Robust broadcast of real-time video over wireless network

Claudio Greco

ParisTech
INSTITUT DES SCIENCES ET TECHNOLOGIES
PARIS INSTITUTE OF TECHNOLOGY

TELECOM ParisTech

2012-ENST-032

EDITE – ED 130

Doctorat ParisTech

T H È S E

pour obtenir le grade de docteur délivré par

Télécom ParisTech

Spécialité « Signal et Images »

présentée et soutenue publiquement par

Claudio GRECO

le 6 juillet 2012

Diffusion robuste de la vidéo en temps réel sur réseau sans fil

—

Robust broadcast of real-time video over wireless network

Directeur de thèse : **Béatrice PESQUET-POPESCU**
Co-encadrement de la thèse : **Marco CAGNAZZO**

Jury

M^me^ **Christine GUILLEMOT**, Directeur de Recherce INRIA, IRISA, Rennes	Présidente
M. Khaldoun AL AGHA, Professeur, LRI, Université Paris Sud, Orsay	Rapporteur
M. Pascal FROSSARD, Professeur, École Polytechnique Fédérale de Lausanne, Lausanne	Rapporteur
M. Adrian MUNTEANU, Professeur, Vrije Universiteit Brussel, Bruxelles	Esaminateur
M^me^ **Roberta FRACCHIA**, Ingenieur de Recherche, Thales Communications	Invitée

THÈSE

Télécom ParisTech
Grande école de l'Institut Télécom - membre fondateur de ParisTech
46 rue Barrault — 75634 Paris Cedex 13 — Tél. +33 (0)1 45 81 77 77 — www.telecom-paristech.fr

To Schatzi

Remerciements

Ce document présente mes travaux de thèse, réalisés au sein du groupe “Multimédia” du Département Traitement du Signal et des Images de Télécom ParisTech.

Je tiens tout d’abord à remercier Béatrice Pesquet-Popescu et Marco Cagnazzo pour avoir assuré la direction de ma thèse, pour leur disponibilité, leurs conseils et la confiance qu’ils ont voulu m’accorder depuis le début.

Je remercie également Christine Guillemot pour avoir accepté de présider mon jury de thèse, Khaldoun Al Agha et Pascal Frossard pour leur contributions en tant que rapporteurs, ainsi que Roberta Fracchia et Adrian Munteanu, examinateurs.

Je remercie enfin toutes les personnes avec qui j’ai eu le privilège de collaborer et qui ont, elles aussi, contribué à la bonne réusite de cette thèse. À toute ces personne je souhaite aussi une bonne continuation, en m’adressant plus particulièrement aux membres du département TSI de Télécom ParisTech.

Résumé

Introduction

Au cours de cette dernière dizaine d'années, l'intérêt pour la diffusion en temps réel de séquences vidéo sur réseaux sans fil ad-hoc a grandi sans cesse, en raison de l'attrayante propriété d'être capable de déployer un système de communication visuelle à tout moment et en tout lieu, sans la nécessité d'une infrastructure préexistante.

Une large gamme d'applications –des opérations militaires et de sauvetage, jusqu'aux applications commerciales, éducatives, récréatives– a été envisagée, ce qui a créé un grand intérêt pour toutes les technologies concernées.

L'objectif de cette thèse est de fournir un service efficace et robuste de diffusion vidéo en temps réel sur réseaux mobiles ad-hoc, avec un aperçu des solutions disponibles à ce moment, et de proposer de nouvelles solutions qui permettraient de surmonter les limites de celles actuelles.

Nos contributions touchent à plusieurs aspects du paradigme *mobile vidéo streaming*, ce qui est un sujet très vif dans la communauté des chercheurs, en particulier dans le cas du transport dans un réseau de pairs. Pourtant, la plupart des protocoles proposés pour la construction de réseaux *overlay* sont hérités des paradigmes de partage de fichiers et ne sont pas à même de garantir une qualité de service satisfaisante pour la transmission de la vidéo en temps-réel.

Notre recherche vise à proposer des techniques capables de créer, de manière distribuée et en temps-réel, des réseaux overlay optimaux en termes de qualité du flux vidéo reçu par les utilisateurs en tenant compte des réseaux sous-jacents respectifs, soit dans le cas d'un réseau filaire, soit dans le cas d'un réseau sans-fil.

Toutes nos approches visent à améliorer la qualité de la communication visuelle, en termes de distorsion et de robustesse, tout en réduisant le retard perçu par les utilisateurs.

1 Codage Vidéo

Dans le premier chapitre de cette thèse, nous introduisons le concepts de base du *codage vidéo*, un sujet de très grande importance dans le domaine des télécommunications, qui entre en jeu à chaque fois que l'on affronte des problématiques de stockage et transmission

de signaux vidéo.

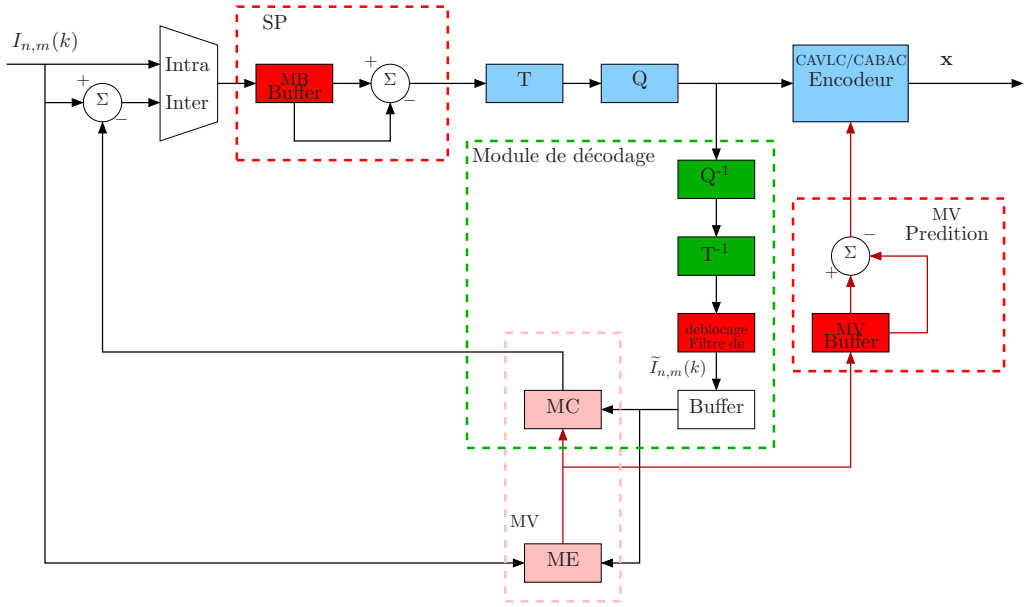


Figure 1: Structure de l'encodeur MPEG-4 part 10/H.264 AVC.

Tout d'abord, nous donnons une définition formelle du problème du codage vidéo et de ses objectives ; ensuite, nous présentons les principales outils utilisées dans le technique de codage ; enfin, nous passons en revue les normes en matière de codage vidéo les plus importantes, et en particulier la norme MPEG-4 part 10/H.264 AVC (dont le schéma de l'encodeur est montré en Figure 1), ce qui est –au moment de la rédaction de cette thèse– la norme de codage vidéo la plus avancée.

Les concepts introduits dans ce chapitre seront fondamentaux pour comprendre les enjeux de la transmission vidéo en temps réel, les difficultés que l'on s'attende de rencontrer et la faisabilité des solutions proposées.

2 Codage par Descriptions Multiple

Dans le deuxième chapitre, nous traitons le sujet du *Codage par Descriptions Multiples* (ou MDC, de l'anglais Multiple Description Coding), un paradigme de codage conjoint source-canal qui fournisse un compromis entre efficacité de l'encodage (en termes de débit pour une qualité donnée), robustesse et complexité.

Nous présentons les principes de base du MDC et en particulier ses applications aux techniques de codage vidéo présentées auparavant pour les rendre plus adaptées à une diffusion vidéo sur un réseau non-fiable, tel que une réseau sans-fil ad-hoc.

Ces technique sont classifiés selon l'approche classique en technique

Nous présentons aussi une nouvelle technique de codage vidéo par descriptions multiples proposée par nous mêmes, qui fournit une qualité vidéo acceptable, même en présence

d'un taux élevé de pertes. Cette technique sera un des composants de base de système de diffusion robuste de la vidéo en temps réel sur réseau sans fil présenté dans les chapitres suivantes.

Notre technique est basée sur le concept de *interpolation temporelle compensée en mouvement* (ou MTCI, de l'anglais Motion Compensated Temporal Interpolation), un concept qui a vu son origine dans le domaine du codage vidéo distribué.

En particulier, nous proposons d'utiliser des technique d'interpolations qui on été proposées dans le contexte du projet DISCOVER (DIStributed COding for Video sERvices), un projet européen financé par le programme FP6 IST de la Commission Européenne.

La première technique est illustrée dans la Figure 2. Soit $I(k)$ la trame à estimer, c'est-à-dire, appartenant à la description manquante. Son estimation $\widetilde{I}(k)$ est produite en utilisant les trames adjacentes $I(k-1)$ et $I(k+1)$. Étant donné que nous appliquons l'algorithme d'interpolation dans le contexte MDC, on peut supposer que les trames adjacents à celle étant interpolée sont disponibles à partir de la description reçue.

Tout d'abord, les trames de référence sont filtrées spatialement pour lisser le bruit et les contributions des fréquences les plus élevées. Ensuite, un algorithme de *block matching* est exécuté pour trouver un champ de vecteur mouvement entre les images $I(k-1)$ et $I(k+1)$. A suivre, une outre estimation de mouvement bidirectionnel est effectuée pour trouver le mouvement entre $I(k)$ et ces références.

Considérons maintenant un bloc de pixels centré sur la position $\mathbf{p}_2$. Soit $\mathbf{v}$ le champ de vecteurs mouvement de $I(k+1)$ vers $I(k-1)$, $\mathbf{u}$ ce de $I(k)$ vers $I(k-1)$, et $\mathbf{w}$ ce de $I(k)$ vers $I(k+1)$. Les vecteurs de mouvement calculés par l'estimation de mouvement en avant est $\mathbf{v}(\mathbf{p}_2)$ et il pointe à la position $\mathbf{p}_2 + \mathbf{v}(\mathbf{p}_2)$ dans la trame $I(k-1)$.

Le modèle sous-jacent suppose vitesse constante et donc mouvement linéaire, c'est-à-dire,$\mathbf{u}(\mathbf{p}_2 + \frac{1}{2}\mathbf{v}(\mathbf{p}_2)) = \frac{1}{2}\mathbf{v}(\mathbf{p}_2)$. Afin d'éviter les occlusions dans l'image compensée, il est nécessaire d'estimer $\mathbf{u}(\mathbf{p}_2)$. Pour cette position, le vecteur le plus proche du centre de bloc est considéré. Dans la figure, ça corresponde à $\mathbf{v}(\mathbf{p}_3)$, car $\|\mathbf{p}_2 - \mathbf{q}_3\| < \|\mathbf{p}_2 - \mathbf{q}_2\|$, où $\mathbf{q}_i = \mathbf{p}_i + \frac{1}{2}\mathbf{v}(\mathbf{p}_i)$.

En conclusion, dans ce cas, l'algorithme DISCOVER va choisir :

$$\mathbf{u}(\mathbf{p}_2) = \tfrac{1}{2}\mathbf{v}(\mathbf{p}_3) \quad \mathbf{w}(\mathbf{p}_2) = -\tfrac{1}{2}\mathbf{v}(\mathbf{p}_3) \tag{1}$$

Enfin, deux autres étapes de traitement sont appliquées sur les champs de vecteurs de mouvement : premier, les vecteurs sont raffinées autour de la position dans l'équation (1) ; seconde, les champs sont régularisées par un filtre médian pondéré. On obtient ainsi un couple de champs à utiliser pour la compensation de mouvement de $I(k-1)$ et $I(k+1)$. La moyenne des images compensées sera l'estimation $\widetilde{I}(k)$ de $I(k)$.

Le décodage central est réalisé comme une combinaison linaire bloc-par-bloc des deux images fournies par les deux décodages latéraux, l'une reçue et l'autre interpolée. Le coefficients optimaux sont choisis à encodeur et envoyés de manière efficace au décodeur

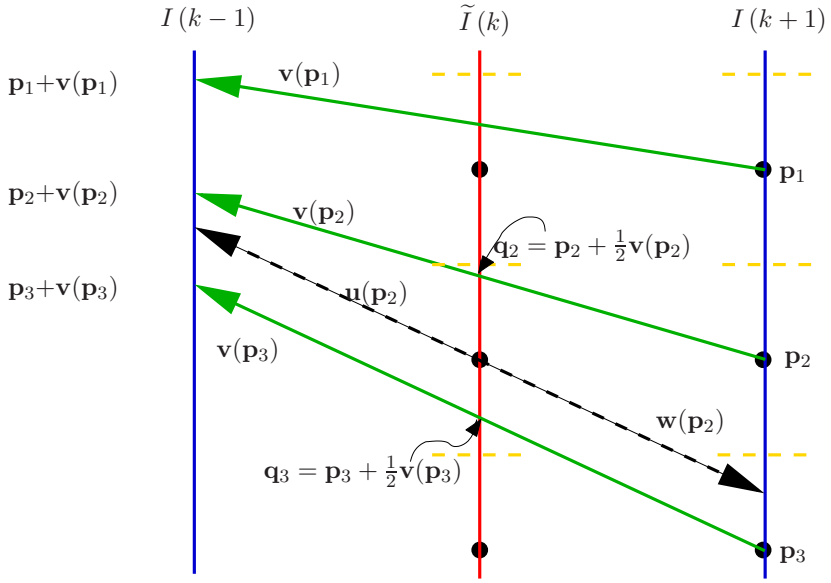


Figure 2: Estimation bidirectionnelle de mouvement dans DISCOVER.

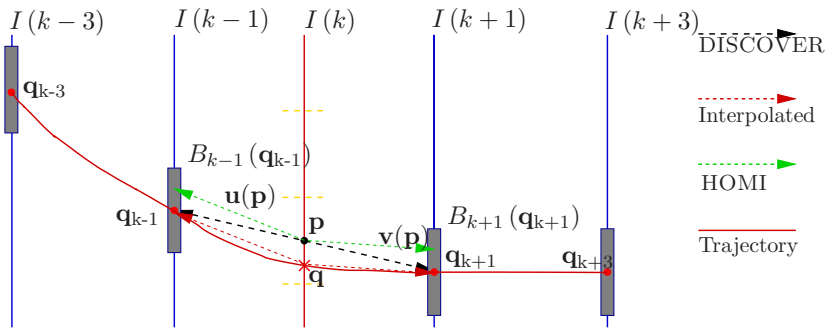
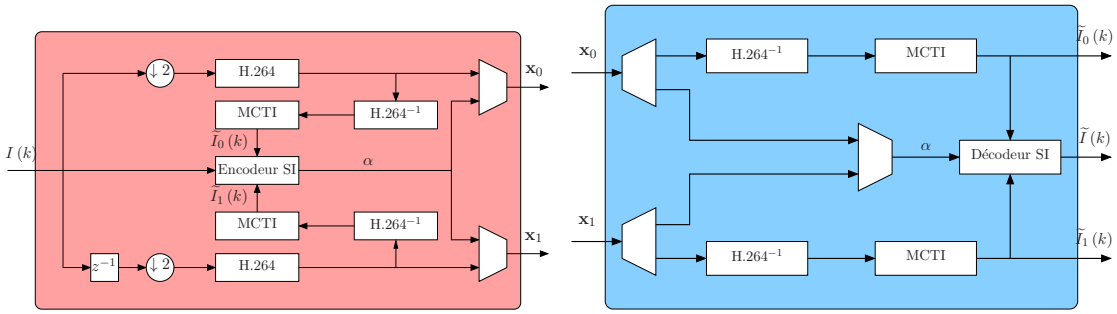


Figure 3: Algorithme d'estimation de mouvement d'ordre supérieur. La position du bloc dans la trame courante est estimée en interpolant sa trajectoire dans le deux bloc précédents et le deux bloc suivants.



(a) Encodeur (b) Décodeur

Figure 4: Schéma du système de vidéo-MDC proposé.

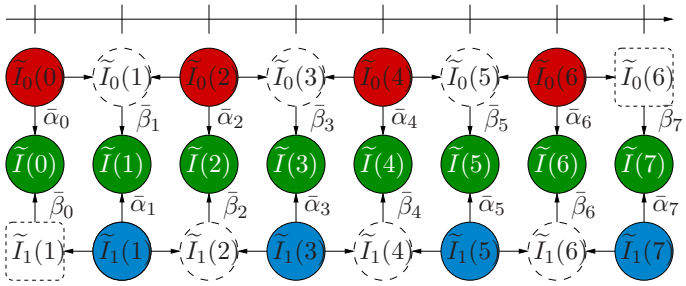
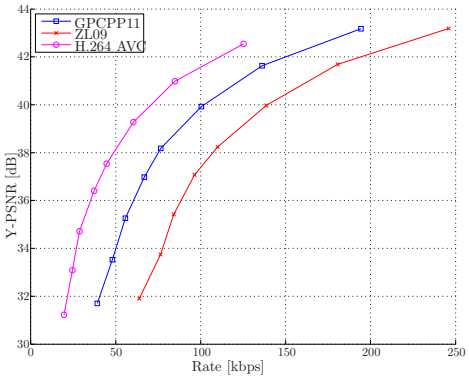


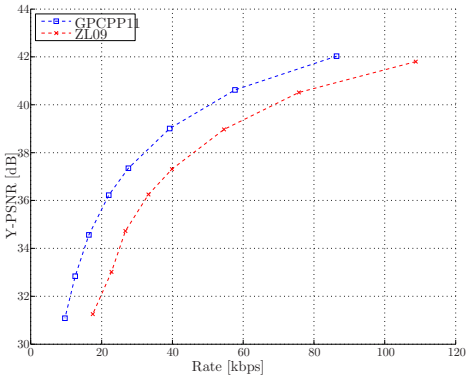
Figure 5: Structure du décodeur central.

comme information adjacente.

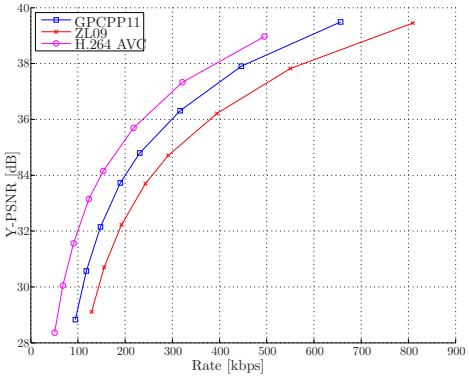
Cette technique montre un gain remarquable par rapport à l'état de l'art des méthodes de la même famille. Les résultats relatives aux développement de cette technique ont été présentés dans deux congrès internationaux, notamment le IEEE Workshop on Multimedia Signal Processing des années 2010 et 2011.



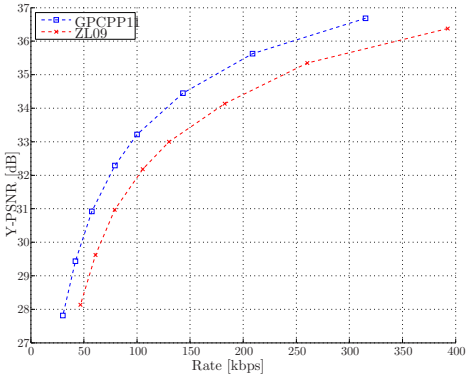
(a) “Akiyo” (Décodeur central)



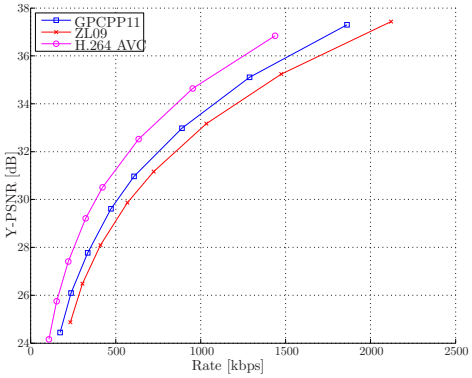
(b) “Akiyo” (Decodeur lateral)



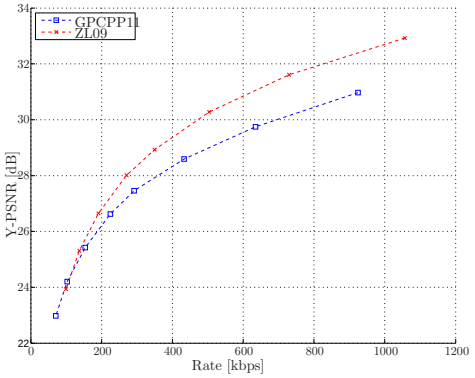
(c) “Foreman” (Decodeur central)



(d) “Foreman” (Decoder lateral)

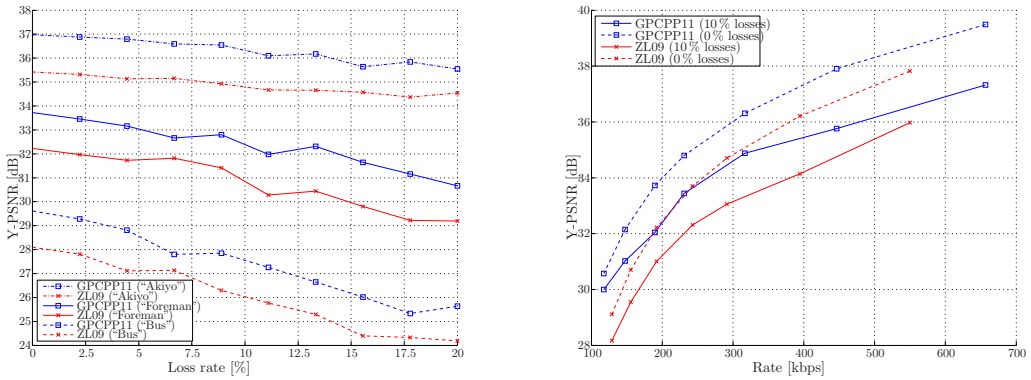


(e) “Bus” (Decodeur central)



(f) “Bus” (Decodeur lateral)

Figure 6: Comparaison entre le schema proposé et la technique de reference, pour plusieures sequences CIF à 30 trames par seconde. Le performace de H.264 AVC sont données pour reference.



(a) Qualité video *vs.* taux de perte pour un debit donné.

(b) Qualité video *vs.* debit pour un taux de perte donné.

Figure 7: Comparaison entre la technique proposée et la technique de reference dans un scenario avec pertes.

3 Protocoles de diffusion vidéo sur réseaux sans-fil ad-hoc

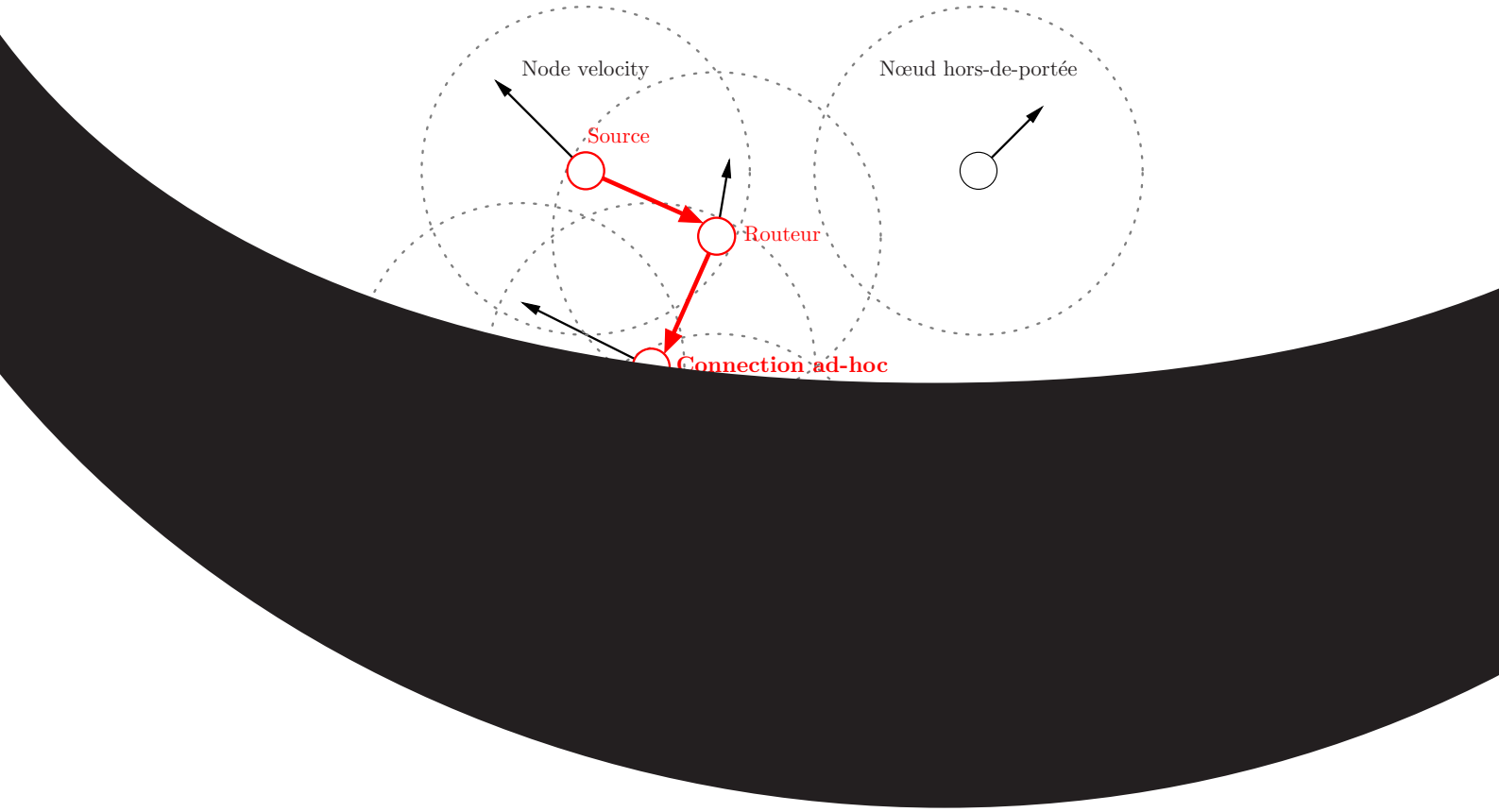
Dans le troisième chapitre, nous présentons l'une des contributions principales de cette thèse, c'est à dire, un protocole *cross-layer* pour la création et le maintien d'un réseau de *overlay*, qui, avec un échange de messages limité, gère de façon répartie un ensemble d'arbres de *multicast*, un pour chaque description du flux.

En utilisant une approche cross-layer, de l'information est échangée entre les différentes couches du protocole de communications, afin de permettre de développer une interaction plus efficace entre elles. Cette approche s'est montre déjà très efficace dans les applications de diffusion de la vidéo en temps réel sur réseau sans fil ad-hoc.

Après avoir présenté le problème de la diffusion vidéo en général et de la diffusion sur réseau sans fil en particulier, nous passons en revue les solutions disponibles dans l'état de l'art, et par un point de vue de la couche application, et par le point de vue de la couche réseau. Nous observons que ces solutions ne sont pas tout à fait satisfaisantes, car elles n'arrivent pas à exploiter au même temps les caractéristiques particulières de la diffusion vidéo et celle des réseaux mobiles.

On a donc évalué la possibilité de parvenir à une solution plus efficace grâce à une approche cross-layer. Avec cette approche innovative, et notamment l'échange d'information entre les couche de liaison de données, réseau et applications, on a pu réaliser un protocole original, nommé ABCD, capable d'assurer la livraison d'un flux vidéo en temps réel à une multitude de nœuds organisés dans un réseau ad-hoc.

Ce protocole a été validé expérimentalement dans une vaste gamme de situations différentes, et nombreuses propriétés du protocole ont été analysées. Dans tous ces scénarios, on a observé que notre protocole est capable d'assurer que le 100 % des nœuds reçoit presque toutes les trames transmises de chaque description, pour une densité du réseau



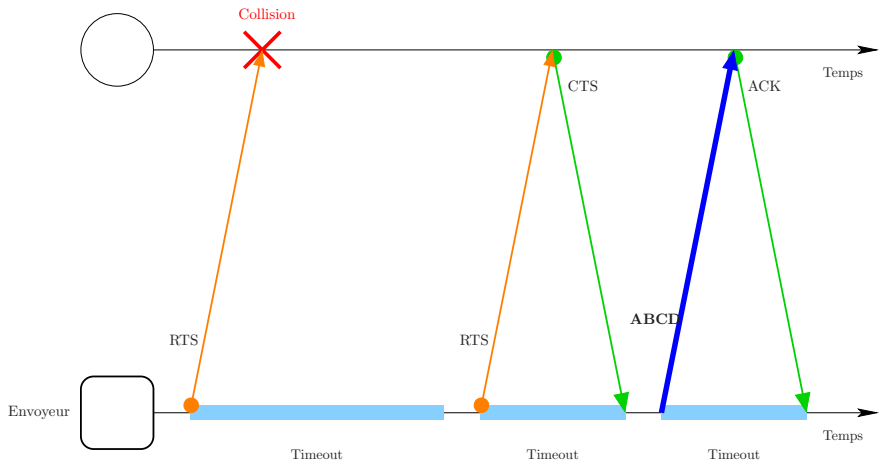
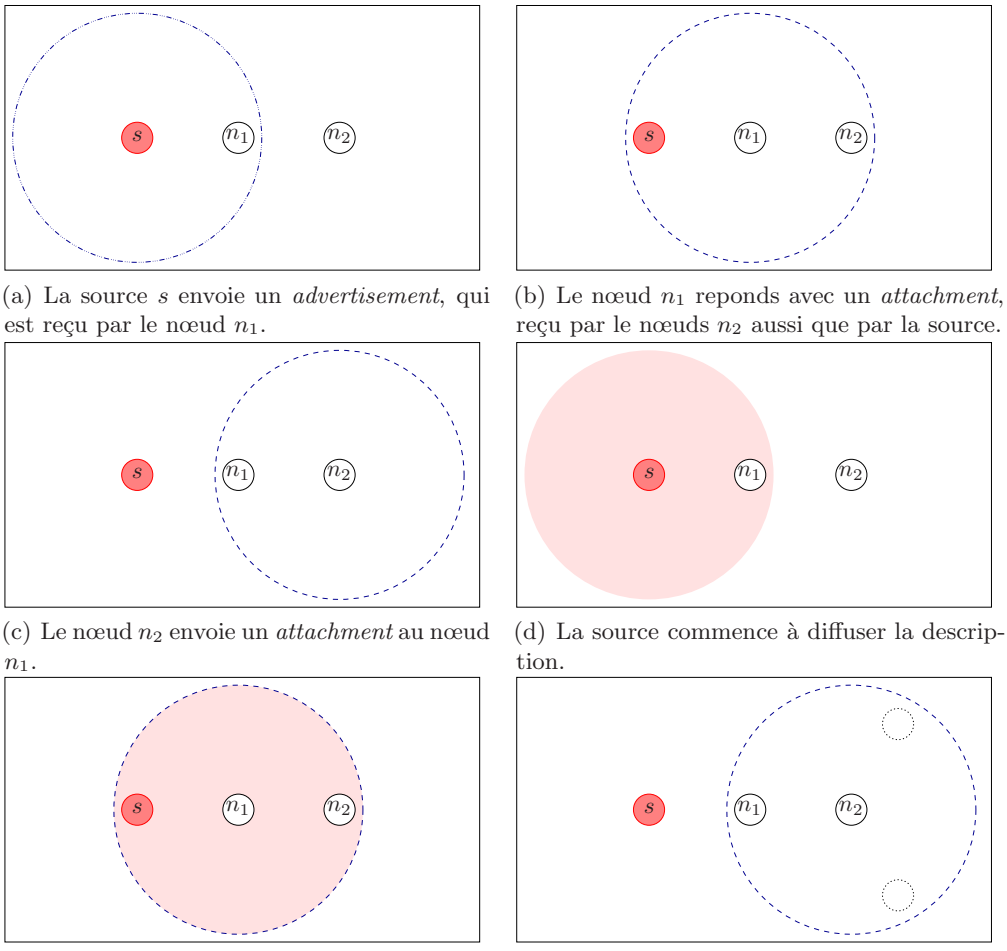


Figure 9: Schema de broadcast fiable. Le pacquet ABCD est envoié avec adresse de destination broadcast.



(a) La source s envoie un *advertisement*, qui est reçu par le nœud n_1.

(b) Le nœud n_1 reponds avec un *attachment*, reçu par le nœuds n_2 aussi que par la source.

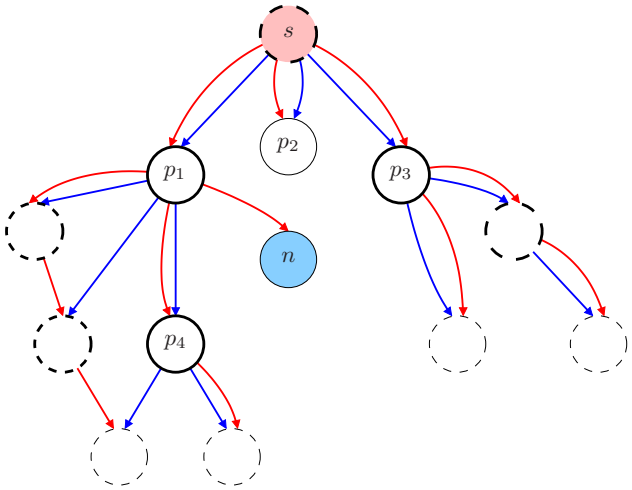
(c) Le nœud n_2 envoie un *attachment* au nœud n_1.

(d) La source commence à diffuser la description.

(e) Le nœud n_1 fait suivre la description au nœud n_2, en ajoutant son *attachment* pour la source.

(f) Le nœud n_2 envoie des *attachment*s periodiques au nœuds n_1, ce qui manifeste l'existence de la description à ses voisins.

Figure 10: Creation de l'overlay dans le protocole ABCD.



(a) Overlay pour deux description, *rouge* et *blue*. Le nœud n doit choisir son parent pour la description *blue*.

Candidat	h	a	d	g
p_1	1	0	1	3
p_2	1	1	0	0
p_3	1	0	0	2
p_4	2	0	0	2

(b) Valeurs usées par le nœud n dans la fonction d'optimisation pour choisir son parent.

Figure 11: Un exemple d'overlay de ABCD.

jusqu'à trois fois plus importante que la densité optimale –c'est à dire, environ 20 nœuds pour ensemble de voisins– et une vitesse de mouvement des terminaux jusqu'à deux fois la vitesse moyenne pour notre applications de référence —c'est à dire, environ 9 m/s. Le délai moyen est toujours inférieur à la centaine de millisecondes si la topologie du réseau change lentement, mais le délai maximum peut augmenter sensiblement si la topologie change brusquement, par exemple dans le cas d'une *flash crowd* ou d'une soudaine phase de mobilité élevée.

Les idées de base de cette technique et leur validation expérimentale ont été l'objet d'une publications sur la revue internationale Inderscience International Journal of Communication Networks and Distributed Systems. Des améliorations suivantes, ont été présentées à la 23ème éditions des *Colloques sur le Traitement du Signal et des Images* (GRETSI 2011) et au congrès IEEE Workshop on Multimedia Signal Processing du 2011.

4 Diffusion vidéo sur réseaux denses avec contraint de delai

Le quatrième chapitre est entierement dedié à une autre de nos contributions originales, notamment, un cadre original de fonctions distribuées pour une *optimisation congestion-distorsion*, qui, grâce à une représentation compacte des informations de topologie, permet aux nœuds d'apprendre la structure du overlay et d'optimiser leur comportement en con-

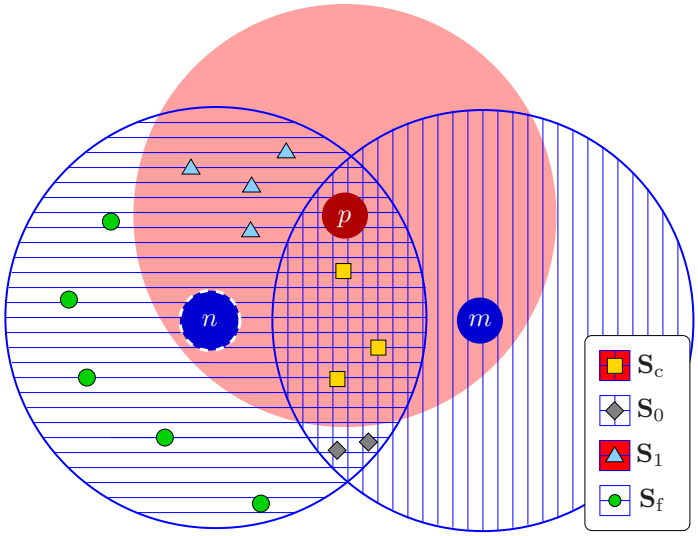


Figure 12: Effet des pertes sur la distortions des nœuds selon leur état.

séquence, ce qui permet une diffusion du flux video fiable en presence de contraints très stricts en termes de delai, meme dans des reseaux ad-hoc très dense.

Tous d'abord, nous donnons une definition formelle du probleme et nous introduisons la notations des parametres avec lesquels on operera. En particulier, nous allons definir un probleme de minimisation contrainte qui reflecte l'effor des nœuds de minimiser la distortion moyenne des nœuds du reseau, sous contrainte de la congestion maximale imposée par le canal.

Le paramentre sur lequel nous allons agir est le nombre d'essais k permis à la couche MAC pour acceder au canal, avan de considérer la transmission echouée. Ce problème de minimisation contrainte peut être mis en forme Lagrangienne de la maniere suivante :

$$k^* \triangleq \arg\min_{k\in\mathbb{N}} \{D(k) + \lambda C(k)\},$$

Ensuite, nous montrons comment les nœuds peuvent estimer les parametres de congestion et distortion grace à un exchange efficace de messages qui exploite les proprietés du protocole de creation de l'overlay.

En particulier nous montrons que la distortion moyenne sur le reseau peut etre estimee efficacement si l'on etablie l'effet des pertes sur la base du nombre de descriptions que les nœuds son déjà en train de recevoir, pas que sur leur arbre de multicast, mais aussi de maniere collaterale à travers des transmission adressees à leur voisins.

Ce transmission collaterales ne sont pas trivales à determiner ; on a donc dû developper un protocol à ce fin qui propage l'état des nœuds vers la sources du flux video, aufin de permetre un optimisation efficate.

Nos simulations on montré que, s'il on impose une contrainte stricte sur le delai, notre technique offre un gain important en termes et de PSNR et de reduction du delai, pour des

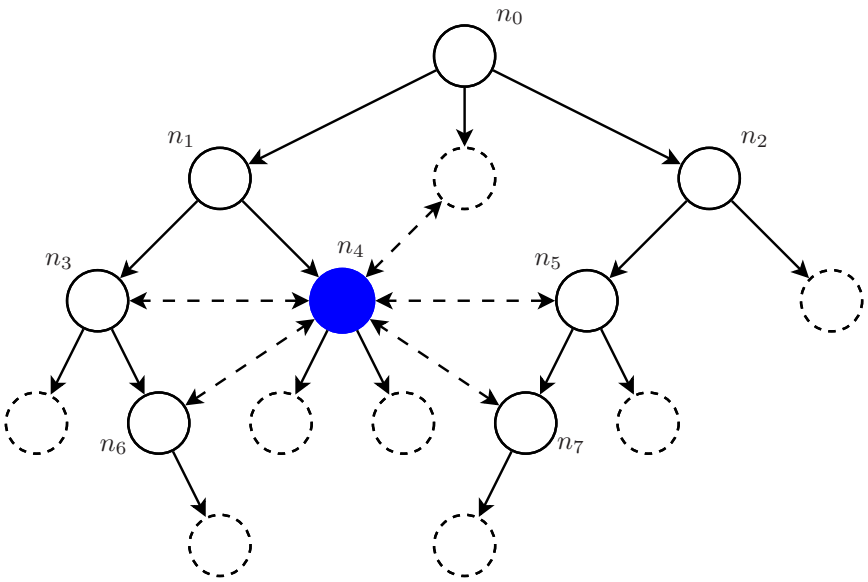


Figure 13: Etat d'un nœud sur la base de ses connexions avec ses voisins.

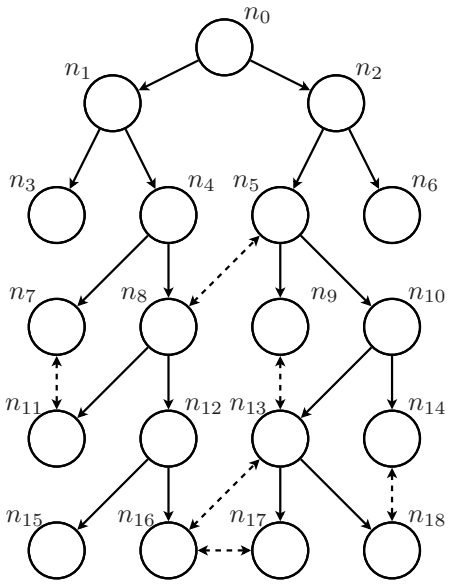


Figure 14: Protocole d'estimations de l'état des nœuds.

debit jusqu'à quelques megabit par seconde, ces deux derniers compatible avec un service de type *conversational*. La methode et les resultat presentés dans ce chapitre ont été l'objet d'une publications sur la revue internationale IEEE Transactions on Multimedia.

5 Codage reseau pour la diffusion vidéo

Dans le cinquieme chapitre, nous traitons le sujet du *Codage Reseau* (ou NC, de l'anglais Network Coding), un paradigme dans lequel, dans le transport d'un paquet qui traverse plusieurs liens, le nœuds du reseau peuvent melanger ensemble plusieurs des paquet d'information qu'ils ont reçu, au lieu de se limiter à en retransmettres de copies. Grace a cette approche, il est possible de transmettre le maximum de flux sur le reseau. Un exemple est montré dans la Figure 15.

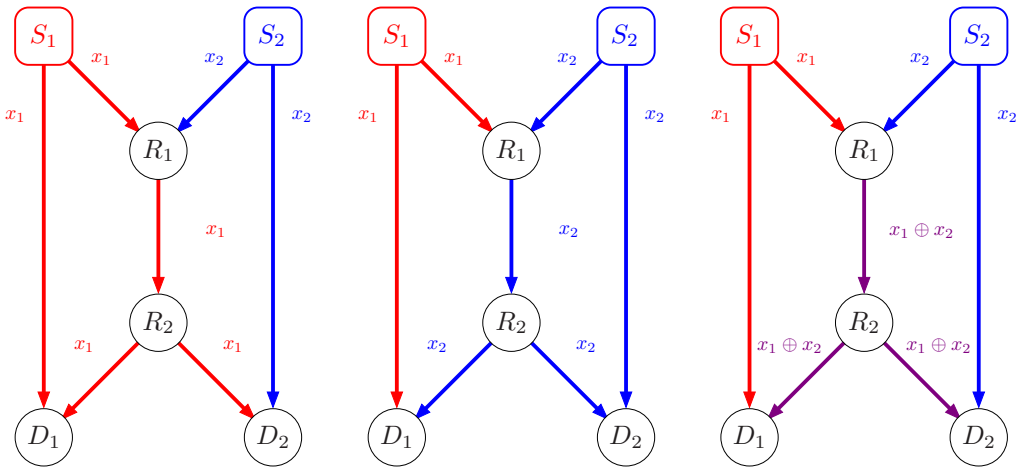
Dans ce chapitre, nous donnons tout d'abord les principes de base du codage reseau et nous passons en revue le technique de codage les plus prometteuse.

Ensuite, nous analysons la possibilité d'appliquer le codage reseau aux flux video, encodé soit en description simple sout en desriptions multiples, pour fournir un service plus robust de diffusion robuste vidéo en temps réel sur des reseaux non-fiables, tels que les réseaux sans fil ad-hoc presentés dans les chapitres precendentes. Dans ce context, nous presentons deux contributions originales.

Pour la premiere technique nous proposons une formulations du probleme de la diffusion d'un flux video encodé par desciptions multiples sur un reseau ad-hoc en termes d'optimisation d'un ensable de coefficients de combinaison des paquets. Ensuite, nous introduison une fonction objectif qui prends en compte les effect qui le decodage d'un nombre reduit de descriptions a sur la distortions totale. Ce cadre de fontions a été integré dans le protocole ABCD presenté dans le troisieme chapitre, ce qui nous permet d'avoir à la fois un graphe acyclique et la necessaire connessaince de l'état des voisins.

Enfin, nous comparons les preformances de notre technique avec le celebre technique de *Codage Reseau Pratique* (ou PNC, de l'anglais Practical Network Coding) combinée avec un *flooding* aleatoire. Nous observons que les limitations sur la taille de la generation par le nombre de desciptions, imposées par la contrainte du delai, a un effet tres negatif sur les performances de la technique de reference, qui par consequent est sistematiquement surpassé par l'approche que nous proposons.

Cette technique et les résultats relatives ont été présentés dans le congrè international IEEE International Conference on Acoustics, Speech and Signal Processing du 2012.



(a) Le message x_1 est envoyé sur le lien *bottleneck*.

(b) Le message x_2 est envoyé sur le lien *bottleneck*.

(c) Une combinaison de x_1 et x_2 est envoyé sur le lien *bottleneck*.

Figure 15: Le *réseau papillon* est un exemple classique des applications du codage réseau. Chaque lien a une capacité d'un message par transmission. Les sources S_1 et S_2 veulent envoyer leurs messages, respectivement x_1 et x_2, aux dexu nœuds D_1 et D_2. Le lien entre R_1 et R_2 est donc un *bottleneck*.

Abstract

During the last decade, real-time video streaming over wireless ad-hoc networks has gathered a steadily increasing interest, because of the attractive property of being able to deploy a visual communication system anytime and anywhere, without the need for a pre-existing infrastructure. A wide range of target applications, from military and rescue operations, to business, educational, and recreational scenarios, has been envisaged, which has created great expectations with respect to the involved technologies. The goal of this thesis is to provide an efficient and robust real-time video streaming system over mobile ad-hoc networks, proposing cross-layer solutions that overcome the limitations of both the application and network solutions available at this time. Our contributions cover several aspects of the mobile video streaming paradigm: a new multiple description video coding technique, which provides an acceptable video quality even in presence of high loss rates; a novel cross-layer design for an overlay creation and maintenance protocol, which, with a low overhead, distributedly manages a set of multicast trees, one for each description of the stream; an original distributed congestion-distortion optimisation framework, which, through a compact representation of the topology information, enables the nodes to learn the structure of the overlay and optimise their behaviour accordingly; and, finally, an integration with the emerging network coding paradigm, which allows an efficient employment of these approaches in networks that suffer of a bottleneck capacity. All these approaches aim at enhancing the quality of the visual communication, in terms of distortion and robustness, with regard to the delay perceived by the users.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ABCD	A Broadcast Content Delivery Protocol
ACK	ACKnowledgement
ALM	Application Layer Multicast
AODV	Ad-hoc On-demand Distance Vector
ARP	Address Resolution Protocol
ATR	Augmented Tree-based Routing
AVC	Advanced Video Coding
BMA	Block Matching Algorithm
CA	Context-Adaptive
CABAC	Context-Adaptive Binary Arithmetic Coder
CAVLC	Context-Adaptive Variable Length Coder
CCR	Conservative CTS Reply
CDF	Cumulative Distribution Function
CDN	Content Distribution Network
CDS	Connected Dominating Set
CIF	Common Intermediate Format
CSMA	Carrier Sense Multiple Access
CSMA/CA	CSMA with Collision Avoidance
CTS	Clear To Send
DAG	Directed Acyclic Graph
DCT	Discrete Cosine Transform

DFT	Discrete Fourier Transform
DISCOVER	DIStributed COding for Video sERvices
DSDV	Destination-Sequenced Distance Vector
DSR	Dynamic Source Routing
DVC	Distributed Video Coding
DWT	Discrete Wavelet Transform
EWNC	Expanding Window Network Coding
FGS	Fine Granularity Scalability
FMO	Flexible Macroblock Ordering
GF	Galois Field
GOP	Group Of Pictures
GPS	Global Positioning System
HDTV	High Definition TeleVision
HEVC	High Efficiency Video Coding
HNC	Hierarchical Network Coding
HOMI	High-Order Motion Interpolation
ICMP	Internet Control Message Protocol
IP	Internet Protocol
IQR	InterQuartile Range
ISDN	Integrated Services Digital Network
ISP	Internet Service Provider
JPEG	Joint Photographic Experts Group
KLT	Karhunen-Loéve Transform
LCM	Linear Code Multicast
MAC	Medium Access Control
MANET	Mobile Ad-hoc NETwork

MC	Motion Compensation
MCTI	Motion-Compensated Temporal Interpolation
MD	Multiple Description
MDC	Multiple Description Coding
MDCT	Multiple Description Correlating Transform
MDQ	Multiple Description Quantisation
MDVC	Multiple Description Video Coding
ME	Motion Estimation
MINIMA	Mutual INformation IMprovement Algorithm
MPEG	Moving Picture Experts Group
MSE	Mean Squared Error
MV	Motion Vector
MVF	Motion Vector Field
NA	Network Awareness
NC	Network Coding
NCT	Normalised Connection Time
NCV	Network Coding for Video
OLSR	Optimised Link-State Routing
PDF	Probability Density Function
PHY	PHYsical Layer
PMF	Probability Mass Function
PNC	Practical Network Coding
PSNR	Peak Signal-to-Noise Ratio
QFE	Quantised Frame Expansion
QoE	Quality of Experience
QOLSR	QoS-enhanced OLSR

QoS	Qualtiy of Service
QP	Quantisation Parameter
RAD	Random Assessment Delay
RD	Rate-Distortion
RDO	Rate-Distortion Optimisation
RGB	Red, Green, and Blue
RLNC	Random Linear Network Coding
RPGM	Reference-Point Group Mobility
RTS	Request To Send
RTT	Round Trip Time
SAD	Sum of Absolute Differences
SD	Single Description
SDC	Single Description Coding
SI	Side Information
SNIR	Signal-to-Noise-plus-Interference Ratio
SNR	Signal-to-Noise Ratio
SP	Spatial Prediction
SSD	Sum of Squared Differences
SSIM	Structural SIMilarity
SVC	Scalable Video Coding
TCP	Transmission Control Protocol
TORA	Temporally-Ordered Routing Algorithm
UEP	Unequal Error Protection
WiFi	Wireless Fidelity
WZC	Wyner-Ziv Coding
XOR	eXclusive Or
Y-PSNR	Luminance-component PSNR

Introduction

Context and Motivation

During the last few decades, thanks to relatively low-cost broadband connections and the high computational power available to the average user, jointly with the advancements in digital signal compression, the majority of Internet services has shifted from text-based to multimedia.

Looking at the evolution of the Internet, we observe that the first *killer applications*, *i.e.*, those applications that have determined the success of the Internet, have been text-based. The wide acceptance of the Internet as a global communication and information tool took place through the introduction of the e-mail and chat services, that quickly interconnected people for both business and leisure; the bulletin board service, that first allowed users to upload and download software and data, post and read news and bulletins; the web browsing, for news and information retrieval, on-line shopping, and – more recently – the social networking.

With the subsequent larger diffusion of broadband connections, and thanks to the introduction of the *peer-to-peer* paradigm, the users have grown progressively accustomed to *multimedia* services — at the beginning, mostly music (in the celebrated *MP3* format) and images (in *JPEG* and *JPEG 2000*). All this belongs to the the past: nowadays the Net is dominated by video content.

Apple Inc. has reported to have sold over 30 million of videos through its iTunes Store just between 2005 and 2006[1]; nowadays it has sold over 200 million TV episodes, of which over one million in high definition, and several million of feature-length films[2]. During the same period of time, the popular video hosting service offered by YouTube has passed from serving about 100 million video viewings *per day* in 2006[3], to 1 billions in 2009, and 2 billions in 2010[4]. Moreover, according to the *Global Internet Phenomena Report* compiled by Sandvine Inc. in autumn 2011[5], the most successful American provider of on-demand Internet streaming media, Netflix Inc., is nowadays the largest source of North American

[1] Jim Feeley, *Video everywhere*, PC World, April 2006.
[2] Statistics compiled by Apple Press Info `http://www.apple.com/pr`, consulted in April 2012.
[3] *YouTube serves up* 100 *millions videos a day online* (Reuters), USA Today, July 2006.
[4] Statistics compiled by `website-monitoring.com`, consulted in April 2012.
[5] `http://www.sandvine.com/news/global_broadband_trends.asp`, consulted in April 2012.

Internet traffic, accounting for almost 25 % of the aggregated traffic.

The volume of traffic over the Internet due to video content has been growing exponentially, and is expected to continue growing.

As David Hsieh (vice-president of Cisco Systems Inc. for Tele-Presence and Emerging Technologies) has pointed out in an interview[6] "video is invading all aspects of our lives [...]. Today, over half of all Internet traffic (51 %) is video". According to his estimations, in three years, digital video content will account for 90 % of Internet traffic.

These days, video options are proliferating at an astonishing pace: everything, from Hollywood films and TV programmes to clips from ordinary users, is available to whomever is connected to the Internet, whether with a laptop, a notebook, or a tablet. More recently, even mobile phones – built with advanced computing ability and connectivity and commonly referred to as *smartphones* – can constantly access video content with news, sports, and video segments. The new frontier of network communications lies in this new paradigm: "Video Anywhere at Anytime".

In this scenario, personal video communication systems based on relatively inexpensive hardware (such as a web-cam connected to personal computer, or a smartphone or tablet with a built-in camera) and providing transmission over packet-switched networks of compressed video have become affordable to the general public. *Video-conferencing* has made significant inroads into business, education, medicine, and media, reducing the need to travel to bring people together. An important aspect of mobile video-conferencing is its ability to transfer to a wide audience a live conversation wherein non-verbal (*i.e.*, visual) information is as an important component of the conversation as the verbal part.

Nowadays, this can be achieved through *ad-hoc networking*, a technology that provides unprecedented means to deploy a sophisticated, interconnected, and interoperable video dissemination system for communication, surveillance, reconnaissance, and order dispatching in environments without pre-existing communication infrastructure.

The possibilities for a dynamic, flexible, and infrastructure-less real-time video dissemination service are limitless. Such a service could be used to centrally coordinate the efforts of the disaster-relief personnel in an emergency situation, *e.g.*, rescue forces searching for, and provisioning aid to, people who are in distress or imminent danger, such as in the aftermath of a hurricane, an earthquake, or a building collapse.

Less dramatically, infrastructure-less real-time video dissemination can be used in business meetings, to offer multimedia presentations to the participants, or in a college lesson, to display video support material to the students. It can be used to allow users to watch football games or popular movies in the underground, or in a building. It could be even used to add an "augmented reality" flavor to sport, musical, and entertaining events in general, giving the possibility to the spectators to enjoy the event they are attending from different points of view (*e.g.*, with a continuous feed from cameras on the playing field or

[6] Andy Plesser, *Cisco: Video will Account for* 90 % *of Net Traffic in Three Years*, October 2011.

on the stage).

However, as we shall see in detail in the following section, ad-hoc video distribution is anything but a trivial task, as it requires an understanding of several problems, located at several layers, to which many studies have been dedicated.

The goal of this thesis is to identify the main challenges facing the design of a flexible and robust video dissemination system over ad-hoc networks, establishing the level of technical development available at this time in the field, and propose innovative and efficient ways to design systems that go beyond the limitation of the current technology.

Challenges and Objectives

Even though the technology involved has greatly advanced during the past few years, a great deal of further improvement is still needed both in the domain of video coding and in the domain of networking.

During the work conducted in preparation of this thesis, we identified several challenges that have to be dealt with in order to provide an efficient and robust broadcast service of real-time video over wireless ad-hoc network.

Mobile ad-hoc networks, as wireless networks in general, have a significantly lower capacity and a much higher expected packet loss rate than wired networks. This is due to several factors: the physical transmission over the channel is less reliable, due to power attenuation, fading, shadowing and multi-user interference, and other physical issues resulting in a time- and location-varying channel conditions. Also, mobile terminals rely on battery power, which is a scarce resource, and are far less dependable than Internet servers, routers, and clients.

This calls for a video coding technique that on the one hand is effective in the sense of reducing the bitrate needed for transmission for a given video quality, and on the other hand is capable to provide a graceful degradation in presence of losses.

Moreover, aside from the inherent communication problems of wireless networks in general, a video dissemination service over ad-hoc networks must also cope with the lack of centralised control that makes possible to deploy the network without pre-existing infrastructure. One of the main challenges is therefore to provide an *overlay network* (*i.e.*, a logical network superposed to the physical one that provides routing capabilities) in an environment where nodes can connect and disconnect dynamically, be located at arbitrary positions, and move in a random manner. The agents of the overlay have to self-organise in order to identify newly connected or moved-in-range devices and transmit the information about their existence wherever it is needed for optimal route selection; this information must also be kept up-to-date throughout the network, and each node should contribute to the healing of the overlay any time another node disconnects or moves out of its range.

