers, which would garantee a certain amount of receivers to have the nominal quality, contrary to current systems which do not garantee the nominal quality to any receiver.

For future work, many improvements can be brought as to the work achieved in this thesis.

Concerning the SARQ scheme:

- A theoretical study needs to be conducted in order to express the performance of the SARQ scheme so that a better control of the throughput/quality through the threshold of the test is possible. It would then be possible to determine the value of the threshold that would achieve throughput such-and-such and/or PSNR such-and-such. The first step of this work would consist in expressing the probability of false alarm and/or the probability of detection and/or the bit/packet error rate and/or the average number of transmissions (hence the throughput) and can apply to any type of communication (not just video) depending on the design needs. The second step would consist in expressing the PSNR as a function of the other quality criteria (probability of false alarm, bit/packet error rate) and will apply only to video transmission applications.

- The more practical situation of variable packet and header sizes may be considered.

- A mechanism allowing to determine which MAC packets are involved in the transmission of a given NALU should be defined. This is necessary to determine which

MAC packets should be rejected and retransmitted when application layer rejects the NALU after robust decoding and hypothesis testing.

- A mechanism allowing to know which portion of a NALU was correctly received (and thus need not be robustly decoded) and which portion was not needs to be defined.
- The implementation of the mechanism on a practical system should be carried out and the resulting performance evaluated. Besides, it would be particularly interesting to combine the SARQ and APP delivery mechanisms (defined in this thesis) with the Header Recovery techniques proposed in (101)(102).

Regarding Point-to-Multipoint systems:

- Packets with a variable size should be considered.
- More realistic channel models should be considered, for in practical situations wireless unicast channels are not memoryless (errors have tendency to occur in bursts) and not independent from one another, as assumed in this work.

Bibliography

[1] IETF RFC 793 "Transmission Control Protocol," Sept. 1981.

[2] IETF RFC 1889 "RTP: A Transport Protocol for Real-Time Applications," Jan. 1996.

[3] IETF RFC 768 "User Datagram Protocol," Aug. 1980.

[4] IETF RFC 791 "Internet Protocol," Sept. 1981.

[5] 3GPP TS 04.60 V8.27.0. "General Packet Radio Service (GPRS); Mobile Station (MS) - Base Station System (BSS) interface; Radio Link Control/Medium Access Control (RLC/MAC) protocol," Release 99, Sept. 2005.

[6] 3GPP TS 23.107 V7.1.0. "Quality of Service (QoS) concept and architecture," Release 7, Sept. 2007.

[7] 3GPP TS 25.301 V7.1.0. "Radio interface protocol architecture," Release 7, April 2007.

[8] 3GPP TS 25.308 V7.2.0. "UMTS HSDPA overall description. Stage 2," Release 7, Mar. 2007.

[9] 3GPP TS 25.322 V7.2.0. "Radio Link control (RLC) specification. Stage 2," Release 7, Sept 2006.

[10] 3GPP TS 25.212 V7.4.0. "Multiplexing and channel coding (FDD)," Release 7, Mars 2007.

[11] 3GPP TR 25.950 V4.0.1. "UTRA High Speed Downlink Packet Access," Release 4, July 2005.

[12] 3GPP TR 25.321 V7.4.0. "Medium Access Control (MAC) protocol specification," Release 7, Mar. 2007.

[13] 3GPP TS 25.323 V7.4.0. "Packet Data Convergence Protocol (PDCP) protocol specification," Release 7, April 2007.

[14] 3GPP TS 23.246 V6.9.0 "Multimedia Broadcast/Multicast Service (MBMS); Architecture and Functional Description" Release 6, Dec. 2005.

[15] 3GPP TS 22.246 V7.0.0 "Multimedia Broadcast/Multicast Service (MBMS) User Services. Stage 1" Release 7, Sept. 2005.

[16] G. Kui, X. Kai, W. Charles (Thomson Licensing) "Scalable video coding streaming and transmission mechanism of the same system," *European patent EP 1 742 476 A1*, Jan. 2007.

[17] K. Premkumar and A. Chockalingam "Performance Analysis of RLC/MAC and LLC Layers in a GPRS Protocol Stack," *IEEE Transactions on vehicular technology*, vol. 53 No.5, Sep. 2004.

[18] B. Adamson,C. Bormann, M. Handley and J. Macker , "Negative-acknowledgment Oriented Reliable Multicast (NORM) Protocol," *Request for Comments: 3940*, Nov. 2004.

[19] J. W. K. Wong and V. C. M. Leung, "Improving end-to-end performance of TCP using link layer retransmissions over mobile internetworks ," *In Proc. of the IEEE International Conference on Communications (ICC)*, pp. 324-328, Jun. 1999.

[20] Y. Bai, A. T. Ogielski and G. Wu, "Interaction of TCP and radio link ARQ protocol," *In Proc. of the IEEE Vehicular Technology Conference (VTC)*, pp. 1710-1714, Sept. 1999.

[21] M. Assaad, M. Jouaber and D. Zeghlache, "Effect of TCP on UMTS/HSDPA system performance and capacity," *IEEE Global Telecommunications Conference (GLOBECOM)*, vol. 6 ,pp. 4104-4108, Nov-Dec. 2004.

[22] IEEE 802.16-2004 standard "Local and Metropolitan Area Networks Part 16 : Air Interface for Fixed Broadband Wireless Access Systems Amendment for Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in License Bands," October 2004.

[23] IEEE 802.16e-2005 amendment "Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in License Bands," Dec. 2005.

[24] C. Reader, "History of MPEG Video Compression - Ver. 4.0," *Joint Video Team (JVT) doc*, 2002. JVT-E066.

[25] ITU, CCITT, "Recommendation IT-81, Information Technology - Digital Compression and Coding of Continuous-Tone Still Images - Requirements and Guidelines (JPEG)," 1992.

[26] R. D. Kell. "Improvements relating to electric picture transmission systems," *Technical Report, British Patent* no. 341.811, 1929.

[27] ITU-T, "Codec for videoconferencing using primary video group management," *Technical Report ITU-T Rec. H. 120* version 1, 1984.

[28] A. Habibi, "Hybrid coding of pictural data," *IEEE Transactions on Communications* vol. 22, no. 5, pp. 614-624, May 1974.

[29] ITU-T, "Video codec for audiovisual services at px64 kbits/s," *Technical Report ITU-T Rec. H. 261* version 1, nov. 1990.

[30] ISO/IEC JTC 1, "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbits/s - part 2: Video." *Technical Report* ISO/IEC 11172-2 (MPEG-1), Mar. 1993.

[31] ISO/IEC JTC 1/SC 29, "Generic coding of moving pictures and associated audio information: Systems," *Technical Report* ISO/IEC 13818-1 (MPEG-2 Part 1), 1996.

[32] ITU-T, "Video coding for low bit rate communication," *Technical Report ITU-T Rec. H. 263* version 1, Nov. 1995.

[33] ISO/IEC JTC 1, "Coding of audio-visual objects - part 2: Video," *Technical Report* ISO/IEC 14496-2 (MPEG-4 visual version 1), April 1999.

[34] ITU-T and ISO/IEC JTC 1, "Advanced video coding for generic audiovisual services," *Technical Report* ITU-T Rec. H.264, and ISO/IEC 14496-10 AVC, Nov. 2003.

[35] K. Sayood. "Introduction to Data Compression, 2nd edition," *Morgan Kaufmann*, San Francisco, 2000.

[36] A. K . Jain "Fundamentals of Digital Image Processing," *Prentice Hall*, Englewood Cliffs, NJ, 1989.

[37] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H. 264/AVC video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology* vol. 13, no. 7, pp. 560-576, July 2003.

[38] ITU, "Methodology for the subjective assessment of the quality of television pictures," *Technical Report* ITU-R BT. 500-11, 2004.

[39] I. Richardson, "H.264 and MPEG-4 Video Compression: Video Coding for Next-Generation Multimedia," *John Wiley and Sons*, 2003.

[40] H. S. Malvar, A. Hallapuro, M. Karczewicz and L. Kerofsky, "Low-complexity transform and quantization in H. 264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology* vol. 13, no. 7, pp. 598-603, July 2003.

[41] G. Bjontegaard and K. Lillevold, "Context-adaptive VLC coding of coefficients," *Technical report, JVT*, 2002.

[42] D. Marpe, H. Schwarz and T. Wiegand, "Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard," *IEEE Transactions on Circuits and Systems for Video Technology* vol. 13, no. 7, pp. 620-636, 2003.

[43] D. Marpe, G. Blattermann and T. Wiegand, "Adaptive codes for H.26L," Technical report, JVT, 2002.

[44] P. List, A. Joch, J. Lainema, G. Bjontegaard and M. Karczewicz, "Adaptive deblocking filter," *IEEE Transactions on Circuits and Systems for Video Technology* vol. 13, no. 7, pp. 614-619, July 2003.

[45] D. J. C. MacKay, "Information Theory, Inference, and Learning Algorithms," *Cambridge University Press*, Cambridge, 2003.

[46] T. Richardson and U. Urbanke, "Modern Coding Theory," *Cambridge University Press*, 2008.

[47] P. Salama, N. Shroff, E. J. Coyle, and E. J. Delp, "Error concealment techniques for encoded video streams," *In Proc. of ICIP*, pp 9-12, October 1997.

[48] W. Zeng and B. Liu, "Geometric-structure-based error concealment with novel applications in block-based low-bit-rate coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 9, no. 4, pp. 648-665, June 1999.

[49] H. Sun and W. Kwok, "Concealment of damaged block transform coded images using projections onto convex sets," *IEEE Transactions on Image Processing*, vol. 4, pp 470-477, April 1995.

[50] M.-J. Chen, L.-G. Chen, and R.-M. Weng, "Error concealment of lost motion vectors with overlapped motion compensation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7, no. 3, pp. 560-563, 1997.

[51] E. Asbun and E. J. Delp, "Real-time error concealment in compressed digital video streams," *In Proceedings of PCS*, 1999.

[52] Y. O. Park, C.-S. Kim, and S.-U. Lee, "Multi-hypothesis error concealment algorithm for H.26L video," *In Proceedings of ICIP*, pp. 465-468, 2003.

[53] B. Jung, B. Jeon, M.-D. Kim, B. Suh, and S.-I. Choi, "Selective temporal error concealment algorithm for H.264/AVC," *In Proceedings of ICIP*, 2004.

[54] Y.-K. Wang, M. M. Hannuksela, V. Varsa, A. Hounrunranta, and M. Gabbouj, "The error concealment feature in the H.26L test model," *In Proceedings of ICIP*, volume 2, pp. 729-732, 2002.

[55] H. Sun and J. Zedepski, "Adaptive error concealment algorithm for MPEG compressed video," *In Proceedings of VCIP*, pp 814-824, 1992.

[56] W.-Y. Kung, C.-S. Kim, and C.-C. J. Kuo, "Spatial and temporal error concealment techniques for video transmission over noisy channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 7, pp. 789-803, July 2006.

[57] V. Buttigieg and P.G. Farrell, "A MAP decoding algorithm for variable-length error-correcting codes," *In Codes and Cyphers : Cryptography and Coding IV*, pp 103-119, Essex, England, 1995. The Inst. of Mathematics and its Appl.

[58] J. Hagenauer, "Source-controlled channel decoding," *IEEE Transactions on Communications*, vol. 43, no. 9, pp. 2449-2457, 1995.

[59] S. Kaiser and M. Bystrom, "Soft decoding of variable-length codes," *In Proceedings of ICC*, vol. 3, pp. 1203-1207, New Orleans, 2000.

[60] L. Perros-Meilhac and C. Lamy, "Human tree based metric derivation for a low-complexity soft VLC decoding," *In Proceedings of ICC*, 2002.

[61] R. Thobanen and J. Kliewer, "Robust decoding of variable-length encoded markov sources using a three-dimensional trellis," *IEEE Communications Letters*, vol. 7, no. 7, pp. 320-322, 2003.

[62] T. Tillo, M. Grangetto, and G. Olmo, "A flexible error resilient scheme for JPEG 2000," *In Proceedings of MSP*, pp. 295-298, 29 Sept.-1 Oct. 2004.

[63] H. Nguyen, P. Duhamel, J. Brouet, and D. Rouffet, "Robust VLC sequence decoding exploiting additional video stream properties with reduced complexity," *In Proceedings of ICME*, pp. 375-378, June 2004, Taipei, Taiwan.

[64] C. Bergeron and C. Lamy-Bergot, "Soft-input decoding of variable-length codes applied to the H.264 standard," *In Proceedings of MSP*, pp. 87-90, 29 Sept.-1 Oct. 2004.

[65] G. Sabeva, S. Ben Jamaa, M. Kieffer, and P. Duhamel, "Robust decoding of H.264 encoded video transmitted over wireless channels," *In Proceedings of MSP*, pp. 9-12, Victoria, Canada, 2006.

[66] C.M. Lee, M. Kieffer, and P. Duhamel, "Soft decoding of VLC encoded data for robust transmission of packetized video," *In Proceedings of ICASSP*, pages 737-740, 2005.

[67] R. Bauer and J. Hagenauer, "On variable length codes for iterative source/channel decoding," *In Proceedings of DCC*, pp 272-282, Snowbird, UT, 1998.

[68] R. Thobaben and J. Kliewer, "On iterative source-channel decoding for variable-length encoded markov sources using a bit-level trellis," *In Proceedings of SPAWC*, Rome, 2003.

[69] H. Nguyen and P. Duhamel, "Iterative joint source-channel decoding of variable length encoded video sequences exploiting source semantics," *In Proceedings of ICIP*, 2004.

[70] H. Nguyen and P. Duhamel, "Compressed image and video redundancy for joint source-channel decoding," *In Proceedings of Globecom*, 2003.

[71] C. Lamy and S. Merigeault, "Procédé de correction d'une trame erronée par un récepteur," *French patent* no. 0206501, 2002.

[72] J. W. Nieto and W. N. Furman, "Cyclic redundancy check (CRC) based error correction method and device," *US Patent* US 2007/0192667 A1, Aug. 16 2007.

[73] J. K. Wolf, "Efficient maximum-likelihood decoding of linear block codes using a trellis," *IEEE Transactions on Information Theory*, no. 24 pp. 76-80, 1978.

[74] J. B. Anderson and S. Mohan, "Source and Channel Coding : An Algorithmic Approach," Kluwer, 1991.

[75] R. E. Blahut, "Theory and Practice of Error Control Codes," Addison-Wesley, Reading, MA, 1984.

[76] H. Jenka, T. Stockhammer, and W. Xu, "Permeable-layer receiver for reliable multicast transmission in wireless systems," *In Proceedings of WCNC*, vol. 3, pp. 1805-1811, vol. 3, 13-17 March 2005.

[77] H. Jenkac, T. Stockhammer, and W. Xu, "Cross-layer assisted reliability design for wireless multimedia broadcast," *EURASIP Signal Processing Journal*, 2006.

[78] C. Bormann, C. Burmeister, M. Degermark, H. Fukushima, H. Hannu, L.-E. Jonsson, R. Hakenberg, T. Koren, K. Le, Z. Liu, A. Martensson, A. Miyazaki, K. Svanbro, T. Wiebke, T. Yoshimura, and H. Zheng, "Robust header compression (ROHC): Framework and four profiles", 2001.

[79] G. Caire, G. Taricco and E. Biglieri, "Bit-interleaved coded modulation," *IEEE Transactions on Information Theory*, vol. 44, pp. 927-946, May 1998.

[80] E. Zehavi, "8-PSK treillis codes for a Rayleigh channel," *IEEE Transactions on Communications*, vol. 40, pp. 873-884, May 1992.

[81] U. Hansson and T. Aulin, "Bit-interleaved coded modulation on the time continuous Rayleigh fading channel," in *the IEEE Int. Symp. on Information Theory*, p. 339, Cambridge, MA, Aug. 1998.

[82] T. May, H. Rohling and V. Engels, "Performance analysis of Viterbi decoding for 64-DAPSK and 64-QAM modulated OFDM signals," *IEEE Transactions on Communications*, vol. 46, no. 2, pp. 182-190, Feb. 1998.

[83] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260-269.

[84] J. K. Omura, "On the Viterbi decoding algorithm," *IEEE Transactions on Information Theory*, vol. 15, no. 1, pp. 177-179, January 1969.

[85] G. D. Forney, Jr., "The Viterbi algorithm," in *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268-278, March 1973.

[86] L. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Transactions on Information Theory*, pp. 284-287, March 1974.

[87] P. Robertson, E. Villebrun and P.Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proceedings of the Int. Conference on Communications*, pp. 1009-1013, Seattle, June 1995.

[88] A. Viterbi, "An intuitive justification and a simplified implementation of the MAP decoder for convolutional codes," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 2, pp. 260-264, Feb. 1998.

[89] M. Fossorier, F. Burkert, S. Lin and J. Hagenauer, "On the equivalence between SOVA and Max-Log-MAP decodings," *IEEE Transactions on Communications*, vol. 2, no. 5, pp. 137-139, May 1998.

[90] J. Hagenauer and P. Hoeher, "A Viterbi Algorithm with soft-decision outputs and its applications," in *Proceedings of the GLOBECOM conference*, vol. 3, pp. 1680-1686, Dallas, Nov. 1989.

[91] C. Berrou and A. Glavieux, "Near optimum error correcting-coding and decoding : Turbo codes," *IEEE Transactions on communications*, vol. 44, no. 10, pp. 1261-1271, Oct. 1996.

[92] C. Berrou, A. Glavieux and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding : Turbo codes," in *Proceedings of the Int. Conf. on Communications*, Geneva, Switzerland, May 1993, pp. 1064-1070.

[93] C. Douillard, M. Jézéquel, C. Berrou, A. Picard, P. Didier and A. Glavieux, "Interative correction of intersymbols interference : Turbo-equalization" *European Transactions on Telecommunications*, vol. 6, no. 5, pp. 507-511, Sept. 1995.