Furthermore, the overlay should be *scalable*, *i.e.*, it should be able to provide an acceptable level of service even in presence of a large number of devices, when the simultaneous

transmission by several nodes is likely to generate *congestion*. This is a particularly important when the overlay is designed to deliver a live video stream, where the latency of data is crucial to the application.

Structure of this manuscript

This manuscript addresses all these challenges, providing, for each aspect of the ad-hoc streaming service, a state-of-the art of the solutions currently available and proposing novel solutions, that have been designed, implemented, and tested during this thesis and that led to several publications on scientific journals and international conferences. In more detail, the rest of this manuscript is organised as follows.

Chapter 1 — Video coding

In this chapter, we introduce the fundamental concepts of *video coding*, and present the notation that we will be using throughout the rest of the manuscript, by giving a formal definition of the video coding problem and of its goals. Also, we conduct a rapid study of the basic tools used in video coding and provide a brief overview of the most relevant video coding standards. This chapter does not contain original contributions, as its purpose is merely to provide an acquaintance with the video coding principles in general, with the most recent technologies, and with the tools and concepts that we shall use in the rest of this thesis.

Chapter 2 — Multiple description video coding

In this chapter, we discuss the concept of *multiple description coding* (MDC), a coding technique meant to provide a trade-off among coding efficiency, robustness toward losses, and computational complexity. We present the basic principles of MDC, and detail how it can be used to provide a video stream with robustness toward errors and losses when transmitted over unreliable networks where no retransmissions are tolerated, such as the wireless ad-hoc networks. Also, we present in this chapter our own contribution to the field: a multiple description video coding technique that will constitute one of the building blocks of our mobile video streaming system. This original contribution is based on the use of a motion-compensated temporal interpolation technique for side reconstruction and a block-wise linear combination of the two descriptions for central reconstruction, with an efficient encoding of the combination coefficients. We present an extensive experimental validation of our contribution, analysing a wide range of relevant features under a variety of conditions.

Chapter 3 — Video streaming protocols for ad-hoc networks

In this chapter, we focus on *mobile ad-hoc* environments, and how they may be used to provide a *live streaming* service to a set of uncoordinated mobile devices. We study the problem of video streaming, with particular focus on mobile networks, and give a review of currently available solutions, from both an application and a network point of view. In this chapter, we also present one of the main contributions of this thesis, *i.e.*, an efficient protocol, designed with a cross-layer approach, for video streaming over ad-hoc networks.

Chapter 4 — Streaming over dense networks with low-latency constraints

This chapter is entirely dedicated to another original contribution of this thesis, *i.e.*, a framework designed to allow video streaming under stringent delay constraints in congested ad-hoc networks. In particular, we study how a transmission protocol, such as the one introduced in the previous chapter, can be integrated with a *congestion-distortion optimisation* framework. This consists in solving a constrained minimisation problem that jointly takes into account the average video distortion of the users and the congestion of the channel. We also provide efficient distributed techniques to estimate the parameters needed for the optimisation. Our experimental validation shows that this contribution grants a consistent gain in terms of both video quality and latency.

Chapter 5 — Network coding for video delivery over unreliable networks

In this chapter, we present the emerging *network coding* technique, introducing the theory behind it and providing an overview of the most promising techniques, with particular attention to the video streaming and mobile network applications. We also study how this paradigm shift can be applied to our scenario, integrating it with the techniques presented in the previous chapters, and present our own original contributions in this field. In our first contribution, we extend the cross-layer protocol presented in Chapter 3 by adding coding capability to the nodes, *i.e.*, allowing nodes to send linear combinations of the descriptions. The second contribution consists in a per-hop transmission policy, based on the joint use of network coding and multiple description coding, that provides a good trade-off between loss resiliency and decoding delay. Both techniques have given promising results in terms of performance of the streaming system.

Chapter 1

Video coding

In this chapter, we shall introduce the basic concepts of *video coding*, a topic of great importance in the field of telecommunication engineering, used whenever digital storage and transmission of video signals occur [Bov00].

First of all, in Sec. 1.1 we shall give a formal definition of the video coding problem and of its goals. Then, in Sec. 1.2, we shall introduce the basic tools used in video coding. Finally, in Sec. 1.3, we shall briefly review the most relevant standards in video coding.

1.1 Definition and purpose

In this section, we shall introduce the concept of video coding or *video compression*. The definition of video coding system will be introduced, along with its objectives.

Uncompressed colour images are represented digitally providing a triplet of values, referred to as *colour channels*, for each sampling point. These values are interpreted as coordinates in some colour space. The RGB colour space is commonly used for display, but other spaces, such as YCbCr, are used in the context of image and video coding.

The YCbCr model, also referred to as YUV, although the term is rigorously used for analog encoding of colour information, defines a colour space in terms of the *luma component* (Y), and blue-difference and red-difference chroma components (Cb and Cr). The luma component is similar to luminance, but different in that light intensity is non-linearly encoded using gamma correction [Bov00, Ch. 1].

For the sake of simplicity, we are assuming here that we deal with grayscale sequences, meaning that only the luma or luminance component needs to be encoded. However, the extension to the case of colour coding is usually quite straightforward.

First of all, let us introduce some definitions and notations. We define a *video frame* as a bi-dimensional signal $I_{n,m}$. A video frame is an image: the indices $n \in \{0, 1, \cdots, N\}$ and $m \in \{0, 1, \cdots, M\}$ are space coordinates identifying a sampling point in the image, while the values of I, in $\{0, 1, \cdots, 2^b - 1\}$, represent the luminance intensity in each point.

The samples of a frame are referred to as *picture elements*, or *pixels*. A *video sequence* is a three-dimensional signal $I_{n,m}(k)$ consisting in a sequence of frames, with $k \in \{0, 1, \cdots, K\}$ being the time variable.

Video coding is the process of compacting the data contained in $I_{n,m}(k)$ into a smaller number of bits [RM94, Ric03], in order to ease storage and transmission. The transmission of a *raw*, *i.e.*, uncompressed digital video sequence would require unsustainable bitrates. For instance, a standard definition television signal (*i.e.*, a video signal of 768×576 pixels per frame, at 30 frames per second) would require about 200 Mbps to be transmitted and a 2 hours movie would require about 180 GB to be stored. This has made necessary the introduction of video compression. More formally, we want to be able to design a pair of coupled systems – an *encoder* and a *decoder* – often described as *codec* (Fig. 1.1). The encoder takes the video sequence $I_{n,m}(k)$ as an input and produces its compact binary representation $\mathbf{x}$, known as *encoded bitstream*, which is smaller than the raw video sequence, *i.e.*, represented on fewer bits. The decoder is able to convert the encoded bitstream in a video sequence $\widetilde{I}_{n,m}(k)$, referred to as *reconstructed sequence*, which is an approximation of $I_{n,m}(k)$.

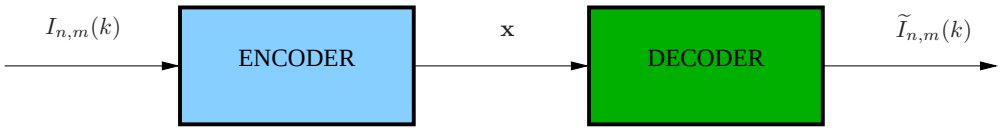


Figure 1.1: General structure of a coding system.

Several aspects must be taken into account to evaluate the performance of a coding process: the *rate* needed to represent $\mathbf{x}$ and the *distortion* of $\widetilde{I}_{n,m}(k)$ with respect to $I_{n,m}(k)$, the *robustness* of the encoded bitstream with respect to losses and errors in the transmission channel, the *delay* in the decoding process, and the computational *complexity* of encoder and decoder.

The rate, which we want to minimise, is related to the number of bits required to represent $\mathbf{x}$. It is usually expressed in bits per picture element (*bits per pixel*, or bpp) or per time unit (*bits per second*, or bps).

The distortion is the degradation introduced by the coding process in the reconstructed sequence. The most popular measures are based on frame error, defined as $d(n, m) = I(n, m) - \widetilde{I}(n, m)$. Among the most common distortion metrics there are: the *sum of absolute differences*, or SAD; the *sum of squared differences*, or SSD; and the *mean squared*

error, or MSE, defined for each frame k as follows:

$$\begin{aligned} \mathrm{SAD}(k) &= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|I_{n,m}(k)-\widetilde{I}_{n,m}(k)\right|, \\ \mathrm{SSD}(k) &= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\left|I_{n,m}(k)-\widetilde{I}_{n,m}(k)\right|^2, \\ \mathrm{MSE}(k) &= \frac{\mathrm{SSD}(k)}{NM}. \end{aligned}$$

For video sequences, the average of these measures over all the frames is considered.

These measures present several advantages: they are simple to compute, they have a simple geometrical interpretation (*i.e.*, they are related to the norm in an (NM)-dimensional space), and they show a good degree of correlation with respect to the human quality perception [WBSS04, WB09]. In particular, SSD and MSE are equivalent, in the sense that both are known if either one is specified.

Given the MSE, it is also possible to assign a quality metric to the reconstruction, in terms of *Peak Signal-to-Noise Ratio*, or PSNR:

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{\left(2^b-1\right)^2}{\mathrm{MSE}}\right),$$

where b is the number of bits per pixel in the original image. For colour sequences, it is possible to apply PSNR on the chrominance components as well. However, in the context of video coding, the metric is usually applied only on the luminance component and also referred to as Y-PSNR.

Fixed a target bitrate R for the encoded sequence, a video coding scheme aims at minimising the distortion D of the decoded sequence. This constrained optimisation problem is commonly approached with the method of Lagrange multipliers. The unconstrained Lagrangian cost function to minimise is:

$$J = D + \lambda R. \tag{1.1}$$

The minimisation of such a cost function J is referred to as *Rate-Distortion Optimisation*, or RDO. Rate-distortion is a major branch of information theory, providing the theoretical foundations for lossy data compression; it was introduced in the seminal works written in 1948 and 1959 by C. E. Shannon, the founder of information theory [Sha48, Sha59] and has been an active topic of research ever since [SW98, WSJ+03].

Even though video codecs are primarily characterised in terms of rate-distortion, other issues must be considered as well in a practical design of a video transmission system, such as *coding delay* and *complexity* [SW05]. The coding delay is the gap between coding order and reproduction order. It originates from the fact that video coders exploit the temporal

redundancies in the video signal through temporal prediction (see Sec. 1.2.1), hence the coding order has to be chosen in a way such that reference frames are coded (and decoded) before the frames predicted upon them. This delay, which is *per se* a nuisance in some applications, also comes with a memory requirement for the decoder, as it has to store the reference frames in a buffer [SMW06]. Another important issue in the design of a codec is complexity, in terms of computational power, memory capacity, and memory access requirements. In general, video coders that present a higher degree of adaptation to the input signal are able to achieve a higher coding efficiency, but at the expense of a higher complexity. As an example, the MPEG-4 part 10/H.264 AVC standard (see Sec. 1.3.2) has a 60 % rate gain for a given quality compared to previous standards, but at the expense of an increased encoding complexity, which is about four times that of MPEG-2 part 2/H.262 and twice that of MPEG-4 part 2 [OBL$^+$04, Ric10]. In general, assessing the complexity of a video coding standard is not a simple task, as its implementation complexity heavily depends on the characteristics of the platform on which it is mapped. It is however possible to evaluate whether a codec complexity level is sustainable for a specific platform, *e.g.*, mobile computing as opposed to wired environments, so that in different environments, different implementations may be adopted. It should also be considered that, as technology advances, a given complexity becomes more and more acceptable.

As we can see, all these factors are mutually influenced, and in general one can be improved only at the expense of the others. However, different applications typically have different requirements in terms of compression efficiency, delay, and complexity (mainly decoding complexity, since encoding is not standardised). For this reason, most modern standards define *profiles* and *levels*, which allow the applications to support only subsets of the standards. The profile defines a subset of features that the decoder shall support, such as compression algorithm, video format, *etc.*. The level, on the other hand, defines a subset of quantitative capabilities, such as maximum bitrate, maximum frame size, *etc.*.

1.2 Video coding tools

In this section, we shall identify the basic components that make up a video coding system and briefly describe the main concepts behind them, before showing their interaction in a typical hybrid video encoder.

1.2.1 Temporal prediction

Temporal prediction aims at eliminating the *temporal redundancy*, *i.e.*, similarities between neighbouring video frames, existing in the video sequence. The prediction consists in an estimation of what the current block could be, given a reference set of frames available at both the encoder and the decoder. The output of the temporal prediction module is subtracted from the current block to produce a *residual block* that conveys the prediction

error and set of model parameters representing the prediction that has been made.

The state-of-the-art in temporal prediction is block-based *motion compensation* or MC, a technique that describes the current block in terms of rigid translations of blocks in the reference frames. The reference frames may be in the past or even in the future, as long as they are also available at the decoder. Each block is predicted from a block of equal size in a reference frame, whose translation is represented by a *motion vector*. A good choice of the motion vectors is crucial to the performance of the codec. The process of estimating the displacement between two frames is commonly referred to as *motion estimation* or ME. The most commonly used technique in ME for video coding is *block matching* or BMA. A block matching algorithm looks for a correspondence of the current block with a block in the reference frame such that it minimises a given distortion metric (*e.g.*, SAD or SSD). Since the motion vectors have to be encoded and sent to the decoder themselves, it is common to actually perform an RD-optimised choice of the motion vectors, *i.e.*, finding the best trade-off between the quality of the prediction and the rate needed to encode the motion vectors. However, it should be noticed that the decoder does not need to know how the choice of the motion vectors has been made, thus the motion estimation algorithms are usually not standardised.

In general, a more accurate temporal prediction can be obtained if more than one reference is used, and the best predictions are obtained predicting *bidirectionally* from both previous and future frames (*B-frames*). This, however, increases the complexity of the codec and requires extra memory resources both at the encoder and the decoder in order to store the references. Furthermore, a delay is introduced in the decoder, since the references used for backward prediction need to be transmitted and decoded before the intermediate frame in display order can be decoded.

Most codecs also allow *sub-pixel motion compensation*, up to the quarter-pixel. It is intuitive that the actual motion in the scene has an arbitrary accuracy, which does not necessarily map well in the grid structure defined by the spatial sampling of discrete images. In some cases, a better prediction may be obtained by matching the current block with an up-sampled (*i.e.*, interpolated) version of the reference frames. In a typical encoder implementation, the block matching algorithm first finds the best "full pixel" (*i.e.*, no up-sampling) motion vector; then, the half-pixel immediately next to these are searched for improvement (factor-2 up-sampling); finally, if required, the quarter-pixel positions next to the best half-pixel are searched (factor-4 up-sampling).

Even though using temporal prediction is very efficient, it does present some drawbacks, as it imposes a dependency relationship among the frames, in the sense that is not possible in general to decode a frame without having decoded all the frames it has been predicted upon. Also, missing a frame or receiving a corrupt version of it would irrecoverably affect the whole decoding process. Furthermore, if a change of scene occurs, the model of displacing block is no longer accurate and highly inefficient. Finally, in some applications, the decoder should be able to seek in the sequence, *i.e.*, to start the decoding at a random

point of the sequence. In order to cope with these situations, the solution is the periodical insertion of frames encoded using intra-frame prediction only (see next section).

1.2.2 Spatial prediction

Spatial prediction aims at eliminating the *spatial redundancy*, *i.e.*, similarities between neighbouring blocks. A prediction for the current block is created from previously coded and reconstructed blocks in the same frame, interpolated along different orientations. For instance, assuming that the blocks are coded in raster-scan order, the blocks in the asymmetric casual half-plane are available for intra prediction [CM04]. The prediction is then subtracted from the current block to generate the residual block, in a similar way to the temporal prediction.

Introduced in MPEG-4 part 10/H.264 AVC as *Intra-Frame Prediction* [Ric03, Ch. 6], spatial prediction has permitted the H.264/AVC intra-frame coding to obtain a higher compression gain than the previous image coding standards, such as JPEG2000 [ARK+04]. In Fig. 1.2, we present the nine prediction modes available in the H.264/AVC standard for 4×4 blocks [Ric03, WSBL03]. Similar modes exist for 16×16 blocks and, if the *Fidelity Range Extensions* (FRExt) [STL04] are used, for 8×8 blocks as well.

1.2.3 Spatial transform

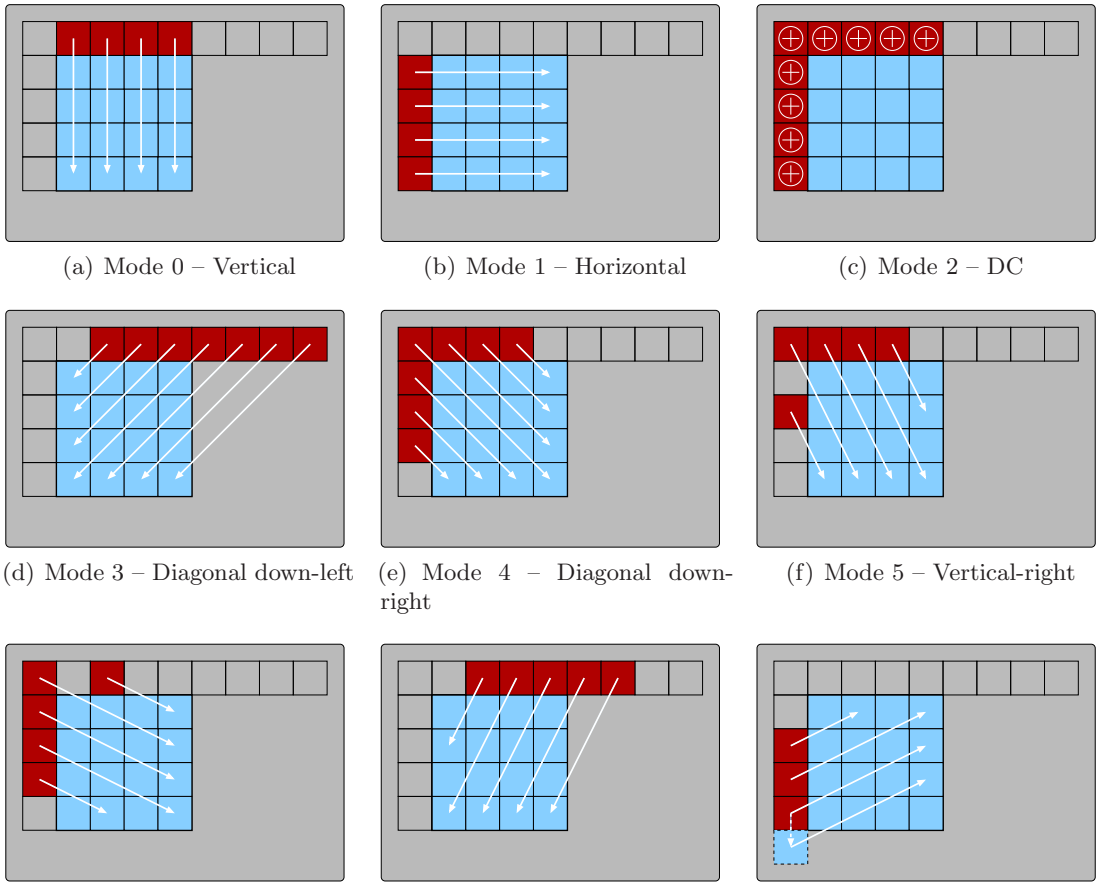
Residual blocks are processed via two-dimensional transform coding, *i.e.*, transformed with a reversible linear application that converts the pixels into another domain, wherein they are represented by transform coefficients. With a good choice of the transform, the transform coefficients should be decorrelated and compact, *i.e.*, they should concentrate most of the energy in a small number of coefficients. Linear transforms commonly employed in video coding include the the Discrete Wavelet Transform (DWT) and the Discrete Cosine Transform (DCT).

The DCT, like any Fourier-related transform, expresses a signal in terms of a sum of sinusoids with different frequencies and amplitudes. The obvious distinctive trait of the DCT with respect to the others Fourier-related transforms is that the former uses only real-valued cosine functions over a compact domain.

In its most commonly used form, the DCT of a mono-dimensional signal x of N elements can be expressed as follows:

$$y(k) = \sqrt{\frac{2}{N}} \sum_{n=0}^{N} x(n) \cos\left(\frac{k\pi}{N}\left(n + \frac{1}{2}\right)\right).$$

The multi-dimensional DCT of a signal can be simply obtained as a separable product of DCTs along each dimension. For example, a two-dimensional DCT of an image X of size $N \times M$ is simply the one-dimensional DCT performed along the rows and then along



(a) Mode 0 – Vertical (b) Mode 1 – Horizontal (c) Mode 2 – DC

(d) Mode 3 – Diagonal down-left (e) Mode 4 – Diagonal down-right (f) Mode 5 – Vertical-right

(g) Mode 6 – Horizontal-down (h) Mode 7 – Vertical-left (i) Mode 8 – Horizontal-up

Figure 1.2: Spatial prediction modes for 4×4 blocks in MPEG-4 part 10/H.264 AVC.

the columns (or *vice versa*), given by the formula:

$$Y(k,l) = \frac{2}{\sqrt{NM}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X(n,m) \cos\left[\frac{k\pi}{N}\left(n+\frac{1}{2}\right)\right] \cos\left[\frac{l\pi}{M}\left(m+\frac{1}{2}\right)\right],$$

where the coefficient $Y(0,0)$ is the DC (*i.e.*, zero-frequency) component and coefficients with increasing vertical (k) and horizontal (l) indices represent higher vertical and horizontal spatial frequencies.

For natural images, this approach has shown strong energy compaction properties, in the sense that most of the energy of the image tends to be concentrated in a few low-frequency coefficients [LG00]. For this reason, the DCT and its approximations play a particularly relevant role in modern standards such as MPEG-4 part 2 and MPEG-4 part 10/H.264 AVC (see Sec. 1.3).

In order to take into account the non-stationarity of natural images, thus providing a higher energy compaction, in these standards the DCT is applied block-wise, *i.e.*, it operates on blocks of $N \times N$ samples, usually with N power of two (4, 8, 16).

1.2.4 Quantisation

To quantise a signal means to map it into another signal with a discrete dynamic, by compressing a range of values to a single value. Fewer bits are needed to represent the quantised signal, as the number of possible values is smaller. However, quantisation is an inherently lossy process, and the quantised signal is bound to present a distortion with respect to the original signal. Since quantisation is the the only source of information loss in the encoding process, the design of an efficient quantiser is crucial to the performance of the codec.

The most simple quantisation technique is *uniform scalar quantisation*, wherein each coefficient is quantised independently. The quantiser divides the range of the original signal into intervals of a chosen step size, and associates at each interval a *reconstruction value* (*e.g.*, its centre). Then, the quantisation is carried on by simply representing each input value with the index of the interval in which it lies. De-quantisation is performed associating at each interval index its reconstruction level.

A more efficient technique is *non-uniform scalar quantisation*, in which the quantisation steps are not the same for all levels. The problem of finding the optimal steps sizes for a non-uniform scalar quantiser has been solved by Lloyd [Llo82] through the well-known Lloyd-Max Algorithm.

It can be shown that, under high-bitrate conditions, the rate-distortion performance of a Lloyd-Max quantiser encoding a zero-mean stationary signal takes the form:

$$D(R) = h\sigma^2 2^{-2R},$$

where σ^2 is the variance of the signal and h is a shape-factor depending on its probability distribution [CT91, Ch. 13].

In image and video coding, quantisation is used in conjunction with spatial transform, as the latter provides a lossless representation of the original signal that is more suitable for quantisation, with many coefficients that can be eliminated as they barely contain any energy.

The advantage of applying an energy-compacting transform before quantisation is usually measured with the *coding gain*, defined as the ratio of the arithmetic and geometric means of the transform coefficients' variances:

$$G = \frac{\frac{1}{N^2}\sum_{l=0}^{N-1}\sum_{k=0}^{N-1}\sigma_{k,l}^2}{\left(\prod_{l=0}^{N-1}\prod_{k=0}^{N-1}\sigma_{k,l}^2\right)^{\frac{1}{N^2}}}.$$

It can be proved that G measures how much gain is achieved from transform coding with respect to quantising directly in the original signal domain at the same rate [Bov00, Ch. 6].

Furthermore, in the case of Fourier-related transforms, where the most energetic coefficients correspond to the low-frequencies, there is little perceptible loss of quality when quantising low-energy coefficients on fewer bits, since the human visual system is less sensible to high frequencies than to low frequencies.

1.2.5 Lossless coding

Even using spatial and temporal prediction, the residual data still present some degree of statistical redundancy. In fact, since the symbols do not have in general a uniform probability, it is possible to assign a shorter representation to more commonly occurring symbols and a longer representation to less commonly occurring ones, thus reducing the total length of the sequence. This technique is known as *lossless encoding*, as it allows an exact reconstruction of the original signal without loss of information. Lossless coding is also sometimes referred to as *entropy coding*, as it aims at producing an output sequence whose length is equal to the *entropy* of the input sequence, *i.e.*, the amount of information contained in the sequence. Two of the most common lossless coding techniques are *Huffman Coding* and *Arithmetic Coding*.

Huffman Coding [Huf52] produces a prefix-free code (*i.e.*, such that the codeword representing a symbol cannot be the prefix of the codeword representing any other) by constructing a binary tree of symbols according to their relative frequency. Huffman Coding is optimal in the sense that, for the same alphabet, any other code cannot have a lower expected length, and its expected length is within one bit of the entropy. However, if the source alphabet is small (*e.g.*, binary), the encoding is efficient only if long blocks of source symbols are used. Unfortunately, Huffman coding is not computationally ideal for long blocks, since the calculation of a code corresponding to blocks of a longer length cannot be extended from a code corresponding to shorter blocks [CT91, Ch. 5].

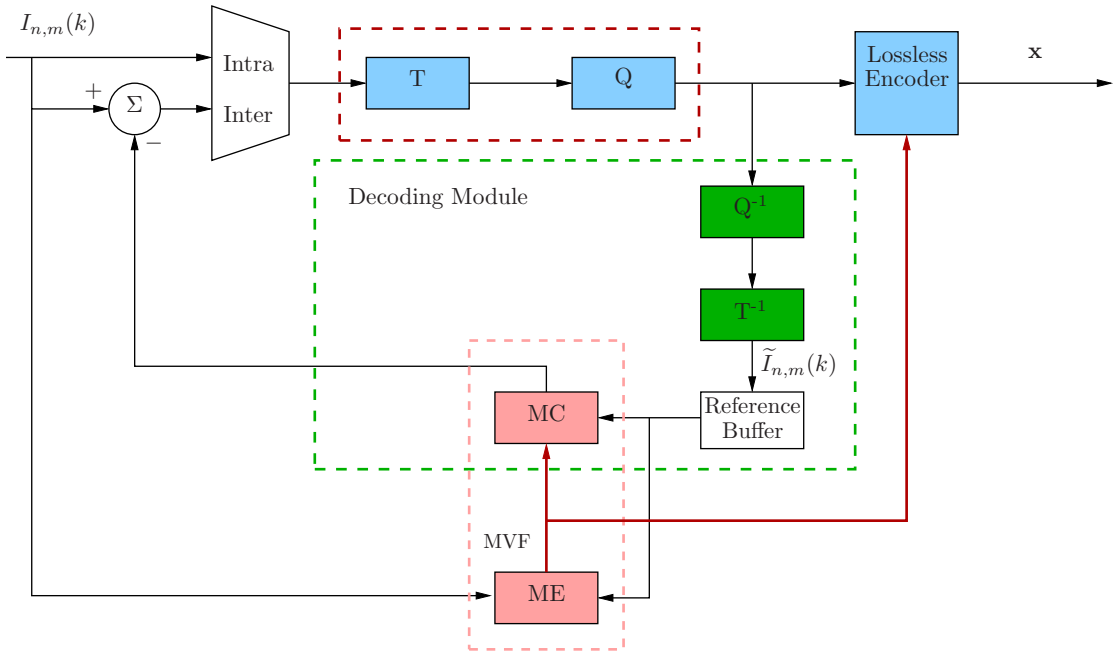
Arithmetic Coding [LR82] differs considerably from Huffman Coding in that, rather than separating the input bitstream into symbols and assigning a codeword to each symbol, it encodes the entire message into a single number, namely a proper fraction, *i.e.*, a rational number between 0 and 1. The coding algorithm, for each input symbol, successively partitions an interval of the number line between 0 and 1 according with the cumulative distribution function of the input alphabet, and retains as new interval the one corresponding to the current symbol. Thus, the algorithm successively deals with smaller intervals. The codeword for the message will thus be any number that lies in the selected interval at the last step. In order to reconstruct the message, the decoder has to recreate how the encoder has successively partitioned and retained each sub-interval, then by comparison establishing in which interval lies the codeword.

Arithmetic coding is actually a little less performing, in terms of compression ratio, with respect to the optimal code, *i.e.*, the Huffman coding applied on blocks as long as the message itself. However, the latter case is in practice computationally unfeasible, whereas

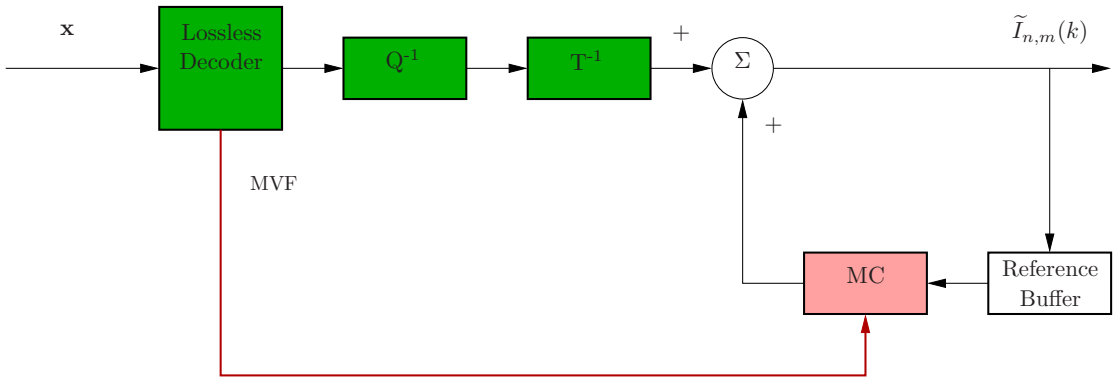
the former admits a particularly simple and fast implementation [LR82].

1.2.6 Hybrid video coding

In Fig. 1.3, we show the scheme of a *hybrid* video codec in order to present how the basic components we presented so far are interconnected in order to design a video coding system. In particular, the basic components of the encoder are shown in Fig. 1.3(a).



(a) Video encoder



(b) Video decoder

Figure 1.3: Structure of a video codec.

The term *hybrid* refers to a video source coding algorithm that uses both temporal prediction, to reduce temporal redundancy, and transform coding, to reduce spatial redundancy.

Video coders usually process the input sequence per block. The size and shape of a block depends on the codec: in early codecs, blocks were always square and their size was

fixed, whereas in more modern codecs the coding unit is usually a macroblock, whose size is fixed, but that can partitioned into smaller blocks. For instance, the upcoming High Efficiency Video Coding (HEVC) standard [SO10] will support macroblocks of size up to 64×64 pixels for motion compensation, and up to 32×32 pixel for spatial transform, and it will allow a very flexible partition of the macroblocks [UAF+10]. As an example, in Fig. 1.8 we show the macroblock partitions allowed in the MPEG-4 part 10/H.264 AVC standard (see Sec. 1.3.2).

For each new macroblock, the encoder operates a *mode selection*, *i.e.*, it determines a set of options that will be used in the encoding. These options include:

- Skip the macroblock encoding altogether, *i.e.*, no information will be sent for this macroblock. The decoder will be able to reconstruct this macroblock satisfyingly just using prediction.
- Encode the macroblock using spatial prediction only, or no prediction (*Intra-mode*, or *I-mode*).
- Encode the macroblock using temporal prediction, either unidirectionally (*P-mode*), or bidirectionally (*B-mode*).

It will also decide whether and how to partition the macroblock in smaller blocks, which prediction mode to use for spatial prediction, which quantisation step to apply, and so forth. The choice of these options should in principle be carried out by minimising the rate-distortion cost function (1.1) mentioned in Sec. 1.1; this is, however, a very demanding task, because of the large number of encoding parameters. For this reason, most codecs implement a sub-optimal but low-complexity mode decision, based on approximating the cost function, reducing the set of tested models, or a combination thereof [Ric10, Ch. 9]. It is important to notice that how the mode selection is implemented is not part of the standard, therefore different implementations can have different performance and different complexities.

In order to meet the users' requirements, in video codecs, temporal prediction can also be limited to a pre-established pattern within a group of successive frames in the coded bitstream, referred to as *Group-of-Pictures*, or GOP. The structure of the GOP is specified at the encoder and inhibits certain prediction modes for the frames. For instance, a user can demand that only Intra-mode is used in a frame, so that, without prediction, possible errors are not propagated, or only unidirectional forward prediction, so that the decoder would not experience a decoding delay due to the backward prediction. The encoder will re-order the frames so that each encoded frame is always transmitted after the frames it is predicted upon.

An example of GOP structure for the MPEG-4 part 10/H.264 AVC standard (see 1.3.2) is shown in Fig. 1.4. Here, frame 0 is Intra-coded, *i.e.*, not predicted upon others; frame 4 is predicted upon frame 0 alone; frame 2 is predicted bidirectionally upon frames 0 and

4, while frame 6 is predicted bidirectionally upon frame 4 and the Intra-frame of the next GOP; all other frames are predicted bidirectionally upon the two neighbouring frames. Notice than the mode selection can always choose to encode a macroblock in I- or P-modes in B-frames, and in I-mode in P-frames, if it deems it more efficient.

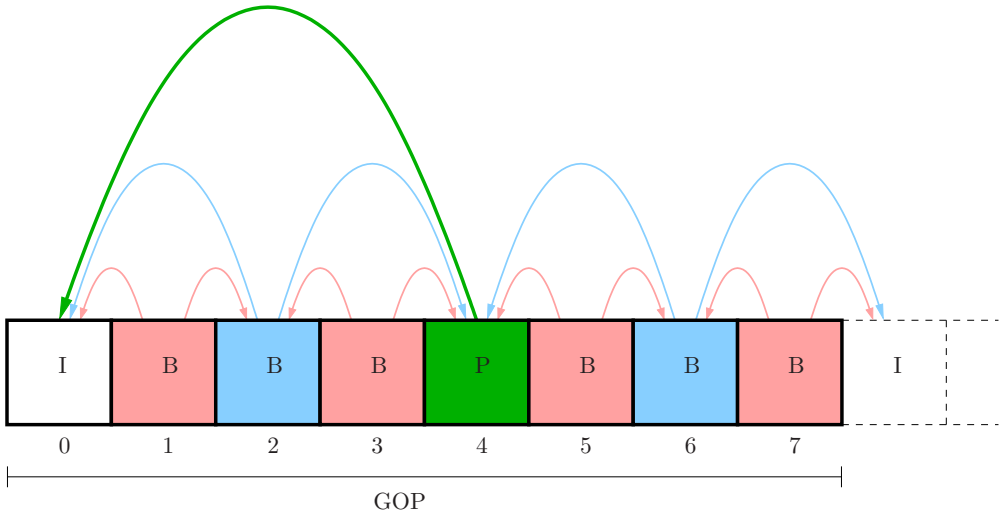


Figure 1.4: Example of structure of a Group-of-Pictures. Arrows show the frames used as references in the temporal prediction.

Since at the beginning of a sequence no reference is available, the macroblocks of the first frame are always encoded in I-mode, *i.e.*, they are transformed and quantised according to the options chosen by the mode selection, then passed to the lossless encoder.

The encoder also includes a local decoder (see Fig. 1.3(a)) that performs exactly the same operations performed at the decoder, reconstructs the whole frame, and stores it in a reference buffer. This is done since, as mentioned above, temporal prediction requires that the frames used as reference are available both at the encoder and the decoder. A mismatched prediction would compromise the decoding process, as the encoder and the decoder would no longer be coupled, a problem known as *drift effect*. Since the decoder does not have access to the frames of the original sequence $I_{n,m}(k)$, the encoder can use as reference only the reconstructed frames $\widetilde{I}_{n,m}(k)$.

The macroblocks of the following frames will be able to use the frames in the buffer as reference for temporal prediction. While the motion vectors will be directly sent to the lossless encoder, the motion-compensated prediction will be subtracted from the current macroblock, and the residual block will be transformed and quantised prior to be sent to the entropy coder. This process is reiterated for each frame in the GOP, and for each GOP in the sequence, until all frames have been encoded.

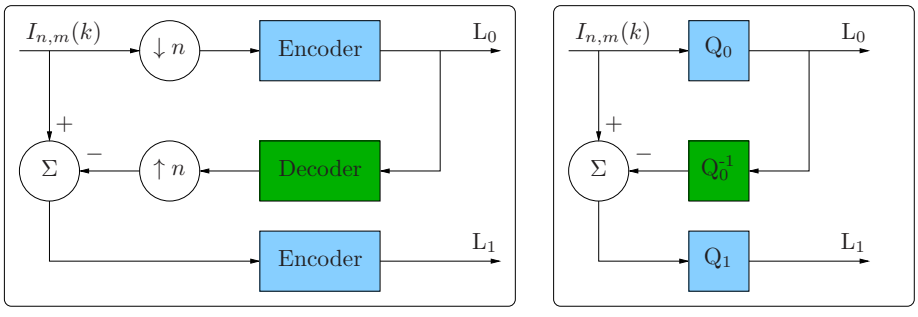
1.2.7 Scalable video coding

Scalable video coding is a coding scheme intended to allow a single encoding of a video sequence, but enabling decoding from partial streams depending on the specific rate and resolution required by a certain application [VdSC07, Ch. 5].

In scalable video coding, the video data are divided into a *base layer* and one or more *enhancement layers*, and is therefore also referred to as *layered video coding*. The layers are designed in such a way that progressive reconstruction at increasingly higher quality is possible. The enhancement layers are useless unless the base layer and all the enhancement layers of lower detail are received. The video client can negotiate with the video server the number of layers it is interested in, according to its quality demands and resource availability. The most important scalable features are spatial resolution, temporal resolution, and SNR.

Temporal scalability is a technique to code a video sequence in a set of layers providing an increasing temporal resolution, with the perceivable effect of a progressive increase of the frame rate. It can be easily achieved skipping some of the frames of the bitstream, with the *caveat* that the dependencies among layers should reflect the dependencies among frames dictated by the temporal prediction. For instance, with reference to the GOP structure depicted in Fig. 1.4, a base layer can be constructed by the even frames and one enhancement layer by the odd ones. This type of GOP structure, known as *hierarchical*, is particularly interesting in this respect, as it allows an easy construction of many layers, based on prediction levels.

In spatial scalability, the layers provide an increasing spatial resolution. A scheme to generate a two-layer spatially scalable video is presented in Fig. 1.5(a). The base layer L_0 is obtained by subsampling the original sequence $I_{n,m}(k)$ of a factor n, then encoding it with a classical (*i.e.*, non-scalable) coder. An enhancement layer L_1 is obtained up-sampling (*i.e.*, interpolating) the reconstructed lower layer and using it as a prediction of $I_{n,m}(k)$. The residual sequence is then in its turn encoded with the classical coder. More layers can be obtained using a higher subsampling factor for the base layer, and progressively up-sampling, up to the original resolution.



(a) Spatial scalability. (b) SNR scalability

Figure 1.5: Scalable coding schemes for two layers.

In SNR scalability, also referred to as bitrate scalability or quality scalability, the layers provide the same spatial and temporal resolution, but an increasing quantisation accuracy. One possible technique to achieve SNR scalability is presented in Fig. 1.5(b). The transform coefficient of the original sequence are encoded using a coarse quantiser Q_0 to obtain the base layer. The quantised coefficients are then de-quantised and used as prediction of $I_{n,m}(k)$. The residue is encoded with a finer quantiser Q_1 in order to obtain the enhancement layer L_1. More layers can be obtaining using a progressively finer quantisation. An alternative technique is to directly use an embedded coding of the transform coefficients, *e.g.*, using bit-plane coding.

Using layered coding, the predictions made by the hybrid coder at one layer should not use any information from higher layers, since this would cause a mismatch or the references used at the encoder and at decoder sides, causing a drift effect. However, a prediction based on the base layer only will always be worse than it could have been if all enhancement layers were allowed in the prediction. This means that scalability combined with hybrid video coding comes at the price of a less performing encoder in terms of rate-distortion. In other words, at each rate, a scalable stream provides in general a lower video quality than the corresponding non-scalable video technique.

Since this inefficiency is mainly due to the prediction loop, which causes a drift problem whenever incomplete information is decoded, some have proposed an alternative approach to the codec structure that is not based on prediction. This approach is based on using temporal filtering instead of temporal prediction, and a multi-resolution transform, such as the Wavelet Transform, applied to both temporal and spatial dimension [AMB+04, TvdSPP05, FPP07].

1.3 Video coding standards

In this section, we shall give a brief review of the most relevant standards in video coding. Two families of standards have pioneered video coding during the 1980s and 90s: the ITU/VCEG H.26x family and the ISO/IEC MPEG family. The two teams have also cooperated to a joint partnership effort known as the Joint Video Team (JVT) for the development of new video coding recommendations and international standards since 2001.

1.3.1 Early coding standards

The Video Coding Experts Group (VCEG) is a study group of the International Telecommunication Union (ITU) responsible of producing *recommendations*, *i.e.*, international standards, on the subject of picture and video coding techniques for conversational (*e.g.*, video conferencing) and non-conversational (*e.g.*, video streaming) application services.

As such, in 1990, the ITU/VCEG emanated the first practical digital video compression

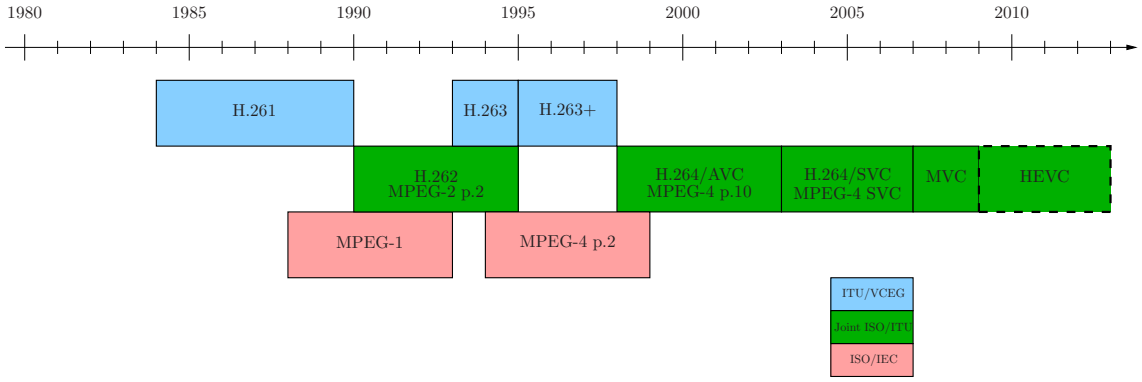


Figure 1.6: Time-line of video coding standards.

standard: H.261 [H2690]. The coding algorithm was designed for transmission over ISDN lines, on which data rates are multiples of 64 kbps, and was able to operate at bitrates between 64 kbps and 2 Mbps [Oku95]. The design of H.261 was a pioneering effort in the introduction of the hybrid video coding model, and all subsequent standards have been heavily influenced by its design. It has been the first standard wherein the concept of macroblock appeared, and to provide temporal prediction to reduce temporal redundancy, even though only integer-pixel motion compensation was supported at the time.

Concurrently, the Moving Picture Experts Group (MPEG), a working group of experts established by the International Organisation for Standardisation (ISO) and the International Electrotechnical Commission (IEC) in order to produce standards for audiovisual compression and transmission, had begun the development of the first complete compression standard for audio and video, released in 1993 as MPEG-1 [MPE93]. It was the first hybrid codec standard to introduce sub-pixel motion compensation for temporal prediction (namely, half-pixel), and the concept of GOP with a fixed coding structure. The MPEG-1 standard implementations are typically limited to bitrates up to $\sim$ 1.5 Mbps (although the standard allows much higher bitrates), and allowed moving pictures and sound to be encoded into a Compact Disc. It was mostly used on Video CD, S-VCD and can still be used for low-quality video on DVD. It was also used in satellite/cable television before MPEG-2 became widespread.

The ITU/VCEG and the ISO/IEC MPEG have on occasion attempted a partnership effort, eventually merging in the standardisation body known as the Joint Video Team (JVT). The standards emanated by the JVT are published as a standard of both organisations, are jointly maintained, and have identical technical content.

A positive factor of the MPEG-1 design had been the generic structure of the standard, which supports a broad range of applications and applications-specific parameters. However, in order to provide a video-coding solution for applications not originally covered or envisaged by the MPEG-1 standard, which operated only at low-quality, the MPEG continued its standardisation efforts with its second phase, MPEG-2 [MPE95], released as

an international standard (ISO/IEC 13818) in 1995. The second part of MPEG-2, *i.e.*, the video compression and encoding technique, was considerably broader in scope and of wider appeal than both MPEG-1 and H.261, supporting interlacing and high definition video [Chi95, Sik97b]. It has been chosen as the compression scheme for digital over-the-air, satellite and cable television signals, and storage media such as S-VCD and DVD. The ITU-T has eventually published a recommendation identical to MPEG-2 part 2, under the name of H.262 [H2695], and the standard is therefore also known as MPEG-2 part 2/H.262.

A notable introduction of MPEG-2 part 2/H.262 was the concept of *profiles* and *levels*. A video profile defines a subset of features that a decoder has to implement to be compliant, whereas the level defines quantitative requirement ranges, such as bitrate, frame size, *etc.* Using profiles and levels it is possible to adapt the standard to specific applications, ranging from mobile streaming to high quality video editing, without having to support the entire standard. MPEG-2 part 2/H.262 was also the first standard to include implementations of layered coding in order to support temporal, spatial, and quality scalability and the first to support multi-view video coding [Ohm05].

A standardisation effort was subsequently launched to handle scalable and multi-resolution compression for HDTV signals in the range of 20–40 Mbps. However, it was soon discovered that similar results could have been obtained through slight modifications to the MPEG-2 part 2/H.262 standard. Thus, the new standard was included as a separate profile in the MPEG-2 part 2/H.262 standard, and shortly thereafter discontinued. As a result, there exist no standard known as MPEG-3.

Later on, the VCEG developed an evolutionary improvement of H.261 based on the experience from the MPEG-1 and MPEG-2 part 2/H.262 standards, released in March 1996 as H.263 [H2696], which provided a suitable replacement for H.261 at all bitrates [Rij96]. The capabilities of H.263 have later been enhanced by several annexes that substantially improved the encoding efficiency and provided other capabilities, such as enhanced robustness against data errors and loss in the transmission channel. These annexes have been released under the name of H.263+.

In 1994, the MPEG group took up the standardisation activity for a new format, resulting in the standard known as MPEG-4, divided into a number of parts [Sch98]. In particular, MPEG-4 part 2 (Visual) [MPE99], released in 1999, defines a video compression standard that employs coding tools with high complexity in order to achieve higher compression factors than MPEG-2 part 2/H.262. As well as an increased compression efficiency, MPEG-4 also offers interactivity and integration of objects of different natures, *i.e.*, the possibility of accessing and manipulating individual objects within the picture. Moreover, it offers possibilities for efficient video storage and for transmission over poor channels, at bitrates between 5 kbps and 4 Mbps, taking into account a wide variety of networks.

One of the most innovative features MPEG-4 part 2 was to treat a video sequence as a collection of *video objects*, as opposed to the traditional view of a video sequence as being

merely a collection of rectangular frames. A video object may occupy an arbitrarily-shaped region, it may exist for an arbitrary length of time, and it may be real or synthetically generated [Ric03, Ch. 5]. This approach enables, for example, to independently encode the background of the scene as a still image; the object within the scene can then be encoded using motion compensated temporal prediction (up to quarter-pixel precision) and transform coding of the residual, with extensions to deal with the shape of their boundaries.

As MPEG-2, MPEG-4 defines a set of profiles and levels for use in specific applications, and provides temporal, spatial, and quality scalability. In particular, MPEG-4 defines *Fine Granularity Scalability* (FGS), which differs from the traditional layered scalable coding techniques in that the enhancement layer can be truncated at any number of bits within each frame to provide a partial enhancement, proportional to the number of bits decoded for each frame [Li01].

MPEG-4 part 2 is also H.263-compatible, in the sense that a basic H.263 bitstream can be correctly decoded by an MPEG-4 decoder [Sik97a, DSK99].

1.3.2 MPEG-4 Part 10/H.264 Advanced Video Coding

The MPEG-4 part 10/H.264 AVC standard [MPE03, H2603], henceforth H.264/AVC, has been released in 2003, and is currently the most powerful video compression standard[1] [WSBL03, OBL+04]. Note that the MPEG-4 part 10 standard defines a different and incompatible format than MPEG-4 part 2 and should not be confused with it.

In this section, we shall briefly describe the most innovative features introduced in H.264/AVC. A global scheme of the encoder is presented in Fig. 1.7. In H.264/AVC, each frame of the input sequence is subdivided into one or more *slices*. The slice is the basic spatial segment that is independent of its neighbours. Thus, errors or missing data from one slice cannot propagate to any other slice within the frame. This also increases flexibility to extend frame types (I, P, B) down to the level of slice types. A slice is composed of a group of macroblocks of size 16×16 that are consecutive in raster scan[2] and are encoded independently from macroblocks in other slices. For each macroblock, depending on the slice type, the encoder may choose among different encoding modes, summarised in Tab. 1.1. In an RD-optimised H.264/AVC encoder implementation, all modes are tried before the one with the best RD performance is selected. There also exist sophisticated low-complexity implementations that deliver performances very close to an RD-optimised coder with a significant reduction in computational complexity [Ric10,

[1] A draft of the successor to H.264/AVC, known as High Efficiency Video Coding (HEVC), is under development by JVT. The final draft, ready to be ratified as an international standard, is expected by January 2013.

[2] Actually, H.264/AVC also provides a tool, known as *Flexible Macroblock Ordering* or FMO, that allows to freely assign each macroblock to a specific slice. This feature is sometimes used to provide a more robust encoding in environments with a high packet loss rate, or to identify a region of interest in the frame that has to be quantised more finely.

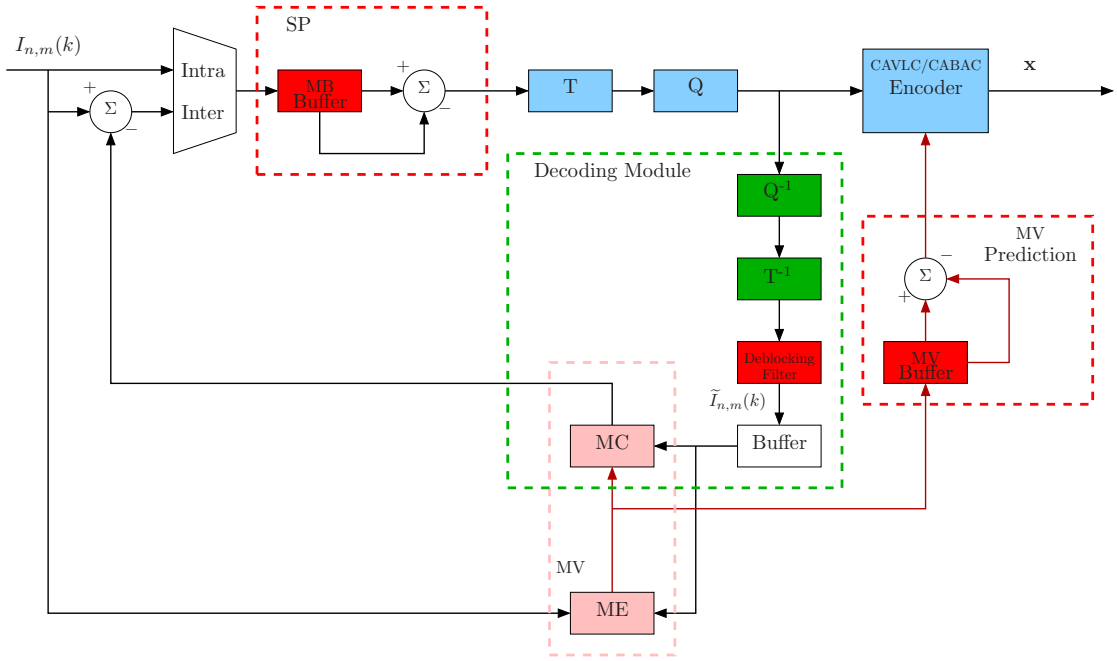


Figure 1.7: Structure of a MPEG-4 part 10/H.264 AVC encoder.

Ch. 9]. The choice of the motion vector is done at the encoder, and is therefore not part

Slice Type	Allowed Encoding Modes
I-Slice	Intra prediction only.
P-Slice	Skip; Intra prediction; Prediction from past slices.
B-Slice	Direct; Intra prediction; Prediction from past slices; Prediction from future slices; Prediction from average of past and future slices.

Table 1.1: Allowed encoding modes by slices type.

of the standard, but it is in practice always performed via BMA with an RD minimisation criterion [ZM00, ZLC02]. The H.264/AVC standard also allows a flexible partitioning of the macroblock to allow a better reconstruction of the movement (shown in Fig. 1.8). Predictive encoding is applied to the resulting motion vectors, *i.e.*, for each vector, the encoded value will be the difference between the vector itself and a prediction based on the median value of its available neighbours.

The residual block resulting from the motion compensation is transformed using a hierarchical transform in order to reduce its spatial correlation. In a first step, the residual image is transformed with an integer 4×4 transform (in Tab. 1.2) having properties similar to the DCT. The advantage of using an integer transform is that an exact reconstruction is possible independently from the precision of the implementation (contrarily to previous

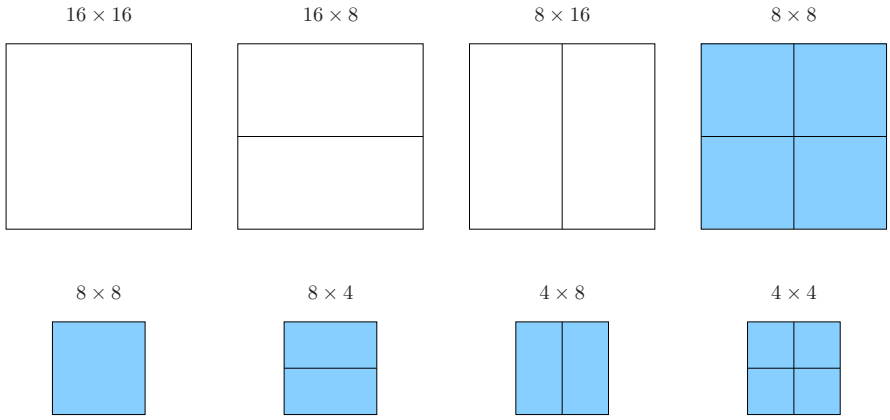


Figure 1.8: Macroblock partitions allowed in H.264/AVC. A 16×16 block may be partitioned in two 16×8 or 8×16 blocks, or four 8×8 blocks. Each 8×8 block may in its turn be partitioned in two 8×4 or 4×8 blocks, or four 4×4 blocks.

standards, such as MPEG-2 part 2 and H.263). If the macroblock is predicted using Intra-mode, the DC (*i.e.*, the frequency 0 coefficients) of the already transformed blocks undergo a second transformation via Hadamard transform [Ric10, Ch. 7]. All transformed coefficients are then quantised using uniform scalar quantisation. The transmission order

$+1$	$+1$	$+1$	$+1$
$+2$	$+1$	-1	-2
$+1$	-1	-1	$+1$
$+1$	-2	$+2$	-1

Table 1.2: The 4×4 integer transforms used by H.264/AVC in residual coding. It can be computed using bit-shifts and sums only.

of coefficients is depicted in Fig. 1.9. If the current macroblock has been predicted in Intra-mode, a block with label -1 is transmitted first. It contains the DC coefficients of all blocks. Then, the blocks containing the AC coefficients (*i.e.*, the non-zero frequency coefficients), labelled 0–25, are transmitted [OBL+04]. Each block of transform coefficient is scanned, *i.e.*, converted to a linear array, before lossless coding. The scan order, depicted in Fig. 1.10, is intended to group together significant quantised coefficients [Ric10, Ch. 3].

At the decoder side, and within the decoding block of the encoder, the frames are reconstructed via motion compensation, inverse quantisation and inverse transform. In H.264/AVC, an in-loop *deblocking filter* is applied to the image thus reconstructed. The deblocking filter is an adaptive low-pass filter applied to the borders of macroblocks in order to remove block-artifacts, due to the block-wise transform, from the reconstructed image. It works as a non-linear adaptive filter, *i.e.*, it is possible to alter the filter strength or to disable it. Since the filtered images will be used as reference for the following motion estimations/compensation, the deblocking filter also improves the quality of the temporal prediction.