[94] A. Glavieux, C. Laot and J. Labat, "Turbo equalization over a frequency selective channel," in *Proceedings of the Int. Symp. on Turbo Codes and Related Topics*, Brest, France, Sept. 1997, pp. 96-102.

[95] V. Lau, "The turbo principle : Tutorial introduction and state of the art," in *Int. Symp. on Turbo Codes and Related Topics*, 2000.

[96] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding" *IEEE Communication Letters*, vol. 1, no. 6, pp. 169-171, Nov. 1997.

[97] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding" in *Proceedings of the Int. Conference on Communications*, June 1999, vol. 2, pp. 858-863.

[98] X. Li and J. A. Ritcey, "Treillis-coded modulation with bit interleaving and iterative decoding," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 4, pp. 715-724, April 1999.

[99] S. ten Brink, J. Speidel, and R. Han, "Iterative demapping for QPSK modulation," *Electronic Letters*, vol. 34, no. 15, pp. 1459-1460, July 1998.

[100] P. Magniez, B. Muquet, P. Duhamel, V. Buzenac and M. de Courville "Optimal Decoding of Bit-Interleaved Modulations: Theoretical Aspects and Practical Algorithms ," in *Proceedings of the 2nd Intl. Symposium on Turbo Codes and Related Topics*, Brest, France, Sept. 2000, pp. 169-172.

[101] C. Marin, Y. Leprovost, M. Kieffer and P. Duhamel, "Robust header recovery based enhanced permeable protocol layer mechanism," *in the Proceedings of SPAWC*, July 2008.

[102] C. Marin, Y. Leprovost, M. Kieffer and P. Duhamel, "Robust MAC-lite and header recovery-based improved permeable protocol layer scheme ," *in the Proceedings of ISSSTA*, August 2008.

[103] C. Marin, "Vers une solution réaliste de décodage source-canal conjoint de contenus multimédia," doctoral dissertation, March 2009.

[104] A. R. K. Sastry "Improving Automatic Repeat Request (ARQ) performance on satellite channels under high error rate conditions," *IEEE Transactions on Communications*, vol. COM-23, pp. 436-438, 1975.

[105] J. M. Morris "On another go-back-N ARQ technique for high error rate conditions," *IEEE Transactions on Communications*, vol. COM-26, pp. 187-189, Jan. 1978.

[106] S. Lin and P. S. Yu "An effective error-control scheme for satellite communications," *IEEE Transactions on Communications*, vol. COM-28, pp. 395-401, Mar. 1980.

[107] P. S.Yu and S. Lin "An efficient selective repeat ARQ scheme for satellite channels and its throughput analysis," *IEEE Transactions on Communications*, vol. COM-29, pp. 353-363, Mar. 1981.

[108] E. J. Weldon "An improved selective-repeat ARQ strategy," *IEEE Transactions on Communications*, vol. COM-30, no. 3, pp. 480-486, Mar. 1982.

[109] S. Lin, D. J. Costello, Jr. and M. J. Miller "Automatic Repeat Request Error-Control Schemes," *IEEE Communications Magazine*, vol. 22, pp. 5-16, Dec. 1984.

[110] S. Lin, D. J. Costello, Jr., *Error Control Coding : Fundamentals and Applications.* Englewood Cliffs, NJ : Prentice-Hall, 1983.

[111] S. B. Calo and M. C. Easton, "A broadcast protocol for file transfer for multiple sites," *IEEE Transactions on communications*, vol. COM-29, pp. 1701-1706, 1981.

[112] M. Mase, T. Katenaka, H. Yamamoto and M. Shinohara "Go-back-N ARQ schemes for point-to-multipoint satellite Communications ," *IEEE Transactions on communications*, vol. COM-31, pp. 583-590, 1983.

[113] J.S. Gopal and J. M. Jaffe, "Point-to-multipoint communication over broadcast links," *IEEE Transactions on Communications*, vol. COM-32, pp. 1034-1044, 1984.

[114] K. Sabnani and M. Schwartz, "Multidestination protocols for satellite broadcast channels," *IEEE Transactions on communications*, vol.COM-33, No.3, pp. 232-240, Mar. 1985.

[115] R. H. Deng, "Hybrid ARQ schemes for Point-to-multipoint over nonstationary broadcast channels," *IEEE Transactions on Communications*, vol. 41 No.9, pp. 1379-1387, Sept. 1993.

[116] J. J. Metzner, "An improved broadcast retransmission protocol," *IEEE Transactions on Communications*, vol. COM-32 No.6, pp. 679-683, Jun. 1984.

[117] K. Sakakibara and M.Kasahara, "A multicast Hybrid ARQ scheme using MDS codes and GMD decoding," *IEEE Transactions on Communications*, vol. 43 No.12, pp. 2933-2939, Dec. 1995.

[118] D. Towsley, "An analysis of a point-to-multipoint channel using a go-back-N error-control protocol," *IEEE Transactions on Communications*, vol. COM-33 No. 3, pp. 282-285, Mar. 1985.

Appendix A : Automatic Repeat ReQuest (ARQ) basic schemes

1 Introduction

Automatic Repeat ReQuest (ARQ) is widely used for error control in data communication systems. This method is simple and provides high system reliability. If a properly chosen code is used for error detection, virtually error-free data transmission can be attained.

When a packet is ready for transmission, a set of parity bits is appended to it. The new packet is then transmitted to the receiver end. The received packet may contain transmission errors.

When a packet is received, the receiver checks the validity of the received data. If the data are valid (from a parity point of view), the received packet is assumed to be error-free and is delivered (with parity bits removed) to the user. If the data are not valid (i.e. the presence of the errors is detected), the receiver discards the erroneously received packet and requests the retransmission of the same packet via a feedback channel. Retransmission continues until the packet is successfully received.

2 ARQ schemes

There are three basic types of ARQ schemes : Stop and Wait (SW) ARQ, Go-Back-N (GBN) ARQ and Selective Repeat (SR) ARQ.

2.1 The Stop-and-Wait ARQ

In a SW ARQ error control system, the transmitter sends a packet to the receiver and waits for an acknowledgement. A positive acknowledgement (ACK) from the receiver indicates that the transmitted packet has been successfully received. A negative acknowledgement (NACK) from the receiver indicates that the transmitted packet has been detected in error, the transmitter then resends the packet and again waits for an acknowledgement. Retransmissoins continue until the transmitter receives an ACK. This is illustrated in fig. 1. Note that "A" means that the transmission will turn out to be successful and the packet will be ACKed whereas "N" means that the transmission will turn out to be erroneous and the packet will be NACKed. The (N)ACK arrives at the transmitter a Round-Trip Time (RTT) after the transmission of the packet.

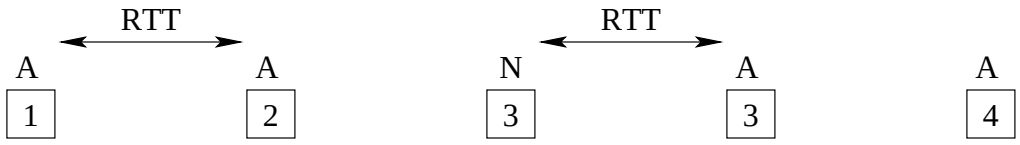

FIGURE 1 – Stop-and-Wait ARQ.

This scheme is simple but inherently inefficient because of the idle time spent waiting for an acknowledgement of each transmitted packet.

2.2 The Go-Back-N ARQ

The GBN ARQ scheme is illustrated in fig. 2. The transmitter continuously transmits packets in order and then stores them pending receipt of an ACK/NACK for each packet.

The ACK/NACK arrives after an RTT. During this interval, $N-1$ other packets are also transmitted. Whenever the transmitter receives a NACK indicating that a particular packet, say packet i, was received in error, it stops transmitting new codewords. Then it goes back to packet i and proceeds to retransmit that packet and the $N-1$ succeeding packets which were transmitted during one round trip delay. At the receiving end, the receiver discards the erroneously received packet i and all $N-1$ subsequently received packets, whether they are error-free or not. Retransmission continues until packet i is acknowledged. In each retransmission of packet i, the transmitter resends the same sequence of packets. As soon as packet i is positively acknowledged, the transmitter proceeds to transmit new packets.

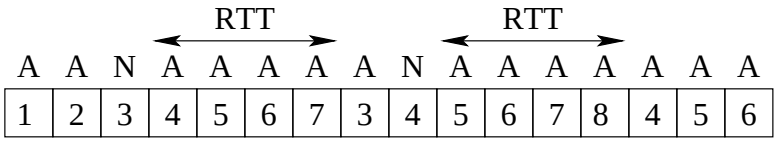

FIGURE 2 – Go-Back-N ARQ.

The main drawback of the the GBN ARQ is that, whenever a received packet is detected in error, the receiver also rejects the next $N-1$ received packets, even though many of them may be error-free. As a result, they must be retransmitted. This represents a waste of transmissions, which can result in severe deterioration of throughput performance.

2.3 The Selective Repeat ARQ

The GBN ARQ scheme becomes quite ineffective for communication systems with high data rate. This ineffectiveness is caused by the retransmission of many error-free packets following a packet detected in error. This can be overcome by using the SR ARQ scheme. In an SR ARQ error-control system, codewords are also transmitted continuously. However, the transmitter only resends those codewords that are negatively acknowledged (NACKed). After resending a NACKed packet, the transmitter continues transmitting new packets in the transmitter buffer (as illustrated in Fig. 3). With this scheme, a buffer must be provided at the receiver to store the error-free packets following a received packet detected in error, because, ordinarily, packets must be delivered to the end user in correct order. When the first NACKed packet is successfully received, the receiver then releases any error-free packets in consecutive order from the receiver buffer until the next erroneously received word is encountered. Sufficient receiver buffer storage must be provided in an SR ARQ system, otherwise, buffer overflow may occur and packets may be lost.

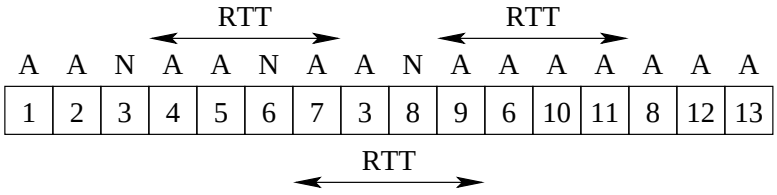

FIGURE 3 – Selective Repeat ARQ.

3 Performance of the basic ARQ schemes

The performance of an ARQ error-control system is normally measured by its throughput efficiency, which is defined as the ratio of the average number of information bits successfully accepted by the receiver per second to the total number of bits that could be transmitted per second.

For simplicity, we assume that the forward channel is a random-error channel with bit error rate ε and that the feedback channel is error-free.

Let n be the size of the packet in bits and k be the size of the information part of the packet (i.e. $n-k$ is the number of parity bits added for error detection). Let P_s be the probability of a successful transmission of a packet. This probability is given by

$$P_s = (1-\varepsilon)^n \tag{1}$$

3.1 Throughput efficiency of a Stop-and-Wait ARQ system

Let λ be the idle time of the transmitter between two successive transmissions, and let δ be the bit rate of the transmitter. Even though the transmitter does not transmit during the idle period, the effect of the idle period on the throughput must be taken into consideration. In one round-trip delay time, the transmitter could transmit $n+\lambda\delta$ bits if it did not remain idle. For a packet to be received correctly, the average number of bits that the transmitter could have transmitted is

$$\begin{aligned} T_{SW} &= (n+\lambda\delta)P_s + 2(n+\lambda\delta)P_s(1-P_s) + \ldots + l(n+\lambda\delta)P_s(1-P_s)^{l-1} + \ldots \\ &= (n+\lambda\delta)P_s(1 + 2(1-P_s) + 3(1-P_s)^2 + \ldots) \\ &= \frac{n+\lambda\delta}{P_s} \end{aligned}$$

Therefore, the throughput of a SW ARQ system is

$$\eta_{SW} = \frac{k}{T_{SW}} = \frac{P_s \cdot (k/n)}{1+\lambda\delta/n} \tag{2}$$

3.2 Throughput efficiency of a Go-Back-N ARQ system

In a GBN ARQ system, retransmission of a NACKed packet involves resending N packets. Consequently, for a packet to be successfully received, the average number of transmissions is

$$\begin{aligned} \overline{M}_{GBN} &= 1 \cdot P_s + (N+1)P_s(1-P_s) + \ldots + (lN+1)P_s(1-P_s)^l + \cdots \\ &= 1 + \frac{N(1-P_s)}{P_s} \end{aligned}$$

Therefore, the throughput of the of a GBN ARQ system is

$$\eta_{GBN} = \frac{k}{n}\frac{1}{\overline{M}_{GBN}} = \frac{P_s(k/n)}{P_s + (1-P_s)N} \tag{3}$$

3.3 Throughput efficiency of a Selective Repeat ARQ system

In a SR ARQ system, for a packet to be accepted by the receiver, the average number of transmissions needed is

$$\begin{aligned} \overline{M}_{SR} &= 1 \cdot P_s + 2 \cdot P_s(1-P_s) + \cdots + l \cdot P_s(1-P_s)^{l-1} + \cdots && (4) \\ &= \frac{1}{P_s} && (5) \end{aligned}$$

Hence, the throughput of a SR ARQ system is

$$\eta_{SR} = \frac{k}{n}\frac{1}{\overline{M}_{SR}} = P_s \cdot \frac{k}{n} \qquad (6)$$

Appendix B : Retransmission mechanisms in an end-to-end connection over a cellular network

1 Introduction

In appendix A, the principle of ARQ and the basic ARQ schemes were discussed. In this appendix, we discuss the practical implementation of ARQ, as well as the different retransmission schemes that are used in the Internet and in cellular networks.

In order to identify the different packets, the transmitter assigns a Sequence Number to each packet. The Sequence Number is generally encoded as n_s bits, and therefore cannot be infinitely large. As a consequence, a cyclically reusable sequence numbering scheme is used. The sequence range is 0 to $2^{n_s} - 1$. The transmitter and the receiver use a window to ensure that the cyclically reusable sequence numbering scheme described above works properly. It can be proved that the appropriate window size are 1 for the Stop-and-Wait scheme, $2^{n_s} - 1$ for the Go-Back-N scheme and 2^{n_s-1} for the Selective Repeat scheme.

Also, in practice, the number of retransmissions of a given packet is limited by a parameter, so that the process does not stall because of a persistent failure in a packet transmission.

In an end-to-end (server-terminal) TCP connection, the retransmissions used are of type Go-Back-N and the transmission is based on a congestion window the size of which is adapted according to different algorithms (slow start, congestion avoidance).

Sections 2 and 3 describe the ARQ mechanisms implemented in 2G and 3G systems respectively. Section 4 describes the ARQ supported by the 802.16-2004 WiMAX in more details. In 802.16-2004 WiMAX ARQ, the stream is partitioned into blocks. Each packet contains one or several blocks, and the Sequence Number contained in the subheaders represents the sequence number of the first block in the packet.

2 Retransmission mechanisms in an end-to-end connection over a 2G network

In 2G systems (GPRS, EDGE), the ARQ is implemented at two levels : The LLC and the RLC sublayers. At the LLC level, a 24-bit CRC called Frame Control Header (FCH) is used for error detection, and the 8-bit LLC Frame Number field of the LLC Frame Header is used for sequence numbering.

At the RLC level, the Block Sequence Number (BSN) is used for sequence numbering. The length of this field is 7 bits in GPRS and 11 bits in EDGE. Note that block here refers to an RLC/MAC PDU called RLC/MAC block. The error detection is carried out using a CRC called Block Control Sequence (BCS). The length of this field is 16 bits when Coding Schemes CS2-CS4 are used and 40 bits when Coding Scheme CS1 is used.

The LLC ARQ operates between the terminal (MS) and the SGSN, whereas the RLC ARQ operates between the terminal (MS) and the Base Station Controller (BSC). Both LLC ARQ and RLC ARQ are of Selective Reject type and both are used in the corresponding Acknowledged Mode.

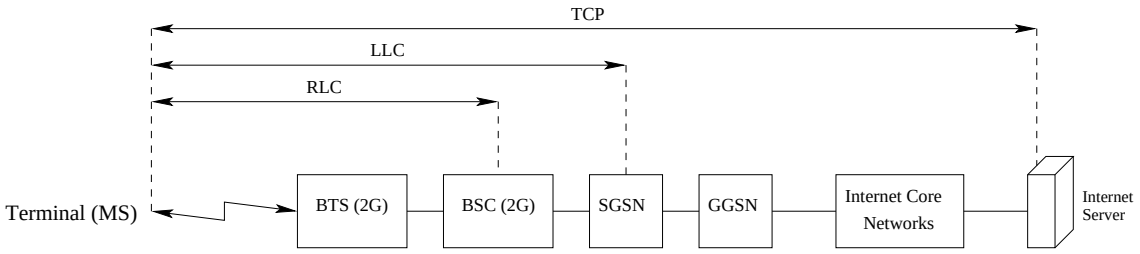

FIGURE 1 – Possible retransmission levels in a TCP connection over 2G networks.

3 Retransmission mechanisms in an end-to-end connection over a 3G network

In UMTS, the ARQ mechanism is part of the RLC protocol. The ARQ is used in the Acknowledged Mode of the RLC protocol. It is of SR type and operates between the terminal (UE) and the Radio Network Controller (RNC). A 12-bit Sequence Number (SN) is used. Error detection is carried out by a CRC the size of which is 24, 16, 12, 8 or 0 bits. The CRC size is signalled from higher layers.

In HSDPA, the ARQ mechanism is implemented in a new (with respect to UMTS) MAC entity called MAC-hs (MAC high speed), which is located in Node B, which represents the layer upgrade when introducing HSDPA on the UMTS radio interface (in UMTS, the Node B is purely a physical element). Layers situated above the MAC-hs layer (MAC-d, RLC, PDCP) are not modified. The ARQ mechanism operates between the Node B and the terminal (UE).

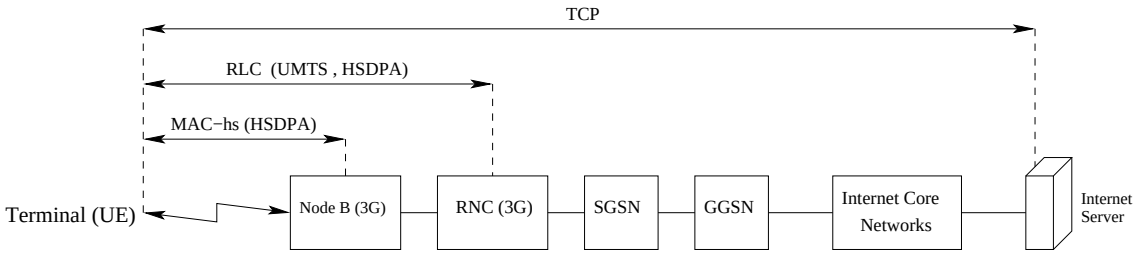

FIGURE 2 – Possible retransmission levels in a TCP connection over 3G networks.

The ARQ technique introduced in HSPDA is called the N-channel Stop-and-Wait ARQ. In the N-channel Stop and Wait, N Stop and Wait (SW) processes are used in parallel and run independently from one another. Using this strategy, the retransmission process behaves as if SR ARQ were used, but since the processes run independently from one another, a persistent failure in a packet transmission doesn't affect the whole transmission but only the transmission in the corresponding logical channel or SW process, whereas in the traditional SR ARQ, a persistent failure in a packet transmission prevents the ARQ window from advancing and prevents all communication.

Up to 8 SW instances may be used. A 3-bit ARQ Id is used to identify the process a packet belongs to and a 6-bit Transmission Sequence Number (TSN) is used to identify the position of the packet in the process it belongs to, all included in the new MAC entity

(MAC-hs or MAC-high speed) header.

4 Retransmission mechanisms in an end-to-end connection over an IEEE 802.16-2004 WiMAX network

The ARQ mechanism is part of the MAC and operates between the Base Station (BS) and the terminal (MS or SS). It is enabled on a per connection basis. The use of ARQ shall be specified and negotiated during connection creation.

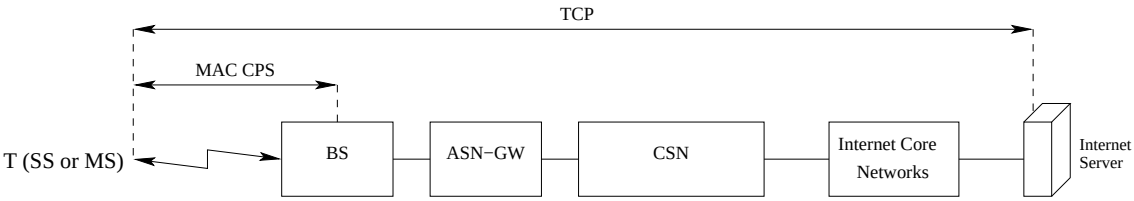

FIGURE 3 – Possible retransmission levels in a TCP connection over a WiMAX network.

A MAC SDU is *logically* partitioned into blocks whose length is specified by a TLV (Type Length Value) parameter called ARQ-BLOCK-SIZE. When the length of the SDU is not an integer multiple of the connection's block size, the final block of the SDU is formed using the SDU bytes remaining after the final full block has been determined. Fragmentation shall occur only on ARQ block boundaries (see Figs. 4 and 5).

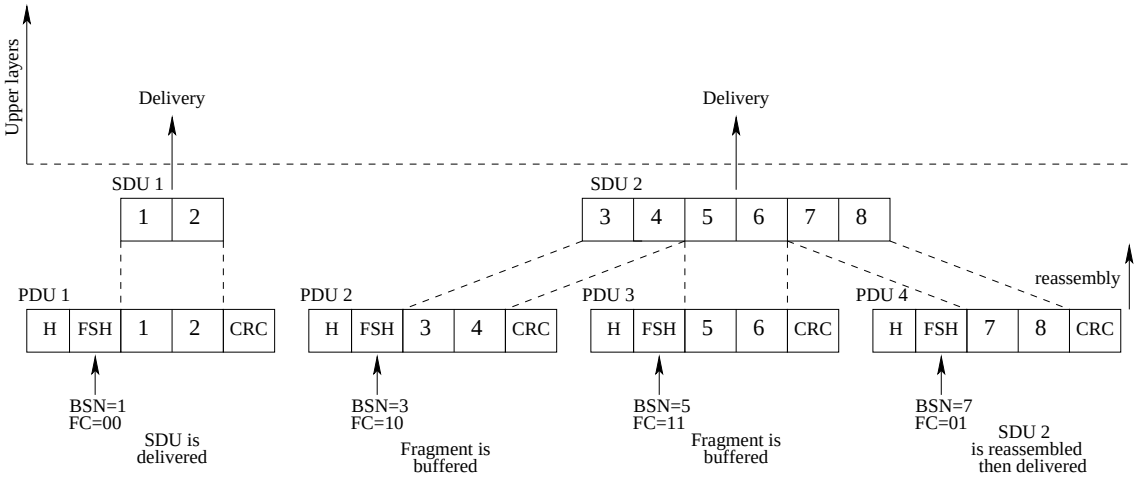

FIGURE 4 – Block partitioning and SDU reconstruction in case packing is not used and ARQ is used.

If ARQ is enabled at the connection, Fragmentation and Packing subheaders contain a BSN (Block Sequence Number), which is the sequence number of the first ARQ block in the data following the subheader (see Figs. 4 and 5). It is a matter of transmitter policy whether or not a set of blocks once transmitted as a single PDU should be retransmitted as a single PDU. Figure 6 illustrates the use of blocks for ARQ transmissions and retrans-

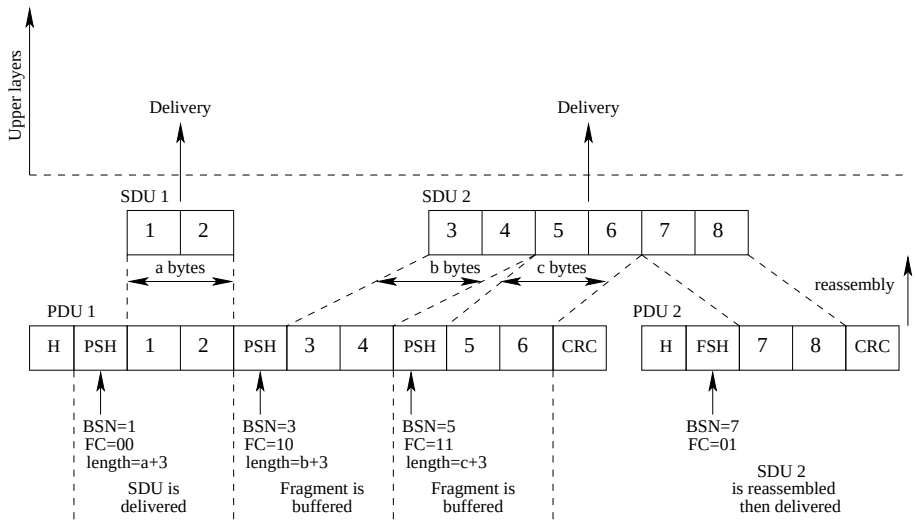

FIGURE 5 – Block partitioning and SDU reconstruction in case packing is used and ARQ is used.

missions ; two options for retransmission are presented : With and without rearrangement of blocks.

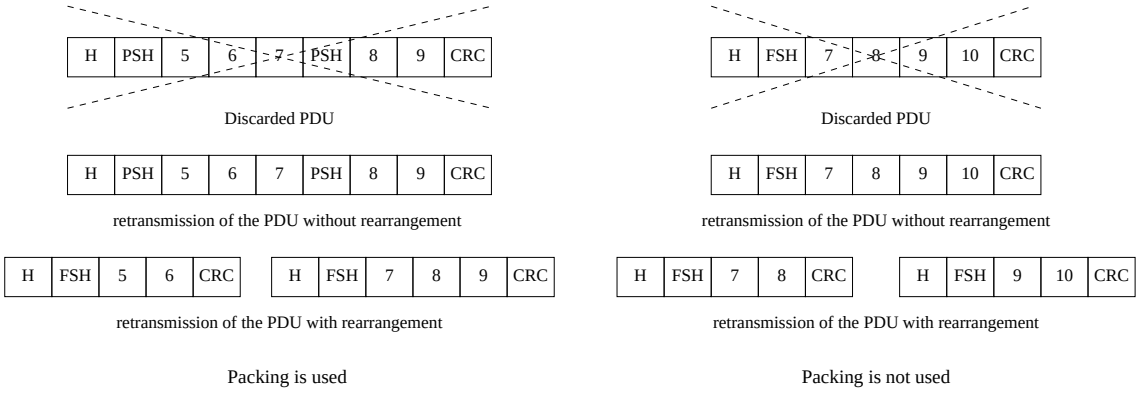

FIGURE 6 – PDU retransmission with and without rearrangement.

In the sequel, for the sake of simplicity, we will only consider that retransmissions are performed without rearrangement, i.e. if a PDU is discarded, the same PDU is retransmitted.

In addition to the ARQ-BLOCK-SIZE, a set of other ARQ parameters define the rules of the ARQ mechanism :

ARQ-BSN-MODULUS ARQ-BSN-MODULUS is equal to the number of unique BSN values, i.e. 2^{11}.

ARQ-WINDOW-SIZE ARQ-WINDOW-SIZE is the maximum number of unacknowledged ARQ blocks at any given time. ARQ-WINDOW-SIZE (WS) shall be less than or equal to half of the ARQ-BSN-MODULUS, that is WS=ARQ-WINDOW-SIZE$\leq$ 1024.

ARQ-RETRY-TIMEOUT ARQ-RETRY-TIMEOUT is the minimum time a transmitter shall wait before retransmission of an unacknowledged block. The interval begins when the ARQ block was last transmitted (see Fig. 7).

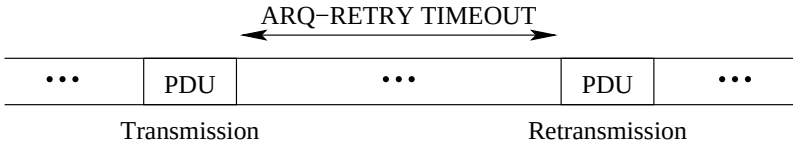

FIGURE 7 – Use of ARQ-RETRY-TIMEOUT.

Transmit window The TRANSMIT-WINDOW-START (TWS) points to the lowest numbered ARQ block that has not been ACKed. The window is advanced when an ACK for the block the BSN of which is equal to TWS is received (see Fig. 8).

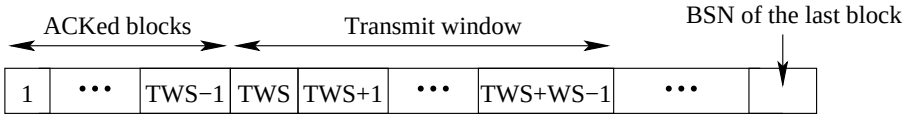

FIGURE 8 – Transmit window.

The size of the transmit window is ARQ-WINDOW-SIZE. Hence the BSN of the next block to send (NBSN) shall be comprised between TWS and TWS+WS-1.

The window is advanced when an ACK for the block the BSN of which is equal to TWS is received.

ARQ-BLOCK-LIFETIME ARQ-BLOCK-LIFETIME is the maximum time interval an ARQ block shall be managed by the transmitter once initial transmission of the block has occurred. If transmission (or subsequent retransmission) of the block is not acknowledged by the receiver before the time limit is reached, the block is discarded (see Fig. 9).

A discard message (DM) is sent following violation of ARQ-BLOCK-LIFETIME. Following the first transmission of the discard message, subsequent discard orders are sent to the receiver at intervals of ARQ-RETRY-TIMEOUT.

Discard orders for adjacent BSN values may be accumulated in a single discard message as in the example below where the discard orders of blocks numbered i, $(i+1)$ and $(i+2)$ are accumulated in the same DM.

Receive window The RECEIVE-WINDOW-START (RWS) points to the lowest numbered ARQ block that has not been marked as correctly received (see Fig. 10).

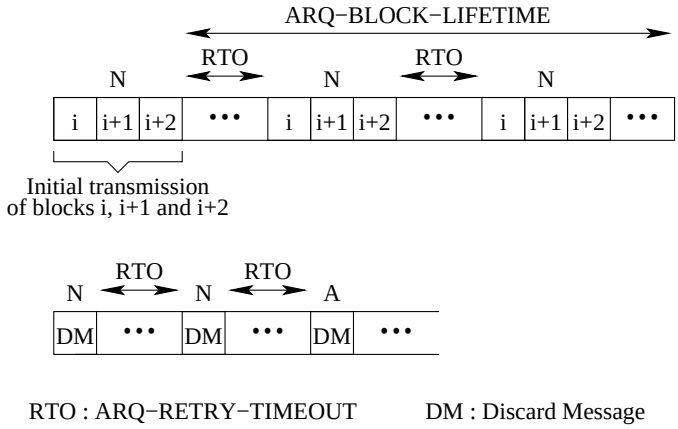

FIGURE 9 – Discard orders following violation of ARQ-BLOCK-LIFETIME (retransmission without rearrangement is assumed).

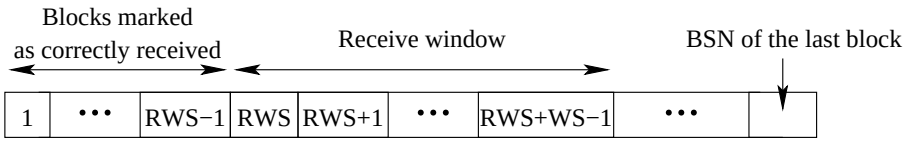

FIGURE 10 – Receive window.

Only PDU's with valid BSN's are ACKed. A valid BSN is comprised between RWS and RWS+WS-1 (only blocks in the received window are ACKed).

The received window is advanced :

1. When the block with the BSN equal to RWS is correctly received.
2. When a discard message (following violation of ARQ-BLOCK-LIFETIME) is received. In this case, the block(s) in question is (are) *marked* as correctly received but will of course not be available at the receiver (see Fig. 11).

ARQ-RX-PURGE-TIMEOUT ARQ-RX-PURGE-TIMEOUT is the time interval the receiver shall wait after successful reception of a block that does not result in advancement of RWS, before advancing RWS (see Fig. 12).

SDU reconstruction and delivery An SDU is reconstructed as soon as all blocks of the MAC SDU have been correctly received (within the defined timeout values). If blocks are marked as correctly received due to timeout violation (PURGE or BLOCK-LIFETIME), the SDU is discarded.

If ARQ-DELIVERY-IN-ORDER is not enabled : The MAC SDU is handed to the upper layers as soon as the MAC SDU is reconstructed.

If ARQ-DELIVERY-IN-ORDER is enabled : The MAC SDU is delivered to the upper layers as soon as it is reconstructed and all MAC SDUs the blocks of which have

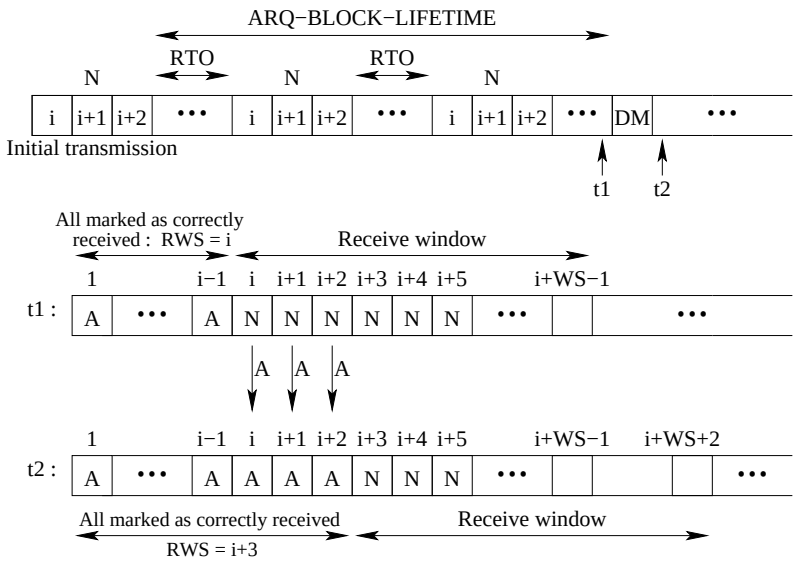

FIGURE 11 – Advancement of the receive window when a Discard Message is received.