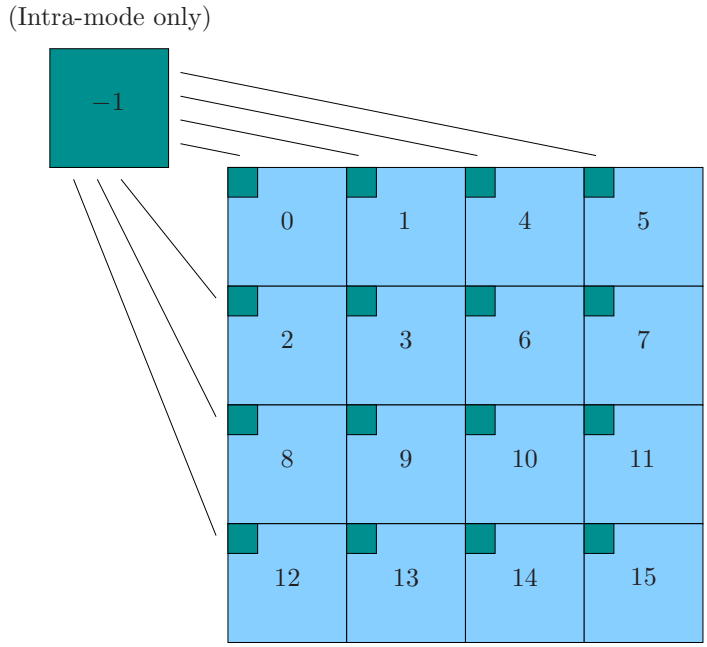


Figure 1.9: Transmission order of transform coefficients of a macroblock (luminance component). If the macroblock is predicted in Intra-mode, a block labelled −1 containing the DC coefficients of all blocks is sent first.

Finally, the H.264/AVC bitstream can be encoded using either binary arithmetic coding or variable-length (Exponential Golomb) coding. The innovation of H.264/AVC is that the lossless encoding is *context adaptive* (CA), *i.e.*, there exist multiple probability modes for different contexts. Context models for each syntax element are defined in the standard. There exist nearly 400 separate context models for the various syntax elements. For each syntax element, the encoder selects which probability model to use, then uses information from nearby elements to optimise the probability estimate. *Context-adaptive binary arithmetic coding* or CABAC is notable for achieving a much better compression performance than most other lossless encoding algorithms used in video encoding, and is one of the primary advantages of H.264/AVC. However, it requires a large amount of processing to decode, and is difficult to parallelise and vectorise. For this reason, Context-adaptive variable-length coding, or CAVLC – which has a lower efficiency, but requires considerably less processing to decode than CABAC – is used on slower playback devices.

In 2003, the MPEG issued a call for proposals on an extension to the H.264/AVC standard to enable the scalable encoding of video sequences, later agreed by the ITU-T. This proposal led to the standardisation of an amendment known as H.264 Scalable Video Coding (SVC), which received its final approval in 2007. The H.264/SVC extension provides three additional profiles: Scalable Baseline, Scalable High, and Scalable High Intra. These profiles are defined as a combination of the corresponding H.264/AVC profiles for the base layer and tools that achieve temporal scalability, spatial scalability, quality scalability and combination thereof [SMW07, WCG+07].

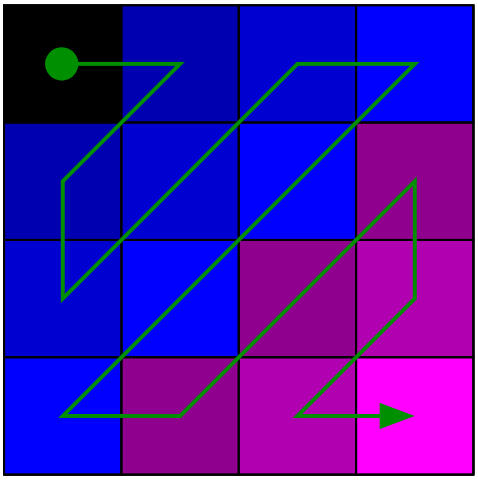


Figure 1.10: Progressive scan order for transform coefficients within a 4×4 block.

Chapter 2

Multiple description video coding

Contents

In this chapter, we shall discuss the *Multiple Description Coding* (MDC) paradigm, a joint source-channel coding technique providing a trade-off among coding efficiency (in terms of compression ratio for a given quality), robustness, and complexity.

First, in Sec. 2.1, we shall present the basic principle of MDC, and in particular how it can be applied to the video coding techniques described in Chapter 1 in order to adapt a video stream to the transmission over unreliable networks, such as the wireless ad-hoc networks. A detailed case study for a wireless video streaming application using MDC will be provided in Chapter 3.

Then, in Sec. 2.2, we shall present a multiple description video coding technique we recently proposed. This technique will make up the basic building block of the video streaming system for wireless ad-hoc network discussed in Chapters 3, 4, and 5. The proposed technique is based on the concept of *motion compensated temporal interpolation*

(MCTI), that we introduce in Sec. 2.2.1. Finally, an experimental study of the technique is presented in Sec. 2.2.3.

2.1 Multiple description coding

Multiple Description Coding (MDC) is a coding framework allowing an improved immunity towards losses in unreliable networks, may losses be due to congestion or other perturbations, in case no feedback channel is available or retransmission delay is not tolerable [Goy01]. Originally proposed at the beginning of the Nineteen Seventies for the speech signal, MDC has since been applied to other fields, such as image coding and video coding. Goyal [GK98], among others, ascribe MDC to the class of joint source-channel coding techniques for erasure channels, since it takes simultaneously into account the possibility of losses and the encoding process, even though in general no explicit loss model is used in the design.

Whereas scalable video (see Sec. 1.2.7) is the state-of-the-art solution when preferential treatment can be performed on packets, *i.e.*, when more important packets can be provided with a better protection against failures, MDC handles the case of multiple independent transmission channels with roughly the same reliability. The main difference between layered video coding and MDC is that, in layered coding, the video signal is encoded in a progressive way and arranged in a hierarchical structure of cumulative layers, so that a layer is only decodable if *all* the lower layers have been received. On the other hand, in MDC, the source generates several versions of the same signal, called *descriptions*. Each description is independently decodable, but with a higher number of descriptions the quality of the reconstruction is improved.

2.1.1 Basic concepts of MDC

The main idea behind MDC is the independent transmission over independent channels of several representations of the same source signal. As we shall see, this imposes a trade-off between robustness and coding efficiency, in terms of compression ratio for a given quality. For the sake of clarity, we shall henceforth assume, without loss of generality, that two descriptions are generated.

A simple two-descriptions transmission system is represented in Fig. 2.1: a source signal (*e.g.*, a video sequence) I has to be transmitted to an end-user, having two independent lossy channels available. The *MD encoder* generates two compressed representations of the signal, $\mathbf{x}_0$ and $\mathbf{x}_1$, referred to as *descriptions* of I. Then, each description is sent over a different channel.

The two descriptions are independently decodable, *i.e.*, each one of them can be used to reconstruct a low-quality version of I. It is noteworthy that this is a crucial difference between MDC and SVC (introduced in Sec. 1.2.7): whereas in a scalable video coding

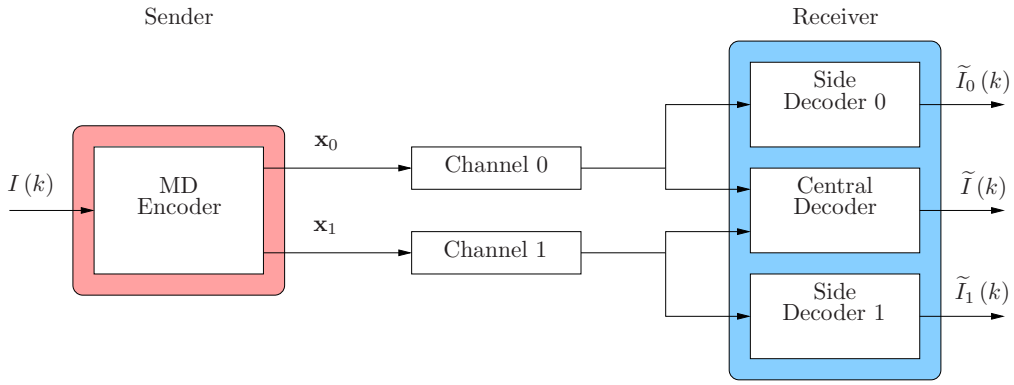


Figure 2.1: Scheme of a two-channels multiple description system.

technique there exist a strictly necessary base layer and a number of optional refinement layers, in MDC all the descriptions are independent of each other and can be used interchangeably. This is important when it is impractical or unfeasible to provide an unequal protection towards errors and losses to the base layer and the refinement layers; on the other hand, using MDC, the mere fact of sending the descriptions over different channels provides a degree of immunity to the stream.

At the receiver side, two different scenarios might occur.

1. Only one description $\mathbf{x}_d$, $d \in \{0, 1\}$, is received correctly, while the other is affected by a loss on the channel. In that case, the decoding process is performed by a *side decoder*, which produces an approximated version of I based only on $\mathbf{x}_d$, here denoted as $\widetilde{I}_d$.

2. Both descriptions are correctly received. The decoding process is performed by the *central decoder*, which produces a reconstructed version of I based on both $\mathbf{x}_0$ and $\mathbf{x}_1$, here denoted $\widetilde{I}$. As a general rule, $\widetilde{I}$ has a lower distortion than both $\widetilde{I}_0$ and $\widetilde{I}_1$.

Notice that this scheme can be easily generalised in the case of more than two descriptions.

A traditional, *i.e.*, not-MDC encoding technique (in literature also referred to as *Single Description Coding*, or SDC), is normally optimised for rate-distortion efficiency, and the redundancy of the representation reduced by the encoding process. Conversely, any MDC technique is inherently affected by a certain degree of redundancy due to the correlation among the descriptions.

The RD characterisation of an MD coded bitstream is a little more articulated than the one presented in Sec. 1.1 for SD streams, as it has to take into account the possibility of either side decoder or the central decoder being used at the receiver. At any bitstream can be associated a quintuple (R_0, R_1, D_0, D_1, D), where R_0 and R_1 are the rates of the two encoded descriptions $\mathbf{x}_0$ and $\mathbf{x}_1$, D_0 and D_1 are the distortions of $\widetilde{I}_0$ and $\widetilde{I}_1$ with respect to I, and D is the distortion of $\widetilde{I}$ with respect to I [Oza80].

An important theoretical result of El Gamal and Cover [EGC82] states that all achievable quintuples when encoding a memory-less Gaussian source with variance σ^2 in multiple

descriptions must satisfy the following relations:

$$\begin{cases} D_0 & \geq & \sigma^2 2^{-2R_0}, \\ D_1 & \geq & \sigma^2 2^{-2R_1}, \\ D & \geq & \gamma\sigma^2 2^{-2(R_0+R_1)}, \end{cases}$$

where γ is a function of the quintuple, and is one only if $D_0 + D_1 \geq \sigma^2 + D$. In other words, only when one or both side distortions are large, the central reconstruction can be very good; otherwise, there is a penalty in the central distortion [Goy01]. This means that, in raw RD-performance, a SD technique usually outperforms an MD-technique if no losses occur, in the sense that, given the total rate of the descriptions, the quality of the reconstruction of an MD codec is lower than the one achievable with a SD codec at the same rate [Oza80, EGC82, VKG03].

However, MDC becomes a viable tool whenever the stream has to be sent over a lossy channel: in this case, the introduction of a controlled redundancy in the MD-stream may be used to provide the end-user with an acceptable quality even if a large part of the stream is lost. A central point in the design of any MDC technique is therefore to tune the degree of correlation among the descriptions: on one hand, redundancy in the representation is what grants this technique its loss resiliency; on the other hand, redundancy implies inefficiency in terms of rate-distortion optimisation. In order to take into account the benefits of loss resiliency, if the loss probabilities of the channels over which the descriptions are to be sent are known, the RD Lagrangian cost function (1.1) discussed in Sec. 1.1 can be generalised as follows:

$$J = (1-p_0)(1-p_1)D + (1-p_0)D_0 + (1-p_1)D_1 + \lambda(R_1 + R_2), \tag{2.1}$$

where p_0 and p_1 are the loss probabilities of channel 0 and channel 1, respectively. If the two descriptions are *balanced*, *i.e.*, $R_0 \approx R_1$ and $D_0 \approx D_1$, and the two channels have the same loss probability p, the cost function becomes:

$$J = (1-p)^2 D + 2(1-p)D_0 + 2\lambda R_0.$$

In the following, we shall briefly review some of the strategies most commonly employed to build an MDC scheme. Since this thesis focuses on video streaming, particular attention will be given to *Multiple Description Video Coding* (MDVC) [WRL05].

Historically, the first MDVC techniques to be proposed inherited from the MDC technique already proven efficient for still images [RJW+99, LPFA00]. These techniques are commonly referred to as *intra-frame* or *spatial* MDVC techniques [WRL05]. Other techniques also exploit the high degree of temporal correlation that video signals present. Even though the latter approach, which we shall refer to as *inter-frame* or *temporal* MDC, shows good results [VJ99, WL02], only a comparatively minor number of works have been

proposed in this sense.

2.1.2 Multiple descriptions with channel splitting

The first MDC scheme to be introduced has been the *channel splitting*, developed in the late 1970s at the Bell Laboratories [CGG+78, Ger79, Mil80], and originally meant for speech signal. Channel splitting, depicted in Fig. 2.2, consists in the partition of the content of the original signal $I(k)$, usually achieved by polyphase subsampling, into a set $\{I_0(k), I_1(k), \cdots I_{N-1}(k)\}$ of signals to be encoded independently in order to generate the descriptions. The reconstruction, in case some descriptions are missing, is generally performed through interpolation.

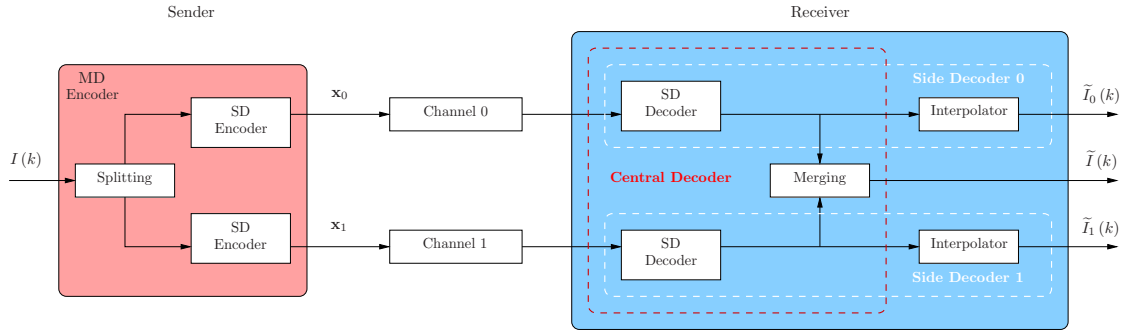


Figure 2.2: General scheme of a two-channel channel-splitting MD system.

In the following, we shall briefly describe how a video signal can be partitioned with temporal or spatial subsampling. These technique generally produce balanced descriptions.

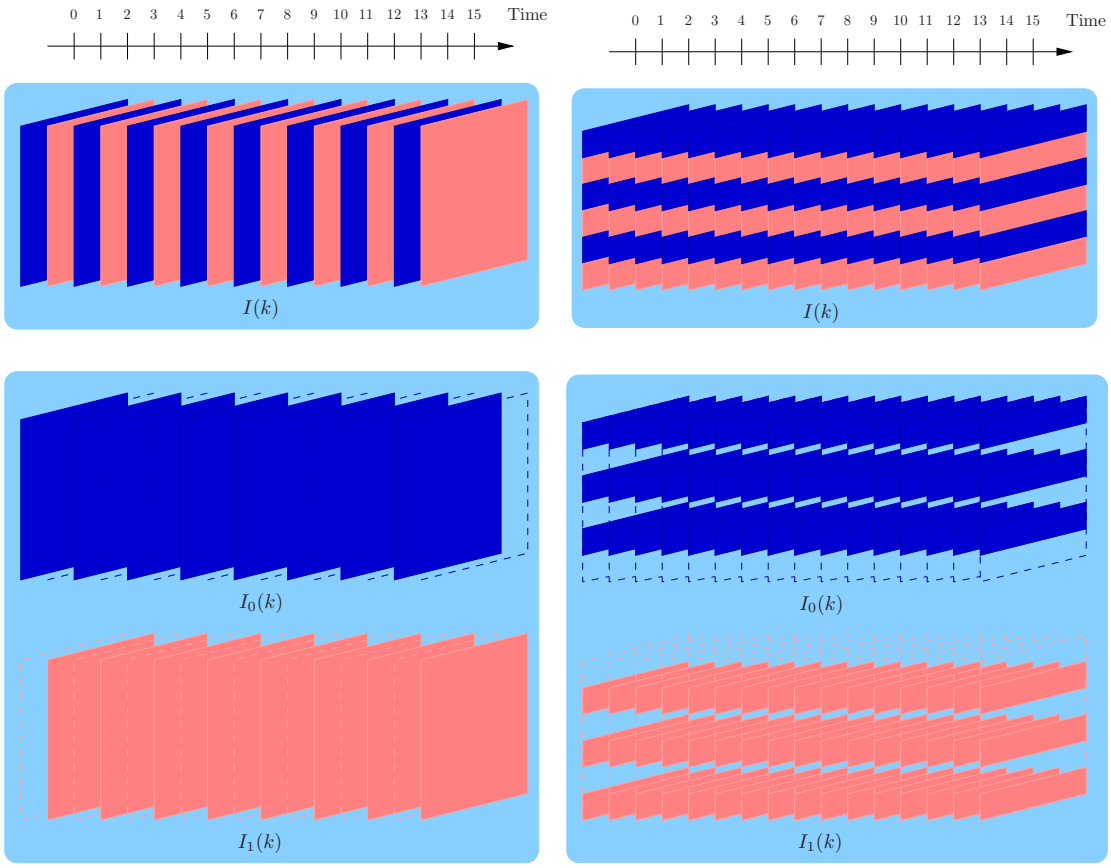
Temporal splitting

Temporal splitting for two descriptions consists in the separation of odd and even frames of a video sequence, as shown in Fig. 2.3(a). The two correlated sources are therefore generated as:

$$\begin{cases} I_0(n,m,k) = & I(n,m,2k) \\ I_1(n,m,k) = & I(n,m,2k+1) \end{cases}$$

The correlation between the sub-streams $I_0(k)$ and $I_1(k)$ depends on the degree of similarity of adjacent frames in the original sequence. When one description is missing, its samples can be approximated by temporally interpolating the other sub-stream. The interpolation technique can be as easy as sample-wise sample-and-hold interpolation, which is equivalent to reducing the frame rate of the reproduction, or a more sophisticated technique such as *motion compensated temporal interpolation*, explained in detail in Sec. 2.2.1.

When both descriptions are received, the two reconstructed sub-streams are merged to create the central reconstruction. The merging technique can be a simple interleaving



(a) Temporal splitting. (b) Spatial splitting.

Figure 2.3: Examples of channel splitting for a video sequence.

of the frames of the two sub-streams, or – as we propose in Sec 2.2.2 – a combination of received and interpolated sequence.

Temporal frame subsampling (or skipping or dropping) is a simple yet efficient technique to produce multiple descriptions. It provides very high compression ratios, especially in regular motion video (such as video conferencing [FFLT05, WRL05]), it is easy to implement, and the descriptions can be encoded as standard-compliant bitstreams. This latter property is particularly important as, even though existing standards can be used to provide MDC, it does not exist to the day a *de iure* standard explicitly addressing multiple description video coding.

Spatial splitting

Spatial splitting [FFLT05] is a second simple technique for building two or more descriptions, consisting in partitioning each individual frame of the video sequence. In Fig. 2.3(b), we provide an example of balanced spatial splitting based on separation of even and odd

rows of the frames. The two correlated sources are therefore generated as:

$$\begin{cases} I_0(n,m,k) = & I(2n,m,k) \\ I_1(n,m,k) = & I(2n+1,m,k) \end{cases}$$

Spatial splitting, as temporal splitting, presents the advantage providing good performance and of being easy to impalement. In this case, the correlation among the descriptions is given by the spatial correlation among neighbouring samples, so the quality of the side reconstructions depends on the regularity of the frames. However, spatial correlation and temporal correlation are very different in nature, and different interpolation techniques are used for side decoding.

2.1.3 Multiple descriptions with transform

The approaches presented so far are based on partitioning the signal in one of the domains it is defined on (time or space). The natural correlation between symbols in the source signal is exploited for reconstruction, *e.g.*, odd samples can be predicted from even samples, and *vice versa*. When such techniques are employed, the degree of correlation among the descriptions depends only on the statistics of the input signal, with barely any control by the designer.

A considerably different approach to MD coding is to actively design a linear transform in order to finely control the degree of correlation between the descriptions of the source signal. This approach is referred to as *MD transform coding*.

MD transform coding represents one of the most performing solutions for multiple description coding [GKAV98, GK01, WOVR01]. It provides good energy compaction properties, resilience to additive noise and quantisation, and great freedom to capture important signal characteristics. The correlation that remains after signal transformation can mitigate the effect of losses, since it offers the possibility to estimate the lost elements based on the received ones.

Pairwise correlating transform

Multiple description pairwise correlating transform (MDCT) was introduced by Wang, Orchard, and Reibman [WOVR01]. In this approach, the multiple description character is achieved by introducing a known correlation between pairs of transform coefficients included in different descriptions.

As we discussed in Sec. 1.2, in conventional (*i.e.*, SD) video coding, spatial redundancy among samples is reduced via two-dimensional transform coding, so that the transform coefficients are less correlated and more compact.

Let us now consider the MD case, where the quantised versions of the transform coefficients are to be sent over two different channels. If one of the channel fails and one

description is lost, being the coefficient poorly correlated, there would be no way to estimate the other.

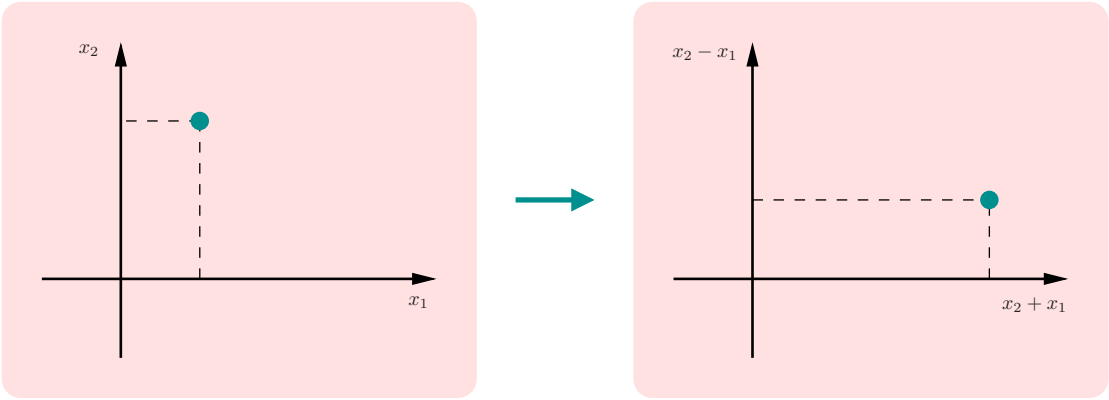


Figure 2.4: Example of correlating transform.

To prevent this, the sequence of coefficients could be processed with a further transformation step, wherein a *known* correlation is introduced in order to allow an estimation of a missing description. An example of correlating transform is given in Fig. 2.4, where a signal $\mathbf{x}$, consisting of two independent Gaussian variables x_1 and x_2, is transformed into $\mathbf{y}$, whose components are $y_1 = x_1 + x_2$ and $y_2 = x_2 - x_1$, which can be shown to be optimal for independent Gaussian sources [WOR97]. This transformation is such that the statistical dependencies between the components of $\mathbf{y}$ allow from any one of them to estimate the original two components of $\mathbf{x}$ to a certain accuracy, and the total estimation error for either component is the same.

In practice, the cascade of a decorrelating and a pair-wise correlating transform is actually implemented as a single linear transform such that coefficients intended to the same description are internally decorrelated, and coefficients intended to different descriptions are correlated with each other[WOVR01].

The correlation between the descriptions improves the side decoder RD performance, as it is now possible to obtain an acceptable reconstruction of all coefficients given one description, but it also degrades the central performance.

This method has been introduced for two descriptions in the context of image coding [WOR97], and subsequently extend to more general mappings to produce an arbitrary number of descriptions [GK98, GK01]. It is worth noticing that, whereas in SDC quantisation is performed after the transformation, quantising *before* applying a correlating transform has been shown to give the best performance [OWVR97, WOVR01].

Redundant signal transform

Another possibility to generate correlated representations of the same source signal via linear transform, is to project the signal onto an over-complete signal dictionary. For

discrete signals, this means that the number of output coefficients will be larger than the number of input signal samples. Then, different subsets of coefficients can be included in different descriptions and sent over independent channels.

Under some mild conditions [DS52], the redundant linear transform is a *frame expansion*, and the representation is known as *quantised frame expansion* (QFE).

While redundant transform had already proven its robustness towards additive white noise and quantisation, Goyal *et al.* [GVK98, GVT98, GKV99] proposed redundant transformation of the input signal as a way to achieve multiple description coding in the context of transmission with an unpredictable number of losses.

In the original proposal, the source signal was estimated from the QFE coefficients as a least-squares problem. Later on, Chou *et al.* [CMW99] investigated how to provide a more efficient reconstruction of the original signal from any subset of quantised coefficients, thus enabling practical reconstructions from over-complete transforms, which had not been possible before. Examples include windowed-DFT based schemes [BDV00] and redundant wavelet based schemes [TPPVdS04, TPPP07], the latter particularly interesting, as they are also inherently scalable.

It is worth noticing that the redundant wavelet based schemes also present the advantage of allowing scalability. An excellent survey on redundant wavelet schemes for multiple description coding of video sequences can be found in [TPPP07].

Kovačević *et al.* [KDG02] also investigated efficient oversampled filter-bank implementations of redundant transform schemes for robust transmission over the Internet.

More recently [RF07], the problem of optimal rate allocation for redundant transform MD schemes has been addressed, adapting the quantisation of the transform coefficients to the importance of the basis functions, the redundancy in the representation, and the expected loss probability on the channel.

2.1.4 Multiple descriptions with quantisation

In *Multiple Description Quantisation* (MDQ), a scalar, vector, or entropy-constrained quantiser is designed to produce a number of descriptions, using a generalised Lloyd-like clustering algorithm [PS01] that minimises a Lagrangian cost function of rate and expected distortions.

MDQ has been one of the pioneering practical approaches to multiple description coding [Goy01]. In the example depicted in Fig. 2.6(a), the MDC character is achieved with two uniform quantisers of step Δ, the second one being offset by half a quantisation interval with respect to the first one. If one description is lost, the source signal is reconstructed from samples quantised with a step of Δ; if both descriptions are received, the resulting quantisation step is $\frac{\Delta}{2}$.

In this scheme, if the side quantisers have b-bit resolution, the central decoder has approximately $(b+1)$-bit resolution. In other words, when no losses occur, the system

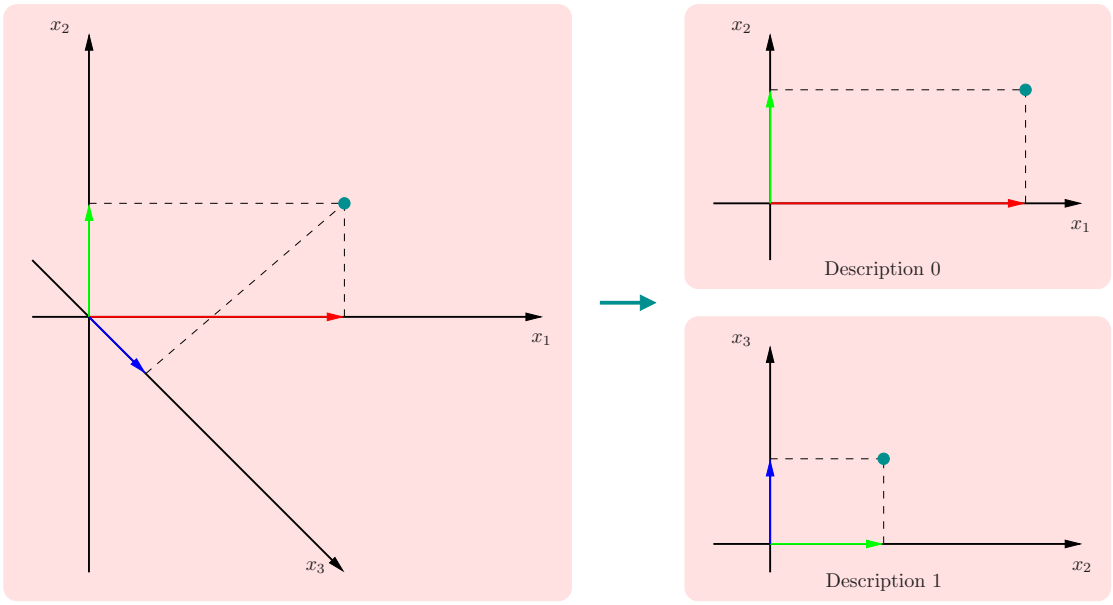


Figure 2.5: Example of redundant transform.

is using $2b$ bits in order to have a resolution of $b + 1$ bit, a very high redundancy that discourages the use of this technique unless the channel loss rates are very high and side reconstructions are very important.

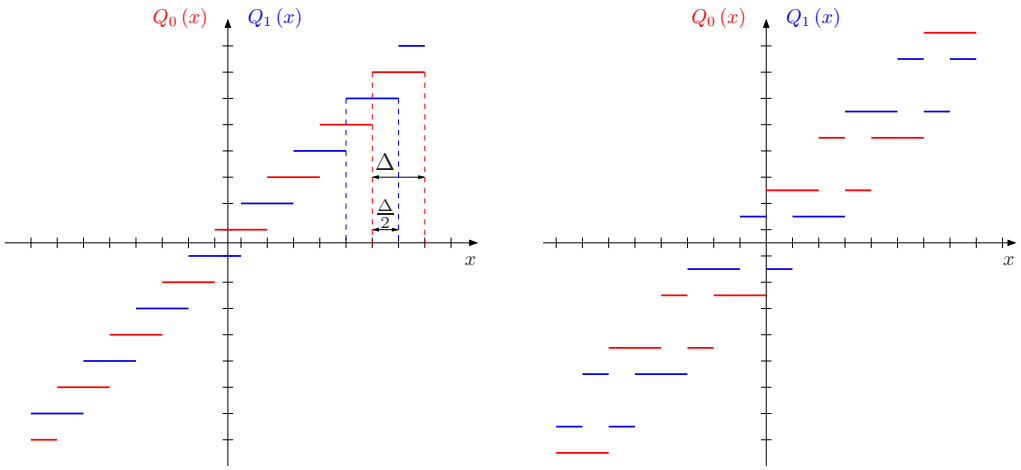
In order to overcome this drawback, Reudink [Reu80] invented several techniques with lower redundancy, such as the non-convex MD-quantiser, exemplified in Fig. 2.6(b).

Later on, Vaishampayan *et al.* [Vai93] independently proposed a theoretical framework for designing MD scalar quantisers with fixed-rate quantisation, subsequently extended to entropy-constrained quantisation [VD94].

The MD-quantisation techniques presented so far are completely agnostic with respect to the nature of the source signal (audio, image, video, *etc.*). More recently [GGP01, CCM08], MDVC schemes based on multiple description scalar quantiser and the H.264 AVC standard have been proposed for reliable real-time video applications.

Furthermore, approaches have been proposed for both video and still images that enable fine-grain scalability as well as robust coding of the input source, based on the joint use of the multi-resolution features of the DWT and an embedded multiple description scalar quantisation [VMG+06, GVM+07].

However, multiple description video coding, like scalable video coding (see Sec. 1.2.7), is known to suffer from *drift effect* when combined with motion-compensated prediction. That is, in presence of losses, the prediction signal available at the decoder may differ from the one used at the encoder, deteriorating the decoding process. In order to solve this problem, Crave *et al.* [CGPPT08, CGPP08, CPPG10] have proposed a robust scheme of multiple description coding with side information that integrates an error control approach in the MDVC scheme applying Wyner-Ziv Coding (WZC) to every other frame of



(a) Offset quantiser (b) Non-convex quantiser

Figure 2.6: Two examples of multiple description scalar uniform quantiser.

each description. In WZC [WZ76, AZG02], a systematic error-correction code is applied on the video frame, possibly encoded using transform coding, but only the parity bits are transmitted. The decoder generates an estimation of the current frame, *e.g.*, by interpolating the adjacent frames. This estimation can be considered as a noisy version of the current frame, that the decoder corrects using the parity bits sent by the encoder.

2.2 Proposed MDC scheme

In this section, we present our recently proposed multiple description video coding technique, which uses a legacy coder (namely, H.264 AVC) and time splitting in order to create two descriptions of a video sequence.

As we discussed in Sec. 2.1.2, an efficient MDVC system can be built by polyphase temporal subsampling of the source signal. This technique is very efficient in terms of total rate of the MD streams with respect to the stream encoded in SD, especially for sequences that present high temporal correlation [FFLT05, WRL05].

However, the distortion perceived by the users will also depend on the quality of the side decoding, *i.e.*, how the receiver reconstructs the skipped frames. It is well known [MPG85] that techniques such as frame freezing or pixel-wise linear interpolation introduce disturbing artifacts. Frame freezing generates rough, un-smooth motion, since the movement of objects is simply not taken into account. On the other hand, pixel-wise interpolation by temporal linear filtering generates blurring in the moving areas, since pixel values of different object are mixed, resulting in the blurring of object boundaries.

In order to remove these artifacts, Wong *et al.* [WA95, WAT96] proposed to use *motion compensated temporal interpolation* (MCTI) to reconstruct the skipped frames with considerably less artifacts. In the following, we shall present two possible approaches to

MCTI and how they can be integrated into a temporal splitting MDVC technique.

2.2.1 Motion compensated temporal interpolation

In MCTI, the motion of objects is compensated for by tracking them between temporal adjacent frames. Known the trajectory of each object, they can be placed at the appropriate location in the interpolated frame. Bidirectional motion estimation is normally used to locate the objects keeping into account the uncovered and the newly-covered regions. The generated motion vector field is then used to construct the interpolated frames.

In this section, we shall present two techniques that implement MCTI inspired by the DISCOVER algorithm [AAD+07], which works under the assumption of constant velocity motion in the scene, and its high-order extension HOMI [PCPP10], which generalises DISCOVER by only assuming linear acceleration.

The DISCOVER motion interpolation

DISCOVER (DIStributed COding for Video sERvices) is a European project funded under the European Commission IST FP6 programme. The DISCOVER project has been devoted to the advancement of *Distributed Video Coding* (DVC) for many years.

DVC is original coding framework that enables the separate encoding of correlated signals with the same compression efficiency as centralised joint compression. This allows to displace the complexity from the encoder to the decoder without loss of performance, which is desirable whenever the encoding has to be carried on by a multitude of simple cheap devices, *e.g.*, wireless sensor networks [VDJ+12].

The theoretical foundations of DVC have been laid more than thirty years ago [SW73, WZ76], but only recently some practical implementations have been proposed. These implementations, however, show a compression performance still quite far from theoretical bounds, making DVC one of the most attracting research issues in the field of digital video coding in these days [GARRM05].

Within the DISCOVER project, a distributed video codec architecture and a set of associated tools have been proposed. Among these tools, we focus on the DISCOVER interpolation algorithm [AAD+07] used at the decoder, and summarised in Fig. 2.7. However, we consider here only the motion compensated temporal interpolation algorithms of DISCOVER and HOMI, as the remaining channel correcting part is not of interest within our scenario.

Let $I(k)$ be the frame to be estimated, *i.e.*, belonging to the missing description. Its estimation $\widetilde{I}(k)$ is produced by using the temporal adjacent frames $I(k-1)$ and $I(k+1)$. Since we apply the interpolation algorithm in the context of MDC, we can assume that the frames adjacent to the one being interpolated are available from the received description.

First, the reference frames are spatially filtered to smooth out possible noise and higher frequency contributions. Then, a block matching algorithm is run in order to find a forward

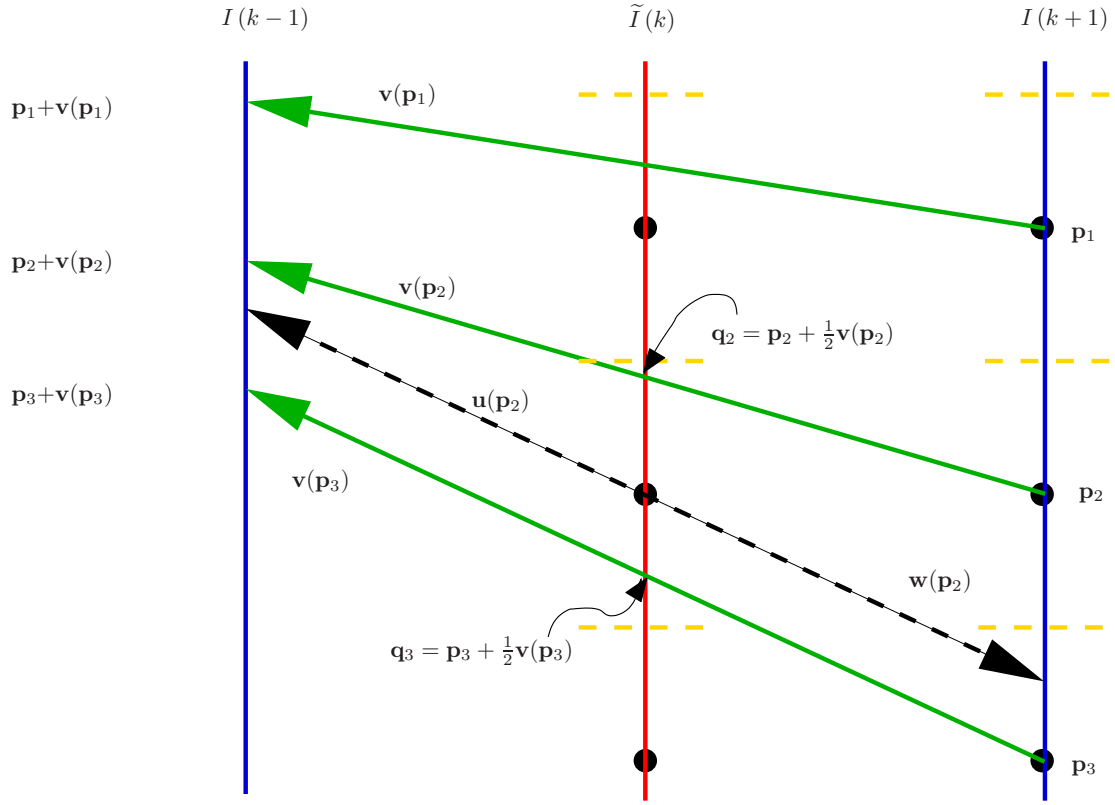


Figure 2.7: Bidirectional motion estimation in DISCOVER. Green solid arrows: results of forward ME. Black dashed arrows: results of bidirectional ME for the block centred in $\mathbf{p}_2$.

motion vector field between images $I(k-1)$ and $I(k+1)$. A further bidirectional motion estimation is performed in order to find the movement between the current $I(k)$ and the references.

Let us now consider a block of pixels centred in position $\mathbf{p}_2$ (see Fig. 2.7). Let $\mathbf{v}$ be the motion vector field from $I(k+1)$ to $I(k-1)$, $\mathbf{u}$ the one from $I(k)$ to $I(k-1)$, and $\mathbf{w}$ the one from $I(k)$ to $I(k+1)$. The motion vector computed by the forward motion estimation is $\mathbf{v}(\mathbf{p}_2)$ and it points to the position $\mathbf{p}_2 + \mathbf{v}(\mathbf{p}_2)$ in the frame $I(k-1)$. The underlying model assumes linear, constant-speed motion, *i.e.*, $\mathbf{u}(\mathbf{p}_2 + \frac{1}{2}\mathbf{v}(\mathbf{p}_2)) = \frac{1}{2}\mathbf{v}(\mathbf{p}_2)$. However, in order to avoid gaps and overlaps in the motion compensated image, it is needed to estimate $\mathbf{u}(\mathbf{p}_2)$. For this position, the vector closest to the block centre is considered. In Fig. 2.7 this is $\mathbf{v}(\mathbf{p}_3)$, since $\|\mathbf{p}_2 - \mathbf{q}_3\| < \|\mathbf{p}_2 - \mathbf{q}_2\|$, where $\mathbf{q}_i = \mathbf{p}_i + \frac{1}{2}\mathbf{v}(\mathbf{p}_i)$.

In conclusion, in this case the DISCOVER algorithm will choose:

$$\mathbf{u}(\mathbf{p}_2) = \frac{1}{2}\mathbf{v}(\mathbf{p}_3) \qquad \mathbf{w}(\mathbf{p}_2) = -\frac{1}{2}\mathbf{v}(\mathbf{p}_3) \tag{2.2}$$

Finally, two further processing steps are applied to the motion vector fields: first, the vectors are refined around the position found in Eq. (2.2); second, the fields are regularised

via a weighted median filter. We thus obtain a couple of fields to be used for motion compensation of $I(k-1)$ and $I(k+1)$. The average of the compensated images will be the estimation $\widetilde{I}(k)$ of $I(k)$.

High-order motion interpolation

The DISCOVER motion interpolation technique performs well for constant-velocity movement, but when between the two closest frames the acceleration is not null, a high order motion interpolation (HOMI, summarised in Fig. 2.8) is necessary to better model the movement [PCPP10].

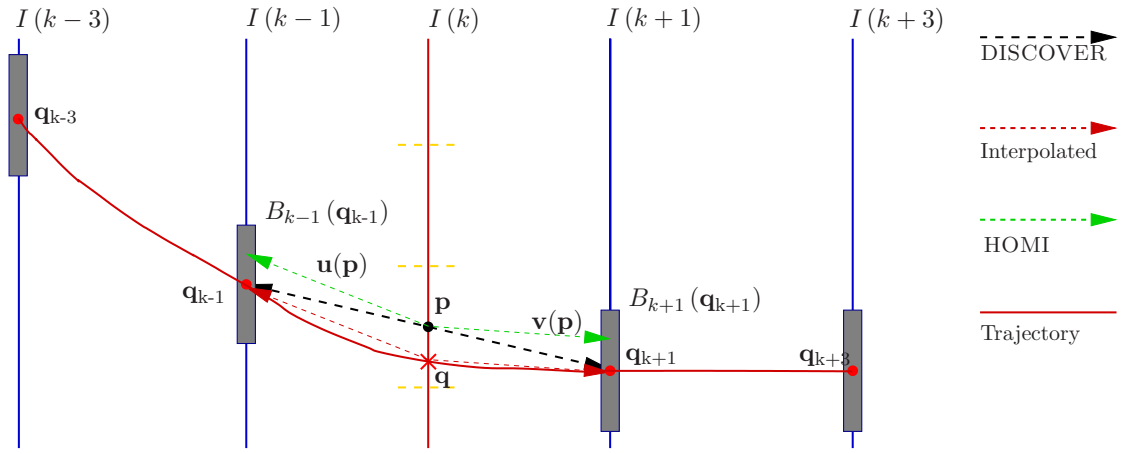


Figure 2.8: High-order interpolation algorithm for motion estimation. The position of the block in the current frame is estimated by interpolating its trajectory in two previous and two following frames.

The main idea behind HOMI consists in using four frames, instead of two, for the interpolation. In this case, the frames that will be used for the interpolation are $I(k-3)$, $I(k-1)$, $I(k+1)$ and $I(k-3)$.

As mentioned above, we apply the interpolation algorithm in the context of a temporal splitting MDVC technique, thus we can assume that frames $I(k-3)$ and $I(k+3)$ are available, since each description has either the even or the odd frames of the original sequence.

First, the DISCOVER algorithm is applied on frames $I(k-1)$ and $I(k+1)$. The result is a pair of motion vector fields $\widetilde{\mathbf{u}}(\cdot)$ and $\widetilde{\mathbf{v}}(\cdot)$ (black dashed vectors in Fig. 2.8). These vector fields are such that, for each point $\mathbf{p}$, the missing frame may be estimated as:

$$\widetilde{I}(k)(\mathbf{p}) = \frac{I(k-1)(\mathbf{p}+\widetilde{\mathbf{u}}(\mathbf{p})) + I(k+1)(\mathbf{p}+\widetilde{\mathbf{v}}(\mathbf{p}))}{2}$$

Let us define the block of the frame $I(k)$ centred in $\mathbf{p}$ as $B_k(\mathbf{p})$. The DISCOVER technique may still be affected by an error, consisting in an incorrect placing of the block $B_k(\mathbf{p})$ in $\widetilde{I}(k)$. Hence, it is reasonable to try and displace the block taking into account frames $I(k-3)$ and $I(k+3)$ as well.

In order to do that, the blocks centred in the points $\mathbf{q}_{\text{k-1}} = \mathbf{p} + \widetilde{\mathbf{u}}(\mathbf{p})$ in $I(k-1)$ and $\mathbf{q}_{\text{k+1}} = \mathbf{p} + \widetilde{\mathbf{v}}(\mathbf{p})$ in $I(k+1)$ are matched with blocks in frames $I(k-3)$ and $I(k+3)$ respectively. Let $\widehat{\mathbf{u}}(\cdot)$ and $\widehat{\mathbf{v}}(\cdot)$ be the corresponding motion vector fields, and let $\mathbf{q}_{\text{k-3}} = \mathbf{q}_{\text{k-1}} + \widehat{\mathbf{u}}(\mathbf{q}_{\text{k-1}})$ and $\mathbf{q}_{\text{k+3}} = \mathbf{q}_{\text{k+1}} + \widehat{\mathbf{v}}(\mathbf{q}_{\text{k+1}})$. It is fair to suppose that the blocks $B_{k-3}(\mathbf{q}_{\text{k-3}})$, $B_{k-1}(\mathbf{q}_{\text{k-1}})$, $B_{k+1}(\mathbf{q}_{\text{k+1}})$, and $B_{k+3}(\mathbf{q}_{\text{k+3}})$ are actually the same block that is moving; hence, the HOMI algorithm interpolates its trajectory to find the position of its centre in time instant k, indicated by $\mathbf{q}$.

The motion vector field from k to $k-1$ is $\mathbf{u}(\mathbf{q}) = \mathbf{q}_{\text{k-1}} - \mathbf{q}$ and the one from k to $k+1$ would be $\mathbf{v}(\mathbf{q}) = \mathbf{q}_{\text{k+1}} - \mathbf{q}$ (red dashed vectors in Fig. 2.8). In order to avoid holes and overlapping blocks, since $\mathbf{q}$ might not correspond to the centre of any block in $I(k)$, HOMI assumes that $\mathbf{u}(\mathbf{p}) \approx \mathbf{u}(\mathbf{q})$ and $\mathbf{v}(\mathbf{p}) \approx \mathbf{v}(\mathbf{q})$ (green dashed vectors in Fig. 2.8), justified by the fact that $\mathbf{p}$ and $\mathbf{q}$ are very close to each other, *i.e.*, they belong to the same object. Then, the missing frame can be estimated for each point $\mathbf{p}$ as:

$$\widetilde{I}^{*}(k)(\mathbf{p}) = \frac{I(k-1)(\mathbf{p} + \mathbf{u}(\mathbf{p})) + I(k+1)(\mathbf{p} + \mathbf{v}(\mathbf{p})))}{2}$$

In principle, this technique could be extended to an arbitrary number of frames preceding and following $I(k)$; however, the number of positions is limited in order to reduce the dependency among frames, which could affect the robustness of the decoding in a lossy transmission scenario. Moreover, if a too large number of frames is used, the interpolated trajectory may be incorrect and unreliable, due to possible mismatches during the block matching.

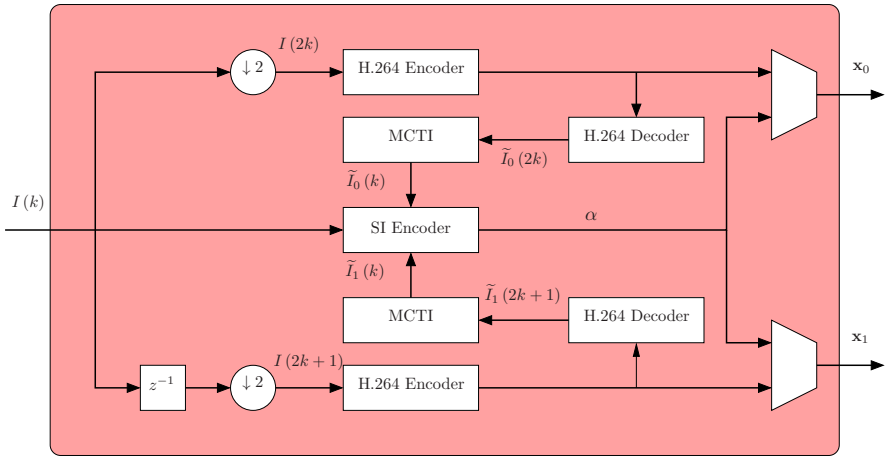
2.2.2 MCTI-based video MDC

Even though the techniques presented above had been originally proposed for *Distributed Video Coding* (DVC) [GPT+07], they can be used to interpolate the reconstructed substream of a temporal splitting multiple description video coding system.

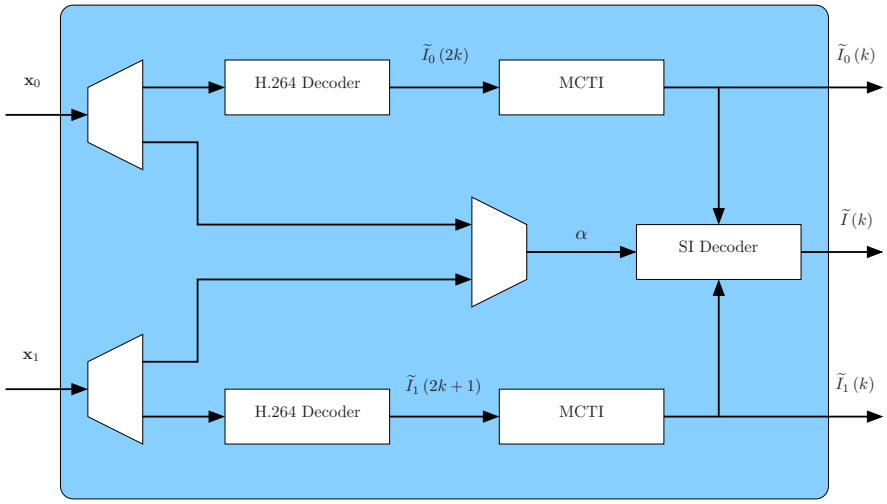
This is a simple technique to obtain an MD codec from a legacy coder without having access to the codec implementation, which is used as a black-box. In the following, we shall explain in detail each component of the proposed MDVC system, depicted in Fig. 2.9. An experimental study for this system will be presented in Sec. 2.2.3.

Encoding

At the encoder side (Fig. 2.9(a)), the original sequence $I(k)$ is first of all split up into even and odd frames. Each sub-sequence is then separately encoded with a hybrid video coder in order to produce the two descriptions, as discussed in Sec. 2.1.2. Our technique is in principle codec-agnostic, as it only works with decoded frames (before encoding or after decoding); however, we shall assume that H.264/AVC, described in Sec. 1.3.2, is used to this purpose.



(a) Encoder



(b) Decoder

Figure 2.9: Scheme of the proposed temporal channel-splitting MDVC system.

The two encoded bitstreams are then locally decoded, delivering the sequences $\widetilde{I}_0(2k)$ and $\widetilde{I}_1(2k+1)$. These are interpolated using a motion compensated temporal interpolation technique to obtain $\widetilde{I}_0(k)$ and $\widetilde{I}_1(k)$. These are exactly the same as the reconstructed descriptions that will be available at the decoder. In the first proposed version of this technique [GCPP10], we employed the DISCOVER technique of temporal interpolation described in Sec. 2.2.1. However, in order to improve the performance for sequences with accelerated movement, we subsequently proposed [GPCPP11] to replace the DISCOVER algorithm with its high-order extension described in Sec. 2.2.1.

The two reconstructed descriptions are used, together with the original sequence, to generate the side information. Let B be a block of fixed size in a frame $I(k)$, and let B_0 and B_1 be the corresponding blocks in $\widetilde{I}_0(k)$ and $\widetilde{I}_1(k)$, respectively. For each block B,

the encoder finds an optimal value $\alpha_{k,B}$, defined as:

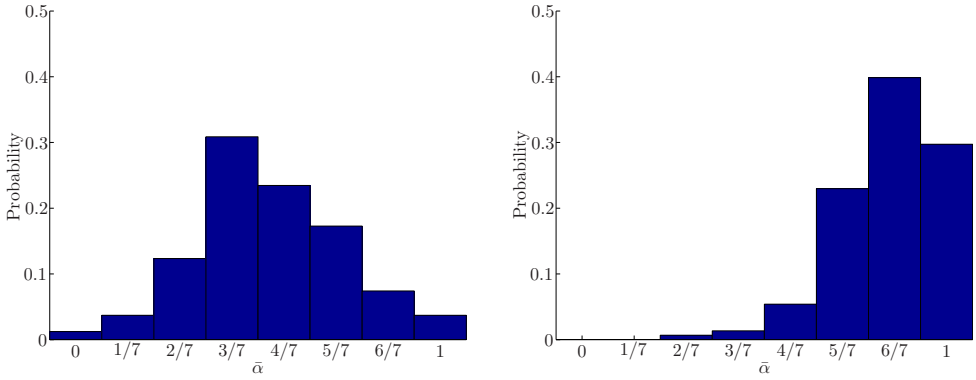
$$\alpha_{k,B} \triangleq \arg \min_{\omega \in [0,1]} \left\{ J\left(\omega,\ \omega B_0 + (1-\omega) B_1,\ B\right) \right\},$$

where $J(\cdot)$ is a cost function depending on the distortion of the convex combination of the two reconstructed blocks B_0 and B_1 with respect to the original block B.

Even though the sequence of optimal relative weights α is theoretically continuous, experimental results show that quantising α on three bits – *i.e.*, eight levels – introduces a negligible distortion on the reconstructed sequence [ZL09, GCPP10]. Let $\bar{\alpha} = Q(\alpha)$ be the three-bits quantised version of α. In order to reduce the bitrate needed to transmit $\bar{\alpha}$, we proposed to use a context-based integer arithmetic coding. We found that there is a statistical dependency of the values of $\bar{\alpha}$ on the quantity $E_{k,B}$, defined as:

$$E_{k,B} = \mathrm{MSE}\left(B_0,\ B_1\right),$$

that is, the distortion between the two descriptions of the block B. When the descriptions are very different, usually the received block is a better representation of the original one than the interpolated block. In other words, for high values of E, the probability mass function of $\bar{\alpha}$ is more concentrated around 1, as shown in Fig. 2.10.



(a) PMF of $\{\bar{\alpha} | E \leq 15\}$ (b) PMF of $\{\bar{\alpha} | E \geq 400\}$

Figure 2.10: Example of the Probability Mass Function of the quantised combination coefficients $\bar{\alpha}$, given high and low values of the context E. (Sequence "Foreman" at ~ 300 kbps.)

Therefore, the context-based arithmetic coding has a rate bounded by $H(\bar{\alpha}|E)$, and $H(\bar{\alpha}|E) < H(\bar{\alpha})$, because of the dependency. However, since the number of possible contexts (*i.e.*, of MSE values) is very high, we risk to incur into a *context dilution* problem: having too many contexts makes it difficult or practically impossible to estimate and update the conditional probabilities of symbols during the encoding process. In order to cope with this problem, we proposed to perform a context quantisation procedure: instead of using a different arithmetic coder for each value of E, we group these values into clusters defined by a quantisation function $Q(E)$. This increases the coding rate, since

$H(\bar{\alpha}|Q(E)) \geq H(\bar{\alpha}|E)$. The difference between the two rate bounds (*i.e.*, the rate penalty) is the mutual information $I(\bar{\alpha}, E|Q(E))$ [CAB10]. For the sake of simplicity, we use a convex quantiser, *i.e.*, the MSE values are grouped into intervals, and only a selection of the quantisation thresholds is needed. These thresholds are chosen by minimising the mutual information $I(\bar{\alpha}, E|Q(E))$. There exist complex iterative techniques to solve this problem, such as the popular *Minimum Conditional Entropy Context Quantisation* [WCX00] or its improved version MINIMA [CAB10], but they are not needed in this case, given the relatively simple structure of the quantiser. We propose instead to use a simple gradient descent algorithm.

In the first proposed version of this technique [GCPP10], $J(\cdot)$ was chosen simply as the distortion of the convex combination of B_0 and B_1 with respect to the original block B. The resulting rate $R(\bar{\alpha})$ for the quantised sequence only depended on its conditional entropy given the quantised context. However, we later [GPCPP11] proposed to estimate $H(\bar{\alpha}|Q(E))$ with a preliminary test, then use the conditional entropy as an estimation $\widetilde{R}(\bar{\alpha})$ of the coding cost $R(\bar{\alpha})$, and to define the minimisation problem as follows:

$$\alpha_{k,B} \triangleq \arg \min_{\omega \in [0,1]} \left\{ \mathrm{MSE}\left(\omega B_0 + (1-\omega)B_1,\ B\right) + \lambda \widetilde{R}\left(\omega\right) \right\},$$

that is, to choose α in an RD-optimal way, where the Lagrangian parameter λ can be chosen equal to the one used by the hybrid coder to encode the sequence. In either case, the entropy-coded sequence of quantised relative weights $\bar{\alpha}$ is sent along with the descriptions as side information.

Decoding

At the decoder side (Fig. 2.9(a)), side decoding is performed on the received bitstreams with a standard H.264/AVC decoder, then reconstructing the missing frames via motion compensated temporal interpolation. If only one description is received, the corresponding reconstructed frames are played out.

If both descriptions are received, central decoding is performed as a block-wise convex combination of the two reconstructed descriptions $\widetilde{I}_0(k)$ and $\widetilde{I}_1(k)$. That is, each block $\widetilde{B}$ of the central reconstructed frame $\widetilde{I}(k)$ is computed as:

$$\widetilde{B} = \bar{\alpha}_{k,B} B_0 + (1 - \bar{\alpha}_{k,B}) B_1.$$

A scheme of the central decoder is shown in Fig. 2.11.

The idea of reusing information from the lower fidelity version of a frame in central decoding can be found in the work of Radulovic *et al.* [RWW+07, RFW+10]. In their work, they argue that although the high fidelity version of the frame is correctly received, it may happen that its reference frames are affected by losses, causing error that may propagate on the received frame as well. On the other hand, it may happen that the

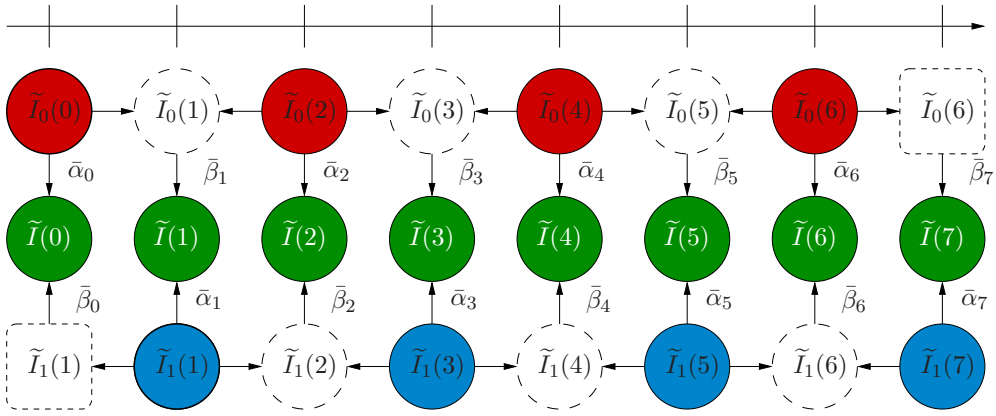


Figure 2.11: Structure of the central decoder. Solid circles represent received frames; dashed circles represent interpolated frames. Horizontal arrows represent interpolation. Vertical arrows represent weighted sum. With $\widetilde{I}_d(k)$ we denote the kth frame of description d. For each k, $\bar{\beta} = 1 - \bar{\alpha}$.

frames from which a lower fidelity frame is generated are error-free, or anyway less affected by transmission errors. In these scenarios, they propose to use a rate-distortion optimal macroblock selection between a higher and lower fidelity version of the frames.

Later on, Liu and Zhu [LZ07, ZL09] pointed out that, instead of choosing either the higher or the lower fidelity block, a convex combination of the two could be used for central reconstruction. In their framework, the lower fidelity version of the frames consisted of coarsely quantised B-frames of lower hierarchical level, while the higher fidelity version consisted in more finely quantised B-frames of higher hierarchical level.

This scheme also shares some similarities with a generic MDC scheme previously proposed by Jiang *et al.* [JO99]. In this scheme, an input signal X is split using a polyphase linear transform into two components Y_0 and Y_1. Then, each component is quantised both with a fine quantiser Q_1 and a coarser quantiser Q_2. The output of both quantisers are multiplexed together for transmission. The decoding scheme, however, is quite different. While Jiang *et al.* propose, when both descriptions are received, to reconstruct the signal using only the finely quantised coefficients, whereas Liu, Zhu *et al.*propose to enhance this information using the coarsely quantised ones as well.

Our scheme differs from both of these schemes, as in our scheme no coarse representation of the complementary description is sent: we propose instead to use an interpolated frame generated at the decoder side, and therefore requires no bandwidth to be transmitted.

2.2.3 Experimental study

In this section, we present an experimental study for the proposed multiple description video coding technique [GPCPP11]. As a reference, we choose the technique proposed by Liu and Zhu [LZ07, ZL09] mentioned in Sec. 2.2.2, which originally introduced the idea of a

temporal splitting MDVC technique performing central decoding as a convex combination of a higher and a lower fidelity version of the same image, namely, two transmitted B-frames of different hierarchical levels. For a fair comparison, in both techniques we used the H.264 AVC reference software JM version 17.0 [Süh11] to encode the sequences.

The key difference between the two schemes is that the frames encoded as lowest level B-frames in the reference scheme are simply skipped in ours, and replaced at the decoder with an interpolation obtained via MCTI. In both schemes, we used a closed Hierachical-B GOP structure, with the size of the GOP relatively small (8 frames). We deemed this scheme to be a suitable choice for transmission over a lossy channel, since it limits error propagation.

It should be noticed that, in the reference scheme, the rate overhead for the side information depends on the entropy of the coefficients α, whereas in our scheme, it depends on the *conditional* entropy of α given the distortion between the two decoded descriptions. The benefit of this strategy of entropy coding can be seen in Table 2.1, where the overheads for the two schemes are presented.

QP	Arithmetic coding	Context-adaptive arithmetic coding	Gain
22	26.30	19.75	25.21%
25	27.89	22.24	20.36%
28	29.26	24.40	16.70%
31	30.27	26.08	13.95%
33	30.86	27.02	12.53%
36	31.71	28.30	10.90%
39	32.40	29.32	9.71%
42	32.61	29.90	8.72%

Table 2.1: Bitrate (in kbps) needed to transmit the side information using arithmetic coding *vs.* context adaptive arithmetic coding based on the distortion between the two decoded descriptions.

In order to compare the rate-distortion performance of the two methods, a set of eight QPs has been selected, namely: 22, 25, 28, 31, 33, 36, 39, and 42. The rate-distortion curves for the video sequences "Akiyo", "Foreman", and "Bus" (CIF, 30 fps) on the first 128 frames are shown in Fig. 2.12.

We measured the gain of our technique with the metric proposed by Bjontegaard, as recommended by the ITU-T VCEG [Bjø01]. For central decoding of the sequence "Foreman" at low bitrates we observe a gain with respect to the reference technique of 1.84 dB in Y-PSNR, corresponding to a reduction of 25.95 % rate. A more extensive comparison over a larger set of video sequences is given in Table 2.2. Notice that, since the sequences are encoded with a fixed QP, the resulting bitrate is higher for sequences with a higher motion content.

The gain we observe can be explained as the HOMI interpolation technique provides

Sequence	Bitrate Range [kbps]	Y-PSNR Gain (Central)	Bitrate Variation (Central)	Y-PSNR Gain (Side)	Bitrate Variation (Side)
akiyo	39 ∼ 67	+4.89 dB	−42.24 %	+3.88 dB	−51.60 %
akiyo	56 ∼ 100	+2.99 dB	−35.59 %	+2.11 dB	−40.73 %
akiyo	77 ∼ 194	+1.69 dB	−29.63 %	+1.42 dB	−34.25 %
hall	58 ∼ 105	+3.24 dB	−34.50 %	+2.58 dB	−49.41 %
hall	83 ∼ 178	+1.45 dB	−27.08 %	+1.33 dB	−37.55 %
hall	127 ∼ 458	+0.63 dB	−21.02 %	+0.74 dB	−35.64 %
foreman	95 ∼ 190	+1.84 dB	−25.95 %	+1.71 dB	−37.90 %
foreman	148 ∼ 317	+1.39 dB	−24.99 %	+1.41 dB	−38.52 %
foreman	231 ∼ 657	+1.12 dB	−23.47 %	+1.23 dB	−41.38 %
city	82 ∼ 190	+1.54 dB	−24.06 %	+1.41 dB	−33.85 %
city	141 ∼ 336	+1.02 dB	−17.40 %	+0.80 dB	−20.10 %
city	238 ∼ 726	+0.68 dB	−12.76 %	+0.25 dB	−07.50 %
flower	154 ∼ 511	+1.08 dB	−20.77 %	+0.91 dB	−22.00 %
flower	342 ∼ 1072	+0.78 dB	−14.88 %	+0.69 dB	−17.13 %
flower	691 ∼ 2322	+0.74 dB	−11.94 %	+0.67 dB	−15.96 %
mobile	156 ∼ 422	+1.06 dB	−20.80 %	+0.80 dB	−21.40 %
mobile	286 ∼ 931	+0.60 dB	−14.67 %	+0.14 dB	−05.66 %
mobile	572 ∼ 2401	+0.64 dB	−13.78 %	−0.9 dB	+2.74 %
stefan	166 ∼ 416	+1.11 dB	−16.04 %	+0.77 dB	−16.98 %
stefan	305 ∼ 786	+0.56 dB	−10.09 %	+0.24 dB	−05.45 %
stefan	544 ∼ 1791	+0.40 dB	−07.18 %	−0.21 dB	+08.10 %
coastguard	93 ∼ 343	+0.64 dB	−19.18 %	+0.59 dB	−25.23 %
coastguard	208 ∼ 830	+0.43 dB	−12.04 %	+0.37 dB	−13.27 %
coastguard	494 ∼ 2058	+0.46 dB	−09.79 %	+0.44 dB	−13.44 %
bus	172 ∼ 473	+0.86 dB	−16.33 %	−0.20 dB	+03.86 %
bus	335 ∼ 890	+0.76 dB	−13.48 %	−0.11 dB	+21.03 %
bus	608 ∼ 1861	+0.69 dB	−11.60 %	−1.13 dB	+35.81 %
football	191 ∼ 574	+0.62 dB	−13.51 %	−0.89 dB	+35.32 %
football	399 ∼ 1062	+0.44 dB	−08.49 %	−1.27 dB	+51.97 %
football	737 ∼ 2083	+0.42 dB	−06.68 %	−1.44 dB	+51.81 %

Table 2.2: Bjontegaard metric of the proposed technique with respect to the reference over various sequences with increasing motion content.

a reconstruction of the missing frames better than the very coarse version provided by the lowest level B-frames in the reference method. Also, whereas our reconstruction is computed at the decoder and does not need to be transmitted, the low quality B-frames of the reference technique, even when a coarse quantisation is used, still need a certain bitrate in order transmit the motion vectors, the mode selections and so on. For a visual comparison of the two techniques, the 9th frame of the video sequence "Foreman" encoded at about 300 kbps with both techniques is shown in Fig. 2.14.

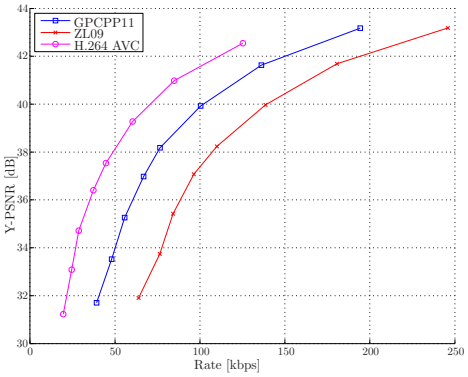
As one could expect, the rate-distortion performance of our scheme, which is based on temporal interpolation, highly depends on the motion content of the video sequence. However, whereas side decoding is severely impaired for sequences with fast movement, central decoding is still efficient, since the low fidelity of the side reconstruction is compensated

with an appropriate value of α.

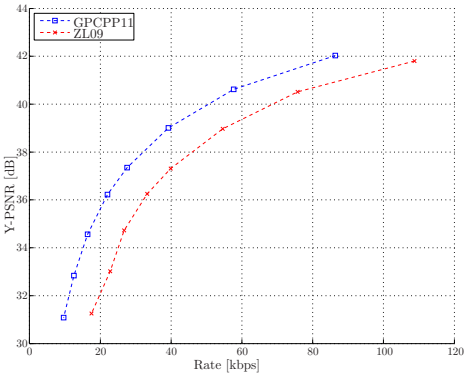
Since we approach MDC as a technique to grant loss immunity to video streams, it is worth providing a performance comparison as a function of the packet loss rate. In a first set of tests, the three video sequences "Akiyo", "Foreman", and "Bus" have been encoded at about 200 kbps. The resulting bitstreams, packetised at one frame per packet, have been affected by packet losses, modelled as independent and identically distributed Bernoulli random variables with probability equal to the loss rate. The same packet loss rate could actually lead to different frame loss rates, since not only the frame or frames in the packet will be lost, but also all the frames predicted upon them (obviously, the losses do not propagate from one description to another). In order to avoid a bias due to the GOP structure, the results presented here are obtained by averaging twenty simulations with the same value of packet loss rate. Both techniques, for each image of the decoded sequence, use central decoding whenever both description are received, side decoding if only one description is received, and concealment (freeze of the last decoded image) if both descriptionss are lost. In the results shown in Fig. 2.13(a), we observe that, as expected, sequences with higher motion content are more affected by packet loss. However, our technique consistently outperforms the reference method.

In a second set of tests, shown in Fig. 2.13(b), the two methods have also been compared over several bitrates for a fixed packet loss rate of 10 %. It can be seen how, at low bitrates, our method at 10 % losses provides a better video quality than the reference method at 0 % losses.

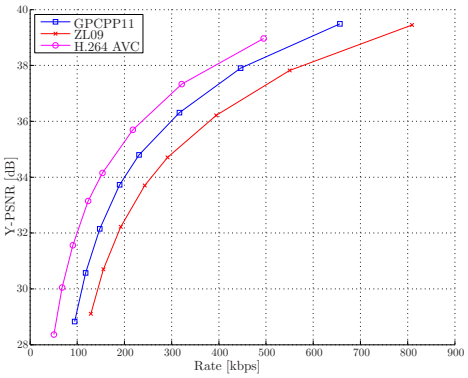
A performance comparison in a more realistic simulation scenario, where the encoded sequences are transmitted over a mobile ad-hoc network using the ABCD protocol, is presented is Chapter 3.



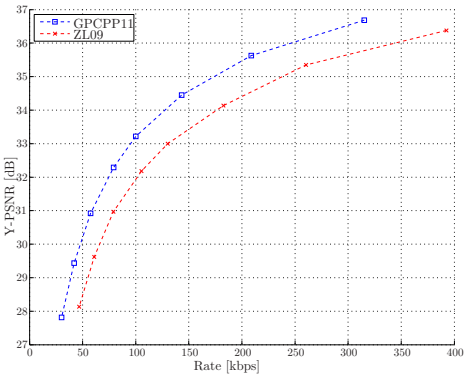
(a) “Akiyo” (Central decoder)



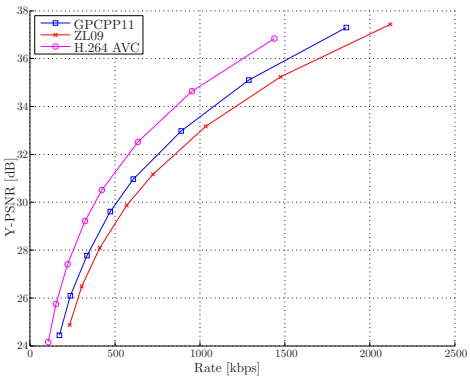
(b) “Akiyo” (Side decoder)



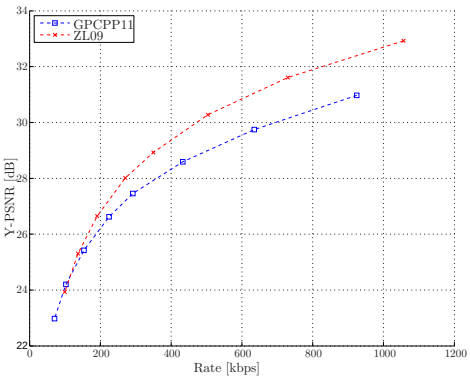
(c) “Foreman” (Central decoder)



(d) “Foreman” (Side decoder)

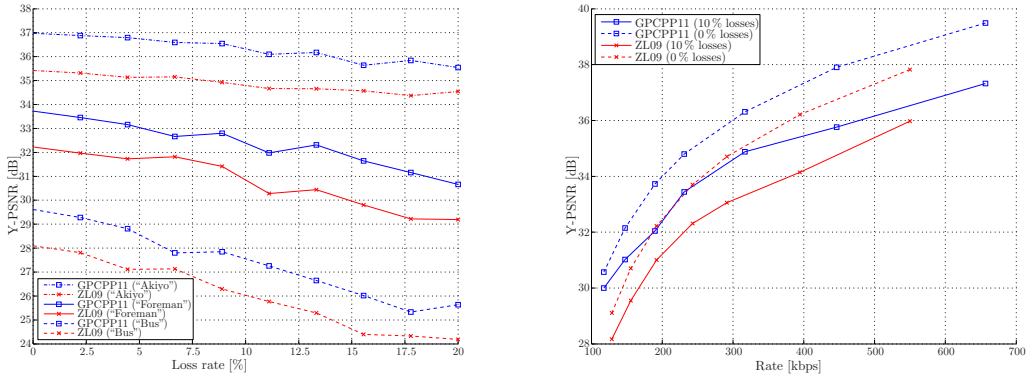


(e) “Bus” (Central decoder)



(f) “Bus” (Side decoder)

Figure 2.12: Comparison of the proposed scheme [GPCPP11] against the Hierarchical B-Picture MDVC scheme in [ZL09] for several CIF sequences at 30 fps. Performance of single description coding with H.264 AVC also reported for reference.



(a) Video quality *vs.* packet loss rate for fixed bitrate(b) Video quality *vs.* bitrate for fixed packet loss rate.

Figure 2.13: Comparison of the proposed scheme [GPCPP11] against the Hierarchical B-Picture MDVC scheme in [ZL09] as a function of the packet loss rate.

(a) GPCPP11, central decoder (35.8 dB)

(b) ZL09, central decoder (35.0 dB)

(c) GPCPP11, side decoder (33.4 dB)

(d) ZL09, side decoder (32.3 dB)

Figure 2.14: Visual comparison of the proposed scheme [GPCPP11] against the Hierarchical B-Picture MDVC scheme in [ZL09]. Sequence "Foreman" at ~ 300 kbps, 9th frame.

2.3 Conclusions

In this chapter, we discussed the subject of multiple coding. First, we introduced the basic concepts of MDC, with a brief overview of the most popular techniques used to design a multiple description coding system.

Then, we presented an original contribution in this field that we have recently proposed. The key idea of this technique is to use a motion compensation temporal image interpolation (inspired to the distributed video coding context) for side decoding. Central decoding is performed using a linear combination of the interpolated image blocks and the received blocks, obtained from the two descriptions. The optimal combination coefficients are computed at the encoder and efficiently sent to the decoder as side information. The proposed technique shows a remarkable gain for central decoding with respect to similar methods available in the state of the art. The first version of this technique, using a linear temporal interpolation, was presented at the 2010 IEEE Workshop on Multimedia Signal Processing (MMSP). We later proposed to replace linear interpolation with high-order motion interpolation, in order to deal with accelerated movement between frames.

The improvements deriving from this new interpolation methods have been presented at the 2011 edition of the MMSP workshop, where we also discussed how this technique can be integrated in a mobile ad-hoc streaming protocol, presented in Chapter 3.

Chapter 3

Video streaming protocols for ad-hoc networks

Contents

In this chapter, we shall discuss one of the main contributions of this thesis, namely, a video delivery protocol for wireless ad-hoc networks, designed with a cross-layer approach to deliver a multiple description coded video stream. Cross-layer design is a popular current research topic, where information is exchanged among different communications layers, in order to enable new and more efficient protocols to be developed [KK05, LBFV$^+$09, LBFM$^+$11, PD11]. This approach has proven to be particularly effective in video streaming applications for wireless ad-hoc networks [FLB10].

We shall first illustrate the problem of video streaming in general and over mobile networks in particular, in Sec. 3.1 and 3.2 respectively. Then, in Sec 3.3, we shall give a review of the state-of-the-art of the available solutions, from both an application and a network point of view. We shall observe how these solutions are often not yet satisfactory, as they fail to exploit at the same time the specific features of video streaming and those of the mobile environment. In Sec. 3.4, we shall investigate how a more efficient solution

can be provided with a cross-layer approach and present our own contribution to the field: the ABCD protocol. Finally, in Sec. 3.5, we present an experimental validation of our protocol, analysing a wide range of relevant features under a variety of conditions.

3.1 Video streaming

During the past decade, there has been a growing demand for applications that offer transmission of video content over unreliable channels such as wireless networks and the Internet, and this demand is expected to continue growing [LWP+07]. Even though these applications are nowadays commonplace, and the technology involved has greatly advanced during the past few years, a great deal of further improvement is still needed before Internet and mobile broadcasting can consolidate their status as viable alternatives to traditional television broadcasting modes. These improvements involve many technical challenges both in the domain of video coding and of networking [RVdSC01].

Video streaming is a video content transmission paradigm wherein the content is continuously received and presented to the end-user, while being delivered by the content provider. It is from this continuous playback, which distinguishes streaming from download-and-play schemes, that most of the design challenges arise. Streaming differs in general from video-telephony and videoconferencing services, which also transmit in real-time, as in video streaming the content might be encoded without the benefit of knowing the state of the channel during transmission [CM06]. Furthermore, streaming is distinguished by its ability to store media data encoded offline, and by its tolerance to a longer playback delay. If by its nature the content cannot be pre-encoded and stored on a server, such as a sport or news event to be broadcast live, the service is usually referred to as *live streaming*.

In a media streaming system, a media server packetises the content into data units in order to transmit them, on demand or following a time schedule, for playback in real time. The clients bufferise the data they receive and begin the playback after a short delay. This delay is introduced to let the client collect an amount of media sufficient to prevent the effect of packet losses and delays from constantly interrupting the play-out of the stream. The choice of the right buffering delay involves a trade-off between the reliability of uninterrupted play-out, the memory resources of the player application, and end-to-end delay itself. It is usually fixed and not depending on the length of the presentation. In most commercial media streaming services[1], which typically operate at 50–1500 kbps, the optimal balance is found in the range 2–15 s [WSBL03]. If data packets are lost, according to the specific application, the data unit may or may not be sent again in another packet [KSG04, CM06]. If the application chooses not to send the packet again, then the end-user may try to reduce the impact of the loss using an *error concealment* strategy, which can be based on redundancy added at the server for this purpose (*forward error con-*

[1] Such as Real Player, available at `http://www.realnetworks.com`, and Windows Media Player, available at `http://www.windows-media.com`.

Figure 3.1: A simple example of video streaming architecture. The video content, may it be available as a live feed, a stored file, or a satellite communication, is processed and stored on a server, which transmits it through the Internet to a multitude of heterogeneous clients.

cealment), on characteristics of video signals itself (*error concealment by post-processing*), or on a dialogue between the server and end-user (*interactive error concealment*) [WZ98].

The inherent requirement of continuous delivery translates to a continuous connectivity requirement between the media server and clients. In many applications, it would be also desirable to have a graceful degradation of received media quality as network environment resources change over time [MLP+03, MCH+07] (see also Chapter 2).

In the following section, we shall discuss a class of mobile mobile environments particularly interesting for video streaming, *i.e.*, namely mobile ad-hoc environments, detailing both the desirable properties and the challenges they present for a video streaming application.

3.2 Wireless ad-hoc architectures

A *mobile ad-hoc network*, or MANET, is a dynamic, self-organising, infrastructure-less network of mobile devices, interconnected by wireless link in a mesh topology [FJL00].

The devices, or *nodes*, of the ad-hoc network move freely and independently in all directions; consequently, the channel conditions of the links and the link themselves may change frequently. Also, individual nodes may connect and disconnect asynchronously, *i.e.*, without previous notice, and therefore their presence cannot be relied upon (*churn problem*). Each node of the network may initiate a communication with any other node. In order to deliver these data packets, other nodes in the network – even if unrelated to the communication – may have to participate in the transmission by forwarding the traffic. In other words, all nodes may at any time be *source*, *sink*, and *router* for a data stream. A simple example of mobile ad-hoc network is depicted in Fig. 3.2.

Ad-hoc networks may be connected to the larger Internet, usually through a *gateway* node, connected via a wired connection, operating as an Internet connection sharing device. However, an ad-hoc network more commonly operates by itself, in order to support a specific application involving the nodes composing it. In fact, the Latin phrase AD HOC literally means "for this", and in this context it is understood as "for this purpose". This paradigm is also defined as *application-driven networking*. A practically relevant case of application-driven networking is a network created by a group of people that use wireless computing to accomplish a collaborative task, which Feeney *et al.* call *spontaneous network* [FAW01]. A spontaneous network reflects the fact that the nodes have chosen to cooperate for some purpose, and can therefore be leveraged into providing full cooperation to the network initialisation and management. Another characteristic of spontaneous

networks is to have a limited extension in both space and time, and that their population – although in principle unpredictable and dynamic – is expected to present relatively infrequent and minor changes over its lifetime [FAW01].

MANETs offer a set of properties – flexibility, ease of deployment, robustness, *etc.*– that makes them applicable in environments without pre-existing communication infrastructure and where deploying it would be too expensive, too long, or simply not feasible. Common examples include students participating in an interactive lecture, business associates sharing information in a meeting, soldiers relaying situational awareness on the battlefield, or emergency disaster-relief personnel coordinating efforts after a hurricane or earthquake [RT99, Ger05]. Due to this possibility to create a network without infrastructure, ad-hoc networking has been defined as "the art of networking without a network" [FJL00].

Since in ad-hoc networks nodes have to operate as routers, the pivotal challenge is to define a protocol that controls the policy by which nodes decide how to route the packets. The main issue is that nodes are unfamiliar with the topology of their networks, and thus have to announce their presence and in turn listen for announcements broadcast by their neighbours, learning about nearby nodes and how to reach them, and possibly announcing to others how to reach them.

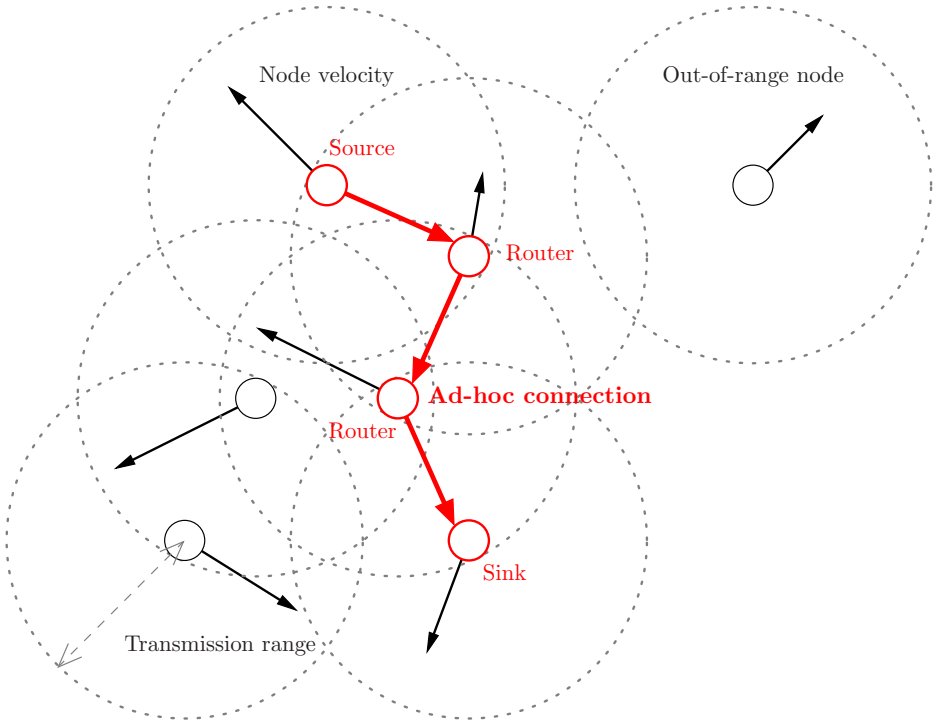


Figure 3.2: A simple example of mobile ad-hoc network: nodes have velocities (black arrows) and a multi-hop ad-hoc transmission established (red arrows).

Such protocols have responsibility of routing packets delivery – including routing through intermediate routers — which is normally implemented at network level (ISO

Protocol	Proposed by	Year	Type	Based on	Advantages	Drawbacks
DSDV	Perkins and Bhagwat	1994	Proactive	Distance vectors	First to solve the routing loop problem.	High overhead. Not suitable for highly dynamic networks.
DSR	Broch, Johnson, and Maltz	1994	Reactive	Source routing	Highly scalable. No routing information stored at intermediate nodes.	High overhead, proportional to the path length. Not suitable for highly dynamic networks.
TORA	Park	1997	Proactive	Source rooted trees	Reduced overhead. Localised reactions.	Does not provide the shortest path.
AODV	Perkins, Belding-Royer, and Das	1999	Reactive	Hybrid	Capable of both unicast and multicast. Low connection setup delay.	Overhead due to the periodic beacon. Stale entries in intermediate nodes can lead to inconsistent routes.
OLSR	Jacquet, Muhlethaler, *et al.*	2001	Proactive	Multi-point relay	Stable and fast. Reduced control overhead.	Unable to track fast moving nodes.

Table 3.1: Comparative overview of routing protocol for ad-hoc networks.

OSI Model Level 3 [Tan03, Ch. 5]), *i.e.*, on top of the Data Link Layer. Specifically, routing protocols are built on top of Data Link's upper sub-layer, known as Medium Access Control, or *MAC*.

The most commonly used MAC Layer for wireless networks is by far the standard IEEE 802.11 and its subsequent amendments, collectively known as 802.11x [BCG05]. The growth of laptops, and more recently personal data assistants and smartphones, implementing 802.11x wireless networking has made MANETs a popular research topic for the last few years. Many works exist evaluating ad-hoc routing protocols and their performances assuming varying mobility models within a bounded space, usually with all nodes within a few hops of each other. Protocols performance is usually measured in terms of packet loss rate, protocol overhead, end-to-end delay, network throughput, *etc.* [BMJ+98, RT99, SGF02, HT09]. In the following we shall give a short overview, in chronological order, of the most used ad-hoc routing protocols, summarised in Tab. 3.1.

The *Destination-Sequenced Distance Vector* Algorithm (DSDV) [PB94] is a hop-by-hop distance vector routing protocol wherein each node periodically broadcasts a routing update. It is based on the classical Bellman-Ford routing algorithm for single-source

shortest paths in a weighted directed graph [Bel58]. DSDV is a table-driven algorithm; each entry in the routing table contains a sequence number, which is even if a direct link is present and odd otherwise. The number is generated by the destination, and the sender needs to communicate the next update with this number. Routing information is distributed between nodes by sending full dumps infrequently and smaller incremental updates more frequently. However, the amount of information that needs to be exchanged makes DSDV unsuitable for networks where topology changes occur frequently. The main contribution of this algorithm was to solve the routing loop problem that traditional distance vector protocols (such as the Bellman-Ford itself) did not.

The *Dynamic Source Routing* Algorithm [JMB01], or DSR, is a reactive source-routing protocol for wireless mesh networks. A reactive routing protocol establishes a route to a destination only on-demand; in contrast, most routing protocols of the Internet are proactive, *i.e.*, they find routing paths independently of the usage of the paths. DSR follows the *source routing* paradigm, a technique in which the sender of a packet – instead of relying on the routing tables at each intermediate node – can partially or completely specify the route that the packet should take through the network, based on the packet's destination. Source routing requires that the complete, ordered address list of the nodes the packet will traverse is carried in the header of the packet itself. In large networks, this may result in high overhead; however, the accumulated path information can be cached by intermediate nodes, so that the learnt paths can be used to route their own packets. Information about the best path to one's destination is gathered in a route discovery phase, based on flooding. The main benefit of source routing is that intermediate nodes need not maintain up-to-date route information, since the packets themselves already contain all the routing decisions. Moreover, the on-demand nature of the protocol eliminates the need for the periodic route advertisement and neighbour detection packets present in other protocols. However, since the routing is decided at the source and never updated along the path, a source with out-of-date information (due to a change in topology between discovery phases) can lead to inconsistent routes.

The *Temporally-Ordered Routing Algorithm* (TORA) [PC97] is a highly adaptive, loop-free, distributed routing algorithm that aims at achieving a high degree of scalability using a "flat" (*i.e.*, non-hierarchical) approach. It is based on the classical "link reversal" routing algorithm for loop-free routes in networks with frequently changing topology [GB81]. TORA builds and maintains a directed acyclic graph rooted at each destination, localising algorithmic reaction to topological changes as much as possible, *i.e.*, avoiding far-reaching control messages. TORA is particularly suited to operate in highly dynamic mobile networking environments as it is a self-stabilising protocol, *i.e.*, it is self-healing in the face of nodes or links failures. However, in order to achieve this, TORA does not use a shortest path solution, which is unusual for routing algorithms of this type. The key contribution of TORA is the localisation of control messages to a very small set of nodes near the occurrence of a topological change.

The *Ad-Hoc On-Demand Distance Vector* [PR99], or AODV, is, as the name indicates, a reactive distance-vector routing protocol. It is essentially based on a combination of both DSDV and DSR. It is similar to DSR in that it borrows the basic mechanism of forming routes on-demand when a transmitting node requests one, but integrates it with the hop-by-hop routing, sequence numbers, and periodic beacons from DSDV in order to assure loop avoidance. Capable of both unicast and multicast routing, AODV provides a synthesis of the benefits of other protocols mentioned above and is nowadays, together with DSR, one of the most commonly used.

The *Optimised Link-State Routing* [JMC+01], or OLSR, is a proactive table based routing. The key idea of OLSR is to define, for each node, a set of *multi-point relays* (MPRs). Rather than declaring all its links, a node only declares only the set of its MPRs, minimising the control traffic. OLSR provides two main functionalities: Neighbour Discovery, through which each node detects the neighbours with which it has a direct link, and Topology Dissemination, with which the node exchange topology control messages with its randomly selected MPRs and maintains topological information about the network. Being a proactive protocol, OLSR performs well in terms of latency to find a route. Moreover, the use of MPRs allows to significantly reduce the number of control messages and retransmission. However, in networks where nodes show a fast mobility, OLSR is unable to track rapid changes of topology, which results in a dramatic drop in performance. To overcome this limitation, Benzaid *et al.* [BMAA02] have proposed an extension of OLSR, called Fast-OLSR, designed to meet the need for fast mobility. Based on the OLSR protocol, Badis, Munaretto *et al.* [MBAAP03, BMAAP04, BAA08] have proposed an innovative link-state QoS routing protocol for ad-hoc networks known as QOLSR. This protocol uses bandwidth and delay heuristic measures in order to select the MPRs, this improving the optimal path in terms of QoS requirements.

Most of the available routing protocols do not operate efficiently with networks of more than a few hundred nodes. Due to the growing importance of ad-hoc paradigm in applications involving a large population of mobile nodes, a great attention has been more recently devoted to ad-hoc routing protocols with satisfactory scalability requirements, such as the *Augmented Tree-based Routing* Algorithm [CFP07], or ATR. ATR utilises an augmented tree-based address space structure and hierarchical multi-path routing in order to solve the scalability problem and to gain good resilience against node failure and link congestion.

Most of the routing protocol mentioned above make large use of broadcast in order to spread the information about possible routes to the nodes of the network; is therefore natural, considering its wide use as a building block for other protocols, the interest in a technique that efficiently delivers a packet from one node to all other nodes in the network. Furthermore, broadcast also has an interest *per se* since many applications, among which video streaming, require one-to-many communication.

In their comprehensive overview on broadcast schemes for ad-hoc networks, Williams

and Camp [WC02] propose to classify the most commonly used techniques to perform multi-hop broadcast, also referred to as *flooding*, into four categories:

- Simple Flooding;
- Probabilistic Flooding;
- Area Based Flooding;
- Neighbour-Knowledge Based Flooding.

Simple Flooding is a widely used scheme for delivering a message to every part of a MANET in order to diffuse link state information for routing purposes. The algorithm starts with the source broadcasting a message to all its neighbours; each of those neighbours, in their turn, re-broadcast the message exactly once, after a random assessment delay (RAD) [MOKM08]. Duplicate messages are dropped without re-broadcasting. The algorithm continues until, eventually, all reachable nodes have received the message. The popularity of this technique has several reasons. First, it is very simple to implement even in devices with little computational power on-board. Also, it is usually very fast, since the message is sent through all possible paths, thus it is sent through the shortest path as well. However, even though this technique may be very effective for diffusing link state information, which has a low bitrate and loose timing constraints, it is dangerously wasteful in terms of bandwidth in the case of general purpose broadcast, *e.g.*, video streaming: the increased load placed upon the network affects both the total available bandwidth and other nodes' resources (*e.g.*, battery power). Also, since more collisions occur, even in a connected network it is not guaranteed to reach all nodes. Furthermore, it is quite sensible to the choice of the RAD. Too short a RAD will affect reliability (because of collisions); too long, it will affect latency (because of the increased average delay). Simple flooding has been proposed as a scheme to achieve multi-hop broadcast and multicast in highly dynamic wireless networks [HOTV99], and there exists an IETF Internet Draft proposing the use of simple flooding for broadcasting and multicasting in ad-hoc networks characterised by low node densities and high mobility [JHMJ01].

Probabilistic Flooding is a technique similar to Simple Flooding proposed to mitigate the aforementioned problems. In this scheme, nodes only re-broadcast with a predetermined probability. It does not require major modifications from Simple Flooding, and in dense networks, where multiple nodes share similar transmission ranges, randomly having some nodes not re-broadcasting saves node and network resources without harming reliability. Collision probability is also reduced, but in sparse networks, where there is much less shared coverage, nodes could not receive all the messages, unless the probability is high. This technique is also unfit for broadcasting a video stream, since there is no control over the path between a node and the source, hence on the QoE of the receivers. When the retransmission probability is set to 100 %, this scheme is identical to simple flooding [PL01].

A different approach is provided by *Area Based Flooding*, wherein the nodes only consider the area covered by a transmission, rather then whether a node exists within that area. If nodes are somehow able to estimate their relative positions, using the signal strength or a GPS, the can infer how much additional area can be covered by their retransmission. Even though they perform generally better than both Simple and Probabilistic Flooding, Area Based methods incur in a disproportionately increased number of retransmissions per packet as the number of source packets or the number of nodes in the network increase.

In all the broadcast schemes discussed so far, the nodes of the network retain barely any information on the topology to decide whether or not to retransmit a packet. *Neighbour-Knowledge Based Flooding* differs in this respect, as each node is required to keep record of its neighbours, usually within one or two hops of radius, and the identity of the nodes it has received packets from. This allows it to determine whether it would reach additional nodes by retransmitting the packet. Neighbour-Knowledge Based methods generally outperform all other schemes here described, and closely approach the theoretical bound. The main drawbacks of these schemes are the higher memory and computational resources required to the mobile nodes, and the need to keep the neighbours' tables up-to-date. If the network changes very dynamically, this could require a significant overhead.

Nevertheless, we found that the Neighbour-Knowledge schemes present the most interesting properties in terms of overall properties with respect to to the design of a video streaming application, and that is the approach we have followed in the design of our protocol. In Sec. 3.4, we shall see in detail how the overhead can be kept low with a wise use of the broadcast medium.

3.3 An overview of streaming protocols for MANETs

In this section, we shall give an overview of the most popular schemes employed in the transmission of video streams over wireless networks, with particular reference to ad-hoc networks. This overview is by no means exhaustive, but it tries to capture the general trends in the design of streaming protocol for ad-hoc networks.

Video streaming in a wireless environments is a challenging task, which can be successfully accomplished only if the coding scheme used to compress the content is both efficient, in terms of rate-distortion, and network-friendly [SHW03]. The H.264/AVC standard and its extensions (see Sec. 1.3.2) offer both of these characteristics, and provide a set of tools that allow to adapt the encoding to the transmission of the coded video data over any kind of network, including wireless [WSJ+03]. These tools include, but are not limited to, a flexible framework that allows the implementation of RD-optimised packet scheduling strategies, and error resilience tools for communication over lossy networks.

One possible solution for video transmission over a multi-hop wireless network is using the scalability features offered by the SVC extension of H.264. However, since in scalable coding the different layers present hierarchical dependencies, a suitable strategy of *Unequal*

Error Protection (UEP) has to be provided. Majumda *et al.* [MSK+02] have proposed such a strategy for both unicast and multicast transmission on wireless networks that minimises an assigned rate-distortion cost function, a solution that efficiently combines scalable coding with forward error correction. However, their solution only applies to video multicast in the last hop, *i.e.*, there is only one hop between the source and the destination.

A solution for the multi-hop scenario has been proposed by Cataldi *et al.* [CGT+10], who suggested to use scalable video coding combined with *rate-less codes*, also known as *fountain codes*. Rate-less codes are such that a potentially limitless sequence of codewords can be generated from a given set of source symbols, and the original source symbols can be recovered from any subset of codewords of size equal to or only slightly larger than the number of source symbols [Lub02]. In particular, they propose a low-complexity class of rate-less codes based on a sliding-window approach applied to Raptor codes. Raptor codes are the first known class of rate-less codes with linear encoding and decoding time [Sho06]. These codes have several properties useful for video applications, and provide better performance than classical rate-less codes. This system has shown a good end-to-end quality and robustness towards fluctuations in the packet loss rate.

Another approach, which delivers from the need to provide unequal error protection, is using multiple description coding (see Chapter 2). This approach, which had already be proven very efficient by Gogate *et al.* [GCPW02] in the transmission of still images over ad-hoc networks, has been investigated by Lin, Mao *et al.* [LWMP02, MLP+03] for video delivery. In their comparative analysis of multipath video streaming techniques over MANETs, they proved that the MDC approach is better suited for ad-hoc environments than scalable video coding. However, in order for MDC to be effective, a suitable routing protocol has to ensure that multiple (ideally, edge-disjoint) paths exist between the source and each receiver.

A class of routing schemes that ensure this property are the *multiple tree* (also *multi-tree*) construction protocols. The basic idea of these schemes is to send each description over a different tree, constructed to be at least partially disjoint with each other so as to increase robustness to loss and other transmission degradations. In order to prove this, Wei *et al.* [WZ07] have proposed two different multi-tree construction schemes, with explicit goal of routing an MD coded video stream. The first scheme constructs two disjoint trees in a serial and distributed fashion, and achieves reasonable tree connectivity while maintaining the trees completely disjoint. The second one, designed to reduce the control overhead and the construction delay, is a parallel protocol that generates nearly-disjoint trees. Their results show that the achieved video quality for either scheme is significantly higher than that of single tree multicast, while it presents only a slightly higher control overhead and a similar forwarding efficiency. While this scheme is aimed only at maximising the disjointness of the trees, our own contribution on this field, presented in Sec. 3.4, generates a multi-tree where the degree of disjointness can be tuned, using a

weighted optimisation function, with other relevant network-related measures, such as the hop count and the number of retransmitting nodes.

While Wei *et al.*, and most of the other works on video transmission over ad-hoc networks existing at the time of their proposal, focused only on network-oriented metrics, such as throughput, delay and packet loss rate, Wu *et al.* [WCWK10] proposed an application-centric approach for real-time video transmission in ad-hoc networks, where expected video distortion is adopted as the routing metric. This is done using a quality-driven cross-layer optimisation that enhances the flexibility and robustness of the routing by jointly optimising the path selection and the video coding, under a given video playback delay constraint. Their results, both theoretical and experimental, demonstrate that this approach outperforms the existing network-centric routing approaches. Although highly interesting, the approach of Wu *et al.* applies to directed acyclic graphs rather than real networks, and network-related problems are out of their scope.

Another important challenge of multi-hop video delivery in ad-hoc network, especially relevant when multi-path routing is used, is to limit the congestion generated by the transmitting nodes. This is a complicate issue, as the transmission policy of a single node streaming a video flow may impact the overall network congestion.

3.4 Cross-layer design of a streaming protocol for MANETs

The rest of this chapter is entirely dedicated to our own original contributions. In particular, in this section, we address the design of a novel network protocol, which has been proposed under the name of *ABCD*: *A Broadcast Content Delivery Protocol* [GC11], able to deliver a video stream emitted by a single source to a multitude of nodes self-organised in a MANET. The video stream is assumed to be composed of a set of mutually refinable sub-streams, *e.g.*, to be encoded in multiple descriptions. This protocol covers the data-link, network, and application layers of a multicast video communication in a cross-layer design. Conversely, the protocol does not deal directly with physical issues, and relies on 802.11 to deal with them instead. However, these issues have been taken into account in its validation (see Sec. 3.5). An experimental validation of this contribution will be given in Sec. 3.5.

In the design of the protocol, we make the following assumptions about the application scenario:

- all nodes share a common wireless channel;
- a message transmitted by a node can be received by all other nodes within its transmission area;
- all links between pairs of nodes are bidirectional;
- messages may be lost, duplicated, and delayed;

- all nodes may participate in forwarding the video stream, *i.e.*, act as routers.

On the other hand, we make no prior assumptions on the number, the geographical distribution, the reliability, and the mobility of nodes. Since nodes are free to move independently in any direction, they might change their links to other nodes quite frequently. These simplifying assumptions are commonplace in the design of broadcast protocols for ad-hoc networks [PL01], as they are either realistic in themselves, or it is easy to modify the design to encompass more realistic assumptions.

One of the most challenging tasks in this design is allowing each node to continuously maintain enough information about the network topology to be able to implement a suitable strategy with respect to the relay of the stream, but with the minimum possible overhead in terms of control messages. Therefore, the set of links that the sub-streams traverse before reaching each destination, commonly referred to as *overlay network*, must be efficient in the sense that the paths from the source to the sink nodes must be as short as possible and the load injected into the network must be as small as possible, *i.e.*, the least possible number of nodes actually relay the stream as they receive it. The rationale behind this goal is two-fold: firstly, reducing the number of copies of the stream injected into the network reduces both congestion and collision probability; and secondly, the nodes typically have limited resources in terms of up-link bandwith and battery power, and must therefore be forced to exploit these resources as seldom as possible.

Being the wireless medium inherently broadcast, the most efficient way to design a multi-hop broadcast seems to be through single-hop broadcast communication, wherein a single relay node can satisfy any number of nodes within its transmission area as long as the packets it sends do not collide. However, the IEEE 802.11x set of standards commonly known by the brand Wi-Fi, which governs wireless networking, was mainly designed for one-to-one communications on *infrastructure networks*, *i.e.*, networks that require the use of an infrastructure device, such as access points or base stations, in order to facilitate communication between client devices. Conversely, one-to-many communication on ad-hoc networks, although supported by the standards, is known to be unreliable and inefficient [NTCS99]. Whereas this is normally not considered to be a major concern for other protocols, we want to make an extensive use of broadcast; therefore, we designed a modified version of the standard 802.11x MAC layer, that provides, for our particular application, a more reliable broadcast communication.

3.4.1 MAC-layer modifications for reliable broadcast

In order to enable a quick discovery of the topology, thus building an efficient overlay with a small control overhead, we want the nodes of our network to be able to read packets not directly addressed to them. The first step in the design of our protocol is therefore extending the standard MAC layer with functionalities that allow an efficient use of the broadcast medium in one-to-many communications over ad-hoc networks. We achieved

this goal implementing a form of *reliable broadcast* that is immune to reservation deadlocks and robust to hidden nodes [MKK01].

There exist in the literature several techniques that offer reliable broadcast schemes for ad-hoc networks [SM00, YYRZ11]. However, the most interesting solutions are the ones based on extending the *Carrier Sense Multiple Access with Collision Avoidance* (CSMA/CA).

CSMA/CA is a channel access method that prevents wireless transmission of a node if another node is transmitting, thus reducing the probability of collision due to the use of a random truncated binary exponential back-off time. This protocol was designed under the assumption that all nodes have the same transmission range. In almost all implementations of IEEE 802.11, unicast communications also implement the optional RTS/CTS handshake, in order to cope with the collisions introduced by the *hidden node* problem [Kar90].

When using the RTS/CTS handshake, also called *Virtual Carrier Sensing*, a node that intends to send data must first initiate the process by sending a *Request to Send* (RTS); the destination node should reply with a *Clear To Send* (CTS). In lack of a CTS, the sender repeats the RTS until it receives a CTS, up to a maximum number of times, referred to as *retry limit*. In situation where a large interference range is expected, some schemes use a Conservative CTS Reply (CCR), wherein a node only replies with a CTS if the receiving power of the corresponding RTS is larger than a certain threshold, even if the RTS is received successfully. With a proper choice of the threshold, all interference area is covered by the RTS/CTS handshake, therefore collisions are totally eliminated, at the price of a reduced throughput. In practice, the same result can be achieved if the effective data transmission range is reduced with respect to the RTS/CTS transmission range [XGB02]. Any other node receiving the RTS or CTS must refrain from sending data for a given time, included in both the RTS and the CTS, thus ensuring that collisions are avoided. Moreover, in most implementation, once the recipient has successfully received the packet, it also sends an *Acknowledgement* (ACK) to the sender, in order to confirm the reception. In lack of an ACK within a given deadline, the sender will retransmit the packet, up to a fixed number of times.

An RTS/CTS handshake only occurs when the size of the packet to be sent exceeds an *RTS/CTS size threshold* (which the standard does not fix) that is typically in the range $0 \sim 2347$ bytes. The threshold is used as the RTS/CTS handshake has an overhead in terms of expected duration of the data transmission.

Reliable broadcast schemes implemented using RTS/CTS are the most interesting ones, as they require almost no change of the MAC layer [Tou98]. If the packet headers remain unchanged, and the modifications are transparent to the receiver, then backward compatibility is ensured.

In our scheme, depicted in Fig. 3.3, we set the destination field of every IEEE 802.11x transmission unit to its broadcast address, which makes any *neighbour* of the sender receive

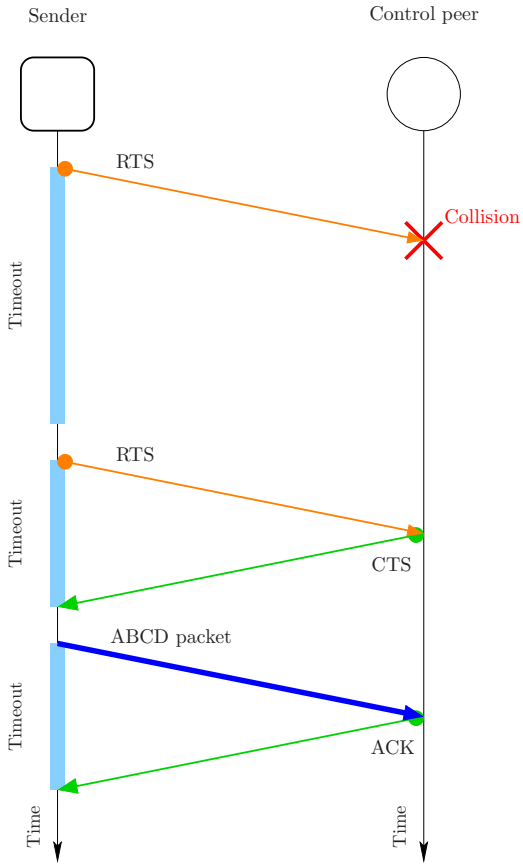


Figure 3.3: Reliable broadcast scheme. The ABCD packet is sent with its destination set to the broadcast address.

its packets. We define as neighbours of a node n all the nodes whose packets can be received by n; if we assume a symmetric channel, the node's neighbourhood is the set of nodes within its transmission area[2]. Setting all destination fields to the broadcast address can be achieved manipulating the *ARP Cache Tables*, where the associations between network addresses and data-link addresses are locally stored. Tampering with the ARP Cache Tables (in jargon, "Cache Poisoning") is a well-known technique, often used by malicious users in order to perform an *ARP spoofing attack* [NNGS10], that we use here for legitimate purposes.

Even though packets are sent in broadcast, we always enforce an RTS/CTS/ACK handshake (*i.e.*, the RTS/CTS threshold is 0). The handshake is performed with a neighbour designated accordingly with the application logic and referred to as *control peer*. The choice of the control peer is discussed in detail in Sec. 3.4.3. The handshake is performed exactly as if the control peer were the recipient of a unicast packet, including the confirmation ACK, with the only difference that, being the destination address set to the broadcast address, the packet is actually received by any neighbour of the sender.

[2]Do notice that the neighbourhood of a node may change over time due to mobility and churn.

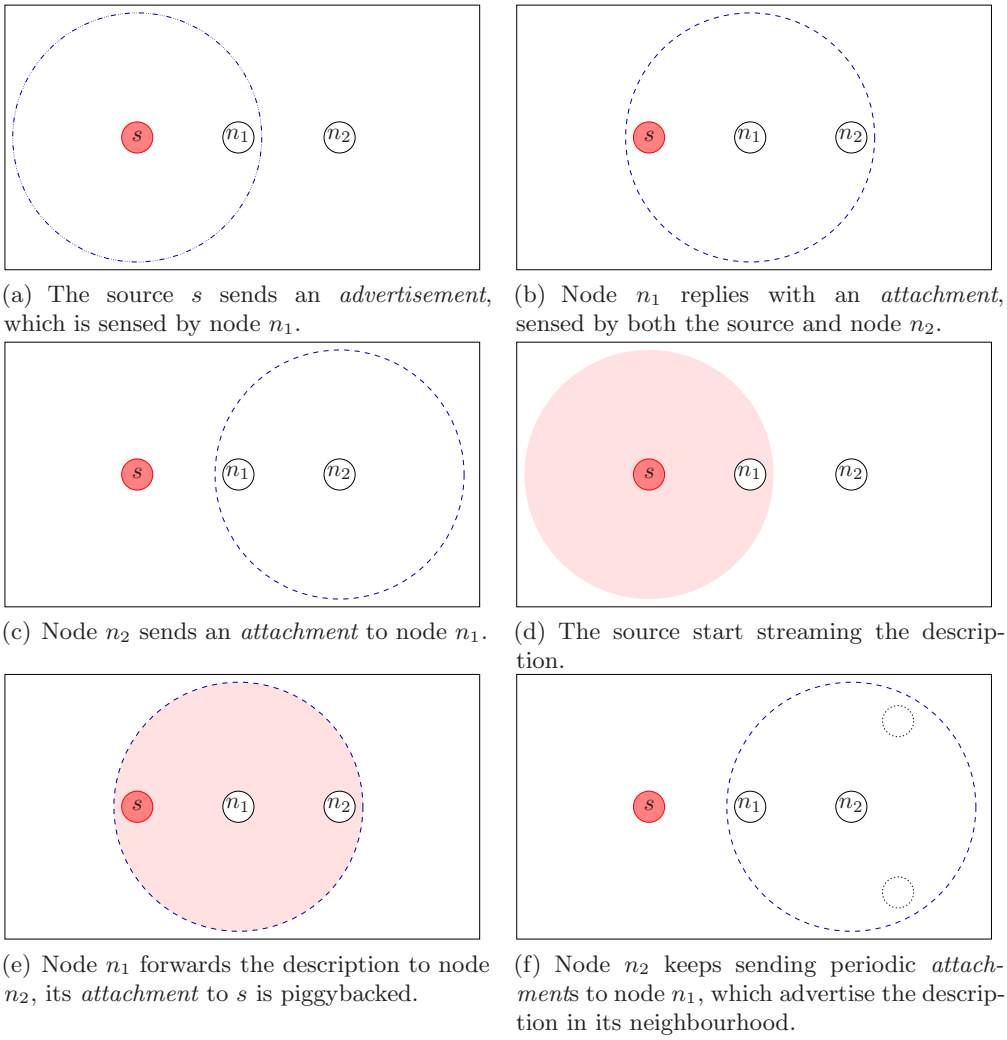
Even though this technique does not grant that *all* neighbours successfully receive the packet (as the ACK is sent by the control peer depending only on *its* state), thanks to the range extension given by the CCR, and with a proper choice of the control peer, medium condition among the receivers are more correlated, making acknowledgement by the control peer a quite reliable estimation of most of the intended receivers. As a result, experimental evidence suggests that the incidence of collisions if such a strategy is implemented is reduced to the point of being negligible in our simulations up to a density of 20 nodes per neighbourhood, which is about three times the optimal density value in terms of normalised network throughput [KS78].