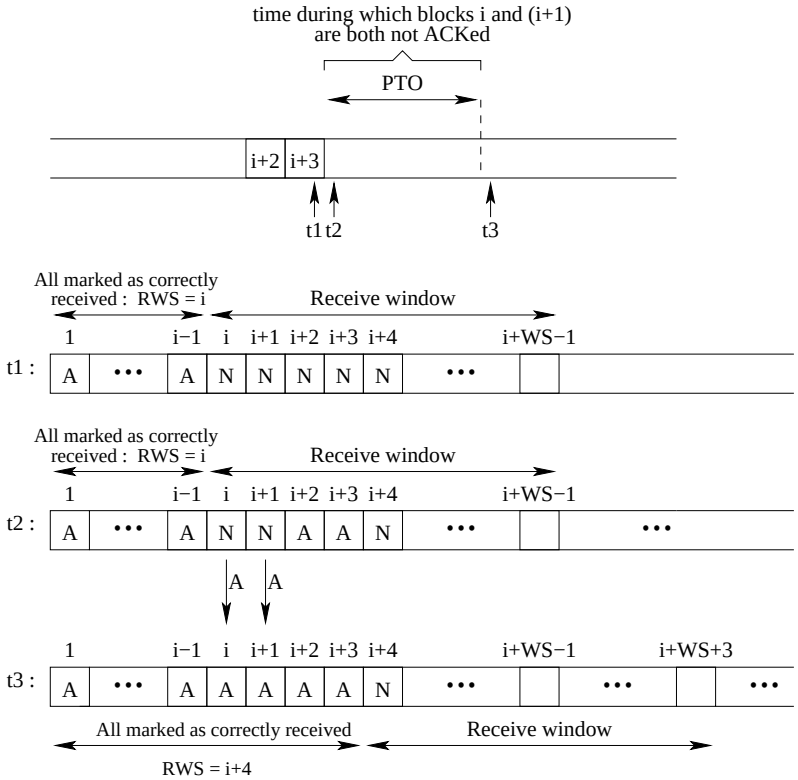

FIGURE 12 – Advancement of the receive window after an ARQ-RX-PURGE-TIMEOUT (PTO).

sequence numbers smaller than those of the reconstructed SDU have either been delivered or discarded.

Appendix C : CAVLC encoding of prediction residuals

1 Introduction

This appendix describes in detail the method used to encode residual, zig-zag ordered 4×4 and 2×2 DC chrominance) blocks of transform coefficients. It is designed to take advantage of several characteristics of quantized 4×4 blocks :

1. After prediction, transformation and quantization, blocks are typically sparse (containing mostly zeros). CAVLC uses run-level coding to represent strings of zeros compactly.
2. The highest non-zero coefficients after the zig-zag scan are often sequences of ± 1 and CAVLC signals the number of ± 1 coefficients (T1s) in a compact way.
3. The number of non-zero coefficients in neighbouring blocks is correlated. The number of coefficients is encoded using a look-up table and the choice of the look-up table depends on the number of non-zero coefficients in neighbouring blocks.
4. The magnitude of non-zero coefficients tends to be larger at the start of the reordered array (near the DC coefficient) and smaller towards the higher frequencies. CAVLC takes advantage of this by adapting the choice of VLC table for the level parameter depending on recently-coded level magnitudes.

The encoding of the blocks follows the 5 steps detailed in the sections below.

2 Encoding of *TotalCoeffs* and *TrailingOnes*

The first VLC, *CoeffToken*, encodes both the total number of non-zero coefficients (*TotalCoeffs*) and the number of T1s (*TrailingOnes*). *TotalCoeffs* can be anything from 0 to 16 and *TrailingOnes* can be anything from 0 to 3. If there are more than three T1s, only the last three are treated as 'special cases' and any others are coded as normal coefficients.

There are 4 choices of look-up table to use for encoding *CoeffToken* for a 4×4 block, three variable-length code tables and a fixed-length code table. The choice of table depends on the number of non-zero coefficients in the left-hand and upper previously coded blocks (N_U and N_L respectively). A parameter N is calculated as follows

– If upper and left blocks are both available, $N = (N_U + N_L)/2$.
– If only the upper block is available, $N = N_U$.
– If only the left block is available, $N = N_L$.
– If neither is available, $N = 0$

The parameter N selects the look-up table (see Figs. 1 and 2) so that the choice of VLC adapts to the number of coded coefficients in neighbouring blocks (*context adaptive*). The first table is biased towards small numbers of coefficients such that low values of *TotalCoeffs* are assigned particularly short codes and high values of *TotalCoeffs* particularly long codes. The second table is biased towards medium numbers of coefficients (*TotalCoeffs* values around 2-4 are assigned relatively short codes). The third table is biased towards higher numbers of coefficients and the fourth table assigns a fixed 6-bit code to every pair of *TotalCoeffs* and *TrailingOnes* pair of values.

3 Encoding of T1s signs

For each T1 (trailing ± 1), the sign is encoded with a single bit (0 for + and 1 for −) *in reverse order*, starting with the highest frequency T1.

$TrailingOnes$	$TotalCoeffs$	$0 \leq N < 2$	$2 \leq N < 4$	$4 \leq N < 8$	$N \geq 8$	$N = -1$
0	0	1	11	1111	000011	01
0	1	000101	001011	001111	000000	000111
1	1	01	10	1110	00001	1
0	2	00000111	000111	001011	000100	000100
1	2	000100	00111	01111	000101	000110
2	2	001	011	1101	000110	001
0	3	0000000111	000011	001000	001000	0000111
1	3	00000110	001010	01100	001001	0000011
2	3	0000101	001001	01110	001010	0000010
3	3	00011	0101	1100	001011	000101
0	4	0000000111	0001111	0001111	001100	000010
1	4	000000110	000110	01010	0001101	00000011
2	4	00000101	000101	01011	001110	00000010
3	4	000011	0100	1011	001111	0000000
0	5	00000000111	00000100	0001011	010000	-
1	5	0000000110	0000110	01000	010001	-
2	5	000000101	0000101	01001	010010	-
3	5	0000100	00110	1010	010011	-
0	6	0000000001111	000000111	0001001	010100	-
1	6	00000000110	00000110	001110	010101	-
2	6	00000000101	00000101	001101	010110	-
3	6	00000100	001000	1001	010111	-
0	7	0000000001011	00000001111	0001000	011000	-
1	7	0000000001110	000000110	001010	011001	-
2	7	00000000101	000000101	001001	011010	-
3	7	000000100	000100	1000	011011	-
0	8	0000000001000	00000001011	00001111	011100	-
1	8	0000000001010	00000001110	0001110	011101	-
2	8	0000000001101	00000001101	0001101	011110	-
3	8	0000000100	0000100	01101	011111	-
0	9	00000000001111	000000001111	00001011	100000	-

FIGURE 1 – VLC tables used for the encoding $TotalCoeffs$ and $TrailingOnes$.

4 Encoding of *Levels*

The sign and magnitude of each remaining non-zero coefficient (*Level*) in the block is encoded *in reverse order*, starting with the highest frequency and working back towards the DC coefficient. The sign and the magnitude are first incorporated in the *Code* parameter according to equations (1) and (2) where $\ll$ represents a 1-bit left shift operation and $\cup$ a logical OR. Thus, the sign is encoded on the least significant bit and the magnitude on the most significant bits.

$$Code = (magnitude - 1) \ll 1 \tag{1}$$

$$Code = Code \cup sign \tag{2}$$

$TrailingOnes$	$TotalCoeffs$	$0 \leq N < 2$	$2 \leq N < 4$	$4 \leq N < 8$	$N \geq 8$	$N = -1$
1	9	00000000001110	00000001010	00001110	100001	-
2	9	0000000001001	000000001001	0001010	100010	-
3	9	00000000100	000000100	001100	100011	-
0	10	00000000001011	000000001011	000001111	100100	-
1	10	00000000001010	000000001110	00001010	100101	-
2	10	00000000001101	000000001101	00001101	100110	-
3	10	0000000001100	000000001100	0001100	100111	-
0	11	000000000001111	000000001000	000001011	101000	-
1	11	000000000001110	000000001010	000001110	101001	-
2	11	00000000001001	000000001001	00001001	101010	-
3	11	00000000001100	00000001000	00001100	101011	-
0	12	000000000001011	00000000001111	000001000	101100	-
1	12	000000000001010	00000000001110	000001010	101101	-
2	12	000000000001101	00000000001101	000001101	101110	-
3	12	00000000001000	000000001100	00001000	101111	-
0	13	0000000000001111	0000000001011	0000001101	110000	-
1	13	000000000000001	0000000001010	000000111	110001	-
2	13	000000000001001	0000000001001	000001001	110010	-
3	13	000000000001100	0000000001100	000001100	110011	-
0	14	0000000000001011	0000000000111	0000001001	110100	-
1	14	0000000000001110	00000000001100	0000001100	110101	-
2	14	0000000000001101	0000000000110	0000001011	110110	-
3	14	000000000001000	0000000001000	0000001010	110111	-
0	15	0000000000000111	00000000001001	0000000101	111000	-
1	15	0000000000001010	00000000001000	0000001000	111001	-
2	15	0000000000001001	00000000001010	0000000111	111010	-
3	15	0000000000001100	0000000000001	0000000110	111011	-
0	16	0000000000000100	00000000000111	0000000001	111100	-
1	16	0000000000000110	00000000000110	0000000100	111101	-
2	16	0000000000000101	00000000000101	0000000011	111110	-
3	16	0000000000001000	00000000000100	0000000010	111111	-

FIGURE 2 – VLC tables used for the encoding $TotalCoeffs$ and $TrailingOnes$ (continued).

The $Code$ parameter is then decomposed into 2 syntax elements : $LevelPrefix$ and $LevelSuffix$. These elements are determined according to equations (3) and (4) where $\gg$ represents a 1-bit right shift and $Code(Shift)$ represents the $Shift$ least significant bits of the parameter $Code$.

$$LevelPrefix = Code \gg Shift \tag{3}$$

$$LevelSuffix = Code(Shift) \tag{4}$$

Finally, the $LevelPrefix$ is encoded using the VLC table in Fig. 3 and the $Shift$ bits of $LevelSuffix$ are appended next.

$LevelPrefix$	Codeword
0	1
1	01
2	001
3	0001
4	00001
5	000001
6	0000001
7	00000001
8	000000001
9	0000000001
10	00000000001
11	000000000001
12	0000000000001
13	00000000000001
14	000000000000001
15	0000000000000001

FIGURE 3 – VLC table used for the encoding of $LevelPrefix$.

$Shift$	Threshold value
0	0
1	3
2	6
3	12
4	24
5	48
6	∞

FIGURE 4 – Threshold values associated to variable $Shift$.

The adaptability principle in the encoding of the *Levels* lies in the variable $Shift$. This value generally changes during the encoding of the *Levels*. It is initialized to 0 at the first step and then incremented if the magnitude of the last encoded coefficient is greater than the threshold value associated to the current value of $Shift$ (see Fig. 4). Given that the magnitudes have tendency to increase as the frequency decreases, this technique proves very efficient.

The decoding operation consists of several steps. The decoder first extracts $LevelPrefix$ and $LevelSuffix$ from the stream based on the current value of $Shift$. Then, by inverting equations (1), (2), (3) and (4), the decoder recovers the *magnitude* and the *sign* of the current coefficient. The decoder updates the value of $Shift$ at each iteration.

$TotalZeros$	$TotalCoeffs$						
	1	2	3	4	5	6	7
0	1	111	0101	00011	0101	000001	000001
1	011	110	111	111	0100	00001	00001
2	010	101	110	0101	0011	111	101
3	0011	100	101	0100	111	110	100
4	0010	011	0100	110	110	101	011
5	00011	0101	0011	101	101	100	11
6	00010	0100	100	100	100	011	010
7	000011	0011	011	0011	011	010	0001
8	000010	0010	0010	011	0011	0001	001
9	0000011	00011	00011	0010	00001	001	000000
10	0000010	00010	00010	00010	0001	000000	-
11	00000011	000011	000001	00001	00000	-	-
12	00000010	000010	00001	00000	-	-	-
13	000000011	000001	00000	-	-	-	-
14	000000010	000000	-	-	-	-	-
15	000000001	-	-	-	-	-	-

$TotalZeros$	$TotalCoeffs$							
	8	9	10	11	12	13	14	15
0	000001	000001	00001	0000	0000	000	00	0
1	0001	000000	00000	0001	0001	001	01	1
2	0001	0001	001	001	01	1	1	-
3	011	11	11	010	1	01	-	-
4	11	10	10	1	001	-	-	-
5	10	001	01	011	-	-	-	-
6	010	01	0001	-	-	-	-	-
7	001	00001	-	-	-	-	-	-
8	000000	-	-	-	-	-	-	-

FIGURE 5 – VLC tables used for the encoding of $TotalZeros$ in 4×4 blocks.

5 Encoding of *TotalZeros*

The *TotalZeros* parameter defines the number of zero coefficients preceding the last non-zero coefficient. The VLC tables of Fig. 5 are used with 4×4 blocks and those in Fig. 6 are used with 2×2 blocks. In both cases, the table is chosen as a function of the value of *TotalCoeffs*. The number of possible values for *TotalZeros* decreases as the value of *TotalCoeffs* increases.

6 Encoding of *RunBefores*

The number of zeros preceding each non-zero coefficient (*RunBefore*) is encoded *in reverse order*. A *RunBefore* parameter is encoded for each non-zero coefficient, starting with the highest frequency, with two exceptions :

1. If there are no more zeros left to encode, it is not necessary to encode any more

$TotalZeros$	$TotalCoeffs$		
	1	2	3
0	1	1	1
1	01	01	0
2	001	00	-
3	000	-	-

FIGURE 6 – VLC tables used for the encoding of $TotalZeros$ in 2×2 blocks.

RunBefore values.

2. It is not necessary to encode *RunBefore* for the final (lowest frequency) non-zero coefficient.

The VLC tables used to encode the *RunBefores* are illustrated in Fig. 7. The table used for the encoding of each *RunBefore* is chosen depending on the number of zeros (signalled by *TotalZeros*) that have not yet been encoded (*ZerosLeft*). *ZerosLeft* is updated after each encoding and represents the adaptation parameter. For example, if there are only two zeros left to encode, *RunBefore* can only take 3 values (0, 1 and 2) and so the VLC need not be more than two bits long. If there are six zeros still to encode then *RunBefore* can take seven values (0 to 6) and the VLC table needs to be correspondingly larger.

$RunBefore$	$ZerosLeft$						
	1	2	3	4	5	6	> 6
0	1	1	11	11	11	11	111
1	0	01	10	10	10	000	110
2	-	00	01	01	011	001	101
3	-	-	00	001	010	011	100
4	-	-	-	000	001	010	011
5	-	-	-	-	000	101	010
6	-	-	-	-	-	100	001
7	-	-	-	-	-	-	0001
8	-	-	-	-	-	-	00001
9	-	-	-	-	-	-	000001
10	-	-	-	-	-	-	0000001
11	-	-	-	-	-	-	00000001
12	-	-	-	-	-	-	000000001
13	-	-	-	-	-	-	0000000001
14	-	-	-	-	-	-	00000000001

FIGURE 7 – VLC tables used for the encoding of $RunBefores$.

Example Consider the encoding of the following 4×4 block

0	3	-1	0
0	-1	1	0
1	0	0	0
0	0	0	0

The 4×4 block is mapped in a zig-zag order to the following 16-element array

0	3	1	-1	-1	0	1	0	0	0	0	0	0	0	0	0

The main parameters are

$TotalCoeffs = 5$ (indexed from highest frequency, 4, to lowest frequency, 0).

$TotalZeros = 3$.

$TrailingOnes = 3$ (in fact there are 4 T1s but only three can be encoded as a special case).

Assuming that the first table in Figs. 1 and 2 is used for the encoding of $CoeffToken$ (i.e $0 \leq N < 2$), the encoding yields the following results

Element	Value	Codeword	Reference
$CoeffToken$	$TotalCoeffs = 5$, $TrailingOnes = 3$	0000100	Fig. 1 (Tab. 1)
T1 sign (4)	$+$	0	
T1 sign (3)	$-$	1	
T1 sign (2)	$-$	1	
$Level$ (1)	$+1$ ($Shift = 0$)	1	Fig. 3
$Level$ (0)	$+3$ ($Shift = 1$)	0010	Fig. 3
$TotalZeros$	3	111	Fig. 5 (Tab. 5)
$RunBefore$ (4)	$ZerosLeft = 3$, $RunBefore = 1$	10	Fig. 7 (Tab. 3)
$RunBefore$ (3)	$ZerosLeft = 2$, $RunBefore = 0$	1	Fig. 7 (Tab. 2)
$RunBefore$ (2)	$ZerosLeft = 2$, $RunBefore = 0$	1	Fig. 7 (Tab. 2)
$RunBefore$ (1)	$ZerosLeft = 2$, $RunBefore = 1$	01	Fig. 7 (Tab. 2)
$RunBefore$ (0)	$ZerosLeft = 1$, $RunBefore = 1$	Not required	Last coefficient

The transmitted bitstream for this block is 000010001110010111101101

Appendix D : Joint Exploitation of Residual Source Information and MAC Layer CRC Redundancy for Robust Video Decoding

Joint Exploitation of Residual Source Information and MAC Layer CRC Redundancy for Robust Video Decoding

Cédric Marin, Khaled Bouchireb, Michel Kieffer, *Senior Member, IEEE*, and Pierre Duhamel, *Fellow, IEEE*

***Abstract*—This paper presents a MAP estimation method allowing the robust decoding of compressed video streams by exploiting the bitstream structure (*i.e.*, information about the source, related to variable-length codes and source characteristics) together with the knowledge of the MAC layer CRC (here considered as additional redundancy on the MAC packet). This method is implemented via a sequential decoding algorithm in which the branch selection metric in the decoding trellis incorporates a CRC-dependent factor, and the paths which are not compatible with the source constraints are pruned. A first implementation of the proposed algorithm performs exact computations of the metrics, and is thus computationally expensive. Therefore, we also introduce a suboptimal (with tunable complexity) version of the proposed metric computation. This technique is then applied to the robust decoding of sequences encoded using the H.264/AVC standard based on CAVLC, and transmitted using a WiFi-like packet structure. Significant link budget improvement results are demonstrated for BPSK modulated signals sent over AWGN channels, even in the presence of channel coding.**

***Index Terms*—Communication systems, MAP estimation, video coding, sequential decoding, codes.**

I. Introduction

WIRELESS channels present a major challenge for high bitrate transmission. Factors such as signal attenuation, multiple access interference, inter-symbol interference, and Doppler shift can heavily degrade signal quality. Consequently, the typical BER encountered in mobile transmission can be several orders of magnitude higher than in wire line (*e.g.*, DSL) transmission.