Such a reduction of the collision probability allows our protocol to perform better than the standard IEEE 802.11 in the rate *vs.* diffusion area trade-off typical of self-limiting multi-hop broadcast in wireless networks [EFLBS07], where a greater number of retransmissions increments the spread of the content, but reduces its throughput because of the limited channel capacity. However, this technique can only achieve this result providing that a suitable control peer is chosen. Our approach being cross-layer, we can select a control peer by exploiting information from layers other than the MAC. Namely, as we shall illustrate in detail in Sec. 3.4.3, we exploit routing information in order to choose the most suited control peer for a transmission. However, this could not be efficiently achieved in a layered protocol so, while extremely useful in our case, this kind of reservation technique could not perform well in all application scenarios, and is not suitable for adoption in the IEEE 802.11x standard family, wherein actions of nodes are in general not coordinated, and a strictly layered approach is maintained.

3.4.2 Overlay creation and maintenance

Once a reliable channel for broadcasting is available, we are able to efficiently build our overlay network on top of it. In our scenario, building an overlay for a mobile ad-hoc network means to define the forwarding strategies of each node, *i.e.*, deciding which nodes will relay which descriptions, if any. The routes that the video packets sent by the source will follow will form a superposition of trees, one for each description; this kind of overlay is referred to in the literature as *multi-tree*.

Multi-tree generating schemes are conceptually close the wider category of schemes based on the *connected dominating set* (CDS). A subset of nodes in a network is a connected dominating set if each node not in the subset is adjacent to at least one node in the subset, and the sub-network induced by the subset is connected. CDS-based schemes attempt to minimise the amount of broadcast traffic by constructing a sub-network called virtual backbone that approximates the minimal CDS [DB97], and allowing only the nodes of the backbone to re-broadcast the traffic. In the case of multi-tree based schemes, the backbone is composed of the inner nodes of the trees (*i.e.*, any node other than the leaves) that, unlike the general case, re-broadcast the traffic unidirectionally.



(a) The source s sends an *advertisement*, which is sensed by node n_1.

(b) Node n_1 replies with an *attachment*, sensed by both the source and node n_2.

(c) Node n_2 sends an *attachment* to node n_1.

(d) The source start streaming the description.

(e) Node n_1 forwards the description to node n_2, its *attachment* to s is piggybacked.

(f) Node n_2 keeps sending periodic *attachments* to node n_1, which advertise the description in its neighbourhood.

Figure 3.4: Creation of the overlay in ABCD protocol.

Figure 3.4 shows how the multi-tree construction proceeds in ABCD. Notice that, even though for the sake of clarity we represent transmission area as circular, this needs not to be the case. All it is required is that if node n_1 is able to receive packets from a node n_2, then node n_2 is able to receive packets from node n_1.

As soon as the source has a video content to deliver, it sends an *advertisement* message for each description of the content (Fig. 3.4(a)).

The source's neighbours, receiving an advertisement message, reply with an *attachment* message. Attachment messages are interpreted as a subscription to the description (Fig 3.4(b)); as soon as the description has at least one subscriber, the source *activates*, *i.e.*, it starts broadcasting the video packets for that description (Fig. 3.4(c)). The subscribing nodes – that we define as the source's *children* – keep sending periodical attachment messages to their *parent* (the source) in order to keep it active: with the help of properly set timeouts, the source would deactivate when nodes turn off or get out-of-range.

Each node that is not in the source's transmission area, but that is in the neighbourhood of at least one of its children, becomes aware of the availability of the descriptions since it receives its *peers*' attachment messages (a peer is any node other than the source). It independently chooses one of these peers as its parent and sends it an attachment message (Fig. 3.4(d)); the node thus chosen will activate, starting to relay the video packets it receives for the description it is active on. Since it now has to send video packets, and – due to symmetric channels – its parent would receive a copy of them anyway, an active peer stops sending separate attachment messages, piggybacking them in the video packets instead (Fig 3.4(e)).

The attachment messages sent by the newly subscribed node will now advertise the description within their neighbourhood, generating other subscriptions (Fig 3.4(f)); this process is reiterated, independently on each description, until all nodes have one parent per description[3].

In conclusion, a node becomes aware of a path to the video resource as it intercepts an attachment message. Quite often, it actually intercepts attachments from multiple nodes (either piggybacked in video packets from an active node or standalone from a subscribed node), and has to decide which peer provides the best path. Even if the node already has a parent, it could become aware of a better path, created by the connection or the mobility of a peer. Therefore, nodes need a metric for the paths through their neighbours, in order to choose the best one. To this end, we designed the following metric, which takes into account the above discussed objectives, that each node evaluates over each candidate parent p (*i.e.*, each node of whom it intercepted an attachment message), then selecting the one that minimises its value:

$$J(p) = \omega_{\mathrm{h}} h(p) + \omega_{\mathrm{a}} a(p) + \omega_{\mathrm{d}} d(p) - \omega_{\mathrm{g}} g(p) - \omega_{\mathrm{q}} q(p), \tag{3.1}$$

where h is the hop count, *i.e.*, the number of hops from the source to the current node; a the is the activation cost, which zero if the candidate parent is already active on the current description and one otherwise; d is a diversity index, *i.e.*, the number of descriptions other than the current one for which the current node is already subscribed to the candidate parent; g is the subscription group size, *i.e.*, the number of peers subscribed to the same candidate parent as the current node; and q is a link quality index, *i.e.*, the exponential moving average of the *signal-to-noise-plus-interference ratio* (SNIR) of the link between the current node and its candidate parent. The ω values are a set of positive real weights used to fine-tune the path choice, constructing a single aggregate objective function as a weighted linear sum of objectives [GCPP11].

The minimisation of the hop count to the source must have the highest priority over all other terms, *i.e.*, hop count minimisation should always be preferred over all other

[3] As we shall see in the following, depending on the parametrisation, they could actually have the same node as parent on all the descriptions, but in the general case they do not.

Symbol	Name	Description
h	Hop Count	Number of hops from the source to the current node.
a	Activation Cost	Zero if the candidate parent is already active on the current description, one otherwise.
d	Diversity Index	Number of descriptions for which the current node is subscribed to a candidate parent (except current description).
g	Subscription Group Size	Number of peers subscribed to the candidate parent for the current description.
q	Link Quality Index	Estimation of the channel quality between the current node and the candidate parent.

Table 3.2: Name and description of the values used in function (3.1).

parameters in the function, since it assures that the overlay graph is acyclic (*i.e.*, a proper tree); also, it is beneficial to the minimisation of the end-to-end delay. This can be done with a proper choice of the value of ω_{h} with respect to the other weights. As a result, in an overlay generated by the ABCD protocol, a node cannot have, in a steady state, a peer in its neighbourhood whose hop count is smaller than that of its current parent, since in that case it would simply switch parent in order to prevent loops.

The number of active nodes per neighbourhood is also minimised, for two reasons: reducing the number of packets injected in the network, hence the congestion, and reducing the average amount of resources demanded to the nodes, which pay an energy cost to relay a description.

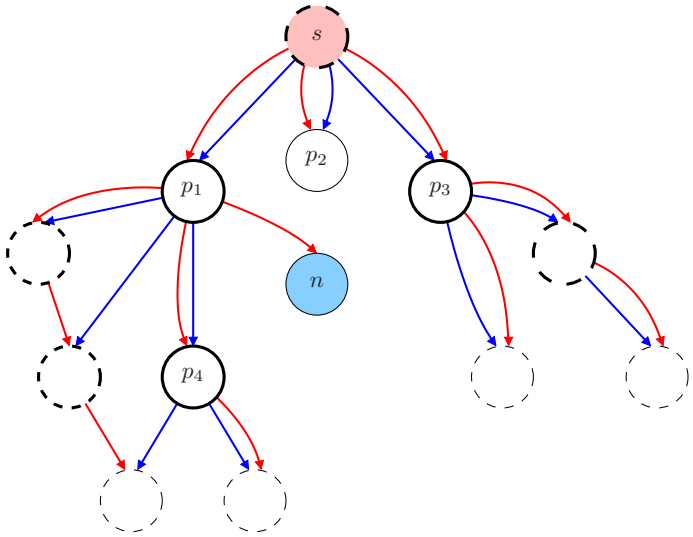
The protocol also aims, by minimising $\omega_{\mathrm{d}}d$, at ensuring path diversity among the descriptions, which is advisable for both fairness and robustness.

The term $-\omega_{\mathrm{g}}g$ in function J implies that the nodes try to maximise the number of peers subscribed to their same parent (*siblings*), in order to concentrate subscriptions on fewer active nodes, making deactivations more frequent.

Finally, the term $-\omega_{\mathrm{q}}q$ implies the maximisation of the link quality with the candidate parent. This is particularly useful in networks presenting node mobility: a reduction of the SNIR can tip a node off that its current parent is moving away from it, prompting the node to switch parent before its current one gets out-of-range.

A summary of the symbols used in function (3.1) is given it Tab. 3.2, whereas an example of the function evaluation is given is Fig. 3.5.

Here, a node n, already connected on its first description, has to choose a parent for the second one. In its neighbourhood, there are four nodes, p_1, p_2, p_3, and p_4, whose state is reported in the table. Since the minimisation of the hop-count is always preferred by the nodes, node p_4 will never be selected, as it is the only one with an hop-count of 2, whereas all other candidates have an hop-count of 1. Node p_2 will also never be selected, as it is *dominated* by p_3, in the sense that none of the parameters of p_2 evaluated in function (3.1) is better than those of p_3, while the activation cost is worse. Depending on



(a) Overlay for two descriptions, *red* and *blue*. Node n has to choose a parent for description *blue*. The neighbours of n are marked with solid border. Active nodes are marked with a thicker border.

Candidate	h	a	d	g
p_1	1	0	1	3
p_2	1	1	0	0
p_3	1	0	0	2
p_4	2	0	0	2

(b) Values used by node n in function (3.1) in order to choose a parent. (Link quality omitted.)

Figure 3.5: An example of ABCD overlay.

the values of the weights ω_{d} and ω_{g}, either p_1 or p_3 will be selected. Namely, we observe that $J(p_1) - J(p_3) = \omega_{\mathrm{d}} - \omega_{\mathrm{g}}$. Therefore, n will choose p_1 if the maximisation of the group size is considered more important in the optimisation function; conversely, p_3 will be selected if more importance is given to the path diversity between the two descriptions.

The set of weights ω used in function (3.1) can be chosen experimentally. In our implementation, we ran several simulations varying parameters such as node density, node mobility, *etc.*, and chose the set that maximised the average PSNR of the decoded video sequence over the nodes in the network. We found out that the choice of weights does not vary too much with respect to the experimental conditions, and that the resulting PSNR values are quite robust. Therefore, we estimated that updating them while the protocol is running would be an unnecessary effort.

3.4.3 Choice of the *control peer*

As mentioned in Sec. 3.4.1, in our reliable broadcast scheme, the incidence of collision is negligible if a proper choice of the control peer is made. In order to "protect" as many nodes as possible, the sender must prefer control peers with the highest number of descendants

n	$T(n)$	$P(n)$
s	12	–
p_1	6	$\frac{7}{12}$
p_2	0	$\frac{1}{12}$
p_3	3	$\frac{4}{12}$

Table 3.3: Example of probability assignment by node s according to function (3.2) for the choice of the control peer, with reference to the topology in Fig. 3.5(a).

in the multicast tree. However, we also want the control peer to change over time, in order to avoid starvation of leaves (the leaves have by definition no descendants, and would therefore never selected in a purely deterministic strategy). To achieve a trade-off of the two goals, we implemented the choice of the control peer as a biased random choice: each child n of the sender s is assigned a probability to be chosen as control peer that is increasing with to the size of the sub-tree rooted in the node n itself; however, even childless nodes are given a small non-zero probability of being selected as control peer. The exact formula for assigning the probability is the following:

$$P(n) = \frac{T(n)+1}{T(s)}, \tag{3.2}$$

where $T(x)$ denotes the number of nodes of the sub-tree rooted in a node x (node x excluded). The numerator of this formula accounts for the elements of sub-tree rooted in n, plus n itself. It is easy to show that summing $P(n)$ up over all nodes n that are children of s, we obtain 1. An example of probability assignment with reference to the topology in Fig. 3.5(a) is given in Tab. 3.3.

Our experiments show that this choice – jointly with the fact that using a Conservative CTS Reply the transmission area for RTS, CTS, and ACK is larger than the area for data, – can eliminate collisions almost entirely, except in moments of peak congestion.

3.4.4 Overhead control

Protocol overhead is the amount of control information transferred for the purpose of directing or controlling the transfer of payload information that reduces the throughput perceived by the user. In the case of ABCD, the protocol overhead amounts to the number of attachments not piggybacked in a video packet[4] that nodes send in order to keep the overlay up-to-date. The number of attachment messages injected in the network must be therefore kept as low as possible; ideally, it should be negligible with respect to the video packets.

We observe that, as topology stays constant, periodic attachments convoy barely any

[4]Even though the piggybacking of an attachment increases the size of a video packet, this is negligible with respect to the size of the payload (a few bytes of extra header). Furthermore, using our reliable broadcast technique, the bottleneck is in the channel access, rather than the amount of data sent.

information. In order to reduce this overhead, we make the attachments more frequent while there are topology changes (like after a parent switch) that need to be propagated, but we progressively reduce their frequency as long as nodes stay subscribed to the same parent. There is, however, a non-zero minimum attachment rate, which grants that the descriptions are advertised, though less frequently, in any neighbourhood.

In all the mechanisms explained so far that ABCD uses in order to build a multi-tree overlay, the behaviour of each node is completely determined by the state of its neighbours. While this is a useful simplification in phase of analysis, as it allows to exactly predict the behaviour of every node, it may prove beneficial to the reduction of the overhead to introducing some degree of controlled randomness.

In fact, observing the evolution of an overlay, we notice that in reaction to a given event, all nodes that witness that event will react in the same way and at the same time. For instance, if a new node n joins the overlay and happens to provide a very good path to the source for its neighbours, all of them will immediately send it an attachment in order to change their current parent, and keep sending it at maximum rate (since they have just switched). If the neighbourhood is crowded, a large number of attachments will be sent within the same neighbourhood almost at the same time; an unnecessary overhead which is likely to cause congestion on node n.

In order to mitigate this effect, we start from the observation that a node n stays active as long as it has *at least one* child (*i.e.*, subscriber). The actual number of children does not change its activation state. In order to reduce the number of "useless" attachments, nodes subscribed to the same parent may send their attachments with a probability decreasing with the number of their siblings, which they know as their parent n include the number of its children g in the attachment messages piggybacked in the video frames.

The probability of sending an attachment must be chosen in such a way that a node with a greater number of children has a greater expected number of received attachments, since the estimation of g that the node includes in its status information, used by its neighbours use to evaluate function (3.1), depends upon the number of attachments that the node receives. In the current implementation, this probability is set to $\Pr\{\text{attachment}\} = 1 - \log_K g$, where K is a parameter chosen to be greater than the maximum value of g one can expect, *e.g.*, 1000 [GCPP11]. As shown in Fig. 3.6, this probability grants that the expected number of attachments is monotonic with the number of children (preserving the order relation of nodes with respect to function (3.1)); yet, the overhead is greatly reduced for a large numbers of children.

This reduction of the attachment rate does leave a minor risk of accidentally deactivating a node, which could leave some nodes unconnected. To further reduce the probability of an accidental deactivation, the control peer for a video packet always sends its attachment message, regardless of the value of g.

Finally, as mentioned above, we would like that when a new path is available, the nodes interested do not switch all at the same time, in order to avoid a sudden peak in congestion.

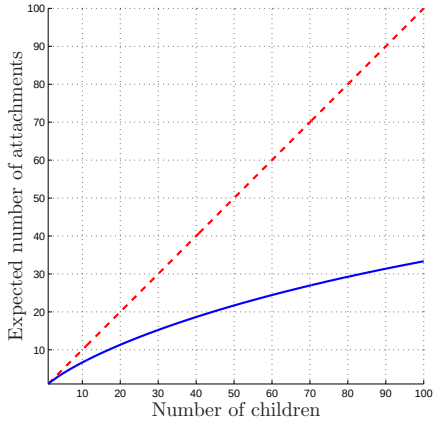


Figure 3.6: Expected number of attachments *vs.* number of children.

Since a coordinated action of nodes would be unpractical, we opted for a probabilistic scheme instead: a node that already has a parent switch to a new one that provides a better path only with a given constant probability $\Pr\{\text{switch}\}$. Since the function (3.1)) is recomputed each time an attachment or a video packet is received, if the value of the path stays unaltered, all the interested nodes will still eventually switch to the new parent, but they will do it within a longer time interval (the expected duration of this interval depends on $\Pr\{\text{switch}\}$). Apart form preventing peak congestion, this mechanism is also beneficial in networks with node mobility: a very good but very transient path, provided by a node "passing by", will not generate major changes of the overlay that would need to be undone as soon as the node gets out-of-range. In the current implementation, the switch probability can be set to a value between 60 % and 90 %.

3.4.5 Protocol packet structure

As a final point, we give here some detail about the structure of the packets generated by the ABCD. All ABCD packets fit in the same general structure, depicted in Fig. 3.7.

All packets contain the ID of the sender and that of the control peer for that packet. We choose to identify ABCD nodes by their MAC address, which is a unique identifier assigned to network interfaces by the manufacturer. This choice is justified by the fact that the ABCD is cross-layer, and communication between the MAC layer and the ABCD protocol is frequent (*e.g.*, for the choice of the control peer), therefore, adding a separate ID assignment system would have added a useless burden to the design.

Following the IDs of sender and control peer, the packet contains a sequence of *node state records*, one per description. A node state record reports the hop-count of the sender for that description (with a special value for $+\infty$ to signal that the sender does not receive the description), the ID of its current parent (set to all zeros if the hop-count is $+\infty$), and the current size of the sender's subscribers group (zero if the sender is not active).

Finally, the packet may contain data from the video stream. In this case, the ABCD

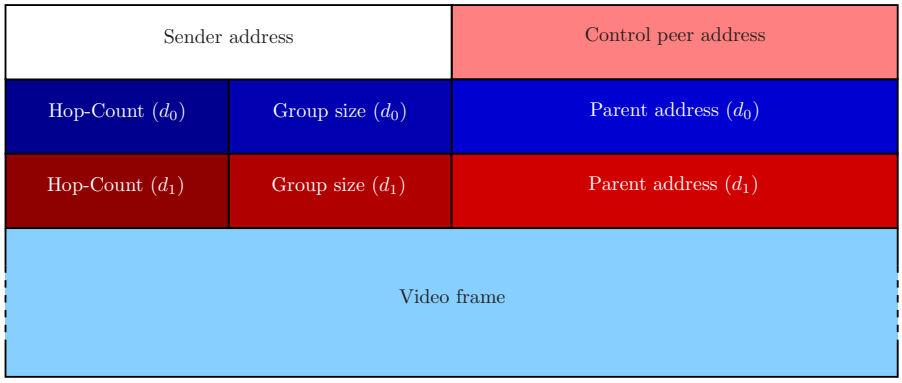


Figure 3.7: Structure of an ABCD protocol packet for two descriptions. Attachment messages do not contain any video frame. Advertisement messages are attachment messages with hop-count set to zero and parent address set to the sender address. A special values is defined to signal an infinite hop-count.

packet has a payload containing the video frame of one of the descriptions. Since the node state record size is fixed, and we assume that the nodes know the number of descriptions in advance, the nodes do not need other information in order to un-marshal the video frame.

For the sake of clarity, in Algorithm 1, we report the pseudo-code of the procedure executed by a node receiving an ABCD packet. Notice that this is a conceptual algorithm, which neglects many details of the actual implementation.

Algorithm 1 Procedure executed by a node n upon reception of an ABCD packet from a neighbour s.

```
1: procedure HandlePacket(pkt)
2:     Let s ←pkt.sender();
3:     for all description d do
4:         p ←pkt.parent(d);
5:         if p = n then                                   ▷ p is the current node
6:             if s is a subscriber of n on description d then
7:                 Refresh the entry of s in the subscribers list of description d;
8:             else
9:                 Add an entry for s in the subscribers list of description d;
10:                Increment own group size counter for description d.
11:            end if
12:        else if s is a subscriber of n on description d then      ▷ s has switched parent.
13:            Remove the entry of s in the subscribers list of description d;
14:            Decrement own group size counter for description d.
15:        else
16:            h ←pkt.hopCount(d);
17:            g ←pkt.groupSize(d);
18:            a ← (g = 0);
19:            d ←number of descriptions where s is parent of n (except d);
20:            q ←SNIR(n, s);
21:            J = +ω_h h + ω_a a + ω_d d − ω_g g − ω_q q;
22:            J_d ←value of J of the current parent on description d;
23:            if J < J_d then
24:                Randomly set the Boolean value x according with Pr{switch};
25:                if x then switch current parent of n for description d to s;
26:                end if
27:            end if
28:        end if
29:    end for
30: end procedure
```

3.5 Experimental study

In this section, we present an experimental study we carried out in order to validate and evaluate the performance of our proposed protocol. The validation has been conducted in a simulated environment, which allowed us to test the protocol under several instances of network topology in a reproducible way. We then analysed the following relevant properties: connection time (in Sec. 3.5.2), delivery rate (in Sec. 3.5.3), end-to-end delay (in Sec. 3.5.4), protocol overhead (in Sec. 3.5.5), and robustness towards mobility (in Sec. 3.5.6).

3.5.1 Simulation setup

We provided an implementation of ABCD protocol in the framework of the discrete event simulator *ns-2* [ns2]. This network simulator is often used in the simulation of routing protocols and is heavily used in ad-hoc networking research, as it models quite accurately the 802.11 MAC layer and implements a shadow/multi-path fading model. Even though

Propagation Model	Two-Ray Ground
Carrier	2.472 GHz (Channel 13)
Transmitted Signal Power	15 dBm
Collision Threshold	10 dB
Receiver Sensitivity	−82 dBm
Nominal Range	25 m

Table 3.4: Specifications of the simulated network adapters.

the PHY model of ns-2 is somewhat inaccurate, its wide availability makes it one of the most commonly used research tools in the field of mobile ad-hoc networking (see, for instance, [CBD02], [LSM07], and references therein). In ns-2, the user describes a network topology by writing a topology scripts and then the main program simulates that topology with specified parameters.

Our implementation consists in a modified version of the 802.11 MAC layer agent to support reliable broadcast, plus a routing agent that implements the application logic. This routing also actually plays the role of transport and network layer, as an integrated cross-layer approach has been followed.

The simulation parameters of the interfaces of the mobile node has been based on the specification of the widely used ORiNOCO 11 b/g card [Pro06], and are resumed in Tab. 3.4.

We set up several network scenarios (resumed in Tab. 3.5), varying the number of nodes and the playground size. The number of nodes is given without considering the source. Three different churns model are used for scenario S10: in S10a, there is no churn, *i.e.*, no connections or disconnections occur during the simulation; in S10b, 100 additional nodes join the network at $t = 9.5\,\mathrm{s}$; conversely, in S10c, 50 randomly chosen nodes leave the network asynchronously at $t = 2.5\,\mathrm{s}$ (the source is never one of the disconnecting nodes). No mobility is simulated in these scenarios, *i.e.*, nodes start at a random position, uniformly across the playground (except from the source, which is always located in the centre), and keep their position throughout the whole simulation, while a separate analysis of mobility is given in Sec. 3.5.6

For each scenario, the results we present in the following have been obtained by averaging the results of at least ten simulations, each with a different random placements of the nodes; this has been done in order to mitigate the effects due to particular topologies (distance from the source, bottlenecks, *etc.*).

3.5.2 Connection time

The first property of the protocol we want to test is the connection time, *i.e.*, how long does it take in order to form the overlay network. Namely, we want to investigate the time necessary for a node to receive at least one description, which allows it to reproduce

Scenario	Number of nodes	Playground size
S1	10	$50 \times 50\,\mathrm{m}^2$
S2	20	$55 \times 55\,\mathrm{m}^2$
S3	30	$60 \times 60\,\mathrm{m}^2$
S4	40	$65 \times 65\,\mathrm{m}^2$
S5	50	$70 \times 70\,\mathrm{m}^2$
S6	60	$75 \times 75\,\mathrm{m}^2$
S7	70	$80 \times 80\,\mathrm{m}^2$
S8	80	$85 \times 85\,\mathrm{m}^2$
S9	90	$90 \times 90\,\mathrm{m}^2$
S10	100	$95 \times 95\,\mathrm{m}^2$
S11	200	$100 \times 100\,\mathrm{m}^2$

Table 3.5: Network scenarios used to test the protocol. Three different churns model are used for scenario S10 (see Tab. 3.6).

Scenario	Churn model
S10a	No churn.
S10b	100 additional nodes join the network at $t = 9.5\,\mathrm{s}$.
S10c	50 randomly chosen nodes disconnect asynchronously at $t = 2.5\,\mathrm{s}$.

Table 3.6: Different churn models used in scenario S10.

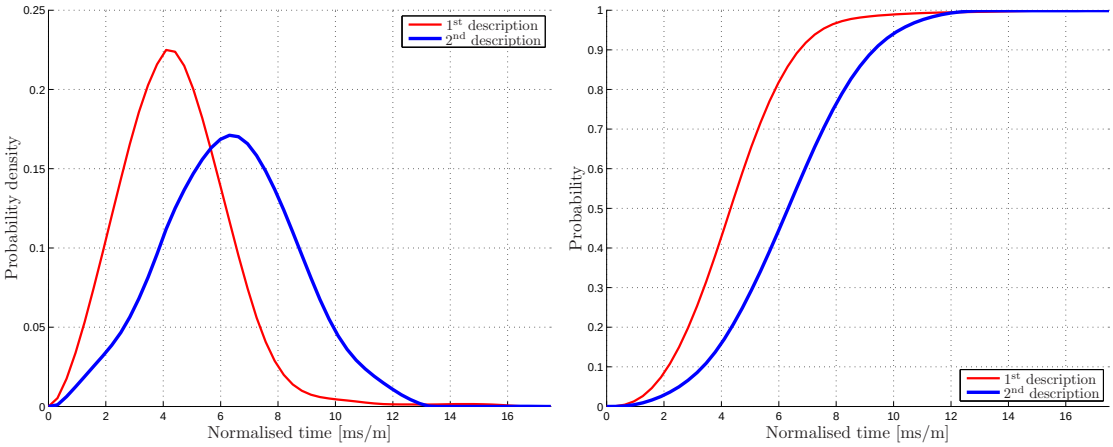
the video sequence at a reduced quality, and to receive both, which allows the node to reproduce the video sequence at the maximum quality.

In Fig. 3.8(a), we show the relative frequencies of the *normalised connection time* (NCT). We define the normalised connection time as the time necessary for a node to receive a description, divided by its distance to the source, on which the number of hops depends. The NCT allows us to make our measurement that is independent on the relative positions of nodes.

In order to give a clearer presentation, the relative frequency of the normalised connection of the nodes has been represented using an estimation of the probability density functions obtained with the Parzen window method [Par62], using all results of all scenarios in Tab. 3.5. We observe that on average nodes connect to their first description in 4.0 ms/m and to both descriptions in 6.0 ms/m. This means, *e.g.*, that on average a node located at 50 m from the source will connect to its first description in 200 ms and to both descriptions in 300 ms.

In Fig. 3.8(a), we also report the cumulative frequency of the normalised connection time. Here, we can observe that almost all nodes connect in less than 10 ms/m to their first description, and in less than 12 ms/m to the second description. This means, *e.g.* that a node located at 50 m from the source will connect to its first description in less than 500 ms, and to both in less than 600 ms. In conclusion, we can say that the protocol is able to connect the nodes in a time that is, both in average and maximum value, within intervals deemed acceptable for most applications.

In Fig. 3.9, we see an example of how an ABCD overlay looks like once that all nodes



(a) Probability density of connection time. (b) Cumulative probability of connection time.

Figure 3.8: Probability of connection time normalised with respect to the distance from the source.

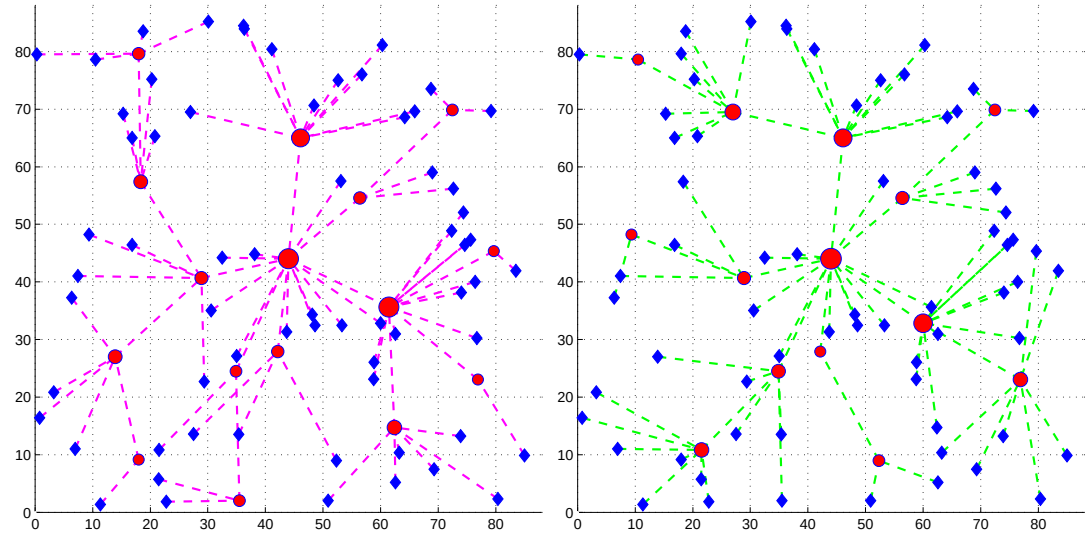
are connected on both descriptions. We notice that, in this overlay, the distance between any two nodes active on the same description is generally in the order of the transmission radius, meaning that there are few active nodes in the same neighbourhood; also, whenever possible nodes choose two different parents for the two descriptions. Both these facts are consistent with our expectations given function (3.1) defined in Sec. 3.4.2.

3.5.3 Delivery rate

Another important propriety of the protocol is its ability to actually deliver the video frames, reducing the number of losses, as this capability affects the quality of the reconstruction of the video sequence that the nodes are capable to achieve.

In Fig. 3.10(a) and 3.10(b), we observe the protocol's delivery performance in scenarios S6 and S11, given as percentage of nodes receiving one or both descriptions. The two selected scenarios present neither connection of new nodes nor disconnections during the simulated time and are useful to study the behaviour of the protocol in a steady state. The node density, expressed as number of nodes per neighbourhood, is of ~ 15 and ~ 30, respectively. We observe that, once the overlay is formed, and all nodes are connected on both descriptions, no frame is lost. Even though a small number of collisions has been observed, it affects only attachments, not video packets.

In Fig. 3.10(c) and 3.10(d), we observe the protocol's delivery performance in more dynamic scenarios, namely S10b and S10c. In S10b, at $t = 0$, 100 nodes connect to the overlay, then, at $t = 9.5\,\mathrm{s}$, 100 additional nodes join the ad-hoc network. We can see that almost all of them receive at least one description as soon as they join the network. This is due to the fact that when a new node joins an existing overlay, if it is in a connected part of the ad-hoc network, it will either be in the transmission area of an active node – in which case, it would be receiving the description without any action taken – or at least



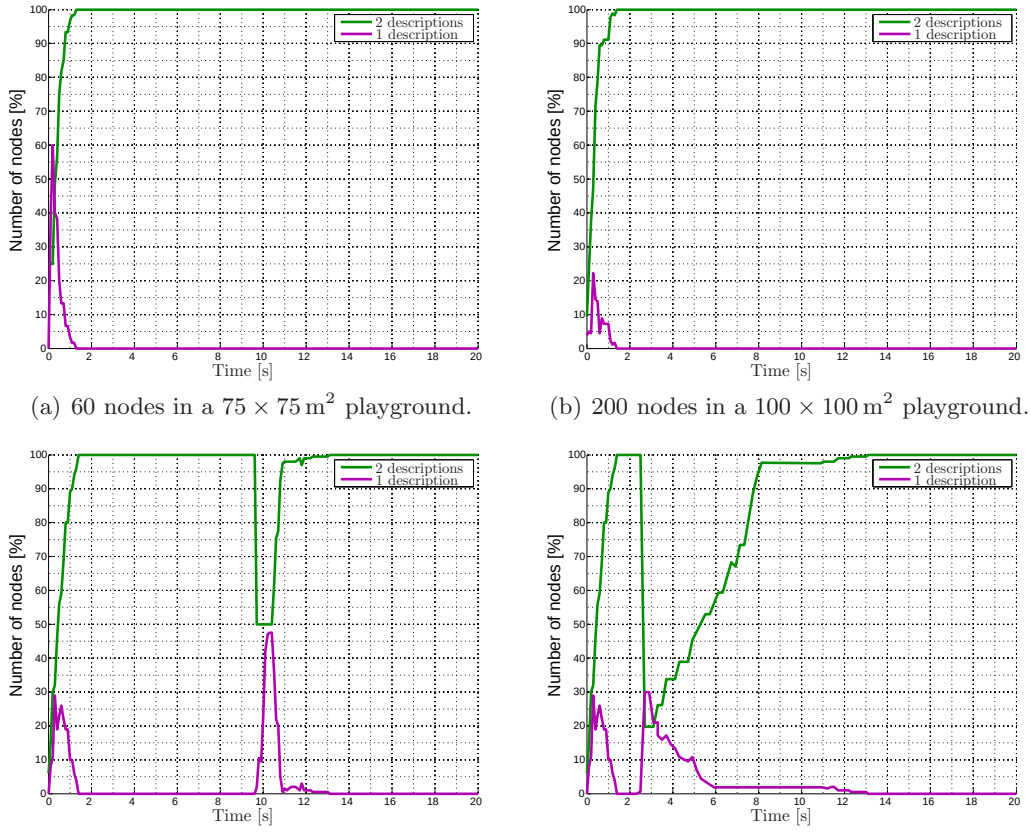
(a) Overlay tree for description *one*.

(b) Overlay tree for description *two*.

Figure 3.9: An example of ABCD multi-tree for two descriptions, in a network with 80 nodes in an $85 \times 85\,\mathrm{m}^2$ playground. Red dots represent active nodes, the size of the spots is proportional to the number of nodes subscribed. The source is located at the centre of the playground.

in the transmission area of a connected node — in which case, it would just need to send an attachment to its designated parent in order to activate it. After the nodes have joined the network, the overlay is adjusted in order to provide nodes with the second description as well, all within the next 2 seconds. Even though the node density is doubled, no frames are lost once the overlay is adjusted. This shows how the ABCD protocol presents interesting properties of *scalability*, *i.e.*, an ability to handle an increased amount of nodes in a graceful manner. This property is inherent to the use of a broadcast medium, wherein transmission to many nodes takes the same resources as transmission to one node.

In S10c, at $t = 0$, 100 nodes connect to the overlay, then, at $t = 2.5\,\mathrm{s}$, 50 randomly chosen nodes (but not the source) leave the ad-hoc network abruptly. We can see that recovering from a massive abrupt disconnection of nodes is much more problematic than recovering from a massive connection of new nodes. Namely, the recovery takes almost twice the time as the initial connection. This is partially due to the fact that the new network has a lower density, so more nodes have to be activated in order to cover the whole network area; however, this is not the only effect. Two major phenomena affect the reconstruction of the overlay: out-of-date information in the nodes' local knowledge of the topology, and increased congestion level. The former phenomenon implies that, upon realising that its current parent has disconnected (which take in itself some time, as a node has to wait for a time-out before assuming that its parent has left) a node tries to connect to the best path available; however, if several nodes have disconnected at once, the new designated parent could have disconnected as well (or it could in its turn have



(a) 60 nodes in a $75 \times 75\,\mathrm{m}^2$ playground.

(b) 200 nodes in a $100 \times 100\,\mathrm{m}^2$ playground.

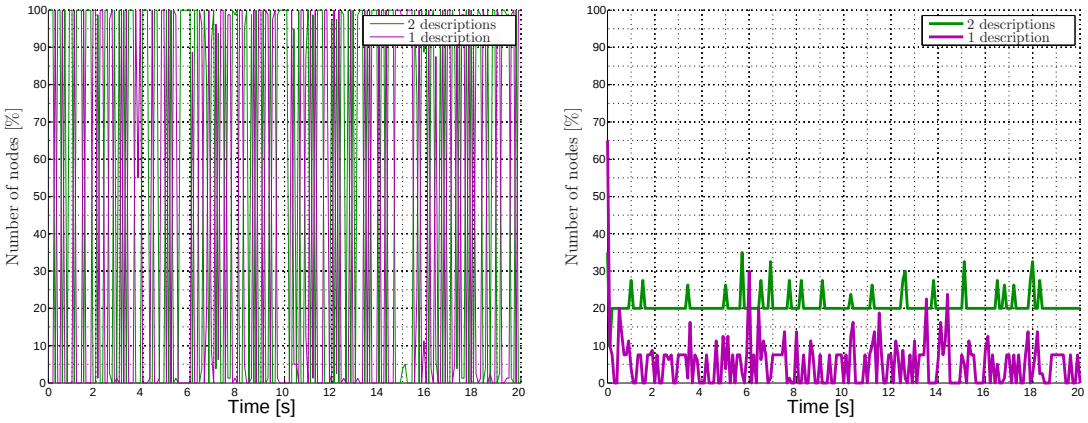
(c) 100 nodes in a $95 \times 95\,\mathrm{m}^2$ playground; 100 more nodes connect at $t = 9.5\,\mathrm{s}$.

(d) 100 nodes in a $95 \times 95\,\mathrm{m}^2$ playground; 50 nodes disconnect at $t = 2.5\,\mathrm{s}$.

Figure 3.10: Frame reception rate *vs.*time.

lost its current parent). Therefore, if the disconnection involves many nodes, a node may try several candidates (each one with an associated time-out) before finding a valid one. Also, due to the latter phenomenon, the new topology is propagated less quickly than in the steady case, as the high number of nodes sending attachments to their new parents (possibly multiple times, before finding a valid one) and at maximum rate (as they have just switched parent) is likely to generate collisions and delay on the attachment themselves. It should be noted, however, that even in the very unlikely scenario of abrupt disconnection of a random half of the nodes composing the network, 30 % of the nodes is still receiving one descriptions, *i.e.*, even though they could possibly suffer a degradation of the video quality, they are still receiving the video sequence, while 20 % is still receiving both, *i.e.*, they are completely unaffected.

In Fig. 3.11, we present for comparison the performance in scenario S10a of two other broadcast protocols, namely *MDC simple flooding* and *MDC probabilistic flooding* [WC02]. With MDC flooding, we refer to a flooding technique wherein the broadcast content is a video sequence encoded in multiple descriptions. The flooding technique is oblivious of the content being broadcast and does not affect the delivery performance; nevertheless, using



(a) MDC simple flooding (b) MDC probabilistic flooding

Figure 3.11: Frame reception rate comparison with two different flooding techniques in scenario S6.

an MDC technique, as opposed to SDC, has an impact on visual quality, which we shall analyse below.

Neither of these flooding techniques rely on an explicit overlay network in order to deliver the content. This lack of structure implies that these techniques are easier to implement and to maintain, as no information about the neighbouring nodes is required. Our protocol, on the other hand, can be classified as a form of *selective flooding* based on neighbour knowledge. Selective flooding means that the protocol chooses, for each description, a subset of the nodes in the network that will behave like in simple flooding for that description (*i.e.*, the active nodes). It is neighbour knowledge based in the sense that the choice of the nodes that have to activate is taken locally by the nodes based on the status of their neighbours and the portion of topology of the network that can be estimated by communicating only within one's own neighbourhood. In order to communicate this knowledge, we pay a cost in terms of packet overhead (*i.e.*, the attachments), but we are able to cover the network with much less packet redundancy (*i.e.*, with less duplicated video packets within each neighbourhood).

As can be seen in Fig. 3.11(a), the lack of coordination of the simple flooding strategy results in a very erratic delivery performance. Since in this scheme *all* nodes are active (*i.e.*, they all transmit the video packets as soon as they received them) collisions are common; our scheme of reliable broadcast could mitigate the effect of collision, but it is hardly applicable, as the lack of structure makes the choice of the control peer not trivial. Also, even if it were possible to select a proper control peer for each node, the congestion of the network would greatly increase, as all nodes are trying to get exclusive access to a shared channel at the same time, resulting in unacceptable delays on the video packets.

Using probabilistic flooding partially improves the performance, as only a share of the nodes is active at any given time; the expected number of active nodes is given by the retransmission probability. The retransmission probability can be used to tune the number

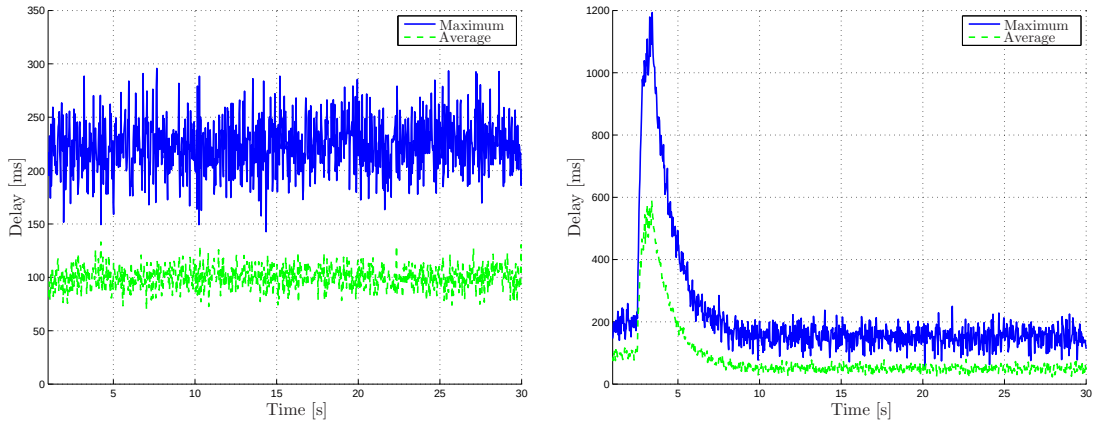
of active nodes in the network: when the probability is high, the coverage of the network (*i.e.*, the number of nodes that can eventually access the content) is increased, but more packets are lost because of collisions or dropped because of excessive delay (depending on whether or not we use a reliable broadcast scheme); conversely, if the probability is low, there are fewer collisions and a smaller congestion, but the coverage is reduced as well. In Fig. 3.11(b), we show the performance of the protocol with a retransmission probability of 47 %, which has been chosen experimentally to maximise the delivery rate. What happens, is that only the central part of the network (*i.e.*, the nodes located around the source) are able to receive the stream consistently, whereas towards the borders of the network reception becomes marginal. We interpret this as an effect of the variable node density in the network: the optimal probability is such that collisions are minimised, but at the border of the network, where density is lower, this probability is too low to assure that at least one node per neighbourhood will be active at each time. This effect could be mitigated if the nodes were somehow able to estimate the local density of the network and choose independent their retransmission probability accordingly; however, such an estimation would take some time in order to converge to a reliable value, which could in turn change very quickly if the network experiences churn or the nodes are moving.

3.5.4 End-to-end delay

For many applications involving video streaming, the delivery rate is not enough to evaluate the performance of a protocol. The video sequence could be supposed to be delivered in a timely fashion, as an excessive end-to-end delay could render the application unusable; we speak in this case of *real-time streaming* or *live streaming*. For instance, we could decide that, for the purposes of our application, a video packet must arrive within a given delay threshold of 250 or 500 ms, otherwise it would be rejected as out-of-date upon reception.

In our simulation we set up the maximum delay to 500 ms. We measure the frame delay as the time elapsed between the moment that the ABCD protocol running on the source passes the packet containing the video frame to the MAC address, and the moment that the ABCD protocol running each peer receives the packet. Any packet received with a higher delay will be rejected at once and will be not retransmitted in any case. A threshold of 500 ms is actually quite high in a real application scenario, but we chose to relax this constraint in our initial validation. A more extensive study of delay using a more stringent constraint is carried out in Chapter 4.

In Fig 3.12(a), we report average and maximum end-to-end delay of video packets *vs.* time in scenario S10a. We observe that average delay is in the order of 100 ms, while maximum delay is lesser than 250 ms, *i.e.*, no video packets are dropped because of lateness with the threshold of 500 ms assumed in the simulations. However, in Fig. 3.12(b), we see that the abrupt disconnection of 50 % of nodes in scenario S10c causes a sudden increase of delay that needs several seconds before settling on more acceptable values. For about



(a) Scenario S10a: 100 nodes in a $95 \times 95\,\mathrm{m}^2$ playground.

(b) Scenario S10c: 100 nodes in a $95 \times 95\,\mathrm{m}^2$ playground; 50 nodes disconnect at $t = 2.5\,\mathrm{s}$.

Figure 3.12: End-to-end delay *vs.*time.

2 seconds, even the average delay surpasses the maximum delay allowed, meaning that most of the frames transmitted in that interval will be eventually dropped. As mentioned in Sec. 3.5.3, this is due to the great number of node activations and attachment exchanges that are performed in order to try and rebuild the overlay network, and results in a relevant number of lost frames for the nodes that did not disconnect. Since most of the frames are not going to provide any benefit anyway, as they are doomed to be dropped because of lateness, it would be beneficial to the network if some nodes actually refrained for some time from sending video packets altogether, allowing a faster fading of the congestion. We shall introduce a novel way to do this in an optimal fashion in Chapter 4.

Both loss rate and end-to-end delay are considered as a measure of the *Quality of Service* (QoS) offered by the protocol. QoS is a concept introduced by the ITU-T in 1994, and is a set of objective measures of a transmission [QoS94]. These measures, however, tell us very few about the actual *Quality of Experience* (QoE) enjoyed by the users, *i.e.*, the metrics that the customer will directly perceive as a quality parameter. In order to estimate the latter, we computed the average objective visual quality, in terms of *Peak Signal-to-Noise Ratio* on the *luminance component* (Y-PSNR), of the node *vs.* the simulation time. Since loss rate and end-to-end delay do not depend on the video coding technique, we have been so far *codec-agnostic*, *i.e.*, we assumed that a generic MDC technique able to produce two balanced descriptions was used, without making further assumptions on the codec. On the other hand, here we need to perform measures on the actual video sequence that the users have decoded, hence, we shall hereby integrate a particular video MDC technique, namely, the one detailed in Chapter 2, in our framework. When both descriptions are lost, concealment is performed by reproducing the last decoded frame.

In Fig. 3.13, we see a comparison of objective video quality among our protocol, a simple flooding technique broadcasting a sequence encoded in MD with our proposed technique, and a simple flooding technique broadcasting a sequence encoded in SD with

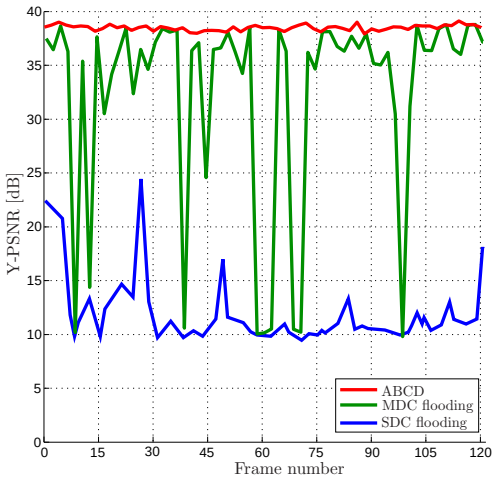


Figure 3.13: Visual quality comparison of broadcast techniques. Quality is measured as the average Y-PSNR for scenario S10a (sequence "Foreman", CIF, 30 fps, 1.8 Mbps).

H.264/AVC.

We see that the Y-PSNR obtained with the ABCD protocol consistently outperforms the multiple description simple flooding, which in turn outperforms the single description simple flooding. While the ABCD protocol is able to provide a stable Y-PSNR of about 38 dB, the multiple description simple flooding presents several significant negative spikes, due to simultaneous loss of the corresponding packets in the two descriptions. However, while using flooding jointly with MDC still provides an almost acceptable average quality, as several losses of video packets are concealed by the use of the side decoder, using single description coding renders the video stream very fragile with respect to losses, an effect which is also exacerbated by the interdependency of frames within the coding structure (*i.e.*, the loss of one frame used for reference in the predictive scheme renders useless the predicted frames even if these are received).

3.5.5 Protocol overhead

Finally, we want to analyse the overhead needed to the protocol in order to exchange the control information used to build and maintain the overlay network.

First, in Fig. 3.14, we quantify the overhead in terms of number of active nodes in the network. The number of active nodes gives an estimation of how many copies of each video packet will be injected in the network. In principle, we want that this number does not grow too quickly with the number of nodes in the network, to assure that the network is not congested; however, a certain amount of redundancy may prove useful as it allows a quick recovery of the overlay in case of node disconnections. We observe that on average, in our scenarios, about one third of the nodes (including the source) is active on at least one description, and about one fifth is active on both. Having a node

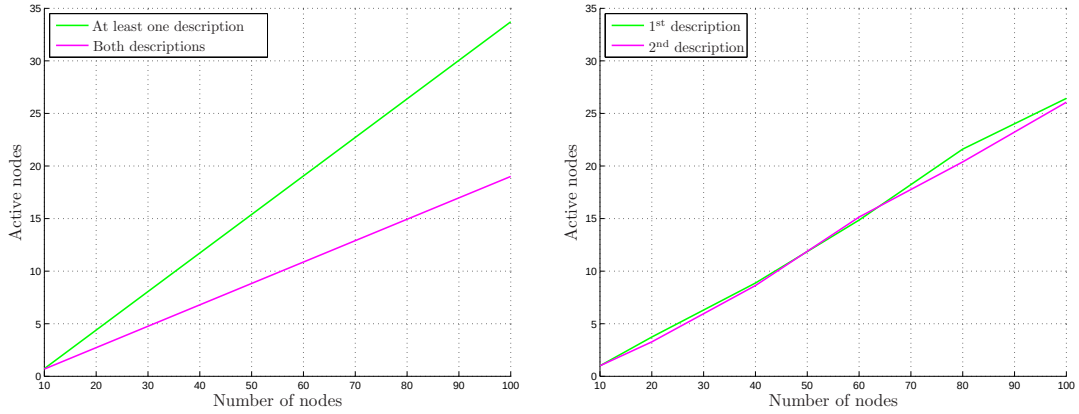


Figure 3.14: Protocol overhead in terms of average number of active nodes.

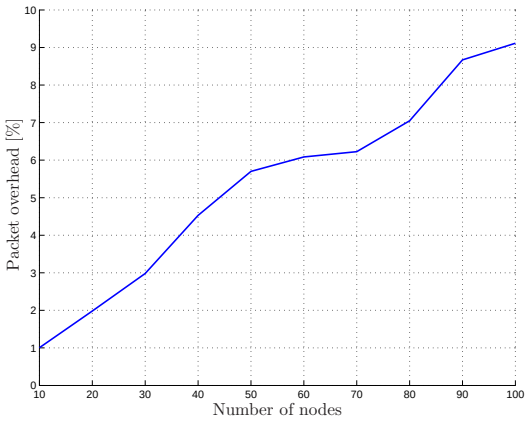


Figure 3.15: Protocol overhead in terms of percentage of attachments; attachments piggybacked in video packets are disregarded.

active on both descriptions does affect the diversity of the paths, leading to a more fragile network in case of disconnection; however, it also reduces congestion, as the node does not have to coordinate its use of the channel with another node, and attachments for the two descriptions can be sent within the same packet.

Furthermore, in Fig. 3.15, we analyse the overhead in terms of percentage of attachments sent with respect to the total number of messages (video packets and attachments). We see that even in scenario S10a, where 100 nodes participate to the network, the packet overhead is below 10 % of the total traffic.

3.5.6 Mobility

In order to test the performance of the protocol in presence of node mobility, two different models have been experimented [GPCPP11]: Random Way-point and Reference Point Group Mobility.

The Reference Point Group Mobility (RPGM) [HOTV99], in particular is well-suited for the kind of application the ABCD protocol is tailored for, *i.e.*, crisis management

service, such as military and rescue operations (*e.g.*, to provide battlefield awareness and data dissemination), business environments, such as video conferencing outside the office (*e.g.*, to brief clients on a given assignment), and recreational contexts (*e.g.*, to allow user to view a live stream of an event they are attending), since, in all these scenarios, the users typically move in groups.

However, we chose here to present the results for the Random Way-point model, as it is the most challenging for a structured protocol, since the nodes move independently.

The delivery rate as a function of the nodes' average speed is presented in Fig. 3.16. We observe that the performance of the protocol is practically unaffected for speeds within the targeted application range (0–5 m/s), and starts to deteriorate only for speeds greater than 9 m/s.

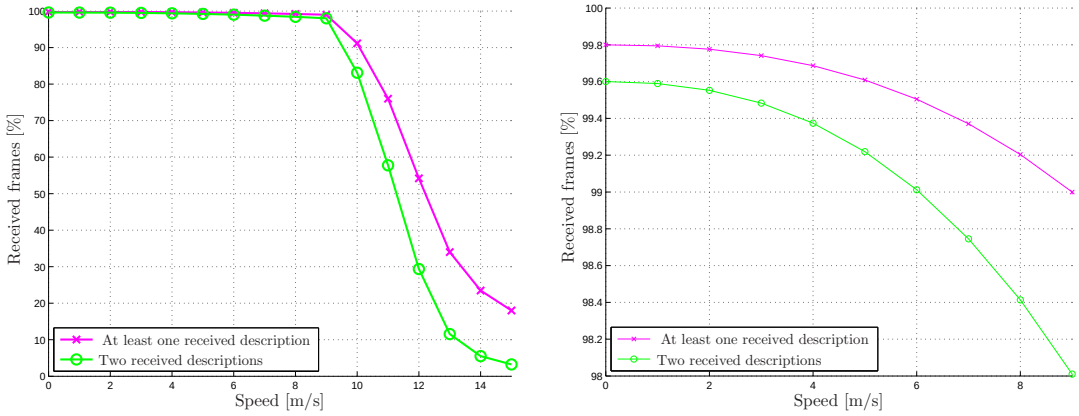


Figure 3.16: Case of mobility with Random Way-point: ABCD delivery rate as a function of average node speed.

3.6 Conclusions

In this chapter, we discussed the subject of video streaming over wireless ad-hoc networks. After having introduced the basic concept of video streaming, we have given an overview of the most popular tools used on MANETs to achieve multi-hop broadcast and multicast, particular attentions to video transmission.

Then, we have presented our own original contribution to this field, *i.e.*, a cross-layer protocol, called ABCD, that uses a distributed multi-objective optimisation function that captures sever properties of a multi-hop broadcast relevant to a video streaming application, such as delay, robustness to link and node failure, congestion, *etc.*

We provided an extensive experimental validation of the proposed protocol, using a suitable simulation environment (*ns-2*), and under several conditions of node density, number of nodes, stream bitrate, and node mobility. We observed that, and in all these scenarios, ABCD has proven to be able to ensure that 100 % of the nodes receive almost all frames of

all descriptions, for a node density up to 20 nodes per neighbourhood (which is three times as high as the optimal density in the sense of trade-off between the number of hops to reach a destination and collisions occurring at each node [RMSM01]) and a node speed up to 9 m/s, which is almost twice the speed of our reference application. The average delay is kept in the order of the hundreds of milliseconds as the topology is slowly changing, but the maximum delay can have much higher peaks if the topology is changing quickly, *e.g.*, a flash-crowd or a high mobility happens. The key ideas of this technique, and the relative results, have been published on the Inderscience International Journal of Communication Networks and Distributed Systems. Subsequent improvements have been presented to the 23rd editions of the *Colloques sur le Traitement du Signal et des Images* (GRETSI) and to the 2011 IEEE Workshop on Multimedia Signal Processing (MMSP).

In order to deal with node densities higher than 20 nodes per neighbourhood, in Chapter 4, we present a Congestion Distortion Optimisation framework that integrates the ABCD protocol, allowing to extend the performance of the protocol to higher densities and more stringent delay constraint.

Chapter 4

Streaming over dense networks with low-latency constraints

Contents

This chapter is entirely dedicated to one of our own original contributions, namely, an extension of the ABCD that allows video streaming, under stringent delay constraints, in highly dense ad-hoc networks using a *Congestion-Distortion Optimisation* (CoDiO) framework.

First, in Sec 4.1, we give a formal definition of the problem introduce the notation for the parameters we will be working with. In particular we will define a constrained minimisation problem reflecting the fact that each node tries to minimise the average distortion of the nodes in the network subject to the constraints imposed by the congestion of the channel. Then, in Sections 4.2 and 4.3, we show how the nodes estimate these two parameters with an efficient message exchange that exploits the properties of the ABCD protocol overlay.

Our experimental validation of the proposed framework, presented in Sec. 4.4, shows that, if a stringent constraint of low delay is imposed, our technique grants a consistent gain, in terms both of PSNR and of delay reduction, for bitrates up to a few megabits per second.

4.1 Congestion-distortion optimisation

As mentioned in Chapter 3, even though the ABCD protocol is able to create a dynamic overlay that is efficient in terms of coverage and packet overhead, it has an inherent limitation due to the broadcast reservation: when node density is very high – globally, or locally due to mobility – or a sudden change in topology occurs – due to a high churn or a very fast mobility – the average delay may become so high that some video frames are received beyond their playback deadline. The actual values for an acceptable delay depend on the buffering strategy and, more generally, on the application; in the following we shall assume that our target application aims at abiding by a conversational pattern, which means that the maximum accepted delay from the video source to the end user is in the order of one hundred milliseconds, and the total (*i.e.*, for all descriptions) bitrate of the stream is in the order of a few megabits per second, a setup consistent with a video-conferencing application. An example of application is a field awareness or an order dispatch service, provided in a military or disaster-relief scenario, where a unidirectional video feed can be provided live to the agents on the field by the central headquarters.

In order to reduce the delay in ABCD, we introduced a Congestion-Distortion Optimisation (CoDiO) criterion in the per-hop forwarding of the protocol; namely, we adjust the retry limit to be passed to the RTS/CTS mechanism of the MAC, in a Co-Di optimised fashion. CoDiO is an approach already proven viable in the design of cross-layer protocols for video streaming on MANETs [SYZ+05]; here we propose a formulation of the problem that takes into account both the inherent redundancy of the overlay and some specific features of the reliable broadcast scheme of ABCD; however, the proposed framework lends itself to be integrated in other tree-based streaming protocols.

The idea of congestion-distortion optimisation in video streaming, as opposed to the traditional rate-distortion optimisation, was introduced to model the effects that self-inflicted network congestion has on video quality [BTB+03, SG04]. This model was introduced considering a wired scenario wherein each node is connected to the video source by a

succession of high-bandwidth shared links and terminating with a bottleneck on the last hop; it was later adapted for the case of unicast streaming over mobile ad-hoc networks by Zhu *et al.* [ZSG05], who proposed a routing algorithm that seeks to minimise the congestion by optimally distributing the traffic over multiple paths. They developed a model that captures the impact on the overall video quality of both the encoding process and the packet losses due to network congestion. Their model has proven to be able to capture the influence of these different parameters, and can be used to predict the end-to-end rate-distortion performance of a multi-hop video delivery system. However, this model provides neither an on-line estimation of the model parameters (*i.e.*, the network conditions), nor a viable extension for multicast streaming.

We start from the observation that the congestion *vs.* distortion trade-off can be adjusted by tuning the retry limit in the RTS/CTS mechanism detailed in Chapter 3. Let us denote with k the value of the retry limit; small values of k would reduce the congestion, since less requests are sent to try and obtain the channel, but the expected distortion would increase, as it would increase the probability of not obtaining the channel and being unable to send the current packet. On the other hand, higher values of k would lower the expected distortion, since the probability of sending the packet is higher, but would also imply a higher congestion due to the channel occupation. We end up with a constrained minimisation problem; specifically, for each video packet, we want to find the optimal value k^* for the retry limit, defined as:

$$k^* \triangleq \arg\min_{k\in\mathbb{N}} \{D(k) + \lambda C(k)\}, \tag{4.1}$$

where $D(k)$ is the expected total distortion over a set of frames depending on the current packet (*i.e.*, contained in the packet or predicted upon it), for all the nodes in the sub-tree rooted in the current node (described in Sec. 4.3), and $C(k)$ the expected congestion of the channel seen by the current node (detailed in Sec. 4.2), both resulting from the retry limit k for the current packet.

The role of the parameter λ in the minimisation problem (4.1) is analogous to that of Lagrange multipliers used in the resolution of continuous constrained optimisation problems. The choice of the parameter λ is detailed in Sec. 4.4, while here we limit to observe that large values of λ will correspond to stricter constraint on congestion, *i.e.*, the minimisation function will privilege the congestion reduction and will therefore be likely to choose a smaller value for k. Conversely, with small values of λ, the function will consider the minimisation of distortion more important, choosing larger values for k.

While the congestion model can be computed locally without need to propagate information through the overlay, since it depends on the channel that the nodes can observe directly, the distortion model offers several challenges. A missing packet affects in general several frames of the decoded video sequence; moreover, the effects are in general different for each node, depending on its ability to receive of the same packet from another path and

the number of descriptions it is currently receiving. Finally, the effects of a loss propagate along the multicast tree, while a node can only communicate with its direct neighbours. These points are discussed in detail in the following.

4.2 Congestion model

With respect to the minimisation problem (4.1), we define the congestion seen by a node n as the number of packets Q_n that cannot be sent (by n or by other nodes) as the channel in the neighbourhood of n is occupied by n itself, times the time $\Theta(k)$ the channel is kept busy by n (the former not depending on k): $C(k) = Q_n\Theta(k)$. Thus, Q_n is the length of a virtual packet queue, distributed among n and its neighbours. This value can be easily estimated if each node includes the length of its transmission queue in the header of its messages (both data and overlay control).

This is different from the assumption usually made in the literature that *all* the neighbours of a node are always willing to transmit at any time, and is based on the knowledge of the neighbours' state, namely, the number of packets they have to send — in line with the ABCD protocol paradigm of piggybacking control information on broadcast packets. Notice that, with this formulation, a node will not refrain from sending a packet for the mere fact of having many neighbours: its neighbours will not be considered when it is known that they do not have a packet to send; conversely, the more a node has neighbours with pending transmissions, the more it will try to reduce the number of packets it sends.

We estimate the time the channel is kept busy by the quantity $\Theta(k)$, defined as:

$$\Theta(k) \triangleq T_{\text{rts}}(k) + P_n(k)T_{\text{pkt}},$$

where T_{pkt} is the time to transmit the data packet — which depends on its size. The term $P_n(k)$ is the probability that n successfully obtains the channel – and thus sends the packet – with at most k tries, which is estimated as $P_n(k) = 1 - (1 - p_n)^k$. The value of $T_{\text{rts}}(k)$ is defined as the expectation of the random variable t_{rts}, which represents the time needed to succeed the RTS/CTS competition and which depends on the number c of collisions occurring, given a retry limit of k: $T_{\text{rts}}(k) \triangleq \mathbf{E}[\, t_{\text{rts}}|k \,]$. Let χ be the random variable representing the number of collisions occurred before obtaining the channel, t_{rts} can be written as:

$$t_{\text{rts}} = (\chi + 1)\, T_{\text{tx}} + \chi\, t_{\text{bo}},$$

where T_{tx} is the time to send an RTS packet including its following inter-frame space (as defined by the IEEE 802.11 standard [CWKS97]), and t_{bo} is the back-off time, also depending on χ. The dependency of t_{rts} on χ suggests to evaluate $T_{\text{rts}}(k)$ as a conditional

expectation with respect to the number of occurred collisions:

$$\begin{aligned} T_{\text{rts}}(k) &= \sum_{c=0}^{k-1} \mathbf{E}\left[\ t_{\text{rts}} | \chi = c\ \right] \Pr\left\{\chi = c\right\} \\ &= \sum_{c=0}^{k-1} \left((c+1)T_{\text{tx}} + c\,\mathbf{E}\left[\ t_{\text{bo}} | \chi = c\ \right]\right) \Pr\left\{\chi = c\right\}. \end{aligned}$$

Given c collisions, the expected value of t_{bo} is $\mathbf{E}\left[\ t_{\text{bo}} | \chi = c\ \right] = \frac{W}{2}\left(2^c - 1\right)$, where W is the size of the contention window as defined by the 802.11 standard [CWKS97]. Therefore, given the probability of having c collisions $\Pr\left\{\chi = c\right\} = p_n(1-p_n)^c$, we can rewrite $T_{\text{rts}}(k)$ as:

$$T_{\text{rts}}(k) = \sum_{c=0}^{k-1} \left[(c+1)\,T_{\text{tx}} + c\,\frac{W}{2}\left(2^c - 1\right)\right] p_n(1-p_n)^c.$$

We observe that, if we define:

$$\Delta T_{\text{rts}}(k) \triangleq \left[kT_{\text{tx}} + (k-1)\frac{W}{2}(2^{k-1} - 1)\right] p_n(1-p_n)^{k-1},$$

then $T_{\text{rts}}(k)$ can be computed using the following difference equation:

$$\begin{cases} T_{\text{rts}}(0) = 0, \\ T_{\text{rts}}(k) = T_{\text{rts}}(k-1) + \Delta T_{\text{rts}}(k). \end{cases}$$

In other words, to estimate the expected time to perform the RTS/CTS mechanism with at most k tries ($T_{\text{rts}}(k)$), we add to the expected time for at most $k-1$ tries ($T_{\text{rts}}(k-1)$) the expected time given $k-1$ collisions weighted with the probability of having $k-1$ collisions ($\Delta T_{\text{rts}}(k)$). This formulation is particularly convenient, since in the Co-Di optimisation several consecutive values of $T_{\text{rts}}(k)$ have to be computed. With this function, each node has now everything it needs in order to estimate the congestion on its channel.

4.3 Distortion model

In this section, we present our distortion model and discuss how its parameters can be estimated and propagated. For the sake of simplicity, we shall assume that the stream consists in two descriptions (d_0 and d_1), hence the overlay consists in the superposition of two different multicast trees. The task of extending the framework for more than two descriptions is conceptually easy, but using more than two descriptions would require that *any* combination of received descriptions should be considered individually, making the exposition long and confused. As a matter of fact, the case of double description coding is the most interesting in practice, and in our reference scenario in particular, as it provides a very good balance between redundancy introduced by MDC, and robustness

to losses [ASPEF01, AWTW02].

Let us consider a node n that has to send a packet r from description d_0. Each description can be decoded independently, but – within the same description – some frames could depend on previously decoded ones if a predictive scheme is employed. Let us call $\mathbf{I}(r)$ the set of video frames that depend on r because they are included in r or predicted upon it. We assume an additive distortion measure, since all the most popular distortion measures are additive or equivalent to additive measures: *e.g.*, the sum of squared differences, SSD, equivalent to the PSNR; the sum of absolute differences, SAD; or the structural similarity (SSIM) index [WBSS04]. Using any of these measures (in particular, we used the SSD), we can define the following quantities: D_c, the cumulative distortion for $\mathbf{I}(r)$, *i.e.*, the sum of the distortion of the frames in $\mathbf{I}(r)$, when the central decoder can be used on these frames, since both descriptions have been received; D_0, the cumulative distortion for $\mathbf{I}(r)$ when decoding only d_0 (only r is received); D_1, the cumulative distortion for $\mathbf{I}(r)$ when decoding only d_1 (only r is lost); and D_f, the cumulative distortion for $\mathbf{I}(r)$ when using a strategy of concealment (as both d_0 and d_1 have been lost).

Let us consider what happens if node n is *deactivated* on description d_0, *i.e.*, it stops transmitting it. Each node in the sub-tree rooted in n belongs to exactly one of the following sets.

- $\mathbf{S}_c$, of size N_c: nodes able to receive both descriptions even if node n is deactivated on d_0;
- $\mathbf{S}_0$, of size N_0: nodes able to receive only d_0 if node n is deactivated on d_0;
- $\mathbf{S}_1$, of size N_1: nodes able to receive only d_1 if node n is deactivated on d_0;
- $\mathbf{S}_f$, of size N_f: nodes unable to receive either description if node n is deactivated on d_0.

In Fig 4.1, we show a node n (identified by the blue circle with dashed border) transmitting description d_0 in its neighbourhood, which in this example is a circle. It should be noted that the actual shape of the transmission area does not affect the protocol nor the definition of the sets $\mathbf{S}_c$, $\mathbf{S}_0$, $\mathbf{S}_1$, and $\mathbf{S}_f$. Node m is also active on d_0 and is not a child of n; while node p is active on the complementary description d_1; the transmission areas of these nodes are also represented.

If we assume that there are no other active nodes in this neighbourhood, considering the intersections of the transmission areas, we see that n has three neighbours in its set $\mathbf{S}_c$ (identified by squares), two nodes belong to set $\mathbf{S}_0$ (diamonds), four nodes belong to set $\mathbf{S}_1$ (triangles); and five nodes belong to set $\mathbf{S}_f$ (circles). Even though this is easily seen on this simplified example, determining the size of these sets is much more complicated when we take into account a sub-tree rooted in n, as discussed below.

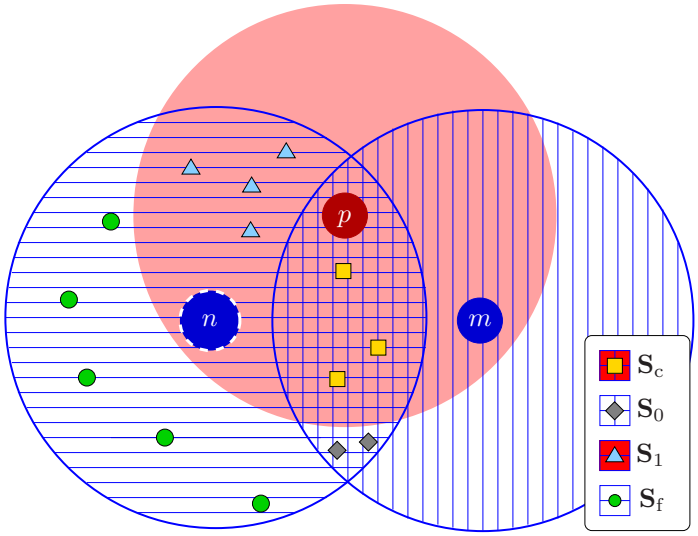


Figure 4.1: Sets $\mathbf{S}_c$, $\mathbf{S}_0$, $\mathbf{S}_1$ and $\mathbf{S}_f$ with respect to node n. Here $N_c = 3$, $N_0 = 2$, $N_1 = 4$ and $N_f = 5$. Only nodes n, m and p are active. The pattern in the background of the nodes' transmission area is parallel lines for description d_0 and solid for d_1.

Knowing the sets $\mathbf{S}_c$, $\mathbf{S}_0$, $\mathbf{S}_{1c}$, and $\mathbf{S}_f$, we would be able to compute $D(k)$ perfectly as:

$$D(k) = D_{\mathbf{S}_c} + D_{\mathbf{S}_0} + D_{\mathbf{S}_1} + D_{\mathbf{S}_f}. \tag{4.2}$$

Unfortunately, in the original protocol, n is not aware of which nodes in its sub-tree belong to which set; we show here how it is possible to estimate how many nodes belong to each set through the up-tree propagation of attachment messages.

For each video packet r, let us define $\eta_n(k)$ the delivery ratio for n given k as a retry limit, *i.e.*, the expected number of nodes in n's sub-tree receiving r sent by n with k as a retry limit, normalised by total number of nodes in the sub-tree. In the following, we shall omit the subscript identifying the node when unambiguous in the context. The delivery ratio models the fact that not all nodes in the sub-tree rooted in n will receive r, since we are dealing with a wireless (therefore lossy) network. We can define the distortion for a set of nodes as the sum of the distortions of all the nodes in the set. The distortion for sets $\mathbf{S}_c$ and $\mathbf{S}_0$ does not depend on the reception of r sent by n, since the nodes in those sets would still receive a copy of r and decode description d_0, and will be therefore omitted as it plays no role in the optimisation function. For set $\mathbf{S}_1$, only the nodes receiving r could decode both descriptions; the others would have to decode only description d_1. Similarly, the nodes in $\mathbf{S}_f$ receiving r could decode at least d_0, while the others would have to use a concealment strategy. Therefore, if we make the assumption that the delivery ratio for r is the same on both $\mathbf{S}_1$ and $\mathbf{S}_f$ (which is reasonable, since the two descriptions are sent independently), the two contributions to the total distortion are:

$$D_{\mathbf{S}_1} = N_1[\eta(k)D_c + (1 - \eta(k))D_1]$$

and

$$D_{\mathbf{S}_\mathrm{f}} = N_\mathrm{f}[\eta(k)D_0 + (1-\eta(k))D_\mathrm{f}].$$

We rewrite the sum of these two contribution as:

$$D_{\mathbf{S}_1} + D_{\mathbf{S}_\mathrm{f}} = +N_1 D_1 + \eta(k)N_1\left(D_\mathrm{c} - D_1\right) + N_\mathrm{f}D_0 + (1-\eta(k))N_\mathrm{f}\left(D_\mathrm{f} - D_0\right).$$

We now introduce $\Delta D_\mathrm{c} = D_1 - D_\mathrm{c} > 0$ and $\Delta D_\mathrm{f} = D_\mathrm{f} - D_0 > 0$, which measure the reduction in the distortion a node experiences, when passing from decoding d_1 only to decoding both, and from decoding no description (concealment) to decoding only d_0, respectively. These quantity are positive, since central decoding always outperforms side decoding, and side decoding always outperforms concealment, in terms of distortion [Goy01, WRL05]. Using these quantities, we rewrite the total distortion in equation (4.2) as:

$$D(k) = [1-\eta(k)]N_\mathrm{f}\Delta D_\mathrm{f} - \eta(k)N_1\Delta D_\mathrm{c} + N_\mathrm{c}D_\mathrm{c} + N_1 D_1 + N_\mathrm{f}D_0 + N_\mathrm{f}D_\mathrm{f},$$

where $N_\mathrm{c}D_\mathrm{c}$, $N_1 D_1$, $N_\mathrm{f}D_0$ and $N_\mathrm{f}D_\mathrm{f}$ do not depend on k, and can be therefore neglected in the optimization in problem (4.1). In order to solve the minimisation problem, a node has to estimate the remaining contribution to the total distortion: the distortion differences ΔD_c and ΔD_f, the delivery ratio $\eta(k)$, and the group sizes N_1 and N_f. Here, ΔD_c and ΔD_f depend on the particular codec used; they can be easily computed at the encoder – where both central and side distortion are known, and concealment distortion can be measured – and included in the video stream as headers, or they can be estimated, such as in our experimental setup; there exists simple and effective distortion models that allow a recursive estimation of the frames distortion [MPP08, MAPPF08, TPP09].

Estimating the delivery ratio and the group sizes is a more challenging task, as this information is distributed and time-varying. In order to have a reliable estimation, nodes have to deduce at least partially the topology of the overlay, beyond their transmission area (wherein they are able to collect information directly). In the following, we present a solution to this problem that relies on a small number of messages exchanged, sent through the same links used for streaming the video.

4.3.1 Group size estimation

In order to solve problem (4.1), nodes have to estimate the group sizes N_f and N_1. To this end, we introduce a model of our ad-hoc wireless network as a simple directed graph $\mathbf{G} \triangleq (\mathbf{E}, \mathbf{V})$, where the vertices $n_i \in \mathbf{V}$ represent the mobile nodes and the edges $e_{ij} \in \mathbf{E}$ the wireless links between two nodes n_i and n_j, *i.e.*, if $e_{ij} \in \mathbf{E}$, n_i and n_j are neighbours (with respect to definition given in Chapter 3 for the ABCD protocol). Note that edges in $\mathbf{E}$ are bidirectional: $e_{ij} \in \mathbf{E} \Leftrightarrow e_{ji} \in \mathbf{E}$.

An overlay produced by ABCD will converge, for each description, to a directed tree

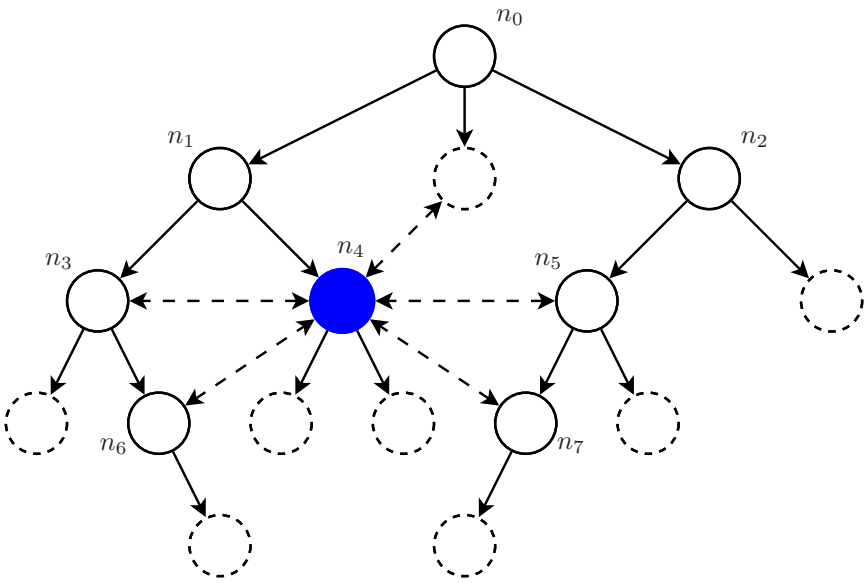


Figure 4.2: Example of directed graph of an ABCD overlay (all nodes, solid lines) for one description. Solid border represents active nodes. Neighbouring relations (dashed lines) are also represented for node n_4 only. Here, $\mathcal{B}_4 = \{n_5, n_6, n_7\}$, $\mu(n_4, n_5) = \mu(n_4, n_7) = n_0$, and $\mu(n_4, n_6) = n_1$. Being $h(n_0) = 0$ and $h(n_5) < h(n_7)$, we designate as foster parent for n_4 the node n_5.

$\mathbf{T}$, rooted in the source (labelled s), and spanning from $\mathbf{G}$. For the sake of simplicity, we shall hereby consider only a snapshot of the overlay topology at the time the packet has to be sent: $\mathbf{T} = (\mathbf{P}, \mathbf{V})$, where $\mathbf{P} \subset \mathbf{E}$ represents the links over which the video stream is sent, *i.e.*, the *parent-child* relation defined in Chapter 3 for the ABCD protocol.

We define now a set of useful relations between two nodes n_i and n_j on the ABCD overlay, and in particular for each description multicast tree, by using binary relations on $\mathbf{G}$: n_i and n_j are *neighbours* (which we denote by $n_i \curlyvee n_j$), if there exists in $\mathbf{E}$ a pair of edges e_{ij} and e_{ji} connecting them; n_i is the *parent* of n_j ($n_i \prec n_j$), if there exists in $\mathbf{P}$ an edge e_{ij}; conversely n_i is a *child* of n_j ($n_i \succ n_j$), if there exists in $\mathbf{P}$ an edge e_{ji}; n_i and n_j are *siblings* ($n_i \sim n_j$), if there exists in $\mathbf{V}$ a node n_k such that there exist in $\mathbf{P}$ two edges e_{ki} and e_{kj}, *i.e.*, they have the same parent. We also define n_i as an *ancestor* of n_j ($n_i \ll n_j$) if there exist a simple directed path from n_i to n_j, in which case we also define n_j as a *descendant* of n_i ($n_j \gg n_i$). Let us also denote by $h(n_i)$ the number of hops that a packet sent by the source has to cross in order to be received by n_i, *i.e.*, the value h used by the neighbours of node n_i that are evaluating it as a candidate parent in the objective function (3.1) of Chapter 3. An example of ABCD overlay for one description, with some examples of relations between nodes, is presented in Fig. 4.2.

In order to estimate N_f and N_1, we analyse the impact of a node skipping the transmission of a packet. We introduce in the protocol the notion of *foster parent*: the idea is to have each node collect information about possible alternative paths from which the description can be received, then transmit it to its current parent, thus allowing the parent

n_i	n_7	n_6	n_5	n_4	n_3	n_2	n_1
δ_i	n_0	n_1	n_0	n_0	n_1	n_0	n_0
c_i	2	2	4	3	4	6	8
x_i	0	0	0	0	0	0	0
φ_i	n_4	n_4	n_4	n_5	–	–	–

Table 4.1: Dependency records generated by the active nodes in Fig. 4.2 and designated foster parents.

to estimate the impact of its decision of not sending a packet. This is a complex problem, as no node has a global view of the topology (nor it should, as propagating the whole topology to all active nodes each time it changes would congest the channel). Our contribution consists in a technique to reliably estimate the group sizes using the nodes' local information plus a small amount of information propagated through the overlay.

Let us call $\mathbf{V}_\mathrm{A} \subseteq \mathbf{V}$, the set of active nodes, *i.e.*, the nodes that are relying the current description, and are therefore parents of at least one node; for each node n_i, we define its set of *candidate foster parents* $\mathcal{B}_i \subset \mathbf{V}_\mathrm{A}$ as:

$$\mathcal{B}_i \triangleq \{\, n_j \in \mathbf{V}_\mathrm{A} \mid (n_i \curlyvee n_j) \wedge (n_i \not\ll n_j) \wedge (n_i \not\gg n_j) \wedge (n_i \not\sim n_j) \}\,,$$

which is the set of active neighbours of n_i that are neither ancestors, nor descendants, nor siblings of n_i (*e.g.*, in Fig. 4.2, $\mathcal{B}_4 = \{n_5, n_6, n_7\}$). Each node has perfect knowledge of its set of candidate foster parents. In fact, since candidate parents are both neighbours and active, the node receives their attachment messages piggybacked in their video packets, and from these it can infer their parent. Comparing its neighbours' parent with its own, it can deduce whether they are siblings. Furthermore, each node obviously knows both its parent and children. Finally, as mentioned inf Chapter 3, it cannot have in its neighbourhood other ancestors than its parent (and, by symmetry, no descendant other than its children).

The existence of a node $n_j \in \mathcal{B}_i$ assures that n_i would stay connected even if the link with its parent were removed. Also, it is possible that it would stay connected even if the link between its parent and its grandparent were removed: this would hold true as long as the grandparent of n_i were not an ancestor of n_j. In order to evaluate the degree of robustness of the path that n_j provides to n_i, we want to investigate the *common ancestors* the two share. To this end, let us observe that any couple of nodes in $\mathbf{V}$ has at least one common ancestor, *i.e.*, the source. However, nodes can have more common ancestors, *e.g.*, two siblings share a number of ancestors equal to their height in the tree. Let us define:

$$\mathcal{C}_{ij} \triangleq \{n_k \in \mathbf{V}_\mathrm{A} \mid (n_k \ll n_i) \wedge (n_k \ll n_j)\}\,,$$

which is the set of common ancestors of n_i and n_j (*e.g.*, in Fig. 4.2, $\mathcal{C}_{4,6} = \{n_1, n_0\}$). In this set, we can identify the *most recent common ancestor* μ_{ij} as $\mu(n_i, n_j) \triangleq \arg\max_{\mathrm{n_k} \in \mathcal{C}_{\mathrm{ij}}} \{h(n_k)\}$, which is the common ancestor of n_i and n_j with the longest path from the source (*e.g.*,

in Fig. 4.2, $\mu_{4,6} = n_1$). Let $n_c = \mu_{ij}$; any of the $h(n_c)$ ancestors of n_c will also be in C_{ij}, and any of its descendants will not. In other words, we can assume that, from n_c to n_i, the two paths (one through its current parent, the other through n_j), are affected by independent failures, *i.e.*, no failure of a single node between n_c and n_i could severe both paths. Therefore, we are interested in the neighbour n_j such that μ_{ij} is as close as possible to the source. This neighbour is called *designated foster parent* and denoted by φ_i. More formally:

$$\begin{cases} m_i & \triangleq & \min\limits_{n_j \in \mathcal{B}_i} h\left(\mu(n_i, n_j)\right), \\ \mathcal{M}_i & \triangleq & \{n_j \in \mathcal{B}_i \mid \mu(n_i, n_j) = m_i\}, \\ \varphi_i & \triangleq & \arg\min\limits_{n_j \in \mathcal{M}_i} \{h(n_j)\}. \end{cases}$$

The rationale behind this choice is to minimise the number of *critical* nodes, *i.e.*, nodes that would cause the failure of both regular and alternative paths if deactivated. In order to find φ_i, a node must be able to compute the number of ancestors it has in common with its neighbours; this can be done by comparison if each node adds in its attachment messages the sequences of its ancestors, which can be easily generated as it suffices that any active node adds its identifier to the sequence it receives from its parent. The amount of data exchanged to propagate the sequences is small, as the ABCD trees tend to be short and wide. It should be noted that, even in the original version of ABCD, upon disconnection of their current parent, nodes could still receive data from paths unsevered by the disconnection. However, this mechanism was implicit and, more importantly, the existence of alternative paths was not propagated through the overlay, therefore other nodes could not rely on this information to make any decision. Even though all alternative paths existing in the original protocol still exist, we *designate* one and advertise information about this designation to allow other nodes to benefit from it.

In the estimation of groups size, we shall assume that each node n_i designates a single foster parent φ_i. The reason is that, even though in principle *any* node in $\mathcal{B}_i$ provides an alternative path, it is unlikely that in case of failure of both the regular and alternative path, due to the disconnection of a node in their common path, other paths in $\mathcal{B}_i$ could still be active, since the protocol tries to concentrate subscriptions on as few nodes as possible, and if a node fails that is a common ancestor of n_i and φ_i, it is likely to be an ancestor of the others nodes in $\mathcal{B}_i$ as well.

The presence of an alternative path allows a node to receive a packet even though its parent decided not to send it or was unable to obtain the channel. These alternative paths determine the groups $\mathbf{S}_1$ and $\mathbf{S}_f$, whose sizes (N_1 and N_f) we want to estimate. In order to do so, we need to spread the information about the existence of these alternative paths through the overlay tree. This information, however, has to be refined while it is spread from the leaves towards the source, in order to prevent congestion. The propagation of the information about alternative paths works as follows. First, each node n_i finds its most recent ancestor in common with its designated foster parent, called *path dependency node*

and denoted δ_i. By convention, a node with no alternative path defines its current parent as path dependency node.

$$\delta_i \triangleq \begin{cases} \mu(n_i, \varphi_i) & \text{if } \mathcal{B}_i \neq \emptyset, \\ n_k \mid n_k \prec n_i & \text{otherwise.} \end{cases}$$

The meaning of δ_i is that n_i has an alternative path that is independent from the current path up to δ_i. See Fig. 4.2 and Tab. 4.1 for an example of generation of the dependency records, with respect to the topology in Fig. 4.2.

Once a node has computed its path dependency node, it has to transmit to its parent the information about the path dependencies. This is done using *dependency records* (see Tab. 4.1), which we define as $\mathbf{d_i} \triangleq [\delta_i, c_i, x_i]$, where c_i is the number of nodes in the sub-tree rooted in n_i sharing the same dependency, and x_i is a flag signalling whether the nodes are also receiving the complementary description. In order to explain how dependency records work, we shall now describe how they are generated and propagated through the tree. Dependency records are generated by each leaf n_i as $\mathbf{d_i} = [\delta_i, 1, x_i]$, one for each description it receives, then sent it to its parent. The parent of n_i interprets this record as follows: for description d_0, n_i has an independent path up to δ_i and it is the only one ($c_i = 1$) in its sub-tree having this dependency node; also, it has (if $x_i = 1$), or has not (if $x_i = 0$), the complementary description d_1. Another similar dependency record is generated for d_1. Let us assume that n_j is the parent of n_i; if n_i has no alternative path or its path dependency node corresponds to the parent of n_j, then n_j updates the record, replacing the path dependency node of n_i with its own. The logic is simple: n_i stays connected as long as n_j is connected; if n_j has a path alternative to its current one, then it is able to restore the path from the source to n_i even if its current path fails. Therefore, for each child's dependency record $\mathbf{d_i}$, the parent node n_j generates an *updated dependency record*:

$$\mathbf{d_i}' \triangleq \begin{cases} \mathbf{d_i} & \text{if } \delta_i \neq n_j \text{ and } \delta_i \not\prec n_j, \\ [\delta_j, c_i, x_i] & \text{otherwise.} \end{cases}$$

In other words, $\mathbf{d_i}'$ represents the dependency record of n_i, with the addition of the knowledge of the topology contributed by n_j: if n_j is the path dependency node for n_i, it can update this information.

Node n_j needs now to transmit an aggregated information about its own alternative paths and its children's. In order to do so, it generates two *aggregate dependency records*, one for the children receiving the complementary description, and the other for those not receiving it:

$$\mathbf{s_j}^{\mathbf{k}} \triangleq \sum_{i \mid n_i \succ n_j \wedge x_i = k} \mathbf{d_i}', \qquad \forall k \in \{0, 1\},$$

where the sum denotes the composition of two records having the same value of the flag

x_i, defined as:

$$[c_i, \delta_i, x_i] + [c_j, \delta_j, x_i] \triangleq \left[(c_i + c_j),\, \arg \min_{\delta_k \in \{\delta_i, \delta_j\}} \{h(\delta_k)\}, x_i\right].$$

This rule of composition respects the semantic of the record: in fact, the number of depending nodes is obtained as the sum of the contribution of each record, while the path dependency node is chosen consistently with the definition of most recent common ancestor given above. In other words, in these records, the new path dependency node for the set of children sharing the same value of x_i is the one with the shortest height. Aggregation of dependency records is needed, as it transforms a series of local views into a more descriptive global information: n_j has to propagate its knowledge up-tree to facilitate its ancestor in making decisions that affect the whole sub-tree. Thus, at higher levels, a global optimisation is performed, using an aggregate information on the descendants. At lower levels, active nodes may or may not operate the same choice as their ancestors, since they will now use a more and more detailed local information.

Finally, n_j will add its own contribution to the proper aggregate dependency record, in accordance with the value of x_j:

$$\begin{cases} \mathbf{d_j}^{\mathbf{x_j}} & \triangleq & \mathbf{s_j}^{\mathbf{x_j}} + [1, \delta_j, x_j] \\ \mathbf{d_j}^{\mathbf{1-x_j}} & \triangleq & \mathbf{s_j}^{\mathbf{1-x_j}} \end{cases}$$

Node n_j will then propagate both $\mathbf{d^0}_j$ and $\mathbf{d^1}_j$.

Using the records received from its children, an active node n_i can estimate the group sizes as follows:

$$\begin{aligned}
\widetilde{N}_{\mathrm{c}} &= \sum_{j \in \mathcal{C}_{\mathrm{c}}} c_j && \text{with } \mathcal{C}_{\mathrm{c}} = \{j \mid n_i \prec n_j \wedge \delta_j \ll n_i \wedge x_j = 1\}; \\
\widetilde{N}_{1} &= \sum_{j \in \mathcal{C}_{1}} c_j && \text{with } \mathcal{C}_{1} = \{j \mid n_i \prec n_j \wedge \delta_j \not\ll n_i \wedge x_j = 1\}; \\
\widetilde{N}_{0} &= \sum_{j \in \mathcal{C}_{0}} c_j && \text{with } \mathcal{C}_{0} = \{j \mid n_i \prec n_j \wedge \delta_j \ll n_i \wedge x_j = 0\}; \\
\widetilde{N}_{\mathrm{f}} &= \sum_{j \in \mathcal{C}_{\mathrm{f}}} c_j && \text{with } \mathcal{C}_{\mathrm{f}} = \{j \mid n_i \prec n_j \wedge \delta_j \not\ll n_i \wedge x_j = 0\}.
\end{aligned}$$

In other words, n_i verifies, for each child n_j, if the dependency for the alternative path of n_j is satisfied, *i.e.*, if the path dependency node declared in the record is one of its ancestors (rather than n_i itself); if this is the case, then it is understood that c_j nodes in the sub-tree rooted in n_j are able to receive d_0 even if n_i is deactivated on that description. These c_j nodes are therefore accumulated – depending on the value x_j declared in the record – either on $\widetilde{N}_{\mathrm{c}}$ (if $x_j = 1$) or on $\widetilde{N}_0$ (if $x_j = 0$). On the other hand, if the path dependency node is not an ancestor of n_i (*i.e.*, it is n_i itself), then the alternative path

is invalidated, and the c_j nodes are assumed to be unable to receive d_0 if n_i deactivates; they are therefore accumulated on either $\widetilde{N}_1$ (if $x_1 = 1$) or $\widetilde{N}_\mathrm{f}$ (if $x_1 = 0$).

In Tab. 4.2 we present an example of group sizes estimation, with respect to the overlay depicted in Fig. 4.3. For the sake of simplicity, let us assume that $x_i = 0$ for all nodes. The estimation of the group size differs slightly from the actual values (namely for n_4 and n_5, with an error of one unit), as nodes are unable to determine their common ancestor over *all* the paths of their designated foster parent. In the example, n_7 assumes n_4 as dependency node for its alternative path through n_{11}, therefore n_4 invalidates it when estimating the group sizes; however, even if n_4 deactivates, n_7 could still actually receive the description through the path $n_0 \to n_2 \to n_5 \to n_8 \to n_{11} \to n_7$, *i.e.*, through the alternative path of n_8. However, even with this simplification – which reduces the amount of information exchanged – the reliability of the estimation is not affected much, as we see in the example, where the small errors on the estimation are corrected upper in the tree. This is a point of strength of the protocol: errors do not propagate through the whole tree; there may be local errors, but they tend to be corrected as dependency records propagate, as they are enriched with new topological information.

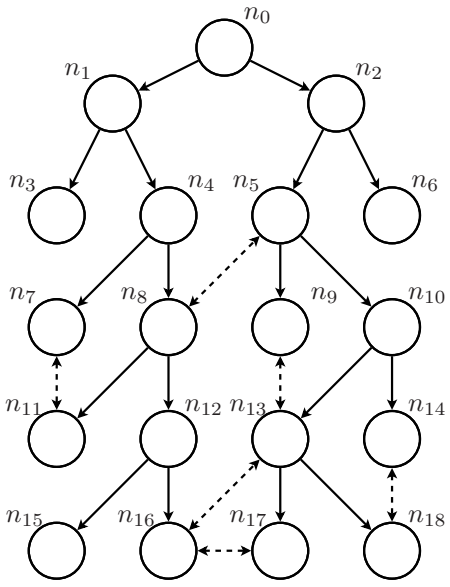


Figure 4.3: Example of alternative paths on an ABCD overlay for one description (only active nodes are depicted). Neighbouring relations (dashed lines) are also represented, for couples of nodes not connected by a parent-child or sibling relation.

4.3.2 Delivery ratio estimation

We shall now discuss how $\eta_n(k)$ is estimated. To this end we make the following assumptions:

1. Before each video packet is sent, the sender transmits an RTS message to the control peer.

n_i	n_{18}	n_{17}	n_{16}	n_{15}	n_{14}	n_{13}	n_{12}	n_{11}	n_{10}	n_9	n_8	n_7	n_6	n_5	n_4	n_3	n_2	n_1	n_0
φ_i	n_{14}	n_{16}	n_{13}	–	n_{18}	n_9	–	n_7	–	n_{13}	n_5	n_{11}	–	n_8	–	–	–	–	–
δ_i	n_{10}	n_0	n_0	n_{12}	n_{10}	n_0	n_0	n_4	n_0	n_5	n_0	n_4	n_2	n_0	n_0	n_1	n_0	n_0	–
c_i	1	1	1	1	1	3	3	1	5	1	5	1	1	7	7	1	9	9	19
$\widetilde{N}_0$	–	–	–	–	–	1	1	–	3	–	4	–	–	**5**	**5**	–	7	7	0
N_0	–	–	–	–	–	1	1	–	3	–	4	–	–	6	6	–	7	7	0
$\widetilde{N}_\mathrm{f}$	–	–	–	–	–	1	1	–	1	–	0	–	–	**1**	**1**	–	7	7	0
N_f	–	–	–	–	–	1	1	–	1	–	0	–	–	0	0	–	1	1	18

Table 4.2: Values used for estimation of groups size in Fig. 4.3. We report here the designated foster parents φ_i, the fields of the dependency records, δ_i and c_i ($x_i = 0$ for all nodes), the estimated values $\widetilde{N}_0$ and $\widetilde{N}_\mathrm{f}$, and the actual values N_0 and N_f.

2. If the sender receives a CTS message from the control peer, then it gains exclusive access to the channel and the video packet shall be received correctly by the control peer.

3. If the control peer has correctly received a video packet, to which it replies with an ACK message, then all designated receivers have correctly received the same packet; *i.e.*, when the sender receives an ACK from the control peer, it can be inferred that all the children received the data packet.

The first assumption is enforced by the ABCD protocol itself, the other two are common assumptions justified by the way the RTS/CTS/ACK mechanism of IEEE 802.11 works. In practice, there exists a minor fraction of nodes not receiving a video packet even if it has been acknowledged by its control peer, in certain topologies. However, these events are always limited in number of both nodes and packets, since the ABCD parent switch mechanism tends to avoid these topologies in the first place. Also, in a scenario with node mobility, these pathological topologies are necessarily transient. Of course, in a wireless environment, there is always the possibility that one or more descendants of a node do not actually receive the packet because of fading. However, on one hand 802.11 provides several tools to reduce this problem. On the other, the only effect that our assumption could have on the optimisation process is a slight overestimation of $D(k)$, which does not necessarily translate into a wrong selection of the retry limit, since the group size estimation can be affected by a small error of the opposite sign, and k is selected into a discrete and relatively small set, therefore small variations of D are drowned by the quantisation on k. Finally, the soundness of these assumptions is supported by experimental evidence both in the articles proposing broadcast reservation [MKK01, YYRZ11] and in tests performed on ABCD itself [GC11].

Let us consider a node n that has at least one child but no grandchildren. We call p_n the probability of n obtaining the channel with a single try, which, under the assumptions made above, the node can estimate by just using its video packets as probes for the channel around itself, with an exponentially-weighted moving average of the number of received

ACKs divided by the number of RTS sent. Node n estimates $\eta_n(k)$ as $1-(1-p_n)^k$, then sends the value of $\eta_n(1) = p_n$ to its parent. Note that here we are also assuming that if one child of node n receives a packet, all of the children of n also do, consistently with the assumption made above.

Let us now consider a second node m that has at least one grandchild, *e.g.*, the parent of n; m will receive from each of its children c its delivery ratios η_c and the number of its descendants g_c.

If with a single try m were able to obtain the channel, the message it would send could be received by its g_m children and, by inductive hypothesis, any node c child of m would reach in its turn $\eta_c(1)g_c$ of its g_c descendants. The value $\eta_m(1)$, can therefore be estimated as:

$$\eta_m(1) = p_m \times \frac{g_m + \sum_{c \succ m} \eta_c(1) g_c}{g_m + \sum_{c \succ m} g_c}.$$

This formula can be read as follows: if m obtains the channel (which happens with probability p_m), then the packet is received by its g_m children, plus (on average) $\eta_c(1)g_c$ descendants through each child c, out of the total number of its descendants ($g_m + \sum_{c \succ m} g_c$). All other values of $\eta_m(k)$ are then estimated as $\eta_m(k) = 1 - [1 - \eta_m(1)]^k$.

The values of g_c are already part of the node state piggybacked in attachment messages and the values of η_c can also be transmitted in the same way, so no congestion is generated in order to transfer this information. Also, piggybacking in attachment messages assures us that the information is always up-to-date in case topology changes.

The estimation of the delivery rate completes the set of values needed required to solve the constrained minimisation problem (4.1). With all the parameters available, each node can decide its value of k for the current packet by simply evaluating $J(k)$ forall k up to a maximum value. The choice of the maximum k is presented in the following section, together with an extensive experimental validation of the proposed framework.

4.4 Experimental study

4.4.1 Experiment settings

In this section, we present the results of the performance tests of the proposed CoDiO extension [GCPP12], in comparison with the conventional ABCD implementation [GC11]. Like in Chapter 3, the ad-hoc mobile network has been simulated using the *ns-2* discrete event simulator [ns2].

A set of nine video sequences ("Akiyo", "Foreman", "Mobile", "Football", "Bus", "Flower", "City", "Coastguard", and "Stefan"; CIF at 30 fps; concatenated then looped to match the total simulation time of 300 s) has been encoded in multiple descriptions using the

channel splitting technique presented in Sec. 2.2, with a coding rate of about 1.8 Mbps, resulting in an average PSNR of 39.94 dB for central decoding and 35.20 dB for each of the two side decoding. The PSNR values for each sequence are reported in Tab. 4.3.

Sequence	Central PSNR	Side PSNR
"Akiyo"	43.96 dB	42.71 dB
"Foreman"	40.62 dB	37.41 dB
"Mobile"	38.48 dB	33.56 dB
"Football"	40.01 dB	31.62 dB
"Bus"	38.89 dB	31.28 dB
"Flower"	39.67 dB	35.11 dB
"City"	39.71 dB	36.30 dB
"Coastguard"	38.67 dB	35.26 dB
"Stefan"	39.48 dB	33.53 dB
Average	39.94 dB	35.20 dB

Table 4.3: Sequences used in the simulations. Average PSNR for side and central decoding are reported for reference.

The parameter λ in the minimisation problem (4.1) has been chosen experimentally to have a value of $\lambda = 1.4$, by maximising the average video quality (in terms of PSNR) of the decoded sequences over a large set of simulations. However, our preliminary tests with several others values of λ in a large interval showed that the technique is quite robust with respect to the choice of this parameter, in the sense that under small perturbations of the value of λ, only small variations of the average video quality are observed.

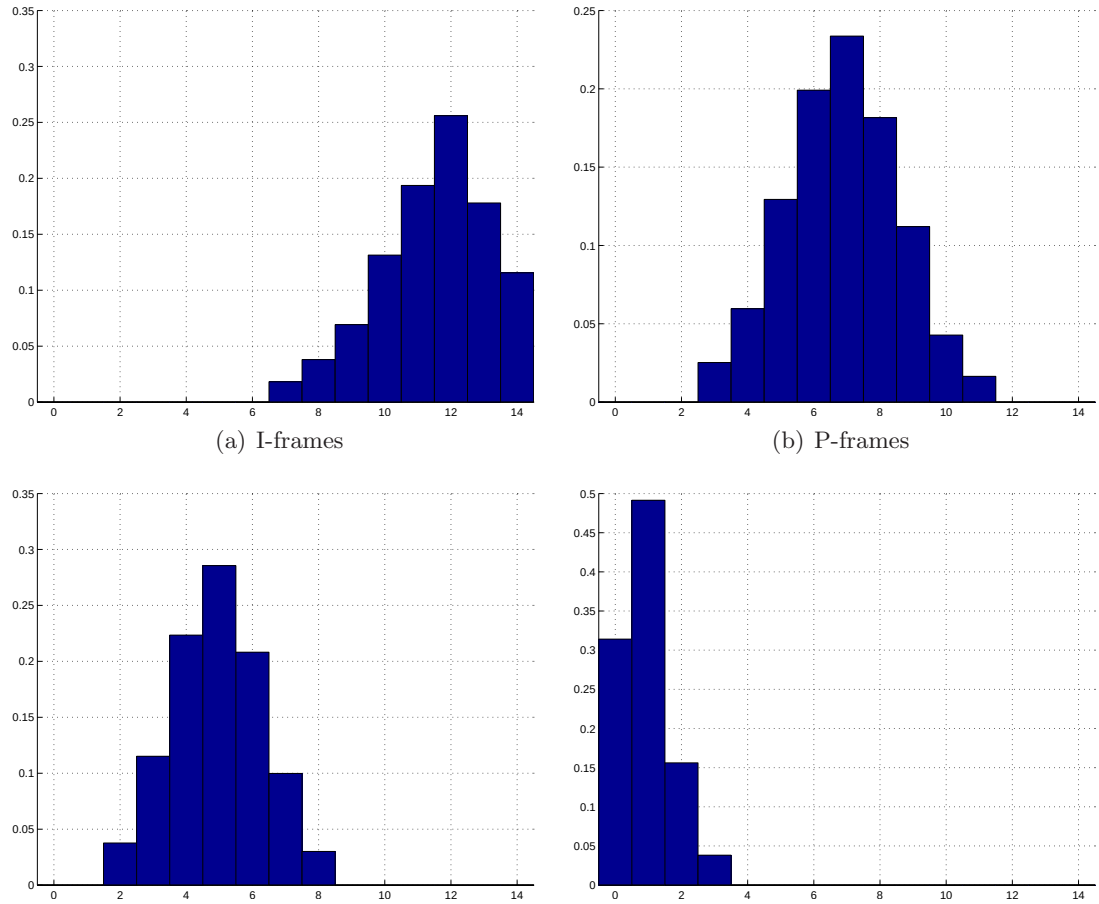
Also, for evident implementational reasons, problem (4.1) cannot be solved testing all $k \in \mathbb{N}$; in practice, it is reasonable to assume that the optimal value of k must lie in a interval $[0, k_{\max}]$, with $k_{\max}$ relatively small, since a time too long to gain access to the channel would result in the packet being dropped for lateness. In order to find a suitable value for $k_{\max}$, we ran several simulations with very large values of $k_{\max}$, in order to be sure that the optimum is not missed; we found that the optimal values for k always lay between 0 and twice the limit prescribed by the standard, $2 \times 7 = 14$; the following simulations are therefore run with $k_{\max} = 14$.

In these experiments, we compare the two versions of the protocol (*i.e.*, plain ABCD and ABCD with CoDiO extension) in a network with 100 nodes and a density of 40 nodes per neighbourhood. This is an extremely high density, chosen in order to appreciate the capability of the proposed framework to deal with very harsh conditions of the network (*e.g.*, a group of rescuer rushing towards the injured in a disaster, or a maniple of soldiers converging on a target). Tests performed at different densities showed that the lower the density is, the more similar the performances of the two protocols are, which was to be expected, as the proposed framework is specifically designed for high densities. Also, as mentioned in Sec. 3.5, this is also the maximum density in which the conventional ABCD

protocol is able to deliver 100 % of the frames.

The simulation time is 5 minutes, but the first 5 seconds are not considered, in order to compare the performance of the protocols in a steady state.

As a first observation, in Fig 4.4, we show the normalised histogram of the occurrences for the selected values of the retry limit k. For the sake of clarity, an histogram is presented for each types of frame in the GOP, which gives a rough estimation of the different contribution to the overall distortion. We observe that for I-frames and P-frames, whose loss have a great impact on distortion, especially since they are used for reference for other frames, the distribution of values of k is concentrated around high values, reflecting the fact that the framework is trying to provide them with a greater protection against losses. Conversely, for B-frames the optimisation typically chooses smaller values, as their contribution to the overall distortion is too small to justify the congestion generated by repeated tries to access the channel. In the following, we shall analyse how this choice of k impacts the delay and the distortion of the decoded video sequence.



(a) I-frames

(b) P-frames

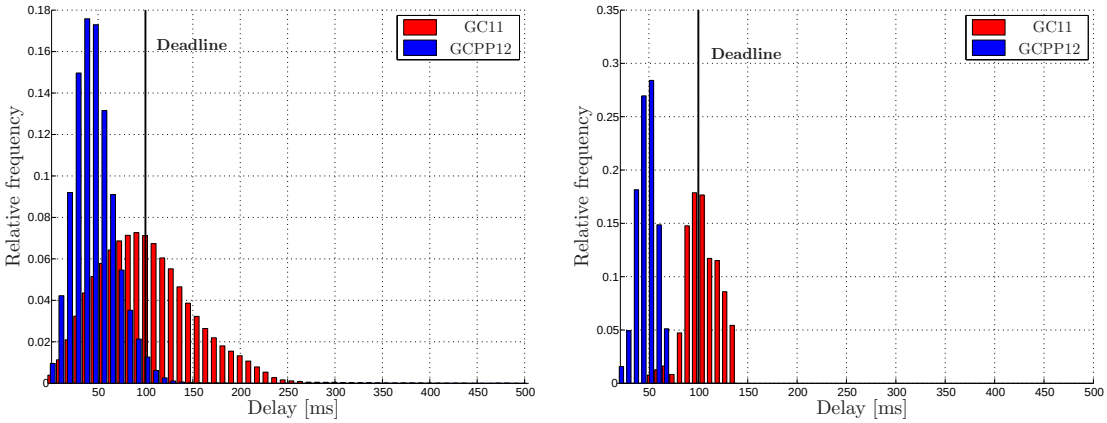
(c) B-frames (first prediction level)

(d) B-frames (second prediction level)

Figure 4.4: Normalised histogram of the occurrences for the selected values of the retry limit for the different types of frames.

4.4.2 Delay analysis

In Fig. 4.5(a), we compare the histogram of the frame delay for the two versions of the protocol, collected from all the nodes in the simulation. Note that any frame with a delay higher than 100 ms (vertical bar) would not abide to the conversational pattern, and is therefore dropped. We can notice that, in the version of the protocol without CoDiO extension, more than one half of the frames are too late to be decoded (55 %), while in the proposed version, only a light tail of the histogram (2.7 %) crosses the deadline. This means that, in the reference technique, one half of the received packets are dropped as useless, even though the channel resources for their transmission have been spent. On the other hand, using the proposed extension, the cost-benefit analysis provided by the congestion-distortion optimisation allows the nodes to exploit the channel more efficiently.



(a) Delay of a generic frame.

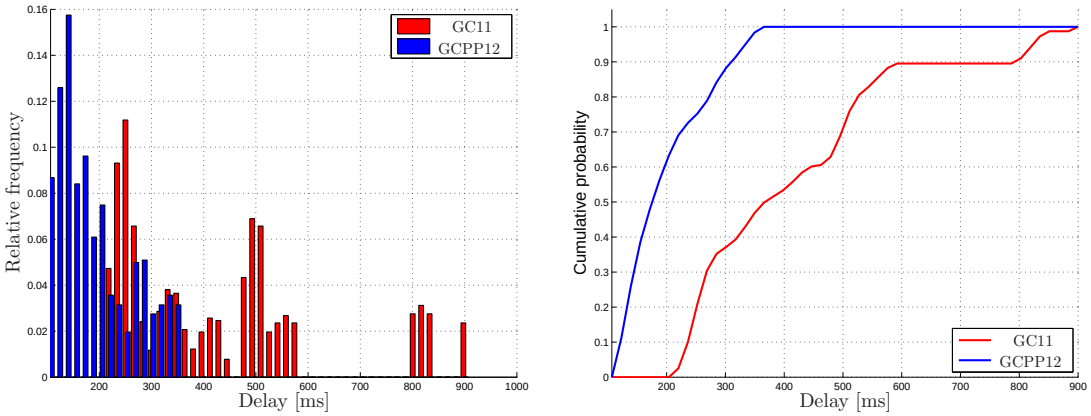
(b) Average delay of frames at a generic node.

Figure 4.5: Histogram of frame delay. The vertical bar marks the maximum delay for frame decoding in the case of conversational pattern.

From the distribution of the the average delay of frames at a generic node, shown in Fig. 4.5(b), we also observe that, using the reference ABCD version, about one half of the nodes have an average delay higher than the 100 ms threshold, while using the proposed extension no node experiences an average delay higher than 70 ms.

We also computed the maximum delay experienced by each node, reported in in Fig. 4.6. For the sake of clarity, both the relative frequency (Fig. 4.6(a)) and the cumulative distribution ((Fig. 4.6(b))) of the maximum frame delay are reported.

We observe that, because of the congestion, with ABCD all nodes experience at least once a delay larger than 200 ms. Using CoDiO largely limits this problem, and many nodes never experience a large delay, over 40 % of nodes never experience a delay larger than 150ms, 75 % of nodes never experience a delay larger than 250 ms, and no node has a single frame delayed more than 350 ms. Moreover, as we observe in Fig. 4.5(b), even nodes that experience a relatively high maximum delay, have a much lower average delay.



(a) Histogram of maximum frame delay.

(b) Cumulative distribution of maximum frame delay.

Figure 4.6: Maximum delay perceived by a generic node.

In other words, the maximum delay is very unlikely.

Being discarded because of lateness is of course not the only reason for a frame not to be decoded by a node. A frame could have not been scheduled for sending in the first place (which can happen in the proposed extension only), or have exhausted all the requests granted without obtaining the channel (both versions). This results in the fact that for some images a node receives both, or one, or none of the descriptions. To keep into account all these effects, in Tab. 4.7 we compare the two versions of the protocol in terms of percentage of use of central decoding, side decoding, or concealment. This has been done labelling, in the network simulation, the video packets with the description they belong to and the frames they contain, then verifying off-line for each node and for each frame whether one, both or none of the packets were received before the deadline.

The reference technique uses central decoding for 73 % of frames and side decoding for 19 %, while the proposed technique uses central decoding 94 % of frames and side decoding for 5 % (concealment is used for the remaining frames). This result is mostly network-related and is almost completely independent on the MDC technique used (it is affected only by the inter-frame dependency due to the predictive structure, see the definition of **I** in Sec. 4.3); this because we assume that using a particular MDC technique affects only the length of video packets, which has a negligible effect on contention over channel access.