High efficiency video transmission is usually dependent on the compression mechanism applied to the image stream [28]. Nevertheless, the compressed video flow is very sensitive to transmission errors. A single error can lead to a decoder desynchronization resulting in a total loss of remaining picture information or to inter-image error propagation due to inter-picture coding. Consequently, the video stream incoming in the video decoder has to be nearly error-free.

In wireless transmission, the received signal may be heavily corrupted and is not directly usable by the video decoder. A first solution to alleviate this problem consists in grouping data into packets protected by an error-detection code (CRC or checksum) [5], [16]. Packets for which integrity is not ensured at receiver side may then be retransmitted. Nevertheless, retransmissions may become difficult in scenarii with strong delay constraints (*e.g.*, for visiophony), or even impossible when broadcasting data (*e.g.*, in satellite television).

In such situations, the standard solution is to make use of very strong error-correction codes (*e.g.*, turbo codes, LDPC) at the *Physical* (PHY) layer combined with packet-erasure codes (*e.g.*, Reed-Solomon) at intermediate protocol layers [19], [26]. However, due to the high channel variability, redundancy is rarely optimally dimensioned. It may be oversized when the channel is clear, reducing the bandwidth allocated for the data. In contrary, some corrupted packets cannot be recovered in bad channel conditions and are lost. Error-concealment techniques [9], [15] may then be used by the source decoders at the *Application* (APL) layer. They exploit the redundancy (temporal and/or spatial) in the decoded multimedia stream for estimating the missing information. However, even if very efficient for providing a video of acceptable visual quality, error concealment cannot replace a clean reception in terms of quality.

In the last years, joint source-channel decoding techniques have been proposed to correct damaged packets. Such methods involve robust source decoders, which exploit the inherent redundancy in the received packets for correcting errors. Several types of redundancy have been identified. Constraints in the syntax of variable-length codes [7], [8], [14], [24], [31] have been used first. Then, the properties due to the semantic of the source coders have been combined along with the syntax redundancy to improve the performance of robust decoders [4], [22], [27], [32]. Redundancy associated to the packetization of encoded data have been introduced in [18]. Recently, information introduced by the channel codes have been jointly employed together with the residual redundancy through iterative decoding processes [3], [21], [30]. These joint schemes improve the decoding performance when compared to classical schemes.

This paper focuses on robust decoding of video data in a downlink situation. We propose a sequential decoding algorithm jointly exploiting the syntax and semantic properties of the encoded video stream together with the redundancy at MAC layer provided by the CRC. Here, the CRC is not only used to detect errors but is also considered as an error correcting code. This CRC based decoding approach has been presented in [17], [23], [29] for correcting erroneous

Manuscript received March 9, 2009; revised November 18, 2009; accepted February 10, 2010. The associate editor coordinating the review of this paper and approving it for publication was V. Bhargava.

The authors are with the L2S – CNRS - SUPELEC - Univ Paris-Sud, France (e-mail: kieffer@lss.supelec.fr).

Digital Object Identifier 10.1109/TWC.2010.090238

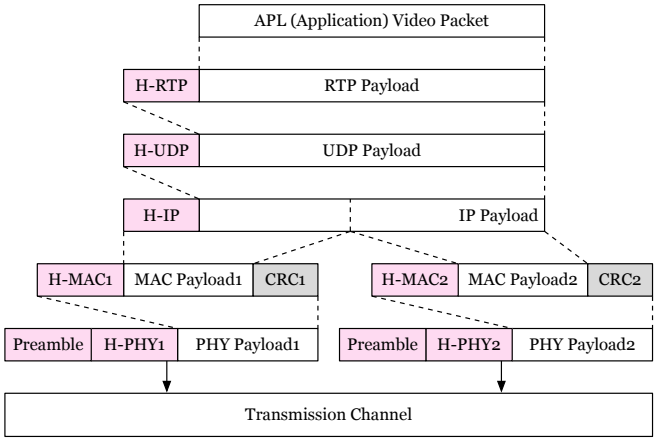

Fig. 1. Protocol stack for video transmission over WiFi.

packets. The main contribution of this paper is to make a simultaneous usage of the CRC and the source redundancy to improve the video decoding performance. This paper is based on a variety of techniques (soft decoding of block codes [2], sequential decoding [1], source decoding depending on syntax and semantic of bitstream [4]) which are combined to attain our objective.

Note that all robust techniques introduced above require soft information to be delivered from the PHY layer to the APL layer. Obviously, this does not correspond to a classical structure of the decoder, and requires the use of some additional tools, some of which being proposed elsewhere by the same authors. In particular, we proposed a header recovery technique exploiting the intra-layer and inter-layer redundancies along with the CRCs or checksums in [20]. With this technique, the header is very likely to be correctly decoded even for poor SNRs, and the payload may be forwarded to the upper layers, resulting in a *permeable* protocol stack. A complementary work, introducing a *transparent* network architecture, may be found in [10]. In this paper, we assume that, due to the use of such techniques, the headers are correctly received, and we concentrate on the evaluation of the payload (*i.e.*, the reception of the video)

This paper is organized as follows. After a brief description of the permeable protocol stack model in Section II, Section III describes the derivation of the decoding metric and proposes a general sequential decoding method. Reduction of complexity is presented in Section IV. Finally, the simulation results are described in Section V before drawing some conclusions.

II. Model of Permeable Protocol Stack

Multimedia packetized transmission usually relies on a multi-layer architecture [16] based on the RTP/UDP/IP stack.

Figure 1 illustrates an example of the segmentation and encapsulation mechanisms implemented at each protocol layer in the case of video transmitted with a WiFi radio interface [11] (802.11 standard). The data processed by the PHY layer are forwarded to the MAC layer which checks their integrity with the help of some CRC. For corrupted packets, a retransmission is requested. Correctly received data are assembled to form the binary stream that is then fed to the video decoder (at APL layer) after removal of IP, UDP, and RTP protocol headers.

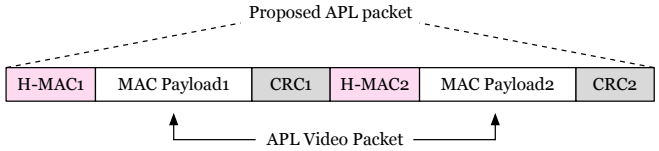

Fig. 2. New format of the APL packet at the input of the decoder. In this example, the original APL packet has been fragmented in two MAC packets.

A protocol stack design where the PHY, MAC, and APL layers of the receiver work very closely together is presented here. Three changes are required to implement the proposed solution:

- The PHY layer includes a *Soft-Input Soft-Output* (SISO) channel decoder for processing the incoming protected data. The soft information are transmitted to the next layer.
- In the MAC layer, the CRC check is deactivated and no retransmission is allowed. Complete MAC packets (composed of header, payload, and CRC) are transferred to the upper layer for being integrated in the payload of IP packets.
- The MAC header and MAC CRC which usually are not transmitted by the IP, UDP, and RTP layers, are now assumed to be available at the APL layer in the form of *soft* values.

These changes require some information to be available everywhere inside the receiver and are compatible with the usual transmission structure: only the receiver has to be modified, and both the transmitter operations and the signal sent are unchanged. As outlined above, they are facilitated by using the robust header recovery and permeable layer mechanisms presented in [10], [20]. Here, the headers are assumed to be available without errors at all layers.

With these modifications, the APL layer receives a succession of MAC packets, containing soft information (provided by the PHY layer). The format of data received by the APL layer is depicted in Fig. 2.

In the proposed architecture, the CRC still plays some error-detection role, used to minimize the computational complexity: its first use is to deactivate the robust decoding process (which is computationally expensive) when:

1) normal CRC check is successful,
2) the quality of soft information provided by the lower layer is too poor, *i.e.*, when the signal power is smaller than a pre-defined threshold. In such a case, the packet is discarded (or retransmitted, see [6]).

The next section presents the analytical derivation of the decoding metric which may be used for robust reconstruction of the transmitted video sequence. We then propose a sequential decoding algorithm based on this metric.

III. Group-Based Sequential Decoding

A. Notations

The symbols produced by a video coder before entropy coding are assumed to be generated by a source $\mathcal{S}$, which

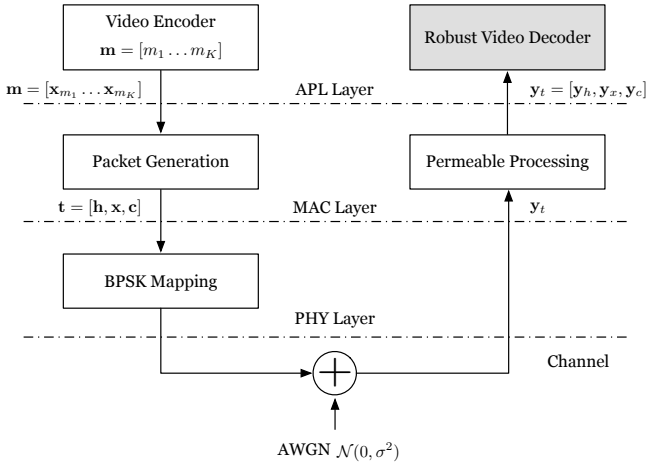

Fig. 3. Overview of the transmission scheme.

has to satisfy some semantic rules. Consider a vector $\mathbf{m} = [m_1 \dots m_K]$ of K symbols generated by this source. The entropy coder associates a variable-length codeword $\mathbf{x}_{m_i}$ to each component m_i of $\mathbf{m}$, $i = 1 \dots K$, which is then mapped onto a binary sequence $\mathbf{x} = [\mathbf{x}_{m_1} \dots \mathbf{x}_{m_K}]$, with

$$\sum_{i=1}^{K} \ell(\mathbf{x}_{m_i}) = \ell(\mathbf{x}). \tag{1}$$

In (1) and in what follows, $\ell(\mathbf{v})$ denotes the length in bits of the vector $\mathbf{v}$. Thus, $\mathbf{x}$ has to be compliant with the syntax of the variable-length code (VLC) and with the semantic rules of the source $\mathcal{S}$.

At MAC layer, a header $\mathbf{h}$ is added at the beginning of the *payload* $\mathbf{x}$, resulting in a concatenated vector $\mathbf{d} = [\mathbf{h}, \mathbf{x}]$. A CRC $\mathbf{c}$ is then computed from the data $\mathbf{d}$ and appended to $[\mathbf{h}, \mathbf{x}]$ to form a MAC *packet*. This set of information is collected in a vector $\mathbf{t} = [\mathbf{h}, \mathbf{x}, \mathbf{c}] = [\mathbf{d}, \mathbf{c}]$, where $\mathbf{c} = \mathcal{F}(\mathbf{d})$, $\mathcal{F}$ being a generic encoding function.

The computation of $\mathbf{c}$ depends on some generator polynomial $g(z) = \sum_{i=0}^{\ell(\mathbf{c})} a_i z^i$ characterizing the CRC [5]. A systematic generator matrix $\mathbf{G} = [\mathbf{I}, \mathbf{\Pi}]$ may be associated to $g(z)$. Using $\mathbf{G}$, $\mathbf{c}$ may be determined by a recursive processing over the $\ell(\mathbf{d})$ bits of $\mathbf{d}$ as follows

$$\mathbf{c}^{j+1} = \mathcal{F}(\mathbf{d}^{j+1}) = \mathbf{c}^j \oplus (d_{j+1} \cdot \boldsymbol{\pi}(d_{j+1})). \tag{2}$$

In (2), $\mathbf{d}^j = [d_1 \dots d_j, 0 \dots 0]$, $\boldsymbol{\pi}(d_j)$ is the j-th row of $\mathbf{\Pi}$, *i.e.*, the parity vector related to d_j, and $\oplus$ represents the XOR operator. At initialization, $\mathbf{c}^0$ is set to $\mathbf{0}$. After $\ell(\mathbf{d})$ iterations, the vector $\mathbf{c}^{\ell(\mathbf{d})}$ contains the CRC value related to $\mathbf{d}$ (*i.e.*, $\mathbf{c}^{\ell(\mathbf{d})} = \mathbf{c}$).

In our model, vector $\mathbf{t}$ is then BPSK-modulated and transmitted over an AWGN channel that corrupts the modulated packets with a Gaussian noise of zero mean and variance σ^2. At the receiver, the observed vector is $\mathbf{y}_t = [\mathbf{y}_h, \mathbf{y}_x, \mathbf{y}_c]$, where $\mathbf{y}_h$, $\mathbf{y}_x$, and $\mathbf{y}_c$ are the observations of $\mathbf{h}$, $\mathbf{x}$, and $\mathbf{c}$ respectively. $\mathbf{y}_t$ contains the observations of $\mathbf{t}$ and represents a segment of the APL packet depicted in Fig. 2. An overview of the transmission scheme is illustrated in Fig. 3.

In practice, $\mathbf{x}$ is usually organized in groups of codewords (*e.g.*, texture information of a block or a macroblock), which are assumed to be encoded independently[1]. Let $\mathbf{a}_1 \dots \mathbf{a}_E$ be the E groups of codewords composing $\mathbf{x}$, *i.e.*, $\mathbf{x} = [\mathbf{a}_1 \dots \mathbf{a}_E]$. The lengths $\ell(\mathbf{a}_e)$, for $e = 1 \dots E$, are supposed to be transmitted reliably as side information to the decoder. In the following, these lengths are called *position markers*. Using these markers, the decoding of each block may be performed separately by synchronizing the decoder over the corresponding portion in the received packet.

B. Decoding Algorithm

Assuming that the header $\mathbf{h}$ has been correctly received, the optimal MAP estimator $\widehat{\mathbf{a}}_e$ for the e-th group is given by

$$\widehat{\mathbf{a}}_e = \arg \max_{\mathbf{a}_e \in \Omega_a^e} P(\mathbf{a}_e | \mathbf{h}, \mathbf{y}_x, \mathbf{y}_c), \tag{3}$$

where Ω_a^e is the set of *valid* combinations of $\mathbf{a}_e$, *i.e.*, compliant with the syntax of the VLC and the semantic of the source. Since Ω_a^e is not well structured, obtaining $\widehat{\mathbf{a}}_e$ would require constructing the $2^{\ell(\mathbf{a}_e)}$ possible combinations, keeping only the valid sequences (belonging to Ω_a^e), then evaluating $P(\mathbf{a}_e | \mathbf{h}, \mathbf{y}_x, \mathbf{y}_c)$ for each of them. When $\ell(\mathbf{a}_e)$ is large (which is usually the case since this reduces the overhead due to the transmission of the side information), a sequential decoder is involved in order to reduce the decoding complexity [1].

Consider the n-th step of the decoding of group e. One may write

$$\mathbf{x} = [\mathbf{b}_e, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{r}_{e,n}],$$

with :

- $\mathbf{b}_e = [\mathbf{a}_1 \dots \mathbf{a}_{e-1}]$, the bits of the first $e-1$ groups. Note that for the decoding of $\mathbf{a}_e$, $\mathbf{b}_e$ is considered as a random vector and not as the decoded bitstream obtained previously.
- $\mathbf{u}_{e,n}$, the first bits of $\mathbf{a}_e$ for which a set of valid combinations $\Omega_u^{e,n}$ has been evaluated at step $n-1$ by the decoder.
- $\mathbf{s}_{e,n}$, a vector for which, regardless of the syntax of the VLC and the semantic of the video coder, $2^{\ell(\mathbf{s}_{e,n})}$ binary combinations are possible. Let $\Omega_s^{e,n}$ be the set of these sequences.
- $\mathbf{r}_{e,n}$, the $\ell(\mathbf{r}_{e,n})$ remaining bits of $\mathbf{x}$. These bits have not yet been processed by the decoder but they are involved in the computation of the CRC.

Figure 4 illustrates the considered structure of the packet. The observations associated to these four vectors are $\mathbf{y}_b^e$, $\mathbf{y}_u^{e,n}$, $\mathbf{y}_s^{e,n}$, and $\mathbf{y}_r^{e,n}$. Moreover, let $\Omega_{[u,s]}^{e,n} \subset \Omega_u^{e,n} \times \Omega_s^{e,n}$ be the set of valid pairs $[\mathbf{u}_{e,n}, \mathbf{s}_{e,n}]$.

At the n-th step, the sequential decoding algorithm evaluates

$$\begin{aligned} P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n} | \mathbf{y}_b^{e,n}, \mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{y}_r^{e,n}, \mathbf{h}) \propto \\ P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{y}_b^{e,n}, \mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{y}_r^{e,n} | \mathbf{h}). \end{aligned} \tag{4}$$

for each $[\mathbf{u}_{e,n}, \mathbf{s}_{e,n}] \in \Omega_u^{e,n} \times \Omega_s^{e,n}$. In (5), one may write

$$\begin{aligned} P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{y}_b^{e,n}, \mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{y}_r^{e,n} | \mathbf{h}) = \\ \sum_{\mathbf{b}_e} \sum_{\mathbf{r}_{e,n}} P(\mathbf{b}_e, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{r}_{e,n}, \mathbf{y}_b^{e,n}, \mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{y}_r^{e,n} | \mathbf{h}). \end{aligned} \tag{5}$$

[1]In realistic situations, the groups of codewords belonging to a specific class of video coding parameters are correlated. However, we consider here that the existing dependencies are small and may be neglected.

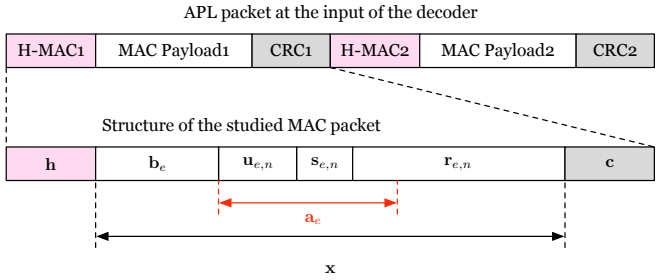

Fig. 4. Partitioning of the received MAC packet at the n-th iteration for the e-th group.