4.4.3 Video quality analysis

How the use of a decoding strategy reflects on the video quality depends on the codec used and the concealment strategy employed; therefore, in order to analyse the effects of the optimisation on video distortion, we need to define the video codec we use. We test the video quality with respect to an MDC technique proposed by the authors [GCPP10],

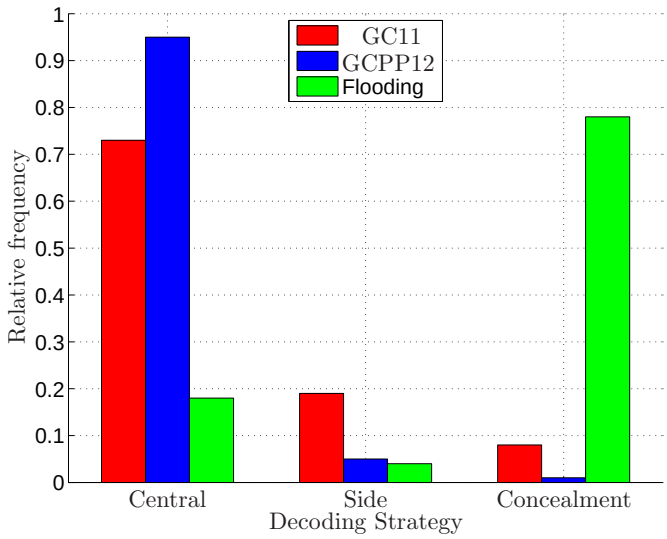
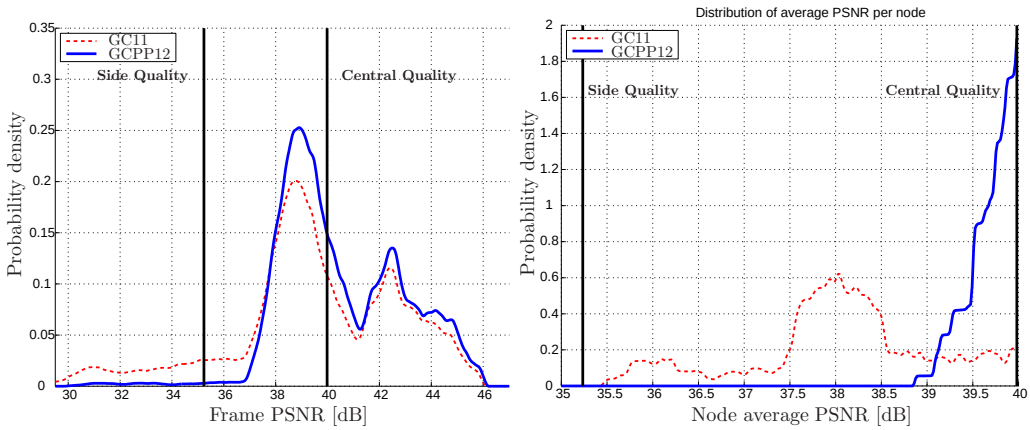


Figure 4.7: Percentage of usage of decoding strategies.

employing frame freezing as concealment. In Fig. 4.8(a) we compare, for the two versions of the protocol, the probability density functions of PSNR per frame, *i.e.*, considering the PSNR of each frame decoded by each node as a realisation; as in Chapter 3, the results are presented using an estimation of the probability density functions obtained with the Parzen window method [Par62].



(a) Probability density of the PSNR of a frame at a generic node.

(b) Probability density of the PSNR of the video sequence at a generic node.

Figure 4.8: Probability density function of the PSNR. The vertical bars in mark the average video PSNR for the sequence when decoded with side (leftmost) and central (rightmost) decoder.

Both distributions have roughly the same shape; however, in the reference scheme, nodes decode a generic frame with a low PSNR with a higher probability, for an average of 38.2 dB, while in the proposed technique nodes are much more likely to decode with a

high PSNR, for an average of 39.7 dB, *i.e.*, 0.24 dB short of the average central quality, clearly better than the reference. This means that using the reference technique in this configuration, the effects of unreliable transmission are visible to the user, whereas in the proposed technique they are barely perceptible.

Now we consider, for each single node, the average PSNR of the locally decoded video sequence. In Fig. 4.8(b) we show the distribution of the per-node PSNR; here, we see how the reference technique corresponds to an almost flat distribution, bounded between the average quality of side and central decoding. This implies that the video quality is acceptable for all nodes, but some of them have an average PSNR much smaller then others. On the other hand, the proposed technique has a much more peaked distribution, meaning that all nodes achieve a very high quality, with the modal value corresponding to the maximum quality. In other words, even though some frames are decoded with a relatively small PSNR (as shown in Fig. 4.8(a)), this hardly happens repeatedly to the same nodes; as a consequence, almost all nodes have an average PSNR over the sequence that is close to the maximum. The difference in dispersion can be better quantified by the inter-quartile range, presented in Tab. 4.4: in the reference technique the 25th node, ordered by decreasing PSNR, has an average PSNR almost 1.0 dB higher than the 75th, whereas in our proposed technique this range is in the order of 0.3 dB.

Quartile	$\mathcal{Q}_1$	$\mathcal{Q}_2$	$\mathcal{Q}_3$	IQR
GC11	37.47 dB	37.92 dB	38.46 dB	**0.99** dB
GCPP12	39.52 dB	39.64 dB	39.84 dB	**0.32** dB

Table 4.4: Quartiles of the average PSNR with the corresponding inter-quartile range (IQR).

In summary, these results show a significant gain both in terms of PSNR and in average end-to-end delay, while the delivery rate is kept close to 100 %, making the technique suited for conversational video applications over mobile ad-hoc networks. These results have been obtained in experimental conditions of high bitrate, high density, and large number of nodes, *i.e.*, conditions prone to generate a severe congestion on the channel; also, a stringent constraint on delay has been imposed. Tests have been performed in less harsh scenario as well, but – even though the proposed technique is never out-performed by the reference technique – the gain is less and less significant in situations where congestion is less relevant (because a longer delay is accepted) or less likely to occur (because the node density and the bitrate are small); this of course depends on the fact that this framework is designed to grant conversational delivery in congested networks, and is unnecessary in more tolerant and less crowded networks.

4.5 Conclusions

In this chapter, we addressed the delivery of a video stream, encoded in multiple descriptions, in a mobile ad-hoc environment, while dealing with low-latency constraints. This kind of application is meant to provide an efficient and reliable video communication tool in scenarios where the deployment of an infrastructure is not feasible, such as military and disaster relief applications.

Based on the ABCD protocol as presented in Chapter 3, we introduced a cross-layer congestion control strategy, where the MAC layer is video-coding aware and dynamically adjusts its transmission parameters (namely, the RTS retry limit) via constrained optimisation of congestion and distortion.

The main challenge in this approach consists in providing a reliable estimation of congestion and distortion, given the limited information available at each node. We proposed models for congestion and distortion that take into account the video coding structure as well as the topology of the overlay network. In particular, for distortion estimation, we introduced an efficient way to propagate information about possible paths alternative to the multi-tree overlay, in order to classify the nodes in groups differently affected by the loss of a packet. The total distortion is then estimated by weighting the expected distortions of each groups with the estimated number of receiving nodes in each group. This allows making a reliable prediction on the consequences of sending a packet with a particular retry limit, thus optimising video transmission in a CoDiO sense.

Our simulations show that, if a stringent constraint of low delay is imposed, our technique grants a consistent gain, in terms both of PSNR and of delay reduction, for bitrates up to a few megabits per second, compatible with a conversational service. This technique, and the relative results, have been accepted for publication in the IEEE Transactions on Multimedia.

Chapter 5

Network coding for video delivery over unreliable networks

Contents

In this chapter, we shall discuss *Network Coding* (NC) [ACLY00], a paradigm in which, in order to rely traffic for a multi-hop communication, the nodes of a network may combine several packets of information together for retransmission, instead of merely relaying the packets they receive. This approach has proven to be able to achieve the maximum possible information flow in a network.

First, in Sec. 5.1, we shall present the basic principles of NC, and review some of the most promising approaches to achieve it. Then, in Sec. 5.2, we shall see how network coding can be applied to video streams, encoded with both single and multiple descriptions, in order to provide a more reliable video streaming application over an unreliable network, such as the mobile ad-hoc networks presented in Chapter 3.

Our own contribution to this field, a novel combined multiple descriptions and network coding based technique for real-time video delivery over ad-hoc networks, will be presented

in Sec. 5.3. Finally, an experimental validation of our proposed contributions is presented in Sec. 5.4.

5.1 Network Coding

Since the very beginning of the design of the Internet Protocol (IP), one of the key guidelines has been that, in a multi-hop communication, intermediate nodes do not alter the contents of packets: the problem of routing was limited to selecting whether or not to send a copy of the received packet, and through which output link [CS82, Tan03, KR04].

Even though this view has been respected for many years, this has recently been superseded by the introduction of *Network Coding* (NC) [ACLY00].

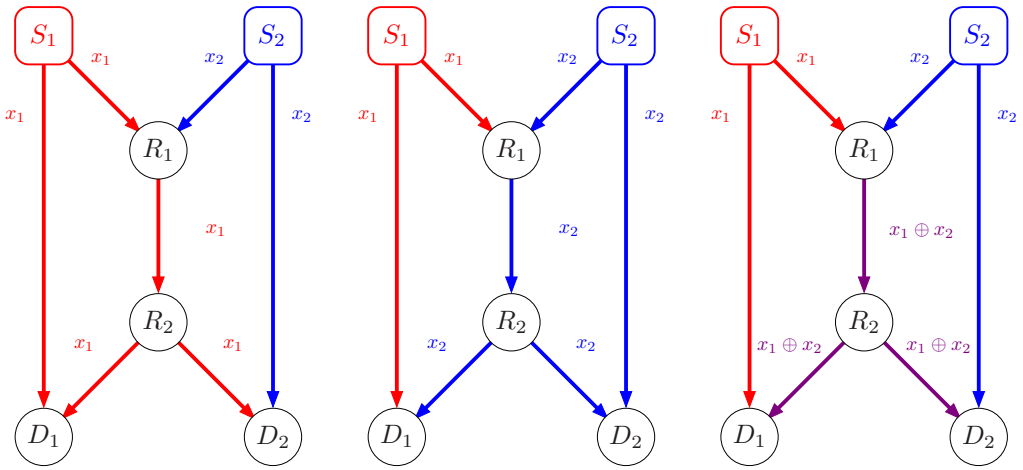
Network coding is a generalisation of routing wherein each packet sent over an output link of a node can be a mixture of the packets received from the node's input links rather than a mere copy. Mixing packets at intermediate nodes, an operation referred to as "coding", has proven beneficial to networking in several ways, improving the throughput, minimising the delay, and granting error resilience — providing that a suitable decoding strategy is available at the receiver.

One of the aspects of networking that has shown to gain a clear advantage in using NC over traditional routing is multicasting communication. With one of the most celebrated results in network coding theory, namely the *Max-Flow-Min-Cut Theorem* for network information flows, Ahlswede *et al.* [ACLY00] have pointed out that coding the packets within a network, under loose hypotheses on the coding function, allows a source to multicast an information flow at a rate approaching the capacity of the smallest minimum cut between the source and any receiver, while the same result cannot be achieved through traditional routing. The minimum cut between a source and a receiver is the smallest set of links that one must remove from the network in order to make the receiver unreachable from the source.

This result has obviously created a great research interest in network coding. Because of the wide range of applications that could benefit from it, such as video streaming, distributed information storage, and content delivery, many research communities have approached NC from a multitude of different points of view, such as graph theory, information theory, channel coding theory, and optimisation theory.

In this section, we shall introduce the basic concepts of network coding and present the most relevant theoretical results in terms of maximisation of the rate in a multicast transmission.

Other studies [DEH+05, KK08] have also regarded NC as a tool to provide resiliency toward errors and erasures over unreliable networks in a distributed fashion. However, these studies are outside of our scope, and shall not be discussed in this thesis.



(a) Message x_1 is sent over the bottleneck link.

(b) Message x_2 is sent over the bottleneck link.

(c) A combination of x_1 and x_2 is sent over the bottleneck link.

Figure 5.1: The butterfly network. Each edge of the graph represents a directed link with capacity of one message per transmission. Source nodes S_1 and S_2 want to transmit messages x_1 and x_2, respectively, to both sink nodes, D_1 and D_2.

5.1.1 Basic principles of network coding

The concept of network coding first appeared in the work of Ahlswede *et al.* [ACLY00], starting from the famous *butterfly network* example, depicted in Fig. 5.1.

Given the topology in Fig. 5.1, they consider the problem of two sources, S_1 and S_2, wanting to deliver their respective messages x_1 and x_2 to two destinations D_1 and D_2. All links are assumed to have a capacity of one message per transmission. If intermediate nodes R_1 and R_2 were only forwarding the messages they receive, at every transmission they could either deliver x_1 to both D_1 and D_2, and x_2 to D_2 only (Fig. 5.1(a)), or conversely x_2 to both, but x_1 to D_1 only (Fig. 5.1(b)). In other words, the link between R_1 and R_2 becomes a bottleneck. However, if node R_1 is allowed to send a combination of x_1 and x_2, *e.g.*, the bit-wise *exclusive-or* (XOR), both receivers can obtain both messages with a single transmission per node, as shown in Fig. 5.1(c). Exclusive-or is a logical operation that corresponds to addition modulo 2; it presents the useful property that $(x_1 \oplus x_2) \oplus x_2 = x_1$ and $(x_1 \oplus x_2) \oplus x_1 = x_2$, thus allowing the reconstruction of both messages if either one is received together with the combination of the two.

In conclusion, while with the routing approach the two messages are delivered to all nodes with at least two successive transmission (also referred to as *rounds*), using NC in the butterfly network, in a single round we can multicast *two* messages, which is the capacity of the minimum cut between each source and each destination.

This result, just presented for the butterfly network, can be generalised in the case of a point-to-point communication network in which a source multicasts information to a certain set of destinations. Known as the *Max-Flow Min-Cut Theorem for Network Information Flows*, it is one of the most important results in network coding theory that

expands a similar result of graph theory [BM08].

5.1.2 The Max-Flow Min-Cut Theorem for Network Information Flows

Let us model the communication network as a *directed acyclic graph* (DAG) $\mathcal{G} = (V, E, C)$, where V is the set of vertices, corresponding to the nodes in the network, E is the set of direct edges, corresponding to the links between nodes, and C is the set of capacities associated to each link.

An s–t cut in a graph is a partition of V into two sets S and T, such that $s \in S$, $t \in T$, $S \cup T = V$, and $S \cap T = \emptyset$. Let us denote with $\mathbf{tail}(e)$ and $\mathbf{head}(e)$ the starting and ending point of a link e. The cut set associated to $\mathcal{G}$ is the set:

$$\chi = \{e \in E \mid (\mathbf{head}(e) \in S) \wedge (\mathbf{tail}(e) \in T)\},$$

that is, the set of edges going from S to T. The capacity of the s–t cut is the sum of the capacities of the edges in χ.

The minimum s–t cut, *i.e.*, the one with the minimum capacity, is an important characterisation of the network, as it represents the bottleneck in the communication between s and t.

The Ford-Fulkerson algorithm [FF56] can be used to find this amount of flow and the edge-disjoint paths from s to t that can carry it. This solution correspond to the largest valid routing, *i.e.*, the one carrying the maximum flow, from the source s to the sink t.

The Max-Flow Min-Cut Theorem for Network Information Flows states that in a flow network the maximum amount of information units that can be transferred from a source s to a set of sinks T is equal to the capacity of the smallest minimum s–t cut for all $t \in T$.

If we represent an unencoded information unit with an element of a finite field F_q, where q is the size of the field, then a message of h units can be represented by a vector $\mathbf{x} \in F_q^h$. Using network coding, for any link $e \in E$ of the network, the message is propagated as a *coded* symbol $f_e(\mathbf{x}) \in F_q^h$, with the encoding mechanism specified by the set of functions $f_e(\cdot)\, \forall e \in E$.

The Max-Flow Min-Cut Theorem for Network Information Flows assures that there exist a set of coding of functions $f_E(\cdot)$ such that the maximum flow in the multicast case can be achieved, providing that the set vectors $\mathbf{x}$ is defined over a finite field F_q with a big enough size q. For instance, in the example of the butterfly network, the maximum flow can achieved as long as $q \geq 2$.

However, this theorem only gives a theoretical upper bound for the achievable multicast rate of the network, while it does not provide a constructive way to achieve it. A great deal of effort has been therefore put into providing constructive solutions for different scenarios.

For instance, Gastpar and Vetterli [GV02] considered the physical model of a wireless network under a relay traffic pattern, *i.e.*, with only one active source/sink pair and all other nodes assisting this transmission. They proposed a code constructions algorithm

leading to achievable rate determined by the Max-Flow Min-Cut Theorem, by allowing for arbitrarily complex network coding.

5.1.3 Linear Network Coding

A great leap in the network coding field has been made when Li *et al.* [LYC03] proved, although this result has been already known as a conjecture for some time, that with a proper choice of q, the upper bound determined by the Max-Flow Min-Cut Theorem can be achieved using linear functions only. In other words, each encoding function $f_e(\mathbf{x})$ can be a linear combination of the type:

$$f_e(\mathbf{x}) = \mathbf{w}_e^\top \mathbf{x},$$

so that the network code is completely specified by the set of *coding vectors* $\mathbf{w}_e \, \forall e \in E$.

The proof has been provided through a constructive algorithm, referred to as *Linear-Code Multicast* (LCM), that sorts the links in topological order and assigns them a coding vector such that it is linearly independent with respect to all previously assigned coding vectors. A direct acyclic graph is said to be in topological order if, for each edge going from a node u to a node v, u appears before v in the ordering [CLRS09, Ch. 6].

Building on the work of Li *et al.*, Koetter and Médard have introduced an algebraic framework for network coding [KM03] that extends network coding to arbitrary networks and robust networking. In the specific case of linear network coding, they also found necessary and sufficient conditions for the feasibility of any given set of connections over a communication network. Introducing an algebraic framework is a powerful tool, as it allows to use well-established theoretical results from algebra in solutions to network problems. However, their results provide an algorithm to construct a solution to the network coding problem that belongs to the NP-hard class of problems, and that is therefore not practically feasible for large networks.

In order to cope with this complexity, Jaggi, Sanders, *et al.* [JSC+05] considered communication networks modelled as acyclic delay-free graphs with edges of integer capacities, and studied the single-source multicast problem. They provided a deterministic polynomial time algorithm, and even faster randomised algorithm, for designing linear codes tolerant to edge failures, operating over finite fields much smaller than those previously proposed.

5.1.4 Practical Network Coding

Despite their indisputable theoretical value, all the results presented so far require a high degree of knowledge of the topology of the network, a centralised decision of the coding strategies and a fixed assignment of these strategies to the nodes of the network. All these requirements are hardly met on most real communication networks.

In most communication networks, the structure, the topology and the traffic demands may change quickly and drastically, while the information about those changes propagates with a certain delay. In wired networks, the edge capacities may vary due to changing traffic conditions and congestion. In wireless networks, they may vary in time due to fading channels, interference and node mobility. Also, each change in the network would require the computation of a new set of optimal combination operations, with the associated computational cost.

Therefore, subsequent work has investigated how to design algorithms capable of solving the multicast problem that could be implemented in practice.

Ho *et al.* argued that in real networks, with cycles and delays, the coding functions can be assigned in a distributed fashion by performing *Random Linear Network Coding* (RLNC) [HMS+03, HKM+03, HMK+06], *i.e.*, choosing the coefficients of the coding vectors independently and randomly over a suitable finite field.

In particular, they proved that the probability of a randomly chosen set of coefficients to ensure decodability at the receiver does not depend on the maximum multicast rate, and that this probability can be made arbitrarily close to one by working with a large enough field size q.

However, Ho *et al.* did not provide a method to transmit the coding and decoding functions, *i.e.*, the coding vectors, to the nodes where they take place.

In order to fill this gap, Chou *et al.* [CWJ03] presented the first practical approach to RLNC that takes into account the transmission of the coding vectors, which has become since then the most popular solution for RLNC. Under the name of *Practical Network Coding* (PNC) it has shown promising results with respect to several problems in multimedia applications (as we shall see in Sec. 5.2).

In order to show how PNC works, let us define, for each intermediate node $v \in V$ of a direct acyclic graph $\mathcal{G} = (V, E, C)$, its set of *incoming* edges $\Gamma_{\text{in}}(v) = \{e' \mid \mathbf{head}(e') = v\}$ and its set of outgoing edges $\Gamma_{\text{out}}(v) = \{e \mid \mathbf{tail}(e) = v\}$, as depicted in Fig. 5.2.

Each outgoing edge $e_j \in \Gamma_{\text{out}}$ of v carries a symbol $y(e_j)$ that is a linear combination of the symbols $y(e'_i)$ carried over its incoming edges $e'_i \in \Gamma_{\text{in}}$, *i.e.*:

$$\forall e_j \in \Gamma_{\text{out}}, \qquad y(e_j) = \sum_{i|e'_i \in \Gamma_{\text{in}}} m_{ij} y(e'_i)$$

with $m_{ij} \in F_q$. The coefficient vectors $\mathbf{m}_j = [m_{ij}]$ for each outgoing edge e_j represent the *local coding vector* of v along that edge.

By induction, the symbols $y(e'_i)$ are in their turn combinations of the symbols received by the node that has emitted them, and so forth, up to the source symbols $x_1, x_2, \ldots, x_h$. Therefore, any symbol transmitted by any node can be expressed as a linear combination

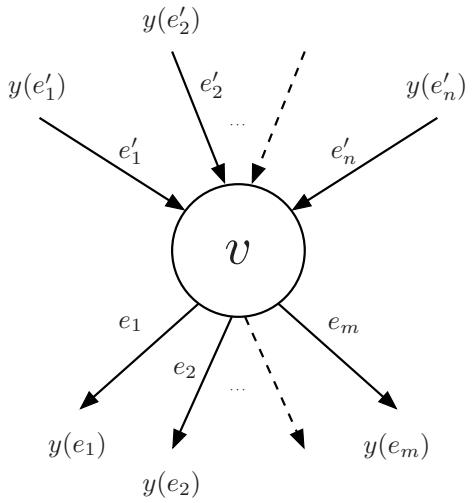


Figure 5.2: Intermediate node in a network with coding capability. For each outgoing edge $e_j \in \Gamma_{\text{out}}(v) = \{e_1, e_2, \ldots, e_m\}$ the transmitted symbol $y(e_j)$ is a linear combinations of the input symbols $y(e'_i)$ carried over the incoming links $e'_i \in \Gamma_{\text{in}}(v) = \{e'_1, e'_2, \ldots, e'_n\}$.

of the source symbols:

$$\forall e_j \in \Gamma_{\text{out}}, \qquad y(e_j) = \sum_{i=1}^{h} g_{ij} \mathbf{x}_i$$

with $g_{ij} \in F_q$. The coefficient vectors $\mathbf{g}_j = [g_{ij}]$ represent the *global coding vector* along each outgoing edge e_j, and can be determined recursively knowing the global encoding vector over the all the links of the path between the source s and node v, and assuming that the coding vectors used by the source corresponds to the ith unit vector, *i.e.*, that a source s_i transmits over all its outgoing links the unencoded symbols $\mathbf{x}_i$.

In this scenario, any receiver v with at least h incoming edges can recover the source symbols $x_1, x_2, \ldots, x_h$ if the matrix $G_t = [\mathbf{g}_j]$ of global coding vectors has rank h. This can be achieved with high probability if each coefficient of the local encoding vectors is chosen randomly from a finite field of sufficient size q. In order to be able to invert the code at any receiver, the global coding vectors used to generate the packets are included in the packets themselves. Since any coding vector of coefficients taken from a field F_q is represented on $h \lceil \log_2 q \rceil$ bits, the field size has to be chosen taking into account the conflicting objectives of reducing the packet size and increasing the decoding probability. Experimental results [CWJ03] show that using a Galois Field of relatively small size (*e.g.*, GF(2^8) or GF(2^{16})) works well in most scenarios and the probability of a node not being able to decode the message becomes negligible, while the overhead due to the prepended coefficient is kept small with respect to the payload.

If the source message is composed of many units, as it is often the case in a real scenario, in order to reduce the number of coefficients in each coding vector, Chou *et al.* propose to divide the data stream is into *generations*, each one consisting of k consecutive data packets of fixed size, with k much smaller than the size of the message. Only packets

coming from the same generation can be combined at intermediate nodes.

In those nodes, the received packets are stored in separate buffers per generation. When a sending opportunity occurs, a new packet is generated by combining, with random coefficients, all the packets in the current generation.

The operations needed to recover the source message can be found at the receiver by inverting the global coding matrix, *e.g.*, by Gaussian elimination, as soon as k independent coding vectors, also termed as *innovative vectors*, have been received.

The choice of the generation size is not trivial, as it involves a trade-off between decoding delay and rate overhead, *vs.* the decoding probability: on one hand, the decoding delay is negatively affected by a large generation size k, since a sink node has to wait for k independent packets in order to decode. Furthermore, the generation size also determines, together with the field size, the overhead due to the inclusion of coding vectors in the packet headers, which should be kept small, and particularly so in wireless networks, where packets are smaller and the overhead may become prohibitive. On the other hand, a large size of the generation affects positively the performance of the network coding in terms of throughput, as more symbols are coded together (the Max-Flow Min-Cut Theorem assures that the maximum multicast flow is achieved only for $h \to \infty$).

5.2 Network Coding for multimedia applications

In the previous section, we have discussed the topic of network coding from a graph-theoretical point of view. Little emphasis has been given on the several options that the ISO/OSI or the TCP/IP protocol stacks offer regarding the level wherein NC should be integrated [MF09].

Most of the solutions available perform network coding either at the Application Layer (OSI Layer 7), or at the Data-Link Layer (OSI Layer 2). Usually, the former approach is followed when coding is allowed only among packets belonging to the same multicast session, referred to as *intra-session* network coding. The latter usually allows combination of packets from different sessions (multicast or unicast), thus referred to as *inter-session* network coding.

In the following, we shall give an overview of both categories in general, and in particular with respect to their applications to video multicast.

Intra-session network coding, implemented at the application layer with little effort, has been proven beneficial for large scale *peer-to-peer* (P2P) content distribution. In this kind of application, the source splits the content into small blocks, termed *chunks*, that are transmitted to each end-user in parallel. Once it has received a chunk, a user can trade it with anyone else interested in it against another chunk. Since in some architectures the source still distinguish itself for reliability and resources made available to the service, some prefer using the term *cooperative* rather than peer-to-peer [PS02], in the sense that the users are not considered peers, but clients that cooperate in order to alleviate the load

of the sever.

Most cooperative architectures implement a *rarest first* chunk download strategy. This strategy, introduced in the popular peer-to-peer protocol BitTorrent [QS04, YLHX06], attempts to achieve a uniform distribution of the chunks among the nodes by forcing them to always download the chunk that is available on the least number of nodes. This is done in order to prevent a user who has all but a few chunks from waiting too long to complete its download.

However, based on experimental results, Gkantsidis *et al.* [GR05, GMR06] pointed out that the rarest first strategy might still lead to the scarcity of some chunks of content, especially when nodes are close to finishing their download. Since in a cooperative architecture they will attempt to obtain the missing chunks directly from the server, this would cause an unnecessary server overloading. Furthermore, other inefficiencies inherent to P2P systems become prohibitive in large heterogeneous networks especially during flash crowds or in environments with high churn.

They therefore proposed an innovative protocol, named Avalanche, that allows nodes to send a random linear combination of all their available chunks. A first advantage of this network coding scheme is that chunk scarcity is no longer a problem: due to the degree of redundancy introduced by the linear combinations, all nodes are able to finish the download, even in extreme situations. A second advantage can be observed when *tracking* the chunks for download, *i.e.*, finding out which nodes of the overlay have available a given chunk. Using network coding this task is greatly simplified, as a node can only has to determine whether a peer provides an *innovative* packet, which can be done by simply comparing their coding matrices. Avalanche has proven to provide a great benefit in terms of throughput with respect to classical schemes without network coding, especially in heterogeneous networks, where nodes have different upload and download capacities, and the routing approach suffers a great inefficiency due to the "slow" nodes spending their bandwidth downloading from the server chunks that are not useful to their "fast" peers [GMR06].

The cooperative approach, which has originally been proposed for content distribution networks (CDN), has been also applied to live multimedia streaming in large scale networks (such as the Internet) [PWCS02, WXL07, SBG08, WXL10, MGPP+12]. In this scenario, the uniform distribution of chunks is an even more critical point, as the chunk selection strategy has to take into account the play-out deadline of chunks as well, and cannot be limited to a simple rarest first policy [CBM08]. An even larger benefit is therefore to be expected by enabling network coding.

In order to evaluate the benefits and trade-offs involved in using network coding for cooperative live streaming, Wang and Li [WL07a, WL07b] implemented a realistic testbed, called Lava, deployed on a server cluster and able to meticulously emulate peer upload capacities and peer dynamics. They performed a fair comparison between schemes using network coding and schemes using traditional routing, implementing a pull-based cooper-

ative live streaming protocol. Their results suggested that using network coding it would be possible to perform live streaming with a much finer granularity, greatly reducing the bandwidth overhead.

On the basis of these results, they subsequently presented they own streaming algorithm, called R^2 [WL07c]. Instead of adding network coding capabilities to an existing protocol, they designed a new one from scratch, incorporating random network coding with a randomised push algorithm.

In their thorough evaluation with real network traffic and emulated peer upload capacities, they observed that R^2 improved the performance of live streaming in terms of initial delays, resilience to network dynamics, and reduced bandwidth costs on the servers in comparisons with the state-of-the-art of the time in streaming applications.

In self-organising networks, such as MANETs, network coding can be used for wide dissemination of data as a substitute for flooding at the Data-Link layer. An example of the advantages of using NC in multi-hop broadcast over wireless ad-hoc networks is given in the simple scenario depicted in Fig. 5.3. In this scenario, two source nodes, S_1 and S_2 both want to broadcast a message, with node R as a relay. While using classical flooding node R would have to relay each message separately, using network coding it can relay the exclusive-or of the two, thus allowing both nodes S_1 and S_2 to decode each other's message. This is an important result, as energy efficiency (*i.e.*, the amount of battery energy consumed per transmitted bits) is a critical design parameter for mobile networks [NTNB09].

This is an example of inter-session network coding, in the sense that messages x_1 and x_2 may belong to different unicast or multicast sessions, oblivious to R. This approach has proven very efficient in the sense of maximisation of the network throughput both for the of case multiple unicast sessions [KKH$^+$05, KRH$^+$06, KRH$^+$08], and for multicast sessions [WFLB05, FWLB06, FWLB08].

However, as Karbaschi *et al.* have pointed out [KVMAA09], network coding schemes merely aimed at throughput maximisation are unfairly biased. This happens as intermediate nodes, in order to decode their message, need to wait for the reception of the whole generation of encoded packets, some of which they may be not interested in. Karbaschi *et al.*therefore proposed a *fair* mixing strategy that takes into account the decoding delay of each destination. Basically, fair mixing consists in a form of dynamic intra-session network coding, where the session is not statically identified by the content, but rather by its end-point destination.

Furthermore, using inter-session network coding makes every packet dependent on other packets, so that even a single erasure or error, both extremely likely in unreliable environments such as wireless networks, might affect the correct decoding of all packets. For this reason, several research efforts have been devoted to the design of robust strategies for NC, aimed at circumventing this limitation. An excellent survey on this topic has been provided by Di Renzo *et al.* [DRIK$^+$10].

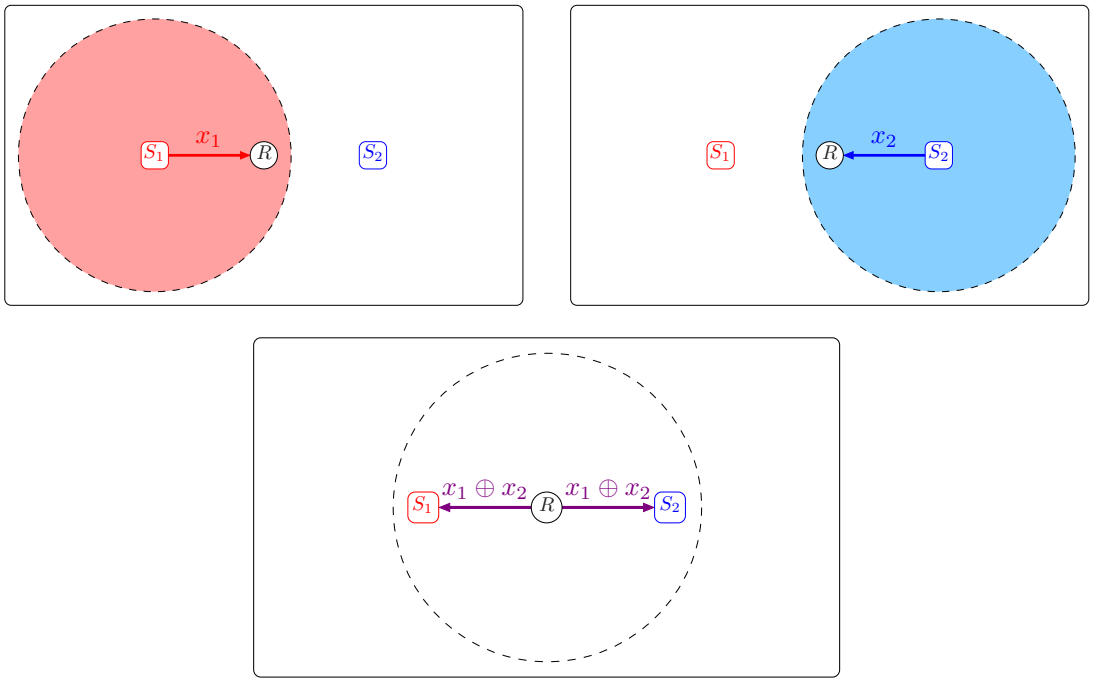


Figure 5.3: Network coding for message exchange in multi-hop wireless network. The number of total transmissions in reduced from 4 to 3 if the relay node R broadcasts a mixture of the two messages x_1 and x_2, received from sources S_1 and S_2, respectively.

The first NC protocol for multiple unicast sessions over ad-hoc networks has been proposed by Katti *et al.* [KRH+06, KRH+08] under the name of COPE. This protocol uses only simple combinations in GF(2), *i.e.*, the coding operations are limited to XORs. COPE, similarly to the ABCD protocol described in Chapter 3, takes advantage of the broadcast medium by allowing nodes to overhear a packet in transmission where they are neither the destination nor a forwarding node. The overheard packets are stored in a buffer for a short time and advertised through periodic *reception reports* to the node's neighbours. With this knowledge, the nodes in the neighbourhood are able to perform *opportunistic coding*, *i.e.*, to maximise the number of packets delivered in a single transmission, while ensuring that each next-hop has enough information to decode its packet.

To deal the reliable broadcast problem, COPE follows the inverse approach with respect to ABCD: instead of forcing an RTS/CTS handshake for broadcast messages and addressing all packets to the broadcast address, it implements a *pseudo-broadcast*, *i.e.*, it only uses unicast transmission, but forces the nodes to be in *promiscuous mode*. Promiscuous mode is the interface mode that causes the controller to pass all traffic it receives to the upper layers, rather than just the ones it is intended to receive (*packet sniffing*). The nodes actually meant to receive the packet are specified within extra header, added in the payload just after the Data-Link header. This is equivalent to the scheme we proposed in Chapter 3 if no other unicast traffic is present in the network. If another unicast session is present, COPE nodes will still overhear all messages and check their payload before even-

tually realise they have to drop them, whereas ABCD nodes (that delegate the reliable broadcast mechanism to the sender, rather than the receivers) would simply ignore them.

The simulation results of using COPE in wireless environments show a relevant improvement in network throughput with respect to the non-coding approach, particularly as the number of flows increases as the number of coding opportunities also increases. However, a higher traffic also increases the network congestion, causing many reception reports to be lost.

Given the good properties shown by network coding both in relaying video content in large scale environments and in implementing a more efficient form of flooding in ad-hoc networks, it is interesting to study how network coding can be integrated in a framework for real time video delivery over wireless ad-hoc networks [FKM+07], such as the one we presented in Chapters 3 and 4.

A possible strategy for video delivery in ad-hoc networks is the joint use of network coding and scalable video coding. As detailed in Sec. 1.2.7, scalable video coding is a video coding paradigm that can account for the different requirements, with respect to quality of service received, and different network conditions are not reliable by allowing the bitrate of the stream to dynamically adapt available bandwidth, which makes it suitable to be used in a video multicast scenario over heterogeneous networks such as MANETs. Since NC also has been shown to improve the throughput in such a network, it is interesting to evaluate how a joint design can further improve the performance in the network.

A drawback of using scalable coding, is that it poses a constraint in the order the layers are to be received, as higher layers can only be used if all previous layers have been received. Therefore, the delivery system must provide some form of unequal error protection in order to ensure that the lower layers are received with a higher probability than the higher. In order to cope with this problem, a viable solution is to use *Hierarchical Network Coding* (HNC) [NNcC07], a technique proposed by Nguyen *et al.* that allows the more important packets to be recovered with higher probability if only a small number of coded packets are received.

Let us assume, for instance, that a video stream is encoded in three layers, L_0, L_1 and L_2, with L_0 the base layer (therefore the most important). Six packets are to be transmitted: p_1^0 and p_2^0, belonging to L_0, p_1^1 and p_2^1, belonging to L_1, and p_1^2 and p_2^2, belonging to L_2 [NNcC07]. Using HNC, each nodes generates the coded packets by randomly choosing one of the following structures:

$$\begin{aligned} P_0 &= g_1^0 p_1^0 + g_2^0 p_2^0; \\ P_1 &= g_1^1 p_1^0 + g_2^1 p_2^0 + g_3^1 p_1^1 + g_4^1 p_2^1; \\ P_2 &= g_1^2 p_1^0 + g_2^2 p_2^2 + g_3^2 p_1^1 + g_4^2 p_2^1 + g_5^2 p_1^2 + g_6^0 p_2^2; \end{aligned}$$

where coefficients g_k^l are non-zero elements randomly chosen from a finite field F_q. Since a receiver has to obtain two packets of type P_0 to recover L_0, four packets of type P_1 to

recover L_1, and six packets of type P_2 to recover L_2, the probability of recovering L_0 is always larger than that of L_1, which is in its turn larger than that of L_2. In this sense, HNC can be considered to provide *Unequal Error Protection* (UEP) to the stream. To fine tune the probability of receiving a certain layer, a node can control the number of coded packets generated for that layer.

This scheme proved to provide a benefit over both classical routing and random linear network coding [NNcC07]. Comparing the three schemes (no network coding, RLNC, and HNC), Nguyen *et al.* observed that if the intermediate nodes do not perform network coding, the time needed to decode any layer is the largest due to the high probability of receiving a duplicate packet. Using RLNC to total time to decode the three layers is the shortest, but the time needed to decoded layers L_0 and L_1 is larger than in HNC. This is because in RLNC all the packets belong to the same generation, and are therefore decoded simultaneously when enough innovative packets are received. Hierarchical network coding, on the other hand, allows the receivers to decode the most important packets earlier, but it suffers an overhead in decoding all less important packets. This is still a good results since, if there is not enough bandwidth, a receiver can be satisfied with the current number of reconstructed layers and instruct the sender to move on to the next chunk.

Generalising the work of Nguyen *et al.*, Vukobratović and Stanković [VS10] have provided an exact decoding probability analysis for the different layers of the source data in random network linear coding designs with unequal error protection. They also provided a viable network coding design framework, called *Expanding Window Network Coding* (EWNC). The key idea of EWNC is to increase the size of the coding window (*i.e.*, the set of packets in the generation that may appear in combination vectors) for each new packet. In this sense, EWNC is a form of HNC where there exists a layer per packet. If in the buffer of the receivers the received coding vectors are kept in row echelon form, using Gaussian elimination, this method provides *instant decodability* of each packet if no losses occur. Thanks to this property, EWNC is preferable over PNC in streaming applications. Even though PNC could achieve almost instant decodability using a small generation size, this would be ineffective in a wireless network, where a receiver could be surrounded by a large number of senders, and if the size of the generation is smaller than the number of senders, some combinations will necessarily be non-innovative. On the other hand, EWNC automatically adapts the coding window size allowing early decodability, and innovativity can be achieved if the senders include the packets in the coding window in a different order. However, these orders should take into account the RD properties of the video stream, as we shall discuss in detail in Sec. 5.3.

Even though UEP scheme can mitigate the problem, the fact that layers have to be received in a predetermined order is still an inherent limitation of scalable video which multiple description coding does not suffer, as discussed in Chapter 2. Therefore, the system design when using NC jointly with MDC has to take into account fewer constraints.

For instance, Ramasubramonian and Woods [RW10] have focused on how a joint use

of NC and MDC can be applied to the problem of throughput maximisation in multicast video streaming. They proposed to encode the video sequence into N descriptions, with the total rate for the encoding, *i.e.*, the combined rate of all descriptions, chosen to match the maximum max-flow of the nodes. The redundancy among the descriptions is chosen depending on the number of descriptions that each node receives, which is assumed to be transmitted to the source with a lossless feedback channel. The N descriptions are then used to generate N linearly independent random combinations and, if the source still has available, linear combinations of the descriptions are generated to fill the capacity (these combinations will of course be non-innovative).

Thanks to the joint use of MDC and NC, users with high max-flow capacity will be able to satisfy their demands, *i.e.*, to receive more descriptions, whatever bottleneck may appear in the network at intermediate nodes. This result would not be possible using scalable coding, since in MDC *any* combinations of a given number of descriptions delivers (approximately) the same video quality, whereas missing a layer in a scalable coding would make it and any following layers useless.

This technique requires that the nodes of the network are able to determine their max-flow capacity, and to transmit it to the source, which dynamically determines the optimal rate allocation for the descriptions. Unfortunately, in many scenarios this is not possible. The source could have to transmit a pre-encoded sequence, and not have sufficient computational power to re-encode it based on the changing conditions of the distribution network. Also, nodes may not be able to determine their max-flow capacity or this could change too frequently to be communicated to the source with little overhead, such as in the case of mobile networks. Also, this framework only considers *lossless* networks with a feedback channel, whereas both MDC and NC are praised for their properties of granting loss immunity in unreliable networks without feedback channels.

In other words, despite its theoretical interest from a video coding point-of-view, this approach translates the problem of network coding into the maximisation of network throughput, and suffers from an over-simplistic model of the networking scenario. The recent trend, for multimedia applications especially, is to build protocols that are implemented as a cross-layer design between the application and network, so to be tailored to the specific media content to be delivered over a specific network type.

In this respect, Seferoglu and Markopoulou [SM07] argued that, when the transmitting live video streams, the network codes should maximise not only the network throughput, but also the video quality. They therefore proposed a video-aware network coding scheme for wireless networks that takes into account the decodability of the code at the receivers as well as the deadlines of video packets and their contribution to the overall video quality.

Their protocol, named *Network Coding for Video* (NCV) identifies at each sending opportunity a *primary packet*, *i.e.*, a packet that must be recovered by a given target node, and a possibly empty set of *side packets*, *i.e.*, packets that it is useful to include in the code in order to minimise the overall video distortion. The set of side packets

could be empty depending on the state of the receivers' buffer as their inclusion in the code might make the recovery of the primary packet impossible at the target node. The selected packets are then coded together with RLNC and sent over the channel with a pseudo-broadcast scheme and exchange information about their buffers using a modified version of the COPE protocol.

The simulation results for NCV show that this scheme remarkably improves the overall video quality compared to both classical routing and COPE, and significantly increases the application level throughput, while achieving a similar level of network throughput.

Seferoglu and Markopoulou subsequently tried to improve over the results of NCV by integrating a rate-distortion optimised packet scheduling framework, which allows to decide the current primary packet in a RD-optimal way, or to drop some packets altogether. Unfortunately, this method can achieve the global optimum for each transmission only if a perfect evaluation of the consequences, in terms of rate and total distortion, is available at each sending opportunity, which would require complete knowledge of the network topology and of the message exchanges among all nodes. However, simulation results also show that NCV performs well in practice with less message exchange and it can be considered an efficient heuristics to the RD-optimised version.

In the NCV protocol little emphasis was given to the creation of the overlay network, and mainly focuses on the per-hop behaviour. To obviate this problem, more recently, Thomos *et al.* [TCF11] have proposed a video streaming solution based on a distributed receiver-driven overlay construction algorithm. Here, the approach is reversed with respect to NCV, in the sense that the prioritisation of packet is not performed as a open-loop optimisation at the sender, but is based on the requests, made by the receivers, of packets of different layers. The sender then chooses the optimal coding strategy based on the requests, the priority of the packets, and the overall contribution to the local distortion (*i.e.*, within its neighbourhood). For overlay networks, a receiver-driven video streaming solution is proposed for video packets belonging to different priority classes. The problem of choosing the network coding strategy at every peer is formulated as an optimization problem of determining the rate allocation between the different packet classes such that the average distortion at the requesting peer is minimised.

The packet classes can correspond to layers in scalable video streams, or can be constructed based on the contribution of each packet to the overall quality of the media content. A class c is defined as the set of packets that are linear random combinations of packets from the c most important layers. The class number is included in a small header in each packet. The protocol follows two stages: first, children nodes compute the optimal coding strategy that their parents should follow based on the available bandwidth, importance of packets in each class and expected loss probability of the link. Then, they send a request message to their parents specifying the number of packets they want to receive from each class. Parent nodes send random linear combinations of packets in the requested classes. Based on the state of its buffer and the local network status, each child

node recomputes the optimal coding strategy and makes another request. In this way the algorithm is receiver-driven and can adapt to the needs of each node and to changing network conditions.

It is important to notice that the authors also proved that the optimisation function used to choose the optimal strategy can be put in convex form and that it can be solved with by means of greedy algorithm, able to find a solution with little computational effort.

According to the experimental results, this scheme substantially outperforms the previous methods in a variety of transmission scenarios, in terms of size of the network, capacity of the links, and packet loss probability.

5.3 Proposed contributions

In this section, we present two of our original contributions in the field of network coding for video streaming over ad-hoc networks.

Studies on the joint use of multiple description coding and network coding for gaussian sources have already shown promising results [IKLAA11]. We focus here on how this scheme can be applied to the case of video content, and in the specific case of a wireless ad-hoc network.

The first contribution consists in a NC extension for the ABCD protocol (presented in Chapter 3), that is able to ensure that each node relays *at most* one description (or a combination of descriptions with equivalent rate), without generating a higher delay or a higher number of active nodes than ABCD.

The second contribution is an RDO framework for video streams encoded with multiple descriptions, adapted to select which frames and in which order should be included in the coding window for network coding.

The experimental validation for both these contributions can be found is Sec. 5.4.

5.3.1 Network Coding extension of the ABCD protocol

We present here a modified version of the ABCD protocol that uses network coding to deploy an efficient video streaming service over ad-hoc networks, when nodes are able to sustain an up-link bitrate sufficient for only one description.

A conceptual example of this is presented in Fig. 5.4. Imagine that a node n_2 is an ABCD network is receiving both descriptions (d_0 and d_1) by two different parent nodes (n_0 and n_1, respectively). Node n_2 also has two other neighbours: n_3, which is receiving only description d_0 by n_0, and n_4, which is receiving only description d_1 from n_1. In this situations, both nodes n_3 and n_4 would send an attachment message to n_2 in order to activate it *on both descriptions*. However, as mentioned in Sec. 5.2, network coding can be used to provide a more efficient message exchange in multi-hop wireless network: if node n_2 activates, but instead of sending both descriptions it sends *a combination* of the

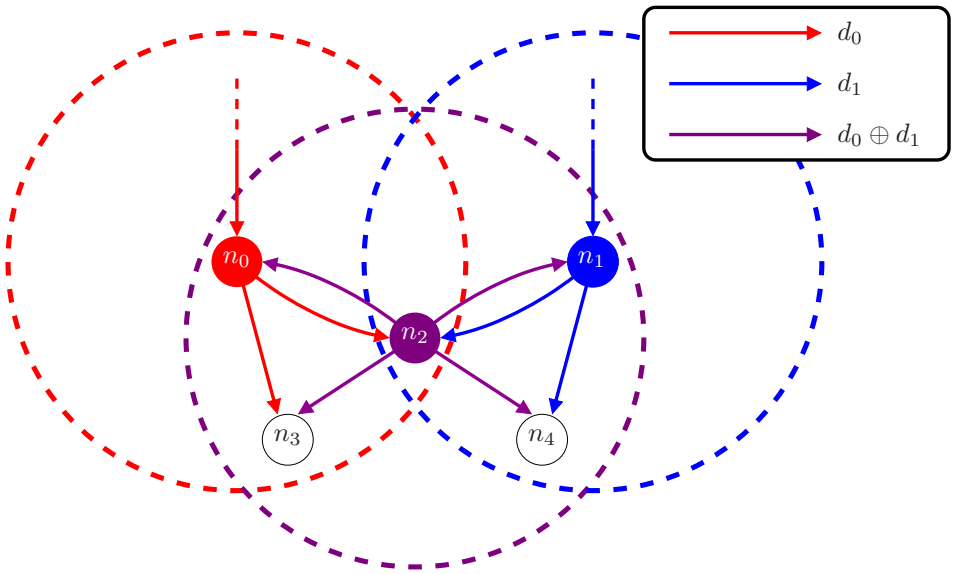


Figure 5.4: Example of combined use of network coding and multiple description coding in a wireless environment. Nodes n_0 and n_1 receive each one exactly one description (d_0 and d_1, respectively) from two different parents (not depicted in the figure). Node n_0 relies description d_0 to nodes n_2 and n_3, while node n_1 relies description d_1 to nodes n_2 and n_4. Without NC, in order to make all the nodes receive all descriptions, node n_2 should necessarily relay both of them. In this scenario, it can achieve the same result in a single round by sending a combination of the two descriptions instead.

two descriptions, *e.g.*, $d_0 \oplus d_1$, both n_3 and n_4 will be able to decode both descriptions, even though only *one* stream is being relayed by n_2. We shall now discuss how this considerations can be generalised, and how a strategy for optimal combinations can be designed and integrated in ABCD.

Let us start our discussion from the model proposed by Chou *et al.* [CWJ03] for Practical Network Coding, where $\mathcal{G} = (V, E, C)$ is a directed acyclic graph having unit capacity and representing the communication network. Let $s \in V$ be the sender, or *source node* and $T \subset V$ the set of receivers, or *peers*. Consistently with the specifications of the ABCD protocol, in our scenario we shall assume that the set of receivers includes any node except the source, *i.e.*, $T = V - \{s\}$.

Let us assume that the video stream is encoded with MDC in N descriptions. At each sending opportunity, the source has to broadcast a video frame x, encoded in packets $x_0, x_1, \ldots, x_{N-1}$ (one per description), to each peer in the network. In the model proposed by Chou *et al.* [CWJ03], each transmitted symbol is associated to the edge (or *channel*) it is carried over. Let us define $\hat{y}(e)$ the symbol carried over channel $e \in E$. Since we want to model a broadcast network, we need to introduce a further constraint. Namely, we want that each node n transmits the same symbol, that we denote $y(n)$, over all its outgoing channels. Once all symbols $y(n)$ are assigned, it is possible to revert to the standard model by imposing:

$$\hat{y}(e) = y(n), \qquad \forall e \in E \mid n = \mathbf{tail}(e).$$

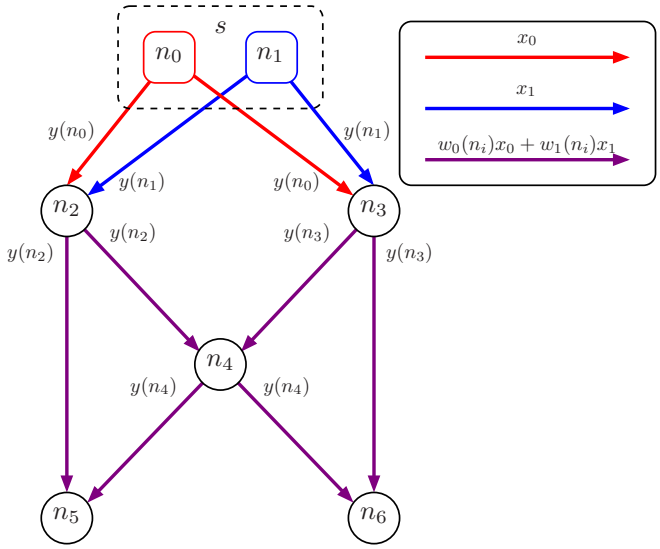


Figure 5.5: Model of wireless network with coding capabilities for two descriptions. The source is modelled with two virtual sources, each one transmitting its associated description. Intermediate nodes transmit the same combination of the two descriptions over all their outgoing links.

Imposing that for all nodes the same symbol is sent on all unitary capacity channels raises a problem: the model does not lend itself to the case of multiple description coding, as the source can transmit a single message. To avoid this problem, we can model the video source as a set $S = \{s_0, s_1, \ldots, s_{N-1}\}$ of N virtual sources, each one having a copy of the outgoing channels of s and emitting exactly one description x_d over all its channels, thus maintaining uniformity in the unit capacity of channels:

$$y(s_d) = x_d, \qquad \forall d \in \{0, 1, \ldots, N-1\}\,.$$

In our model, depicted in Fig. 5.5, by our definition, the set of nodes V does not contain the "original" source s and contains instead the set S. There are thus $M = |V|$ nodes, with N virtual sources and $M - N$ peers.

Using NC, the symbols emitted by a node $n \in T$ must be linear combinations of the symbols carried over the channels entering n. Let us define $\mathcal{I}_n$ the set of nodes m such that a channel exists from m to n. The symbol emitted by n will be in the form:

$$y(n) = \sum_{m \in \mathcal{I}_n} c_m(n) y(m), \qquad \forall n \in T. \tag{5.1}$$

The *local encoding vector* of node n, $\mathbf{c}(n) = [c_m(n)]_{m \in \mathcal{I}_n}$, represents the encoding function of node n along all its outgoing channels $e \in E \mid n = \mathbf{tail}(e)$. The virtual sources do not have any entering channels, but this does not pose a problem, as their emitted symbols are assigned beforehand.

We shall now give a definition of a node's *height* that differs from the one used in Chapters 3 and 4: here, the height $h(n)$ of a node n is the length of the *longest* finite path in $\mathcal{G}$ from any node in S to node n; this induces a partial order on the set V, well defined, as G is acyclic. Let us label the nodes with indices $0, 1, \ldots, M-1$ such that $i < j \implies h(i) \leq h(j)$. It follows that the nodes in S are labelled $0, 1, \ldots, N-1$.

This order is consistent with the propagation of packets outgoing from the source. If we define an *encoding matrix* $\mathbf{X}$ as follows:

$$\mathbf{X}_{ij} = \begin{cases} c_j(i) & \text{if } \exists e \in E \mid i = \mathbf{head}(e), j = \mathbf{tail}(e), \\ 0 & \text{otherwise,} \end{cases}$$

we can rewrite equation (5.1) as:

$$y(i) = \sum_{j<i} \mathbf{X}_{ij} y(j), \qquad \forall i \in \{N, N+1, \ldots, M-1\},$$

while for the virtual sources we impose:

$$y(i) = x_i, \qquad \forall i \in \{0, 1, \ldots, N-1\}.$$

We are interested in the number of packets a node is able to decode. Let us define $\mathbf{x}$ as a vector that has for components all the descriptions of frame x, that is:

$$\mathbf{x} = [x_d]_{d \in \{0,1,\ldots,N-1\}}.$$

Since any emitted symbol $y(i)$ is a linear combination of packets from all the descriptions, there exists a *global encoding vector* $\mathbf{w}(i)$ with N components such that $y(i) = \mathbf{w}(i)^\top \mathbf{x}$. The vector $\mathbf{w}(i)$ for nodes in S has only one non-zero component, corresponding to the description emitted, that is:

$$\mathbf{w}(i) = [\delta_{d,i}]_{d \in \{0,1,\ldots,N-1\}}, \qquad \forall i \in \{0, 1, \ldots, N-1\},$$

where $\delta_{\cdot,\cdot}$ denotes the Kronecker's delta function. For nodes in T, the global encoding vector can be inferred from matrix $\mathbf{X}$:

$$\mathbf{w}(i) = \sum_{j<i} \mathbf{X}_{ij} \mathbf{w}(j), \qquad \forall i \in \{N, N+1, \ldots, M-1\}.$$

Nodes transmit their global encoding vector along with their emitted symbol; the nodes

that receive this information interpret it as a linear equation in the form:

$$\begin{aligned} y(j) &= w_0(j)x_0 + w_1(j)x_1 + \ldots + w_{N-1}(j)x_{N-1} \\ &= \sum_{d=0}^{N-1} w_d(j)x_d. \end{aligned}$$

By collecting symbols and global encoding vectors on its entering channels, a node i is able to construct a system of linear equations:

$$\mathbf{W}(i)\mathbf{x} = \mathbf{y}(i),$$

where $\mathbf{W}(i)$ is a *global encoding matrix* obtained by horizontal concatenation of row vectors $\mathbf{w}(j)$, and $\mathbf{y}(i)$ is a column vector with components equal to $y(j)$, for all j such that $\exists e \in E \mid i = \mathbf{head}(e), j = \mathbf{tail}(e)$

A node i is able to perform central decoding, *i.e.*, to decode all N descriptions, if and only if $\mathbf{rank}\left(\mathbf{W}(i)\right) = N$. However, the rank is not a reliable tool to estimate the number of descriptions used in side decoding, *i.e.*, decoding only a subset of the N descriptions. For instance, a node i having $\mathbf{rank}\left(W(i)\right) = 1$ could be receiving from node j an equation in the form $y(j) = w_0(j)x_0$, which is trivial and allows the decoding of x_0. But it could also be receiving $y(j) = w_0(j)x_0 + w_1(j)x_1$, which is impossible to solve without further information.