Moreover

$$P(\mathbf{b}_e, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{r}_{e,n}, \mathbf{y}_b^{e,n}, \mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{y}_r^{e,n}|\mathbf{h}) = P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n}|\mathbf{h}) P(\mathbf{y}_u^{e,n}|\mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{h}) P(\mathbf{y}_s^{e,n}|\mathbf{y}_u^{e,n}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{h}) P(\mathbf{b}_e, \mathbf{r}_{e,n}, \mathbf{y}_b^e, \mathbf{y}_r^{e,n}, \mathbf{y}_c|\mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{h}). \quad (6)$$

Using the fact that $\mathbf{u}_{e,n}$ and $\mathbf{s}_{e,n}$ do not depend on $\mathbf{h}$ and that the channel is memoryless, (6) becomes

$$P(\mathbf{b}_e, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{r}_{e,n}, \mathbf{y}_b^{e,n}, \mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{y}_r^{e,n}|\mathbf{h}) = P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n}) P(\mathbf{y}_u^{e,n}|\mathbf{u}_{e,n}) P(\mathbf{y}_s^{e,n}|\mathbf{s}_{e,n}) P(\mathbf{b}_e, \mathbf{r}_{e,n}, \mathbf{y}_b^e, \mathbf{y}_r^{e,n}, \mathbf{y}_c|\mathbf{h}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}). \quad (7)$$

Now, combining (5), (5), and (7), one obtains

$$P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n}|\mathbf{y}_b^{e,n}, \mathbf{y}_u^{e,n}, \mathbf{y}_s^{e,n}, \mathbf{y}_r^{e,n}, \mathbf{h}) \propto P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n}) P(\mathbf{y}_u^{e,n}|\mathbf{u}_{e,n}) P(\mathbf{y}_s^{e,n}|\mathbf{s}_{e,n}) \Phi(\mathbf{h}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{y}_b^e, \mathbf{y}_r^{e,n}, \mathbf{y}_c), \quad (8)$$

with

$$\Phi(\mathbf{h}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{y}_b^e, \mathbf{y}_r^{e,n}, \mathbf{y}_c) = \sum_{\mathbf{b}_e, \mathbf{r}_{e,n}} P(\mathbf{b}_e, \mathbf{r}_{e,n}, \mathbf{y}_b^e, \mathbf{y}_r^{e,n}, \mathbf{y}_c|\mathbf{h}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}). \quad (9)$$

In (8), $P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n})$ represents the *a priori* probability of sequence $[\mathbf{u}_{e,n}, \mathbf{s}_{e,n}]$, which is null if $[\mathbf{u}_{e,n}, \mathbf{s}_{e,n}] \notin \Omega_{[u,s]}^{e,n}$. As for the valid sequences, they are assumed to be equally likely *a priori*, *i.e.*, $P(\mathbf{u}_{e,n}, \mathbf{s}_{e,n}) = 1/|\Omega_{[u,s]}^{e,n}|$. Consequently, the metric $\mathcal{M}_e$ associated to a *valid* sequence in group e is given by

$$\mathcal{M}_e([\mathbf{u}_{e,n}, \mathbf{s}_{e,n}] \in \Omega_{[u,s]}^{e,n}|\mathbf{h}, \mathbf{y}_t) = P(\mathbf{y}_u^{e,n}|\mathbf{u}_{e,n}) P(\mathbf{y}_s^{e,n}|\mathbf{s}_{e,n}) \Phi(\mathbf{h}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{y}_b^e, \mathbf{y}_r^{e,n}, \mathbf{y}_c), \quad (10)$$

where $P(\mathbf{y}_u^{e,n}|\mathbf{u}_{e,n})$ and $P(\mathbf{y}_s^{e,n}|\mathbf{s}_{e,n})$ are the likelihoods of $\mathbf{u}_{e,n}$ and $\mathbf{s}_{e,n}$ respectively.

C. Implementation Issues and Complexity

In (10), $\Phi(\mathbf{h}, \mathbf{u}_{e,n}, \mathbf{s}_{e,n}, \mathbf{y}_b^e, \mathbf{y}_r^{e,n}, \mathbf{y}_c)$ is a sum the complexity of which is $\mathcal{O}(2^{\ell(\mathbf{b}_e)+\ell(\mathbf{r}_{e,n})})$. Consequently, the evaluation complexity of (5) for all $[\mathbf{u}_{e,n}, \mathbf{s}_{e,n}] \in \Omega_{[u,s]}^{e,n}$ is $\mathcal{O}(|\Omega_u^{e,n}| \cdot |\Omega_s^{e,n}| \cdot 2^{\ell(\mathbf{b}_e)+\ell(\mathbf{r}_{e,n})})$. $|\Omega_s^{e,n}|$ depends on the number of bits taken into account at the n-th steps and may thus be upper bounded by a constant. The main difficulty comes from $|\Omega_u^{e,n}|$, which is growing exponentially with n. To limit the complexity increase, at each step, only the M most probable sequences belonging to $\Omega_{[u,s]}^{e,n}$ are kept and stored in $\Omega_u^{e,n+1}$. The parameter M allows to tune the trade-off between complexity and efficiency.

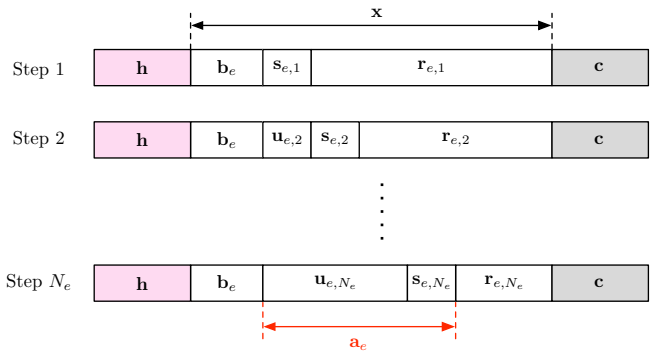

Fig. 5. Evolution of the partitions through the sequential decoding steps for the e-th group.

Figure 5 illustrates the evolution of parts $\mathbf{b}_e$, $\mathbf{u}_{e,n}$, $\mathbf{s}_{e,n}$, and $\mathbf{r}_{e,n}$ through the different steps. The flowchart of the decoding algorithm is depicted in Fig. 6 and explanations are given in the following. Note that the metric $\mathcal{M}_e([\mathbf{u}_{e,n}, \mathbf{s}_{e,n}]|\mathbf{h}, \mathbf{y}_t)$ is computed using (10). A each step, one obtains a suboptimal algorithm the complexity of which becomes $\mathcal{O}(2^{\ell(\mathbf{b}_e)+\ell(\mathbf{r}_{e,n})})$, mainly due to the evaluation of Φ in (10). Section IV describes optimal and suboptimal reduced-complexity algorithms for determining Φ and $\mathcal{M}_e$.

Let N_e be the number of steps necessary to reach the end of group e. The number of bits $\ell(\mathbf{s}_{e,n})$, for $i = 1 \dots N_e$, must thus be adjusted such that

$$\sum_{i=1}^{N_e} \ell(\mathbf{s}_{e,i}) = \ell(\mathbf{a}_e), \quad (11)$$

for all $e = 1 \dots E$. In practice, the first $N_e - 1$ decoding depths are set to a constant value and the last one, *i.e.*, $\ell(\mathbf{s}_{e,N_e})$, is chosen so that (11) is satisfied.

We now describe the complete sequential decoding algorithm for the e-th group. At initialization ($n = 1$), $\Omega_u^{e,1} = \emptyset$. Afterwards, at each step $n > 1$, the algorithm explores the new branches (on $\ell(\mathbf{s}_{e,n})$-bit depth) and only preserves the M most probable extended sequences $[\mathbf{u}_{e,n}, \mathbf{s}_{e,n}]$. These M sequences are temporarily stored in a stack (corresponding to $\Omega_u^{e,n+1}$), before being extended again at the next step.

In Section V, this algorithm is applied to the decoding of H.264/AVC CAVLC sequences.

IV. Practical Evaluation of the MAP Metric

For the sake of simplicity, the exponents e and n are omitted in what follows. Moreover, $\Phi(\mathbf{h}, \mathbf{u}, \mathbf{s}, \mathbf{y}_b, \mathbf{y}_r, \mathbf{y}_c)$ and $\mathcal{M}([\mathbf{u}, \mathbf{s}] \in \Omega_{[u,s]}|\mathbf{h}, \mathbf{y}_t)$ are replaced by Φ and $\mathcal{M}([\mathbf{u}, \mathbf{s}])$.

In (10), only the computation of Φ requires a large complexity. Assuming that the bits of $\mathbf{b}$ and $\mathbf{r}$ are i.i.d. and do not depend on $\mathbf{h}$, $\mathbf{u}$, and $\mathbf{s}$, (9) becomes

$$\Phi = \sum_{\mathbf{b}} \sum_{\mathbf{r}} P(\mathbf{b}) P(\mathbf{y}_b|\mathbf{b}) P(\mathbf{r}) P(\mathbf{y}_r|\mathbf{r}) P(\mathbf{y}_c|\mathcal{F}([\mathbf{h}, \mathbf{b}, \mathbf{u}, \mathbf{s}, \mathbf{r}])). \quad (12)$$

Assuming that all $\mathbf{b}$ and all $\mathbf{r}$ are equally likely *a priori*, the evaluation of (12) requires summing the product of the

likelihoods related to $\mathbf{b}$, $\mathbf{r}$, and their corresponding CRC, over the $2^{\ell(\mathbf{b})+\ell(\mathbf{r})}$ combinations of $\mathbf{b}$ and $\mathbf{r}$. In this section, two reduced-complexity methods are proposed for evaluating (10) based on two evaluations of Φ. The first one provides an exact evaluation of $\mathcal{M}$, whereas the second results in an approximate (but faster) evaluation of the metric.

A. Exact Computation

The CRC can be evaluated recursively over the data $\mathbf{d}$, as shown by (2). More precisely, the value of the CRC associated to the first $j+1$ bits of $\mathbf{d}$ (in short, at time $j+1$) only depends on the value of the CRC at time j and on the $j+1$-st bit of $\mathbf{d}$. Each value of the CRC at time j leads to two different values of the CRC at time $j+1$. Consequently, the evolution of the CRC values according to the bits of $\mathbf{d}$ can be described by a trellis. In this trellis, states correspond to the $2^{\ell(\mathbf{c})}$ possible values of the CRC. Transitions are determined by the bits of $\mathbf{d}$. At each time $j = 1 \dots \ell(\mathbf{d})$, we study the contribution of d_j (the j-th bit of $\mathbf{d}$) to the global CRC.

In our case, $\mathbf{d} = [\mathbf{h}, \mathbf{b}, \mathbf{u}, \mathbf{s}, \mathbf{r}]$. The header $\mathbf{h}$ is assumed to be known and we want to find the best combination of $[\mathbf{u}, \mathbf{s}] \in \Omega_{[u,s]}$ by taking into account the redundancy of the code (given by $\mathbf{c}$). The trellis is thus applied to the portions $\mathbf{b}$, $\mathbf{r}$, and $\mathbf{c}$ for given $\mathbf{h}$, $\mathbf{u}$, and $\mathbf{s}$. This trellis consists in grouping combinations of $\mathbf{b}$ and $\mathbf{r}$ giving the same value of the CRC.

Consequently, (12) may be rewritten as

$$\Phi = \sum_{\mathbf{c}} P(\mathbf{y}_c|\mathbf{c}) \sum_{\mathbf{b},\mathbf{r}|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{u},\mathbf{s},\mathbf{r}])=\mathbf{c}} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b})P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r}). \tag{13}$$

In the sequel, the state associated to a possible value $\mathbf{c}'$ of CRC is denoted by $S(\mathbf{c}')$, $\mathbf{c}'$ being the binary representation of $S(\mathbf{c}') \in \{0 \dots 2^{\ell(\mathbf{c})} - 1\}$. For instance with a 3-bit CRC, if $\mathbf{c}' = [1, 0, 1]$ then $S(\mathbf{c}') = 5$. After some derivations, one can show that (13) may be generalized as follows (see Appendix)

$$\begin{aligned} \Phi &= \sum_{\mathbf{c}'} \left[\sum_{\mathbf{b}|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{0},\mathbf{0},\mathbf{0}])=\mathbf{c}'} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b}) \right] \\ &\quad \left[\sum_{\mathbf{r}} P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r})P(\mathbf{y}_c|\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{r}])) \right] \\ &= \sum_{\mathbf{c}'} \alpha(S(\mathbf{c}')) \cdot \beta(S(\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{0}]))), \end{aligned} \tag{14}$$

with

$$\alpha(S(\mathbf{c}')) = \sum_{\mathbf{b}|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{0},\mathbf{0},\mathbf{0}])=\mathbf{c}'} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b}), \tag{15}$$

$$\beta(S(\mathbf{c}'')) = \sum_{\mathbf{r}} P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r})P(\mathbf{y}_c|\mathbf{c}'' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{r}])), \tag{16}$$

for all $\mathbf{c}', \mathbf{c}'' \in GF(2)^{\ell(\mathbf{c})}$. In (15), $\alpha(S(\mathbf{c}'))$ represents the sum of the probabilities associated to the combinations of $\mathbf{b}$ reaching state $S(\mathbf{c}')$ when starting from state $S(\mathcal{F}([\mathbf{h},\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0}]))$. In (16), $\beta(S(\mathbf{c}''))$ denotes the sum of the probabilities associated to all combinations of $[\mathbf{r}, \mathbf{c}'' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{r}])]$ when starting from state $S(\mathbf{c}'')$. In fact, the evaluation of Φ using (14) is efficiently performed using the BCJR algorithm for block codes [2], [33]. Thus, $\alpha(S(\mathbf{c}'))$ and $\beta(S(\mathbf{c}''))$ are easily evaluated recursively as follows

$$\begin{aligned} \alpha_{j+1}(S(\mathbf{c}')) &= P(b_{j+1}=0)P(y_{b_{j+1}}|b_{j+1}=0)\alpha_j(S(\mathbf{c}')) \\ &+ P(b_{j+1}=1)P(y_{b_{j+1}}|b_{j+1}=1) \\ &\quad \alpha_j(S(\mathbf{c}' \oplus \boldsymbol{\pi}(b_{j+1}))), \end{aligned} \tag{17}$$

with the boundary conditions (at $j = 0$)

$$\alpha_0(S(\mathbf{c}')) = \begin{cases} 1 & \text{for } \mathbf{c}' = \mathcal{F}([\mathbf{h},\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0}]) \\ 0 & \text{for all } \mathbf{c}' \neq \mathcal{F}([\mathbf{h},\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0}]) \end{cases}, \tag{18}$$

and

$$\begin{aligned} \beta_{j-1}(S(\mathbf{c}'')) &= P(r_j=0)P(y_{r_j}|r_j=0)\beta_j(S(\mathbf{c}'')) \\ &+ P(r_j=1)P(y_{r_j}|r_j=1)\beta_j(S(\mathbf{c}'' \oplus \boldsymbol{\pi}(r_j))), \end{aligned} \tag{19}$$

with the boundary conditions (at $j = \ell(\mathbf{r})$)

$$\beta_{\ell(\mathbf{r})}(S(\mathbf{c}'')) = P(\mathbf{y}_c|\mathbf{c}''), \text{ for all } \mathbf{c}'' \in GF(2)^{\ell(\mathbf{c})}. \tag{20}$$

The equations in (17) and (19) are the key for computing $\alpha(S(\mathbf{c}'))$ with a forward recursion over the bits of $\mathbf{b}$ and $\beta(S(\mathbf{c}''))$ with a backward recursion over the bits of $\mathbf{r}$. After $\ell(\mathbf{b})$ iterations, $\alpha_{\ell(\mathbf{b})}(S(\mathbf{c}')) = \alpha(S(\mathbf{c}'))$, and after $\ell(\mathbf{r})$ iterations, $\beta_0(S(\mathbf{c}'')) = \beta(S(\mathbf{c}''))$.

Finally, substituting (14) in (10), one obtains

$$\begin{aligned} \mathcal{M}([\mathbf{u},\mathbf{s}]) &= \sum_{\mathbf{c}'} \alpha(S(\mathbf{c}')) \cdot P(\mathbf{y}_u|\mathbf{u})P(\mathbf{y}_s|\mathbf{s}) \cdot \\ &\quad \beta(S(\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{0}]))) \\ &= P(\mathbf{y}_u|\mathbf{u})P(\mathbf{y}_s|\mathbf{s}) \\ &\quad \sum_{\mathbf{c}',\mathbf{c}''|\mathbf{c}''=\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{0}])} \alpha(S(\mathbf{c}')) \cdot \beta(S(\mathbf{c}'')). \end{aligned} \tag{21}$$

The evaluation of $\mathcal{M}([\mathbf{u},\mathbf{s}])$ consists in summing the probabilities associated to the $2^{\ell(\mathbf{c})}$ paths linking state $S(\mathbf{c}')$ to state $S(\mathbf{c}'')$, such as $\mathbf{c}'' = \mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{0}])$.

The steps for evaluating the global metric (10) with the above mentioned method are summarized below:

Step 1: Initialize $\alpha_0(S(\mathbf{c}'))$ and $\beta_{\ell(\mathbf{r})}(S(\mathbf{c}''))$ according to (18) and (20).
Step 2: Compute $\alpha_j(S(\mathbf{c}'))$, for all $\mathbf{c}' \in GF(2)^{\ell(\mathbf{c})}$ and for all $j = 1 \dots \ell(\mathbf{b})$, by using (17) in a forward way (partial BCJR forward step).
Step 3: Compute $\beta_j(S(\mathbf{c}''))$, for all $\mathbf{c}'' \in GF(2)^{\ell(\mathbf{c})}$ and for all $j = \ell(\mathbf{r})-1 \dots 0$, by using (19) in a backward way (partial BCJR backward step).
Step 4: For each $[\mathbf{u},\mathbf{s}] \in \Omega_{[u,s]}$, compute the metric $\mathcal{M}([\mathbf{u},\mathbf{s}])$ by using (21), recalling that $\alpha(S(\mathbf{c}')) = \alpha_{\ell(\mathbf{b})}(S(\mathbf{c}'))$ and $\beta(S(\mathbf{c}'')) = \beta_0(S(\mathbf{c}''))$.

Hence, one step of the sequential decoding is performed with a complexity $\mathcal{O}((\ell(\mathbf{b}) + \ell(\mathbf{r}) + |\Omega_{[u,s]}|)2^{\ell(\mathbf{c})})$, compared to $\mathcal{O}(|\Omega_{[u,s]}|2^{\ell(\mathbf{b})+\ell(\mathbf{r})})$ for a decoding with a straightforward metric computation.

Remark 1: As presented above, the decoding of $\mathbf{x}$ requires repeating steps 1 to 4 for each portion $[\mathbf{u},\mathbf{s}]$ in $\mathbf{x}$ since the portions $\mathbf{b}$ and $\mathbf{r}$ change according to the position of $[\mathbf{u},\mathbf{s}]$. To optimize the global decoding, as soon as $\mathbf{y}_t$ is received, $\alpha_j(S(\mathbf{c}'))$ and $\beta_j(S(\mathbf{c}''))$ may be computed, for all $\mathbf{c}', \mathbf{c}'' \in GF(2)^{\ell(\mathbf{c})}$ and for all $j = 0 \dots \ell(\mathbf{x})$, and may be stored in matrices $\mathbf{A}$ and $\mathbf{B}$. This is equivalent to

perform a *complete* BCJR algorithm over $\mathbf{x}$: the forward step is performed on $\mathbf{b} = \mathbf{x}$ and the backward step on $\mathbf{r} = \mathbf{x}$. The global decoding of $\mathbf{x}$ begins after this step. As explained previously, each portion $[\mathbf{u}, \mathbf{s}]$ is sequentially decoded by using (21) in which the values of $\alpha(S(\mathbf{c}'))$ and $\beta(S(\mathbf{c}''))$ are extracted from $\mathbf{A}$ and $\mathbf{B}$ depending on the position of the current portion $[\mathbf{u}, \mathbf{s}]$.

Note that in this case, steps 1 to 3 are performed once as a preamble, and step 4 is performed repeatedly for each $[\mathbf{u}, \mathbf{s}]$ in $\mathbf{x}$.

B. Approximate Computation

In practice, most CRCs are larger than 16 bits and the complexity $\mathcal{O}(2^{\ell(\mathbf{c})})$ is too large to allow a real-time implementation of the method presented in Section IV-A. An approximate computation consists in splitting the CRC into m_b partitions of $\ell(\mathbf{c})/m_b$ bits, each partition being assumed statistically independent from the others. A trellis is thus associated to each of the m_b partitions. Thus, $\mathbf{y}_c$ may be written as $\mathbf{y}_c = [\mathbf{y}_{c_1} \dots \mathbf{y}_{c_{m_b}}]$. Using the independence approximation, as explained with more details in [20], the global metric in (21) becomes

$$\mathcal{M}([\mathbf{u}, \mathbf{s}]) = P(\mathbf{y}_u|\mathbf{u})P(\mathbf{y}_s|\mathbf{s}) \prod_{m=1}^{m_b} \sum_{\mathbf{c}'_m, \mathbf{c}''_m | \mathbf{c}''_m = \mathbf{c}'_m \oplus \mathcal{F}_m([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{0}])} \alpha^m(S(\mathbf{c}'_m)) \cdot \beta^m(S(\mathbf{c}''_m)), \tag{22}$$

where $\alpha^m(S(\mathbf{c}'_m))$ and $\beta^m(S(\mathbf{c}''_m))$ represent the probabilities associated to states $S(\mathbf{c}'_m)$ and $S(\mathbf{c}''_m)$ respectively, for $\mathbf{c}'_m, \mathbf{c}''_m \in GF(2)^{\ell(\mathbf{c})/m_b}$, in the m-th trellis.

The total complexity for evaluating (22) is now $\mathcal{O}((\ell(\mathbf{b}) + \ell(\mathbf{r}) + |\Omega_{[u,s]}|)m_b 2^{\ell(\mathbf{c})/m_b})$, at the cost of a slightly suboptimal performance.

Remark 2: To reduce the complexity of the global decoding of $\mathbf{x}$, we can apply the principle introduced in Remark 1 to the new method. In this case, the algorithm generates first the m_b submatrices $\mathbf{A}^m$ and $\mathbf{B}^m$ associated to partition $\mathbf{c}_m$. During the decoding, the values of $\alpha^m(S(\mathbf{c}'_m))$ and $\beta^m(S(\mathbf{c}''_m))$ in (22) are extracted from $\mathbf{A}^m$ and $\mathbf{B}^m$ according to the position of the current portion $[\mathbf{u}, \mathbf{s}]$.

C. Decoding complexity

From the two previous sections, one may evaluate the computational complexity for evaluating (10) as $\mathcal{O}((\ell(\mathbf{b}) + \ell(\mathbf{r}) + |\Omega_{[u,s]}|)2^{\ell(\mathbf{c})})$ with the exact computation and as $\mathcal{O}((\ell(\mathbf{b}) + \ell(\mathbf{r}) + |\Omega_{[u,s]}|)m_b 2^{\ell(\mathbf{c})/m_b})$ with the suboptimal algorithm.

A careful comparison for the decoding of an erroneous payload $\mathbf{x}$ depends on the choice of several parameters, which which also have an impact on the performance. It is shown in the simulation section below that significant improvements can be obtained with respect to the classical reception algorithm even when the complexity is reduced by a factor larger than 4.10^6, compared to an optimal decoding algorithm.

A payload $\mathbf{x}$ is divided in E groups, each group e is processed iteratively in N_e steps. At each step n of the decoding of the e-th group, $\ell(\mathbf{s}_{e,n}) \approx \ell(\mathbf{x})/E/N_e$ bits are thus decoded. As $\Omega^{e,n}_{[u,s]}$ contains only the valid sequences obtained by concatenating the sequences in $\Omega^{e,n}_{[u]}$ with the $2^{\ell(\mathbf{s}_{e,n})}$ possible sequences $\mathbf{s}_{e,n}$, one has $|\Omega^{e,n}_{[u,s]}| \approx \left|\Omega^{e,n}_{[u]}\right| .2^{\frac{\ell(\mathbf{x})}{E.N_e}} = M.2^{\frac{\ell(\mathbf{x})}{E.N_e}}$, since in $\Omega^{e,n}_{[u]}$, only the M best candidates are kept, see Fig. 6. Finally, since $\ell(\mathbf{b}) + \ell(\mathbf{r}) \approx \ell(\mathbf{x})$, the decoding complexity is

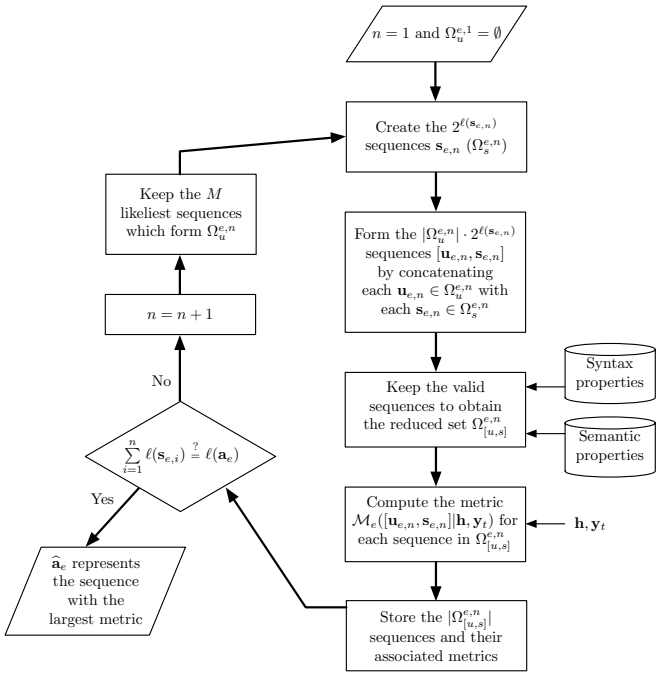

Fig. 6. Proposed sequential decoding scheme.

$$C_{\mathrm{e}} = \mathcal{O}\left(E.N_e.\left(\ell(\mathbf{x}) + M.2^{\frac{\ell(\mathbf{x})}{E.N_e}}\right)2^{\ell(\mathbf{c})}\right) \tag{23}$$

when the exact computation is performed for evaluating (10) and

$$C_{\mathrm{e}} = \mathcal{O}\left(E.N_e.\left(\ell(\mathbf{x}) + M.2^{\frac{\ell(\mathbf{x})}{E.N_e}}\right)m_b 2^{\ell(\mathbf{c})/m_b}\right) \tag{24}$$

when the suboptimal algorithm is used. The tuning parameters are thus E, N_e, M, and m_b in the case of the suboptimal algorithm.

Considering a large (but not too large) value of E, *i.e.*, considering many groups reduces the computational complexity. The price to be paid is an increased overhead, since more position markers are required to localize these groups. The number of decoding steps N_e of a group has also to be optimized to minimize the decoding complexity. When M is increased, the decoding complexity increases, but since more candidates are kept in $\Omega^{e,n}_{[u]}$, the decoder may perform better. The role of m_b has already be discussed in Section IV-B.

V. Simulation Results

In the *extended* profile of H.264/AVC [13], an error-resilience mode is provided. In this mode, the compressed picture data are classified according to their influence on the video quality. Three partitions are defined:

- Partition A contains the headers and the motion vectors of each encoded picture.
- Partition B consists of the texture coefficients of INTRA coded blocks.

- Partition C contains the texture coefficients of INTER coded blocks.

This stream decomposition allows an adaptation of the protection to the sensitivity of the partition to be sent. After compression, each partition is encapsulated in a *Network Abstraction Layer Unit* (NALU) which is delivered to the RTP layer. In our simulation, packets associated to the A partition are assumed heavily protected and correctly interpreted at the receiver. On the other hand, B and C packets are transmitted over a noisy channel and are corrupted by transmission errors. As previously mentioned, these packets contain the texture coefficients of the various 4×4 blocks of a picture. These blocks are encoded in CAVLC [25].

In this paper, we focus on the decoding of the CAVLC sequences included in the B and C packets. Each CAVLC sequence is considered as an independent group of codewords which can be separated from the others by using the position markers, transmitted as side information (see Section III-A). Consequently, the group-based sequential decoding method of Section III may be used for their estimation. Note that in H.264/AVC, the CAVLC sequences are not totally independent (adaptive context) but the existing dependencies are indeed small and are neglected here. The performance of the presented method has been evaluated by simulations and compared to that of two other decoding methods: a standard decoding method and a classical robust decoding method (exploiting only the source properties).

The simulated system consists of a transmitter, a channel, and a receiver. The transmitter uses repeatedly the 5 first pictures of *Foreman.cif* with the IPPPP frame structure and generates the encoded partitions using the CAVLC H.264/AVC video coder. Video packets (partitions) are then processed by the protocol stack defined in Fig. 1. At the MAC layer, IP packets are fragmented in several MAC packets of variable payload size. A CRC of 4 bytes, consistent with the 802.11 standard, is added at the end of each MAC fragment. At the PHY layer of the transmitter, the data are encoded by the convolutional channel coder of the $802.11a$ standard [12]. Next, the coded PHY packets are mapped onto BPSK symbols before being sent over the physical medium. To improve the decoding performance, the aforementioned position markers are sent as side information, indicating the location of each 4×4 encoded texture block in B and C packets. This side information is transmitted in a specific NALU and the markers are compressed using the Exp-Golomb coding of H.264/AVC. The overhead due to the transmission of this redundancy represents about 30 % of the total bitrate. The channel does not degrade the data contained in A packets nor the side information. On the other hand, it does add a white Gaussian noise to the other packets. At the receiver, the data are processed by a SISO channel decoder (BCJR algorithm) and are then delivered to the APL layer (following the permeable mechanism explained in Section II). At the APL layer, three different decoders are considered:

1) A *standard* decoder performs hard decisions on the received soft data and makes usage of position markers to decode each block.
2) A *robust* decoder uses the source properties, the soft data as well as the position markers, but does not use the redundancy provided by the CRC. This decoder exploits the algorithm depicted in Section III, but the metric in (10) does not include the term Φ.
3) A *CRC-robust* decoder combines all the previous sources of redundancy along with the CRC properties through the decoding method presented in Sections III and IV.

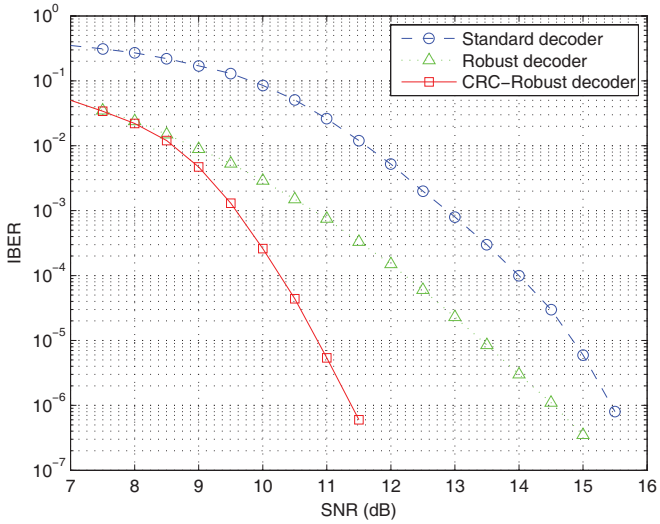

Fig. 7. Image block error rate (IBER) vs SNR for the three types of decoders, with MAC payload size of 120 bytes and deactivated channel coder/decoder at PHY layer.

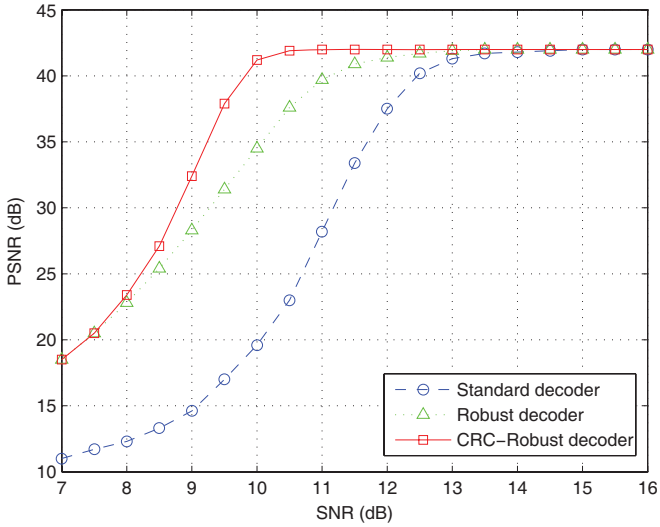

Fig. 8. Decoded image quality (PSNR) vs SNR for the three types of decoders, with MAC payload size of 120 bytes and deactivated channel coder/decoder at PHY layer.

Note that, in our simulations, the two robust decoders use the same stack size $M = 20$ and the same default decoding depth $\ell(\mathbf{s}) = 4$ bits. The size of a group is 11 bits in average. The CRC-robust decoder uses the suboptimal method presented in Section IV-B. For this purpose, the CRC is split into 4 blocks of 8 bits, this allows to reduce the decoding complexity by a factor of more than 4.10^6. The decoding complexity of the exact algorithm is not manageable in this context and is thus not considered.

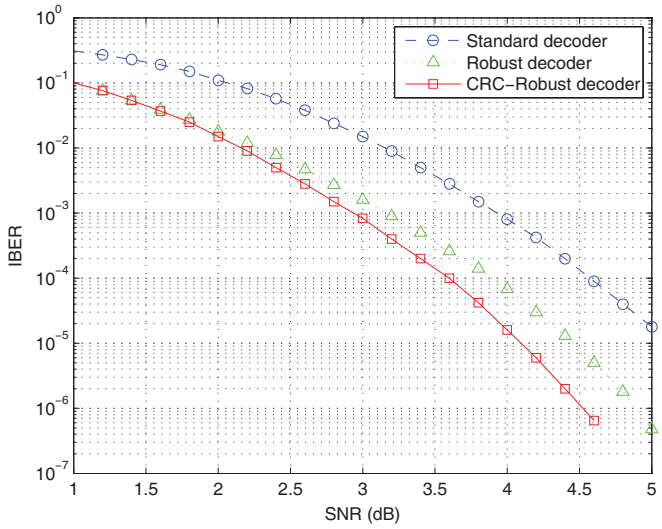

Fig. 9. Image block error rate (IBER) vs SNR for the standard, robust, and CRC-robust decoders. In this case, the 802.11a channel coder/decoder is considered at PHY layer and the MAC layer protocol of the transmitter generates 120-byte MAC payload.

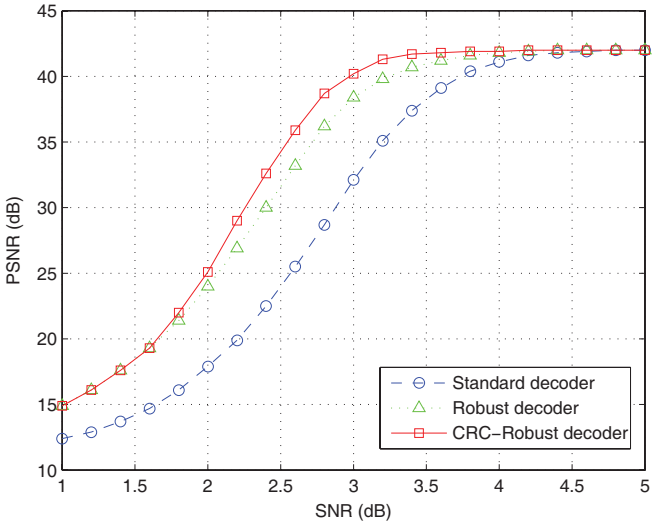

Fig. 10. Decoded image quality (PSNR) vs SNR for the standard, robust, and CRC-robust decoders. In this case, the 802.11a channel coder/decoder is considered at PHY layer and the MAC layer protocol of the transmitter generates 120-byte MAC payload.

Figures 7 to 10 show the evolution of the *Image Block Error Rate* (IBER) and of the *Peak Signal to Noise Ratio* (PSNR) of the decoded video as a function of the SNR for the three different decoders, with and without channel coding. In Figs. 7 and 8, the channel coder/decoder at PHY layer was deactivated. In all figures, the standard, robust, and CRC-robust decoders are compared for a MAC payload size of 120 bytes.

We can notice that in terms of IBER, the standard decoder is outperformed by the two robust decoders independently of the presence of the outer channel code. Moreover, the two robust decoders are equivalent for low SNRs and the CRC-robust decoder outperforms the classical robust decoder above a given SNR threshold. In this region, the coding gain increases with the SNR. This behavior is specific to channel decoding performance: the CRC plays the role of an error-correcting code above this threshold. In our simulations, the threshold is about 8.5 dB in Fig. 7, and 1.8 dB in Fig. 9.

In terms of PSNR, the behavior is almost similar. However, when the channel conditions are good enough, the difference in IBER does not translates in PSNR improvements, since the number of erroneous blocks is very small for all decoders. As a result, the CRC-robust decoder improves over the robust decoder only in a specific SNR range (from 8 dB to 12 dB without channel code and from 1.8 dB to 3.8 dB with a channel code).

Globally, the comparison between Figs. 7 and 8 on one side and Figs. 9 and 10 on the other side, shows that the presence of the convolutional code at PHY layer reduces largely the improvements brought by robust decoders, but that significant improvements are still observed. The robust decoders provide improvements as soon as the convolutionnal code leaves some (and not too many) uncorrected errors in the bitstream.

Figure 11 illustrates the 5-th image of the *Foreman.cif* video sequence, along with its reproductions obtained after this image is transmitted and decoded by the standard, robust and CRC-robust decoders respectively. In this figure, the channel coder/decoder is considered. This result was obtained with a payload size of 120 bytes and at an SNR of 2.8 dB for which the PSNR of the standard, robust and CRC-robust decoders are 29, 35 and 38 dB respectively (see Fig. 10). Obviously, the image obtained with the standard decoder contains many artifacts and is of a very poor quality. On the other hand, the robust decoder strongly improves the quality even though some distortions are still visible. Finally, no visual difference may be noticed between the original image and the image obtained by the CRC-robust decoder.

VI. Conclusion

In this paper, we have presented a MAP estimator for robust video decoding. The decoder jointly exploits the inherent source coder information along with the MAC layer CRC redundancy. The implementation of this MAP estimator was shown to be a combination of a sequential decoding algorithm along with the BCJR algorithm for obtaining appropriate metrics. We applied this method to H.264/AVC decoding of CAVLC sequences. Simulation results show that the information carried by the CRC does improve the decoding efficiency. More precisely, joint use of CRC and source properties becomes interesting above a certain threshold. It should be noted that, the bitrate used for transmitting the side information is rather high (about 30 %) in the presented experiments. We are currently working at reducing this overhead. One possibility is to consider position markers indicating, *e.g.*, the location of each macroblock of 16×16 pixels.

The proposed method could readily be applied to H.264 with a Context Adaptive Binary Arithmetic Coder (CABAC) as entropy code. Nevertheless, the residual redundancy left by the CABAC in the compressed bitstream is less than that left by the CAVLC. The performance improvement provided by the robust decoders would probably be less significant.

Fig. 11. 5-th image of *Foreman.cif* obtained after (a) error-free decoding, (b) standard decoding, (c) robust decoding, and (d) CRC-robust decoding for a SNR of 2.8 dB and a MAC payload size of 120 bytes, with channel coding.

APPENDIX

Below, we detail the derivation of (14). Assuming that the bits of $\mathbf{b}$ and $\mathbf{r}$ are i.i.d. and do not depend on $\mathbf{h}$, $\mathbf{u}$, and $\mathbf{s}$, Φ in (9) may be developed as follows

$$\begin{aligned}\Phi &= \sum_{\mathbf{b},\mathbf{r}} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b})P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r}) \\ &\quad P(\mathbf{y}_c|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{u},\mathbf{s},\mathbf{r}])) \\ &= \sum_{\mathbf{r}} P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r}) \sum_{\mathbf{b}} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b}) \\ &\quad P(\mathbf{y}_c|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{u},\mathbf{s},\mathbf{r}]))\end{aligned} \tag{25}$$

In (25), the sum over $\mathbf{b}$ is a sum over all the possible values that $\mathbf{b}$ can take, each value corresponding to a path in the trellis. On the other hand, any possible path $\mathbf{b}$ ends up at a state $S(\mathbf{c}') \in \{0 \ldots, 2^{\ell(\mathbf{c})} - 1\}$ (*i.e.*, one of the $2^{\ell(\mathbf{c})}$ possible states). As a result, summing over all the possible paths $\mathbf{b}$ is equivalent to summing over all the paths $\mathbf{b}$ that end up at state 0, and all the paths that end up at state 1, ..., and all the paths that end up at state $2^{\ell(\mathbf{c})} - 1$. Hence, (25) becomes

$$\begin{aligned}\Phi &= \sum_{\mathbf{r}} P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r}) \sum_{\mathbf{c}'} \sum_{\mathbf{b}|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{0},\mathbf{0},\mathbf{0}])=\mathbf{c}'} P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r}) \\ &\quad P(\mathbf{y}_c|\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{r}])) \\ &= \sum_{\mathbf{c}'} \sum_{\mathbf{b}|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{0},\mathbf{0},\mathbf{0}])=\mathbf{c}'} \sum_{\mathbf{r}} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b})P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r}) \\ &\quad P(\mathbf{y}_c|\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{r}])) \\ &= \sum_{\mathbf{c}'} \left[\sum_{\mathbf{b}|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{0},\mathbf{0},\mathbf{0}])=\mathbf{c}'} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b}) \right] \\ &\quad \left[\sum_{\mathbf{r}} P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r})P(\mathbf{y}_c|\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{r}])) \right] \\ &= \sum_{\mathbf{c}'} \alpha(S(\mathbf{c}')) \cdot \beta(S(\mathbf{c}' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{u},\mathbf{s},\mathbf{0}]))),\end{aligned}$$

with

$$\begin{aligned}\alpha(S(\mathbf{c}')) &= \sum_{\mathbf{b}|\mathcal{F}([\mathbf{h},\mathbf{b},\mathbf{0},\mathbf{0},\mathbf{0}])=\mathbf{c}'} P(\mathbf{b})P(\mathbf{y}_b|\mathbf{b}), \\ \beta(S(\mathbf{c}'')) &= \sum_{\mathbf{r}} P(\mathbf{r})P(\mathbf{y}_r|\mathbf{r})P(\mathbf{y}_c|\mathbf{c}'' \oplus \mathcal{F}([\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{r}])).\end{aligned}$$

REFERENCES

[1] J. B. Anderson and S. Mohan, *Source and Channel Coding: An Algorithmic Approach*. Kluwer, 1991.

[2] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. 20, no. 2, pp. 284–287, 1974.
[3] R. Bauer and J. Hagenauer, "On variable length codes for iterative source/channel decoding," in *Proc. DCC*, pp. 272–282, Snowbird, UT, 1998.
[4] C. Bergeron and C. Lamy-Bergot, "Soft-input decoding of variable-length codes applied to the H.264 standard," in *Proc. MSP*, pp. 87–90, 29 Sep. 2004.
[5] R. E. Blahut, *Theory and Practice of Error Control Codes*. Addison-Wesley, Reading, MA, 1984.
[6] K. Bouchireb, C. Marin, P. Duhamel, and M. Kieffer, "Improved retransmission scheme for video communication systems," in *Proc. IEEE PIMRC*, pp. 1–5, Cannes, France, 2008.
[7] V. Buttigieg and P. Farrell, "A MAP decoding algorithm for variable-length error-correcting codes," in *Codes and Cyphers: Cryptography and Coding IV*, pp. 103–119. Essex, England: The Institute of Mathematics and its Applications, 1995.
[8] J. Hagenauer, "Source-controlled channel decoding," *IEEE Trans. Commun.*, vol. 43, no. 9, pp. 2449–2457, 1995.
[9] M.-C. Hong, H. Schwab, L. P. Kondi, and A. K. Katsaggelos, "Error concealment algorithms for compressed video," *Signal Processing: Image Communication*, vol. 14, no. 6-8, pp. 473–492, 1999.
[10] J. Huusko, J. Vehkapera, P. Amon, C. Lamy-Bergot, G. Panza, J. Peltola, and M. G. Martini, "Cross-layer architecture for scalable video transmission in wireless network," *IEEE Signal Process.: Image Communication*, vol. 22, no. 3, pp. 317–330, 2007.
[11] IEEE, 802.11 Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications, technical report, 1999.
[12] IEEE, 802.11a part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications, high-sped physical layer in the 5 Ghz band, technical report, 1999.
[13] ITU-T and ISO/IEC JTC 1, Advanced video coding for generic audiovisual services, technical report, ITU-T Rec. H.264, and ISO/IEC 14496-10 AVC, Nov. 2003.
[14] S. Kaiser and M. Bystrom, "Soft decoding of variable-length codes," in *Proc. ICC*, vol. 3, pp. 1203–1207, New Orleans, 2000.
[15] W.-Y. Kung, C.-S. Kim, and C.-C. J. Kuo, "Spatial and temporal error concealment techniques for video transmission over noisy channels," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 16, no. 7, pp. 789–803, 2006.
[16] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach Featuring the Internet*. Addison Wesley, Boston, 3rd edition, 2005.
[17] C. Lamy and S. Merigeault, Procédé de correction d'une trame erronée par un récepteur, French patent no. 0206501, 2002.
[18] C. Lee, M. Kieffer, and P. Duhamel, "Soft decoding of VLC encoded data for robust transmission of packetized video," in *Proc. ICASSP*, pp. 737–740, 2005.
[19] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.
[20] C. Marin, Y. Leprovost, M. Kieffer, and P. Duhamel, "Robust MAC-lite and soft header recovery for packetized multimedia transmission," *IEEE Trans. Commun.*, 2008, submitted.
[21] H. Nguyen and P. Duhamel, "Iterative joint source-channel decoding of variable length encoded video sequences exploiting source semantics," in *Proc. ICIP*, pp. 3221–3224, 2004.
[22] H. Nguyen, P. Duhamel, J. Brouet, and D. Rouffet, "Robust VLC sequence decoding exploiting additional video stream properties with reduced complexity," in *Proc. ICME*, pp. 375–378, June 2004, Taipei, Taiwan.
[23] J. W. Nieto and W. N. Furman, Cyclic redundancy check (CRC) based error method and device, US Patent US 2007/0192667 A1, Aug. 16, 2007.
[24] L. Perros-Meilhac and C. Lamy, "Huffman tree based metric derivation for a low-complexity soft VLC decoding," in *Proc. ICC*, pp. 783–787, 2002.
[25] I. Richardson, *H.264 and MPEG-4 Video Compression: Video Coding for Next-Generation Multimedia*. John Wiley and Sons, 2003.
[26] T. Richardson and U. Urbanke, *Modern Coding Theory*. Cambridge University Press, 2008.
[27] G. Sabeva, S. Ben Jamaa, M. Kieffer, and P. Duhamel, "Robust decoding of H.264 encoded video transmitted over wireless channels," in *Proc. MMSP*, pp. 9–13, Victoria, Canada, 2006.
[28] K. Sayood, *Introduction to Data Compression, Second Edition*. San Francisco, CA: Morgan Kaufmann, 2000.
[29] Q. Spencer, Method of correcting message errors using cyclic redundancy checks, US patent no. 0040644, 2008.
[30] R. Thobaben and J. Kliewer, "On iterative source-channel decoding for variable-length encoded Markov sources using a bit-level trellis," in *Proc. SPAWC*, pp. 50–54, Rome, 2003.
[31] R. Thobanen and J. Kliewer, "Robust decoding of variable-length encoded markov sources using a three-dimensional trellis," *IEEE Commun. Lett.*, vol. 7, no. 7, pp. 320–322, 2003.
[32] T. Tillo, M. Grangetto, and G. Olmo, "A flexible error resilient scheme for JPEG 2000," in *Proc. MSP*, pp. 295–298, Sep. 2004.
[33] J. K. Wolf, "Efficient maximum-likelihood decoding of linear block codes using a trellis," *IEEE Trans. Inf. Theory*, vol. 24, pp. 76–80, 1978.

Cédric Marin was born in Chaumont, France, in 1980. He received the Ing. degree in electrical engineering from the University Paris-Sud, Orsay, in 2003, and a master degree in signal processing in 2004 from the same institution. In 2009, he receives a Ph.D. degree in electrical and computer engineering at the University of Paris-Sud. This Ph.D. was prepared jointly with the Alcatel-Lucent Bell Labs center.

His interests are in the areas of video compression, joint source-channel decoding, iterative decoding, wireless transmission protocol, and cross-layer-design. His thesis focuses on the problematics of video transmission over wireless channels.

Khaled Bouchireb was born in Algeria in 1982. He received the Ing. degree in electronic engineering from the Ecole Nationale Polytechnique, Algiers, Algeria in 2004 and the Master in Digital Telecommunications Systems from the UniversitÈ Pierre et Marie Curie and Ecole Nationale SupÈrieure des TÈlÈcommunications, Paris, France, in 2005.

He is currently with Devoteam Solutions, Levallois, France, working on the operability and optimization of Alcatel-Lucent 3G LTE networks. From early 2006 to late 2008, he was a Ph.D. student at Laboratoire des Signaux et SystËmes, Gif-sur-Yvette, France, working on the optimized delivery of video data over cellular networks. His research interests are in signal processing, digital communications, and cellular networks, with emphasis on multimedia service provisioning and retransmission mechanisms.

Michel Kieffer (M'02 – SM'07) was born in Sarreguemines, France in 1972. In 1995, he received the Agrégation in Applied Physics at the Ecole Normale Supérieure de Cachan, France. He received a PhD degree in Control and Signal Processing in 1999, and the Habilitation à Diriger des Recherches degree in 2005, both from the Univ Paris-Sud, Orsay, France. Michel Kieffer is an assistant professor in signal processing for communications at the Univ Paris-Sud and a researcher at the Laboratoire des Signaux et Systèmes, CNRS - SUPELEC - Univ Paris-Sud, Gif-sur-Yvette, France. From sept. 2009 to sept. 2010, he has been invited professor at the Laboratoire Traitement et Communication de líInformation, CNRS - Telecom ParisTech, Paris.

His research interests are in joint source-channel coding and decoding techniques for the reliable transmission of multimedia contents. He is also interested in guaranteed parameter and state estimation for systems described by non-linear models.

Michel Kieffer is co-author of more than 90 contributions in journals, conference proceedings, or books. He is one of the co-author of the book *Applied Interval Analysis*, published by Springer-Verlag in 2001, and of the book *Joint Source-Channel Decoding: A Crosslayer Perspective with Applications in Video Broadcasting*, published by Academic Press in 2009. He is associate editor of SIGNAL PROCESSING since 2008.

Pierre Duhamel (F'98) was born in France in 1953. He received the Eng. Degree in Electrical Engineering from the National Institute for Applied Sciences (INSA) Rennes, France in 1975, the Dr. Eng. Degree in 1978, and the Doctorat ès sciences degree in 1986, both from Orsay University, Orsay, France.

From 1975 to 1980, he was with Thomson-CSF, Paris, France, where his research interests were in circuit theory and signal processing, including digital filtering and analog fault diagnosis. In 1980, he joined the National Research Center in Telecommunications (CNET), Issy les Moulineaux, France, where his research activities were first concerned with the design of recursive CCD filters. Later, he worked on fast algorithms for computing Fourier transforms and convolutions, and applied similar techniques to adaptive filtering, spectral analysis and wavelet transforms. From 1993 to Sept. 2000, he has been professor at ENST, Paris (National School of Engineering in Telecommunications) with research activities focused on Signal processing for Communications. He was head of the Signal and Image processing Department from 1997 to 2000. He is now with CNRS/LSS (Laboratoire de Signaux et Systemes, Gif sur Yvette, France), where he is developing studies in Signal processing for communications (including equalization, iterative decoding, multicarrier systems, cooperation) and signal/image processing for multimedia applications, including source coding, joint source/channel coding, watermarking, and audio processing. He is currently investigating the application of recent information theory results to communication theory.

Dr. Duhamel was chairman of the DSP committee from 1996 to 1998, and a member of the SP for Com committee until 2001. He was an associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1989 to 1991, an associate Editor for the IEEE SIGNAL PROCESSING LETTERS, and a guest editor for the special issue of the IEEE TRANSACTIONS ON SIGNAL PROCESSING on wavelets.

He was a Distiguished Lecturer, IEEE, for 1999, and was co-general chair of the 2001 International Workshop on Multimedia Signal Processing, Cannes, France. He was also co-technical chair of ICASSP 06, Toulouse, France. The paper on subspace-based methods for blind equalization, which he co-authored, received the "Best Paper Award" from the IEEE TRANSACTIONS ON SIGNAL PROCESSING in 1998. He was awarded the "Grand Prix France Telecom" by the French Science Academy in 2000. He is the co-author of the book *Joint Source-Channel Decoding: A Crosslayer Perspective with Applications in Video Broadcasting*, published by Academic Press in 2009.