Let us assume we have an operator $\mathbf{dec}(\mathbf{W}(i))$ able to infer how many descriptions a node i will be able to decode, given its global encoding matrix $\mathbf{W}(i)$. This operator can be easily implemented in practice by counting the number of trivial equations. We also define a *value* operator $\varphi(\cdot)$ that associates a quality metric to a number of decoded descriptions $\mathbf{dec}(\mathbf{W}(i))$. The choice of the quality metric will depend of course on the requirements of the application. If we use, for instance, the expected PSNR, $\varphi(\cdot)$ should reflect the fact that the PSNR gap between a node not receiving any description and one receiving just one description is bigger than the gap between a node receiving one description and one receiving two of them, *i.e.*, $\varphi(1) - \varphi(0) \geq \varphi(2) - \varphi(1)$.

Given this model, our approach is quite straightforward. At each sending opportunity, each node i inspects the state of the buffers of its neighbours, then it chooses an optimal weight vector $\mathbf{w}^*(i)$ as:

$$\mathbf{w}^*(i) = \arg\max_{\mathbf{w}\in\mathcal{W}_i} \left\{ J(\mathbf{w}) = \sum_{j\in\mathcal{N}_i} \left(\varphi \circ \mathbf{dec}\right)\left(\mathbf{W}(j)\right) \right\}, \tag{5.2}$$

where $\mathcal{W}_i$ is the set of coding vectors available to i, and $\mathcal{N}_i$ is the set of neighbours of i. As we shall discuss in the following, a node may know the state of its neighbours if it uses an overlay maintenance protocol, such as ABCD, that can include this information as

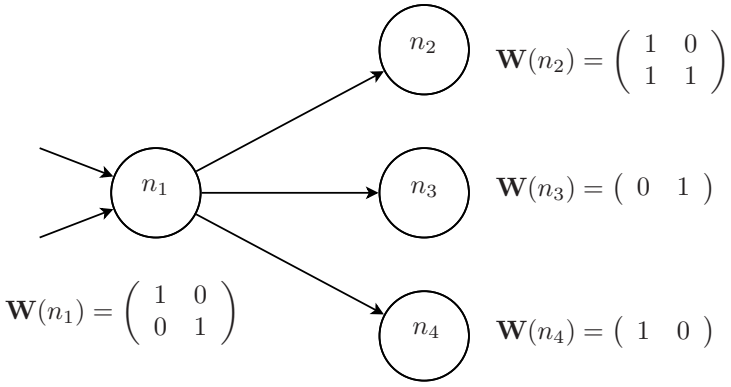


Figure 5.6: Example of optimisation of the weight vector of a node n_1.

an header in the protocol messages. Notice that $\mathcal{W}_i$ is restricted by the symbols actually received by i: a node can only choose to send one of the packets it received or a combination thereof.

The optimisation of the emitted symbol is independent from the decoding capability, *i.e.*, even if a node is unable to decode any description (*e.g.*, if it is receiving just a combination $w_0x_0 + w_1x_1$), it can still choose to forward a combination of what it received, if that would benefit its neighbours.

In Fig. 5.6 we present an example of optimisation performed by a node n_1. Here, $J([1,0]) = \varphi(2) + \varphi(2) + \varphi(1)$, as only the rank of $\mathbf{W}(n_3)$ is affected by the combination, $J([0,1]) = \varphi(2) + \varphi(1) + \varphi(2)$ as only the rank of $\mathbf{W}(n_4)$ is affected, and $J([1,1]) = \varphi(2) + \varphi(2) + \varphi(2)$ as both the rank of $\mathbf{W}(n_3)$ and $\mathbf{W}(n_4)$ are affected. Since $\varphi(2) > \varphi(1)$, the optimal choice is $\mathbf{w}^*(1) = [1,1]$.

There are two main challenges that need to be dealt with in order to use this approach: firstly, a mobile ad-hoc network is hardly a DAG; secondly, the nodes need to inspect the buffer state of their neighbours in order to solve the optimisation problem (5.2).

Both problems are solved using the ABCD protocol: on one hand, we can apply the algorithm not directly on the ad-hoc network, but rather on the overlay network generated by the protocol. As detailed in Chapter 3, the ABCD protocol generates multi-trees, which are indeed DAGs. On the other hand, the control messages sent in order to build and maintain the overlay can be used to propagate the state of the global encoding matrix $\mathbf{W}(i)$ from node i to its neighbours, thus providing the information needed to solve problem (5.2). Furthermore, the ABCD protocol ensures that the nodes have an up-to-date view of the topology and of the state of their neighbours, even in presence of node mobility and churn.

5.3.2 RDO-scheduling for joint NC-MDC streaming over MANETs

The second contribution we present is a per-hop transmission policy aimed at achieving a good trade-off between resiliency to losses and timely delivery.

Recently, Thomos *et al.* [TCF11] have proven that providing a prioritised video delivery

via path diversity and random linear network coding substantially outperforms baseline network coding and rate-less codes with inherent UEP properties.

In order to provide such a prioritisation, we propose to jointly use EWNC [VS10] and video MDC, which we expect to provide loss resiliency to the video stream without affecting the delay.

As mentioned in Sec. 5.2, the efficiency of EWNC highly depends on the order in which the packets are included in the coding window. The original EWNC method was proposed for layered video coding, therefore the priority of the packets was naturally imposed by the dependencies among layers.

Such a strategy is unfeasible in our scenario, as we deal with multiple uncoordinated senders sharing a broadcast medium, and if they all were to choose the same order of packets (*i.e.*, the one imposed by the layered structure), at any given sending opportunity they would send *non-innovative* combinations.

In general, if a prioritisation is optimal, it is also unique, thus all the senders would always transmit dependent combinations, defeating the purpose of using NC. In order to take advantage of the benefits of NC in terms of loss resiliency, we need to generate a variety of schedules, possibly slightly sub-optimal, but with acceptable performances.

The GOP structure of a video coding technique (such as H.264/AVC) leaves a certain degree of freedom in the scheduling, as frames on the same prediction level can be sent in any order (see Sec. 1.3.2). However, this degree of freedom may not be enough to provide a sufficient number of different schedules for the different senders.

Using an MDC technique, it is possible to have multiple senders transmitting packets that refer to the same instant, but that are different nonetheless. Furthermore, corresponding packets of different descriptions are mutually refinable, therefore a node being served by multiple senders will perceive an enhanced video quality.

Using MDC, the pool of frames candidate for inclusion in the coding window is a bi-dimensional *multiple description GOP* (MD-GOP), *i.e.*, a rectangular buffer of size $N \times W$, where N is the number of descriptions and W is the GOP size of each description. An example of MD-GOP is depicted in Fig. 5.7, for 4 descriptions and a GOP structure of each description as the one in Fig. 1.4 of Sec. 1.2, *i.e.*, Hierarchical-B with 4 prediction levels. Notice that, in the buffer, the frames are not ordered by their play-out date, but in encoding order, so that frame dependencies are respected.

The task of the scheduler is to provide an order in which the frames in the MD-GOP are included in the coding window. Since wireless networks are affected by churn and mobility, and the video stream can be interrupted at any moment, it is desirable that any new combination maximises the marginal benefit in terms of RD properties. In other words, at each step, we want the scheduling algorithm to select the frame that optimises an RD criterion for insertion in the coding window.

However, the corresponding frames of different descriptions might have differences in their RD properties, which would still lead to a unique optimal policy of inclusion in the

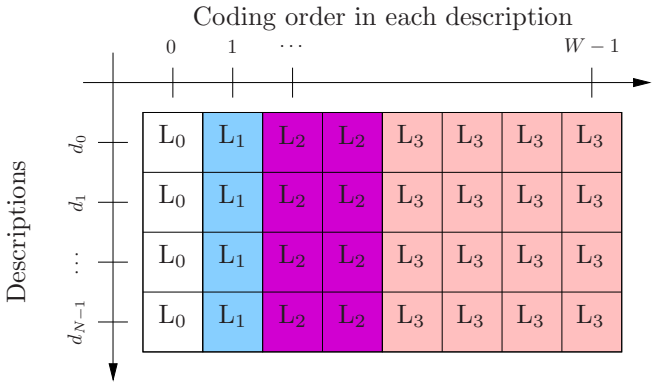


Figure 5.7: MD-GOP for $N = 4$ descriptions and $W = 8$ frames in a Hierarchical B-frame GOP with $L = 4$ temporal prediction levels. Frames are ordered by prediction level.

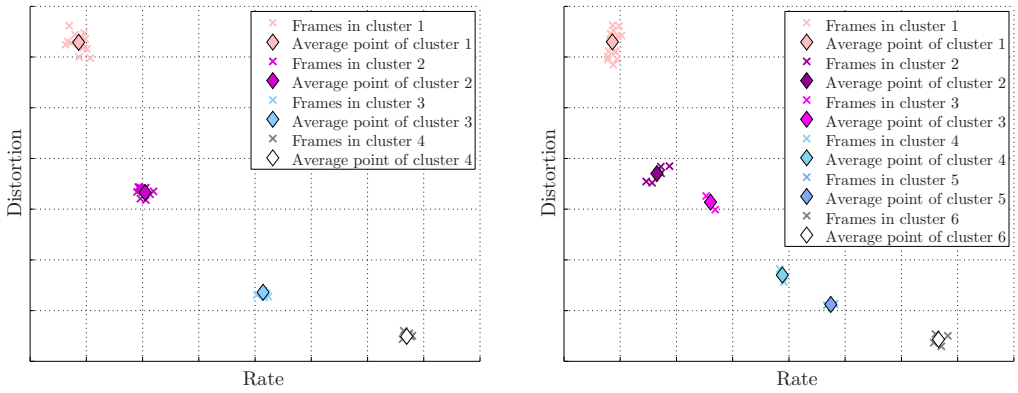


Figure 5.8: Two examples of video frames clustering for RDO-scheduling. Frames with similar operating points are assigned to the same cluster. The RDO-scheduling will consider each frame as having the average operating point of its cluster.

coding window.

In order to obviate this problem, we propose a *clustering* of the video frames, where this term is to be intended with respect to the scheduling policy. The clustering is a classification of the frames that takes place at video source, after the video encoding and before scheduling for transmission. Its purpose is to improve diversity by letting nodes transmitting, at each sending opportunity, a random frame within an cluster.

Clusters are decided *only once* at the encoder, where rate and distortion are known with negligible computational overhead, with frames in the same prediction level. The average rate and distortion of the cluster is then computed, possibly quantised, and added as a header to each frame in the cluster.

In the intermediate nodes, at each sending opportunity, the scheduler can select any cluster whose prediction level is compatible with the scheduling so far, and it chooses the cluster that optimises the RD criterion. Within this cluster, it chooses *randomly* one frame. This frame is added to the coding window, increasing its size by one. The size of the coding window is reset to one with the new GOP.

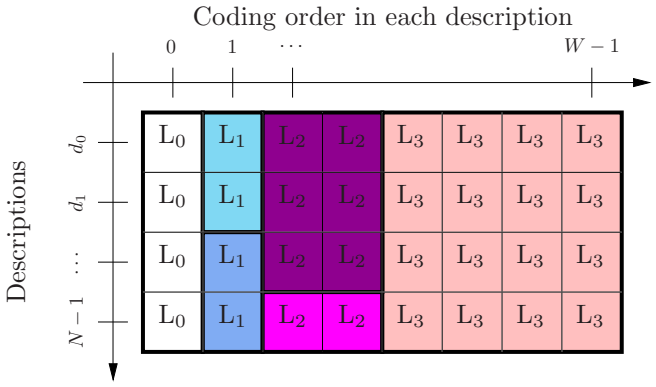


Figure 5.9: Example of MD-GOP clustering. Frames in the same cluster share similar RD properties.

An example of frame clustering of the MD-GOP is presented in Fig. 5.9. There, the frames of the base temporal prediction level for the 4 descriptions have roughly the same RD properties and are therefore assigned to a single cluster. On level L_1, on the other hand, descriptions d_0 and d_1 have similar RD properties between them, but different from descriptions d_2 and d_3, which are in turn close to each other. In this case, two clusters are created containing the frames with similar properties. The same holds true for the level L_2, where descriptions d_0, d_1 and d_2 have been clustered together, while description d_3 was assigned to another cluster. Finally, all frames of level L_3 for all descriptions give similar contributions to distortion and have been assigned to a single cluster.

Large clusters increase the diversity of the scheduling among senders, thus reducing non-innovative packets. However, if clusters are chosen too large, the scheduler will randomly choose among frames with very different values of the objective function, resulting in a sub-optimal performance.

Ideally, the size of the clusters should be chosen according with the expected number of senders that are going to transmit at the same time (so that each sender could select a different frame of the same cluster), which can be roughly estimated with the node density of the network.

In practice, clustering can be performed in several ways. For instance, a coarse but simple scheme is to assign all the frames on the same prediction level to a single cluster. This scheme is independent from the actual RD properties of the sequence and can be easily implemented; nevertheless, it can be quite efficient if the descriptions are actually frame-by-frame balanced. If the corresponding frames of different descriptions have slightly unbalanced properties, then a more sophisticated scheme can be employed.

An example of two different scheduling orders compatible with the clustering in Fig. 5.9 is presented in Fig. 5.10. For the sake of clarity, only the scheduling for the first 16 packets is presented. We can observe that, if only a subset of a cluster is chosen, the two schedulers choose different frames within it. If the whole cluster is chosen, then the frames still differ in the order they are included in the coding window.

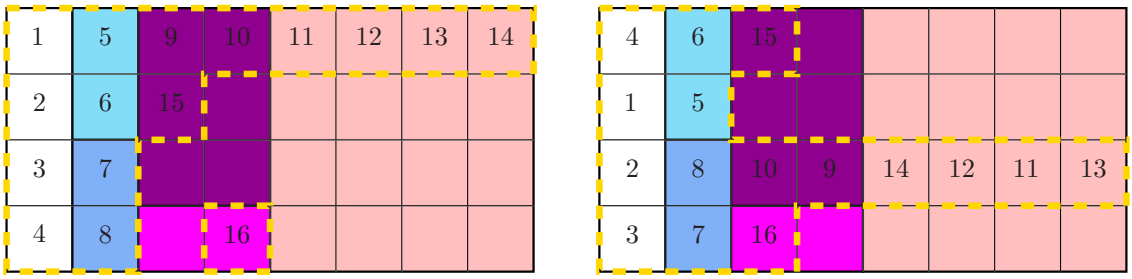


Figure 5.10: Two possible schedules (first 16 packets). The numbers indicate the order in which the frame is included in the coding window. The dashed border identifies which frames have been selected for inclusion in the coding window at the 16th packet.

5.4 Experimental Results

5.4.1 NC extension for ABCD

Here, we present the results of the proposed network coding extension for the ABCD protocol [NGCPP12], and compare them with the result achievable via the Practical NC scheme proposed by Chou *et al.* [CWJ03].

To generate the DAG, we randomly construct a MANET consisting of 100 nodes in a $100 \times 100\,\mathrm{m}^2$ playground. The nodes have the same transmission parameters as in Chapters 3 and 4, with a nominal transmission range of 25 m.

The ABCD protocol [GC11] is run in order to form a directed acyclic overlay. When the source starts broadcasting the stream, the proposed coding strategy is applied at each node. Observing which packets have been decoded, we compute the average Y-PSNR observed by the various users.

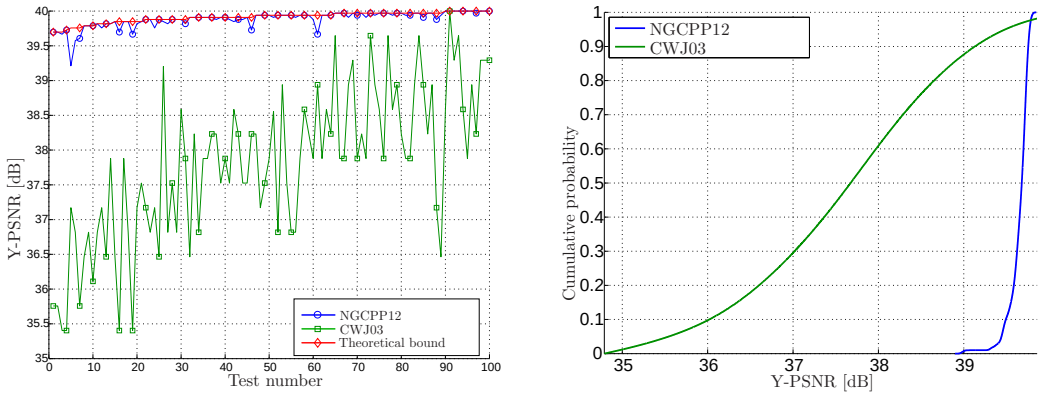
In order to generate the video content we encoded the "Foreman" video sequence (CIF, 30 fps) with the temporal MD technique proposed in Chapter 2, that generates two descriptions, balanced in terms of rate-distortion properties.

The test is repeated 100 times, with different initial position of the nodes, in order to take into account the variability of the network topology. Several tests have been performed with other video contents, but the technique does not appear to be heavily affected by the content.

In Fig. 5.11(a) the results for the average PSNR obtained with our method and PNC. A theoretical bound, obtained by exhaustive exploration of the solutions, is also reported for reference.

For the PNC implementation we assumed that the coding window cannot be set along the time axis, *i.e.*, we do not mix packets with different due-dates, in order to avoid decoding delay, crucial in real-time applications. Therefore, combination of packets can only occur along the descriptions axis, *i.e.*, we mix packets from different descriptions, but having the same due-date. This implies that the length of the coding window equals the number N of descriptions, which is 2 in our test. The coding coefficients are chosen

randomly in GF(256), which has been shown [CWJ03] to give a low probability of building duplicate packets. In order to reduce the number of non-innovative packets injected in the network, we imposed to PNC to use a probabilistic flooding (see Chapter 3), with a retransmission probability (chosen to maximise the average PSNR) of 75 %.



(a) Average PSNR (b) Cumulative distribution of PSNR

Figure 5.11: Comparison between the reference and the proposed technique, for video sequence "Foreman", CIF, 30 fps, 1.8 Mbps.

We notice that the proposed technique performs on average about 2 dB better than PNC. Moreover in Fig. 5.11(b), we report the PSNR cumulative distribution functions for the two techniques. We observe that while in the reference technique the distribution of PSNR is very widespread, using our technique almost all nodes achieve very close, and very high qualities. We ascribe this result to the fact that the efficiency of PNC is considerably affected by the constraints on the length of the coding window (*i.e.*, the fixed size of the generation).

To achieve these qualities, using the reference technique, all nodes collectively injected into the network $4.6 \cdot 10^3$ packets per second on the average, while, with our technique, only $1.2 \cdot 10^3$ packets are sent. We also notice that from one experiment to another, these values do not vary much.

We conclude that, in this scenario, an optimisation technique outperforms a technique based on random coefficients, providing a better video quality to the end users.

5.4.2 RD-optimised scheduling

We present now the results of the proposed RD-optimised scheduling [GNCPP12] and compare them with the results achievable via EWNC applied to an SD-coded stream and EWNC applied on an MD-coded stream, but ordered using a trivial schedule. For SDC, the trivial strategy consists in including the frames in coding order, *i.e.*, by prediction level and, within frames on the same level, play-out order. For MDC, we assume again that frames are included in coding order and, within frames with the same encoding time (*i.e.*, corresponding frames of independently encoded descriptions), the descriptions are

High Bitrate		Medium Bitrate				Low Bitrate			
16	19	22	25	28	31	33	36	39	42

Table 5.1: QPs used in encoding the video sequences.

"Akiyo"	"Hall"	"Foreman"	"City"
"Coastguard"	"Football"	"Stefan"	"Bus"

Table 5.2: Video sequences used in simulations.

selected in a fixed order.

To encode the video sequences, we chose to use 4-descriptions spatial channel splitting technique (see Sec. 2.1.2), *i.e.*, the 4 sub-streams are generated by splitting the original sequence via polyphase down-sampling along rows and columns by a factor of 2. To generate the descriptions, each sub-stream is independently encoded using an H.264/AVC reference encoder JM [Süh11], version 17.0. The encoding algorithm uses the Hierarchical-B closed-GOP structure. A closed-GOP was preferred in order to reduce error propagation in case of losses.

The rate-distortion properties of each frame are exactly measured. Clustering is performed based on temporal prediction level. The average rate and distortion for the frames in each cluster are computed, quantised on on eight bits each, and sent along with the corresponding video frames.

At the decoder side, all the descriptions are independently decoded in order to obtain the N sub-streams, which the receiver interleaves to reconstruct the central sequence. When some descriptions are lost, the receiver oversamples the available sub-streams, interpolating the missing pixels to obtain a good low-resolution frame. When none of the descriptions is available, the loss is concealed using the closest decoded frames.

In order to compare the performance of the method under a variety of inputs, we selected a set of 10 QPs (in Tab. 5.1) and 8 video sequences (in Tab. 5.2) with CIF spatial resolution at 30 frames per second.

The transmission scenario we simulate is depicted in Fig. 5.12. In this scenario, M active nodes $S_m,\ m \in \{1, 2, \cdots, M\}$, intend to transmit the same video sequence $I(k),\ k \in \{1, 2, \cdots, K\}$ to a single receiver R.

In order to allow a clear evaluation of our technique, a discrete-time transmission model is assumed: the time is segmented in *transmission rounds* wherein each active node S_m sends exactly one packet from a predetermined transmission buffer TX_m. Each channel C_m between transmission buffer TX_m and the receiver buffer RX is in general lossy, with independent uniform packet loss probability p_m; the transmissions on different channels do not interfere with each other. At the end of each round, the receiver decodes all the frames available in its buffer RX, generating a reconstructed sequence $\widetilde{I}(k)$. This simple scenario is well suited to model a wireless ad-hoc network where a channel reservation

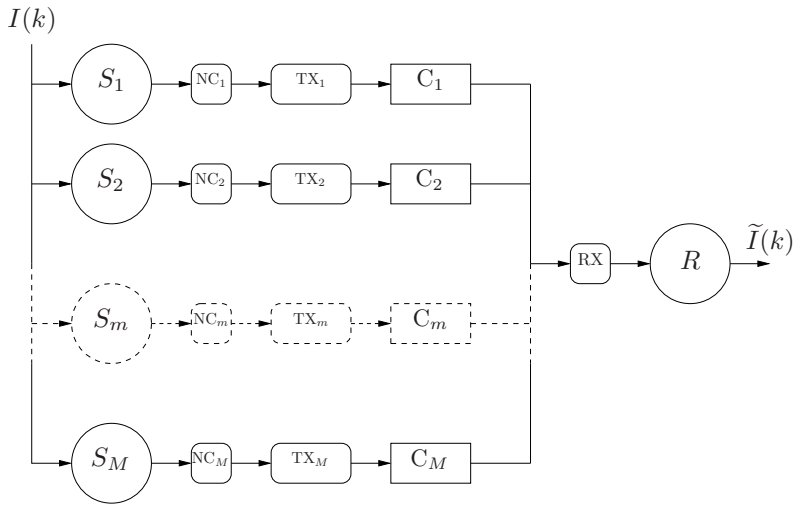
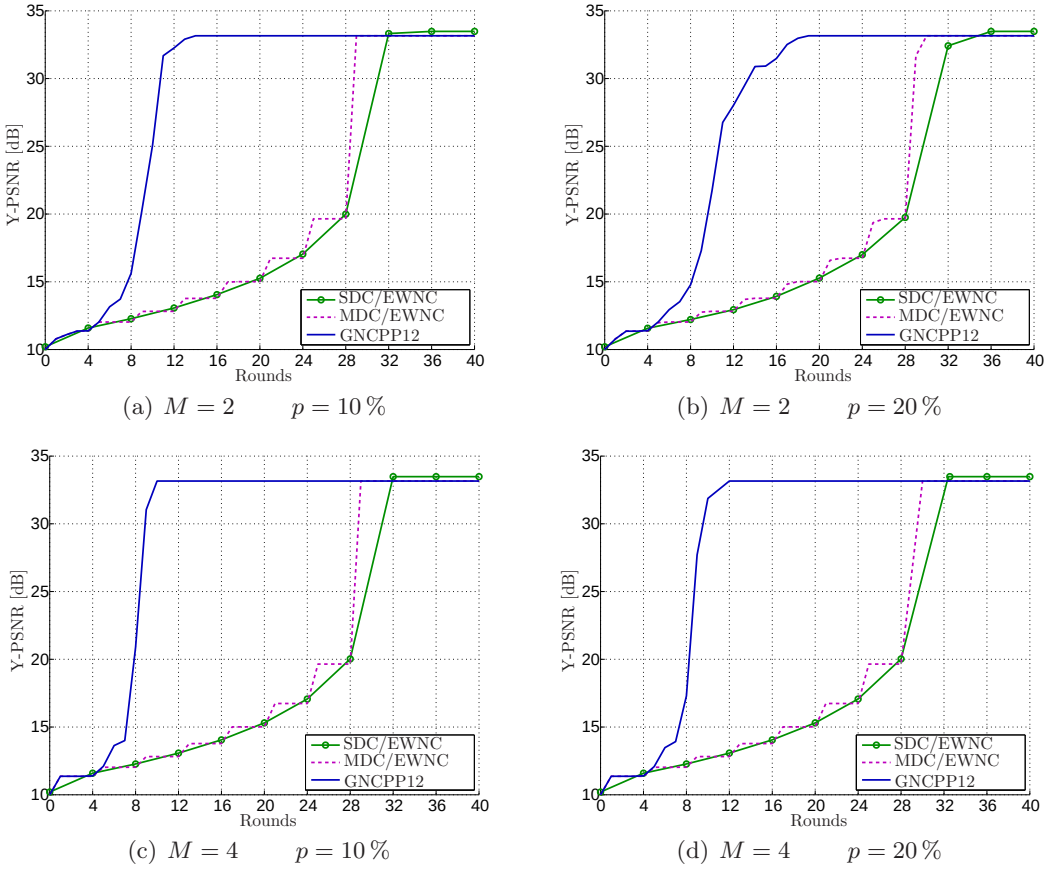


Figure 5.12: Simulated scenario. $I(k)$ and $\widetilde{I}(k)$ are respectively the original and reconstructed frames, $S_m,\ m \in \{1, 2, \cdots, M\}$ are the senders, NC_m their network coding modules, TX_m their transmission buffers, C_m the channels, and RX is the receiver R's buffer.

mechanism, such as the one proposed in Chapter 3, is enforced, providing both discrete-time transmission and channel isolation.

In our simulations, the proposed approach has proven to be able to deliver an acceptable video quality to the receiver in a shorter number of rounds than the reference techniques. As an example, in Fig. 5.13, we report a comparison with the reference techniques under a few different simulation conditions. We observe that, thanks to the variety in the scheduling, our technique is able to reduce the number of linearly dependent coding vectors, and is therefore able to provide a better video quality (in terms of Y-PSNR) in fewer rounds. It should be noted that the final value of the Y-PSNR for the SD-based technique is slightly higher (about 0.5 dB) than that of both MD-based ones, which is a direct consequence of the inherent redundancy among the descriptions of the MD encoding (see Chapter 2). However, this happens after a long enough time (*i.e.*, about 30 rounds), during which MDC/NC has already achieved its final Y-PSNR. We can also observe that the performance of the method benefits from a higher number of active nodes, whereas it is of course negatively affected by the loss rate.



(a) $M = 2$ $p = 10\,\%$ (b) $M = 2$ $p = 20\,\%$

(c) $M = 4$ $p = 10\,\%$ (d) $M = 4$ $p = 20\,\%$

Figure 5.13: Comparison of the average Y-PSNR of the decoded sequences, for M active nodes and packet loss probability p.

5.5 Conclusions

In this chapter, we discussed the subject of network coding. After having given reviewed the seminal articles that defined the basic concepts of NC, we have given a short overview of the state-of-the-art in the field of NC for wireless networks, with a particular attention to network coding for video streaming solutions. Then, we presented two of our own original contribution in this field.

For the first proposed technique, we formulated the problem of broadcasting a video stream encoded in multiple descriptions on an ad-hoc network in terms of finding an optimal set of combination coefficients. Then, we introduced an objective function that takes into account the effect that decoding a reduced number of descriptions has on the total distortion. This framework has been integrated the cross-layer protocol presented in Chapter 3, which provides both an acyclic overlay network and the necessary knowledge of the neighbours' state. Finally, we compared the performance of our technique with the celebrated Practical Network Coding technique combined with a probabilistic flooding. We observed that the limitations of the generation size to the number of descriptions, imposed by the delay constraints, severely affect the performance of the reference technique, which

as a result is consistently outperformed by the proposed approach. This technique, and the relative results, have been presented at the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2012).

The key idea of the second technique is to use an Expanding Window Network Coding scheme in order to guarantee instant decodability to the flow. The frames are included in the coding window in an order determined by an RD-optimised scheduler. In order to reduce the probability of generating non-innovative packets, the source operates a classification of the frames (clustering) that provides a degree of freedom in the choice of the schedule. We compared the performance of our technique with Expanding Window Network Coding applied on both on Single Description and Multiple Description coding, assuming a trivial scheduling order, and (in the case of MDC) limiting the combinations within the same description. We observed that the introduction of the scheduling, jointly with the possibility of mixing packets across descriptions, significantly improves the performance with respect to the reference techniques, in terms of video quality perceived by the user. This contribution has been submitted for publications at the 20th European Signal Processing Conference (EUSIPCO 2012), organised by the European Association for Signal, Speech, and Image Processing (EURASIP).

The results we obtained for both techniques suggest that further research in this direction could be profitable. In particular, we are currently exploring the direction of a joint design of the overlay management protocol, of the optimal choice of network coding coefficients, and the optimal scheduling for inclusion in the coding window, in a scenario where nodes have different up-link capacities and different demands in terms of video quality.

Conclusion and future work

We believe, supported by the observation of the market's trends, that the future of computer networking lies in the combination of two key factors: multimedia content and ubiquitous accessibility.

In particular, we are interested in those scenarios where a live video communication has to be provided in an environment without pre-existing infrastructure, which can occur in a wide range of situations, such as military and rescue operations, business meeting, education, and leisure.

The main purpose of this thesis was to cope with the different challenges that arise in the design of a live video streaming system for mobile ad-hoc networks. This has been put in concrete form by developing several techniques aimed at improving the various components of the system, both from a video coding and from a network point of view.

In the following, we detail our main claims and original contributions formed during the work conducted in preparation of this thesis, and based on the conclusions drawn from our results, we identify different ideas that would be, in our opinion, interesting to investigate.

Video coding technique

In Chapter 2, we have argued that a multiple description video coding technique is the best suited for the applicative scenario of video streaming over mobile ad-hoc networks, for two main reasons. First, the MDC paradigm is able to provide a graceful degradation of the video quality in presence of losses, which are much more frequent in wireless than in wired networks. Secondly, using MDC, each description, or sub-stream, is independent from the others, in the sense that it can be independently decoded (as opposed to scalable video coding, where the sub-streams constitute an hierarchy of layers to be decoded in a predetermined order). This facilitates the task of designing an overlay network for the diffusion of the stream, as the system is free to construct the diffusion tree of each sub-stream independently from the others, thus significantly reducing the complexity. This is particularly crucial in an ad-hoc scenario, as the construction of the overlay that provides routing capabilities to the networks has to be carried out in a distributed fashion by the nodes of the network, without centralised control.

Consistently with this argument, we have proposed a multiple description coding technique following a set of guidelines that make it particularly suited for our target scenario. In particular, our MD coder is based only on pre- and post-processing of the video sequence, and the use of legacy coders (in the presented results, H.264/AVC, but the technique can be applied to any video codec). This choice is justified by the fact that nowadays several companies (Broadcom, Conexant, Realtek, ATI, Nvidia, *etc.*) are producing hardware implementations capable of encoding and decoding H.264/AVC video that are being installed on-board of mobile devices. These implementations can be reused by a pre-/post-processing multiple description coding technique, thus delivering extremely high performances, but with significantly reduced development time and development cost.

Our technique can be categorised as a temporal channel-splitting MDC technique, *i.e.*, is based of the separation and independent encoding of even and odd frames of the video sequence. Side decoding, used whenever only one description is received, is performed using *motion-compensated temporal interpolation* to reconstruct the missing frame. In the first version of our proposed technique, we used a first-order interpolation, that we later replaced with a high-order motion interpolation, in order to be more effective in case of accelerated motion between frames. Central decoding, used when both descriptions are received, is performed as a block-wise linear combination of the two descriptions. The rationale is that the interpolated version of the frame, being generated from the two adjacent frames, can convey information that was not present in the decoded version, as the adjacent frames were not part of its coding process. The combination coefficients are computed at the encoder and encoded in a rate-distortion optimised way.

In our experimental validation, we compared our proposed technique with a similar scheme based on legacy coders, proposed by Liu and Zhu [LZ07, ZL09], that generates two descriptions encoding the video sequence twice with an H.264/AVC coder, but with a different selection of key pictures. Our experimental results show that our technique consistently outperforms our reference, and particularly so in highly lossy environments.

Future work will focus on implementational challenges of this technique, with particular respect to the parallel implementation of the motion estimation [GPP11], made possible by the wide availability of multi-core processors even on-board of mobile devices.

Overlay creation and maintenance

In Chapter 3, we assert that the task of delivering a high-throughput flow, such as a video stream, over a delicate and unreliable network, such as a mobile ad-hoc network, especially in presence of latency constraints, can effectively be addressed only by abandoning the rigid layered design traditional to network protocols and moving on towards a more dynamic cross-layer design.

The first aspect that we investigated was the efficiency of single-hop broadcast transmission in ad-hoc networks. We concluded that, being it outside of the specifications of the

current standards, this service is highly unreliable and inefficient. We therefore proposed a simple modification of the IEEE 802.11x standards based on channel reservation, through the selection of a *control peer*, *i.e.*, a designated receiver of the broadcast communication, responsible for the coordination of the channel, and selected with application-driven criteria. This technique enables a much more reliable channel for broadcast communications, at the price of an increased congestion.

Based on this reliable broadcast scheme, we designed an overlay construction and maintenance protocol, named ABCD, that produces a set of partially edge-disjoint multicast trees (multi-tree); each description of the stream can be transmitted using a different tree. The overlay is the product of a distributed multi-objective optimisation. This optimisation function can be fine-tuned in order to adapt the relative importance of the different objectives (minimisation of the hop-count, minimisation of the number of transmitting nodes, maximisation of the disjointness, *etc.*) to a wide range of possible scenarios. Within the protocol, we also defined a set of strategies to reduce the control overhead, *i.e.*, the number of packets that need to be injected in the network in order to construct the initial overlay, expand it to include newly connected nodes, and repair it in presence of node mobility and disconnections.

From our experimental validation, this protocol has proven to be efficient (in the sense of percentage of packets delivered, video quality perceived by the user, and transmission delay) and to require a small overhead. Moreover, it maintains its efficiency even in presence of mobility of nodes compatible with our application scenario, and also in the case of abrupt failure of a significant share of the nodes.

An interesting perspective for this contribution is to broaden the scope of the protocol to include the physical layer. A study of the literature on the topic led us to conclude that including transmission power control in the parameters of the distributed cross-layer optimisation could in fact prove beneficial to the performance of the streaming application.

Furthermore, an interesting generalisation of the protocol would be to consider a scenario wherein the different nodes have different up-link capacities and different quality-of-service demands. In this scenario, wherein a node could demand to the system more resources than it is willing to provide, it would be interesting to analyse the system from a game-theoretical point of view and to design incentive mechanisms to prevent free-riding, *e.g.*, based on a distributed reputation system.

Congestion-distortion optimisation

Since mobile ad-hoc networks are, by definition, infrastructure-less, they may occur situations where, even with an ideal overlay, the network conditions are simply too harsh to allow the system to deliver satisfying video quality, especially when stringent constraints are imposed over the maximum accepted delay. In order to cope with this situation, we maintain that it is fundamental for the system to be content-aware, as opposed to the

traditional assumption that the delivery network is content-agnostic.

In Chapter 4 we therefore propose to exploit a property of the encoded video, *i.e.*, that different parts of the bitstream provide a different contribution to the overall video quality. We designed a cross-layer *congestion-distortion optimisation* (CoDiO) framework that influences the transmission parameters of the MAC layer based on criteria dictated by the application (*i.e.*, video coding) layer. Namely, for each video frame, the framework decides for how many times the current retransmitting node has to initiate a channel reservation procedure before discarding the frame.

First, the framework establishes the total degradation of the video quality that the users in the network would perceive if the current frame (sent by the current retransmitting node) were lost, by taking into account both the multi-tree overlay, and the multiple paths (generated by the inherently broadcast medium) that are not under direct control of the protocol. Then, for each value of retry limit in a given range, the probability of loosing the frame, and thus the expected distortion, is computed. For each value of the retry limit, the framework also provides an estimation of the congestion generated on the channel, depending on the number of packets that have to be transmitted in the neighbourhood.

The values produced by these two estimations, which are carried out with negligible packet overhead, are combined in a weighted sum function, and the value of retry limit that minimises this function is selected.

This framework, integrated within the ABCD protocol, has been able to grant a consistent gain, in terms both of video quality and of delay reduction, for bitrates up to a few megabits per second and constraint on delay of as low as 100 ms, *i.e.*, in scenarios compatible with video-conferencing service.

Integration with network coding

In the design of the ABCD protocol and of its CoDiO extension, we have been working under the classical routing assumption, *i.e.*, the intermediate nodes of a communication relay the packets they receive without modifying their payload.

Even though this assumption has gone virtually unchallenged for several decades, we find that the promising results obtained in the field of *network coding* (*i.e.*, combination of packets in intermediate nodes) justify a deeper investigation, and we realised that the network coding approach was not alternative, but incremental with respect to our previous contributions.

Thus, in Chapter 5, we present two new recently proposed contributions in this sense. The first contribution consists in a network coding-based overlay refinement, in which ABCD nodes are allowed to send linear combinations of homologous frames of different descriptions. The combinations are chosen independently by each retransmitting node to maximise the video quality of its direct descendants. This scheme has been proven able to ensure that 100 % of the nodes decode almost all frames of all descriptions, with nodes

relaying *at most* one description.

The second contribution is a rate-distortion optimisation framework that operates on the per-hop behaviour of the mobile nodes. It allows to select which frames and in which order should be included in the coding window (*scheduling*), with the objective of minimising the decoding delay and maximising the marginal benefit in terms of video quality. This is justified by the fact that mobile networks are affected by churn and mobility, and the video stream can be interrupted at any moment.

We compared the performance of this scheme with several other techniques, and we observed that the introduction of the scheduling, jointly with the possibility of mixing packets across descriptions, significantly improves the performance in terms of video quality perceived by the user.

Based on the experimental results observed in the validation of these contribution, and based on the conclusions we have drawn from them, we believe that a viable research perspective is the further investigation of a paradigm shift of the work of this thesis towards a network coding perspective.

While our preliminary results, obtained with an integration of our ABCD protocol with network coding turned out to be quite promising, we believe that a much higher performance gain can be achieved by re-designing the protocol, without prejudice to its key principles, in a network coding perspective. In other words, while in our latest works the network coding part and the overlay creation part have been maintained to some degree independent, in future works we shall design the overlay creation and maintenance aware of the network coding.

Annex A – Cooperative solutions for large-scale video diffusion

In this annex, we discuss the topic of large-scale multimedia content distribution, and in particular, we investigate the benefits that the *peer-to-peer* paradigm can bring to this kind of service [JCWF07].

We shall first illustrate, in Sec. A.1 the general problem and its challenges; then, in Sec. A.2, we present an interesting solution in this field: the Orchard algorithm, developed in 2007, which aimed at building a cooperative distribution network meeting video multicast requirements. In Sec. A.3, we identify the limitations of this algorithm, and propose a new algorithm, Cedar, that integrates the functionalities of Orchard with new ones that speed-up the multicast tree construction, building a topologically-aware overlay network that minimises the end-to-end delay. In Sec. A.4, we present an experimental validation of our proposed algorithm. Finally, in Sec. A.5, we draw our conclusions and identify some viable ideas for future work.

A.1 Introduction

Multimedia content distribution has been a widely investigated topic over the last few years. Traditionally, multimedia content has been delivered using the client/server paradigm, which is currently being abandoned, as a server alone could not be able to handle a large number of clients. Nevertheless, the client/server paradigm is still employed; *e.g.*, YouTube uses it for low popularity videos [SRL08].

Content Distribution Networks (CDNs) are also used for distributing multimedia content [KWZ01]. They inherit from the client/server model, but employ multiple servers interconnected through a dedicated infrastructure, with the aim of partitioning the large load [PWCS02]. YouTube, for instance, employs the content distribution networks of Akamai to deliver popular videos to users.

Nowadays, a new paradigm for multimedia content distribution is emerging: video multicast. In this context, two types of multicast are considered: IP multicast and Application Level Multicast (ALM). IP multicast takes advantage of the implementation of multicast functionalities in the network layer, as at these levels the network topology is

best known.

Unfortunately, IP multicast suffers from scaling issues and a lack of support for higher layers functionalities. It also requires costly infrastructural changes, since the routers have to integrate IP multicast [DLL+00]. ALM is thus favoured for its low cost and its simplicity of implementation, which does not require any infrastructural changes, as the routers need only to support IP unicast.

An interesting alternative paradigm for multimedia content distribution is represented by Peer-to-Peer (P2P) networks, which provide inherent self-organisation and resource scalability properties [JCWF07].

Using P2P to distribute multimedia content has been investigated, for instance, within the popular P2P community BitTorrent[1] leading to the appearance of BitTorrent Live Streaming, still under development.

The development of media content distribution over P2P networks has encountered a few setbacks. First, most P2P systems suffer from *free-riding* [YLHX06], *i.e.*, the presence in the system of peers that receive data without relaying them can limit the large scale distribution of the content. Second, in P2P networks, there is the implicit assumption that a peer is connected for a long period of time, due to the traditional use of P2P networks in file sharing; conversely, in video streaming connection times can be quite small, with a high departures rate of peers. Along with high arrival rates, this situation is referred to as *churn* [SR05], a behavioural characteristic that we already introduced in the context of ad-hoc networking in Chapter 3, and that P2P systems must also be resilient to. Finally, a video stream may require a high bitrate, so the bandwidth of users (both in upload and download) must be adequate, which is not always the case.

In this annex, we shall first study an existing solution to the video multicast problem, *i.e.*, the Orchard algorithm, which builds a cooperative distribution network for video multicast using multi-tree overlays and multiple description coding. Then, in Sec. A.3, we identify the limitations of the Orchard algorithm, and propose a set of new functionalities, collectively referred to as Cedar, to speed-up the overlay construction and to reduce the end-to-end delay. An experimental validation of the Cedar algorithm is presented in Sec. A.4, and conclusions are drawn in Sec. A.5.

A.2 The Orchard algorithm

Within the context of cooperative application layer multicast, in 2007 Mol *et al.* have developed the Orchard algorithm [MES07] to deal with the challenges of free-riding, churn, and high bitrates.

Orchard attempts to minimise free-riding using a form of *tit-for-tat*. Tit-for-tat, or *equivalent retaliation*, is a well-known and highly effective strategy in game theory, ori-

[1] Website: `http://www.bittorrent.com/`

ginally proposed for the iterated prisoners' dilemma [AH81]. An agent playing by this strategy will initially cooperate, then respond in kind to an opponent's previous action. If the opponent previously was cooperative, the agent is cooperative. If not, the agent is not. However, the agent is quick to "forgive", *i.e.*, to restore a cooperative behaviour after a retaliation time is over. In this respect, Orchard is inspired to the celebrated BitTorrent peer-to-peer protocol, which uses a tit-for-tat strategy to optimise the download speed [Coh03, AML+05].

Orchard also assumes that Multiple Description Coding is used to encode the stream, in order to guarantee a robust and flexible system (see Chapter 2), so that each description can take a completely different route from the others.

In this section, we present the basics of the Orchard algorithm. First, we describe the general primitives, then we show how the different peers cooperate with each other in order to obtain the descriptions.

In Orchard, each description is designated by a colour. The source, having all the descriptions, is assigned the colour *white*. Conversely, a peer that has just joined the system, and has therefore no descriptions, is designed as *blank*. As soon as it will get its first description, this peer will be assigned with the respective colour, that it will maintain when it will get the others. In other words, even if a node receives all descriptions, it will still be assigned a single colour, *i.e.*, that of the first description it received.

One of the key ideas of Orchard is that, even if a peer has more descriptions, it can only distribute the description corresponding to its own color.

A peer will always try to maintain a neighbour set of m neighbours. The identities of these neighbours are obtained by each peer through an external server, dedicated to this purpose. Furthermore, each peer also stores a list of its neighbours' properties, which include their respective colours. Hence, a peer knows exactly which neighbour to ask to obtain a description which it does not already have.

An Orchard peer, in order to obtain the different descriptions, can propose to its neighbours the following three types of deals:

***Join* deal** If a peer has the source as its neighbour, it could try to strike a join deal, where it simply asks the source for a particular description. If the request meets the source criteria, it will accept and reply with the requested description, without expecting anything in return. This is the only form of free-riding tolerated in the system.

***Exchange* deal** If two peers have different colours and require each other's color, they can simply exchange their descriptions.

***Redirection* deal** The above deals alone are not enough to supply every peer with all the descriptions, simply because the source has only sufficient bandwidth to supply free descriptions to a small set of peers. Hence, a new peer that joins the system

may not be able to obtain a color because the source could be already saturated, and it cannot strike an exchange deal because it has nothing to exchange (*bootstrap problem*). To solve this, the Orchard algorithm provides the redirection deals. Redirections are always based on an exchange. A peer exchanging descriptions may receive a redirection request, upon which – if it accepts – it redirects its outgoing description through the requesting peer, *i.e.*, the requesting peer will receive the description and will send it to the original destination to maintain the exchange structure. The requesting peer will receive its new parent's color if it was initially blank. Redirections through coloured peers are possible as well, but the protocol prescribes that they have to be broken and replaced with a redirection through a blank peer whenever it is possible.

A.3 The Cedar algorithm

In the next section, we present an evaluation of Orchard where we expose its principal weaknesses and propose potential solutions to overcome them.

A.3.1 Limitations of the Orchard algorithm

The exploratory value of the Orchard algorithm notwithstanding, several challenges resulting from the practical deployment of large scale video multicast infrastructure remain unaddressed.

First of all, the Orchard algorithm creates and maintains its ALM trees with barely any regard to network related metrics such as delay and jitter. This means that in a real world scenario, even though all video frames are received by a user, they could still have accumulated so high a delay that they have passed their play-out date, resulting in an impaired video quality.

Secondly, peer are assumed to be cooperative with each others in order to speed up their connection time; this assumption contrasts with the reality, as peers are more likely to be self-interested, rather than cooperative. Cooperation can indeed be achieved, providing that a proper incentive scheme is produced [PVdS10b].

Furthermore, there exist pathological topologies in which a node fails to join the network in a reasonable time. Even with the redirection deals and the relaxation offered by the redirections through coloured peers, deadlock situations are still possible. Indeed, it is possible that the source becomes saturated and the exchanges that can be redirected become rare. In this case, a given peer will spend precious time searching for peers having the demanded descriptions with whom it can strike a successful deal, and consequently, the general multi-tree construction is slowed down.

While it is true that Orchard proposes that a peer changes its color accordingly to the demand of its neighbours to overcome these situations, the fact remains that we can gain

in the multi-tree construction speed if we implement additional functionalities that give, in the first place, more opportunities for peers to obtain descriptions. We may also find that a peer's color change is not necessary anymore.

Another obvious setback in the algorithm is the random selection of neighbours, which may lead to collaborations between geographically distant peers, thus increasing the average end-to-end delay. This can be countered by implementing some form of Network Awareness, to build an overlay network which takes into account the geographical network topology.

Finally, the free-riding problem, at least partially, remains. Although it is extremely minimised in Orchard, it is still present in Join deals, and malicious peers can take advantage of this weakness to get free descriptions. A peer getting a description should always repay with something, even if it is not a description, *e.g.*, it could be some kind of commitment towards the received description, or a high trust value in a reputation management system.

A.3.2 Proposed amendments

We propose here a set of new functionalities to accelerate the multi-tree construction. The aim is to inject more flexibility on the collaboration level in the system to avoid deadlock situations. Thus, we had to find new types of collaborations between peers, to increase the probability of a peer finding another peer with the demanded description and who was willing to collaborate. Note that a peer can also try to strike one of the classic Orchard deals, *i.e.*, these functionalities integrate the Orchard deals, do not replace them.

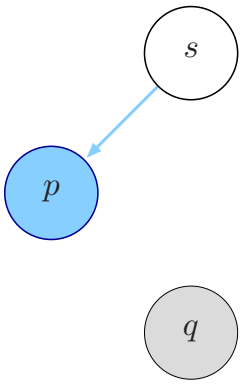
Transferable Joins

In Orchard, an instance of free-riding occurs whenever a Join deal is struck, as the Source delivers a description to the requesting peer without demanding anything in return. In order to reduce this effect, we propose that a peer, in exchange for receiving a description from the source, commits itself to provide this description to another peer upon request, as depicted in Fig. 6.1.

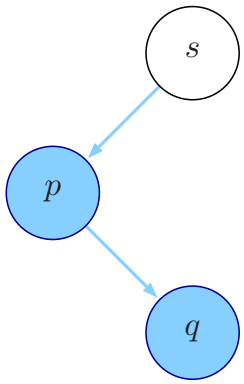
This functionality, that we refer to as *Transferable Joins*, by mitigating the bootstrap problem, allows to considerably speed up the multi-tree construction: while in Orchard the number of available *slots* that allowed direct connections to the server was limited, in Cedar, using the Transferable Joins, we always have the same number of free slots at any time in the system, since a peer that reserves a slot offers its neighbours a new one to connect to. This effectively avoids system saturation.

Super-peers

Super-peers are peers that are very bandwidth resourceful. They can therefore be used as proxy servers, delivering connections to other peers, *i.e.*, they can act as sources within

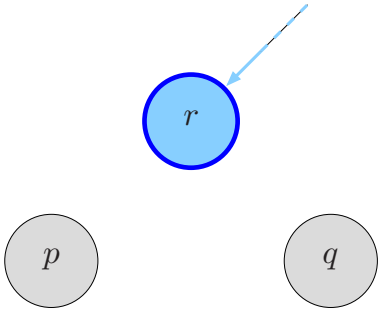


(a) A peer p has a Join deal with the source s for color blue. The source has no more direct connections to offer.

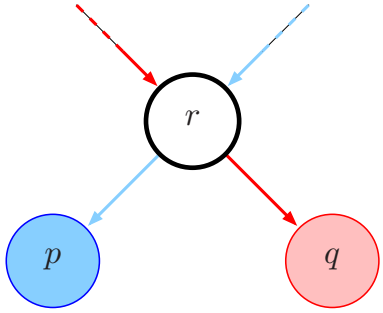


(b) Peer q joins the system. Being unable to strike a Join deal with s, it asks color blue to p, which is obliged to provide it.

Figure 6.1: Transferable Join



(a) Super-peer r is receiving only colour blue. Peers p and q are still blank.



(b) As soon as r receives colour red, assuming these are the only colours, it becomes *white*, *i.e.*, can act as a source to p and q, accepting their Join deals on either colour.

Figure 6.2: Super-peers

their neighbourhood, as depicted in Fig. 6.2.

Once they have all the descriptions, the super-peers can accept Join deals (in Orchard only addressed to the source), thus delivering descriptions to the requesting peers in exchange of their commitment to transfer this description upon request (see Sec A.3.2).

This functionality allows to have hierarchical levels of distribution, where proxy servers negotiate with the main server or source (top level) and clients collaborate with those super-peers (bottom level). The resulting model is more representative of the reality of the topology and can be easily integrated in the framework of P4P [XKSY08]. P4P, or *Proactive Provider Participation for P2P*, is an innovative paradigm in which the Internet service providers (ISPs) cooperate with the P2P software in order to optimise peer-to-peer connections. The key idea of P4P is that the ISPs may, through dedicated servers, provide up-to-date information on how the network is configured, which the data routes the ISP

prefers, and which connections to avoid. The P2P software can then connect to peers that are "closer", in terms of network metrics, instead of choosing peers randomly.

One might argue that the introductions of super-peers also incurs the risk of generating more free-riding in the system, as more free descriptions are distributed. However, we can consider that in return to getting the free description, a peer gives the super-peer a high *reputation* level.

In a reputation management system, each peer in the network is assigned a trust value based on the peer's history of actions, thus aiming to reduce the incidence of malicious behaviours [KSGM03, DPT08, PVdS10b, PVdS10a].

Redirection-to-Exchange Transformation

In Orchard, in order to avoid redirection chains and the consequent latency increase, an Exchange deal could not be redirected more than once.

However, if a situation occurs where an exchange is redirected in both ways through the same peer, then we can consider that the peer getting the redirections is in fact *exchanging* descriptions with the peers it is getting the descriptions from. The advantage in identifying these situations is that Exchange deals, by definition, can be redirected, as depicted in Fig. 6.3.

Special Redirection

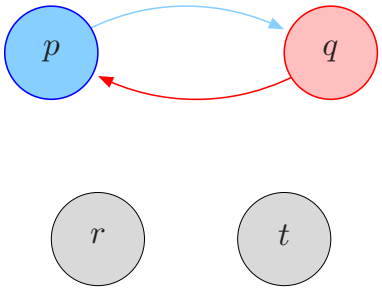
Since an Exchange consists of two distinct transfers of descriptions, it can be redirected at most twice, once in each direction. If a peer succeeds in getting a redirection for one colour (*i.e.*, description), it will most likely be interested in the second colour of the deal as well.

Therefore, instead of letting a peer searching for another peer that has that description, we simply force it to ask for the redirection in the other direction as well. This functionality reduces considerably the search time, thus accelerating the multi-tree construction.

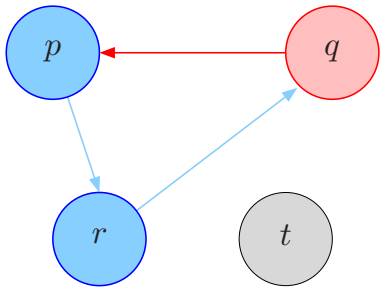
Moreover, if the request is accepted, we find ourselves in a situation where an exchange is redirected in both ways, through the same peer. Consequently, this functionality encourages the occurrence of the situation in Fig. 6.3(c), where the Redirection-to-Exchange Transformation can be operated, which is extremely efficient in reducing the multi-tree construction delay.

A.3.3 Network Awareness

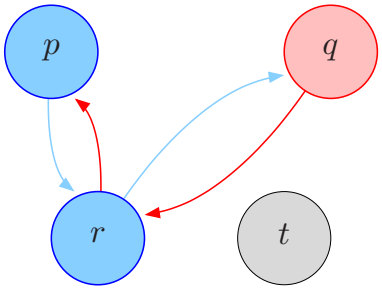
An overlay network is a virtual network, built on top of an underlay network composed of stations, switches, routers, modems and other networking equipments, and which is completely transparent to the user. Links in the overlay network are end-to-end links, each one potentially corresponding, in the underlay network, to a series of point-to-point connections between different equipments.



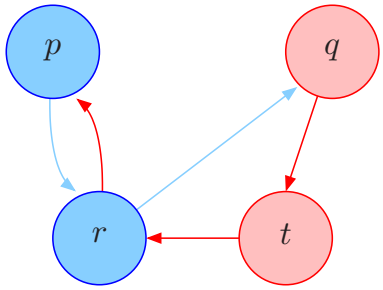
(a) Peers p and q have an exchange deal.



(b) Peer r strikes a Redirection deal with peer p for colour blue.



(c) Peer r also strikes a redirection deal with peer q for colour red. The exchange in Fig. 6.3(a) has been redirected in both ways through the same peer r. For the Orchard algorithm, these deals cannot be further redirected.



(d) Using the Redirection-to-Exchange Transformation, peer t can strike a redirection deal with peer q for colour red, as the deal between q and r is now considered an exchange, and can be therefore be redirected.

Figure 6.3: Redirection-to-Exchange Transformation

This can imply considerable delays, if the peers collaborating on the application level are in fact geographically distant. Thus, we need to find a way to create the overlay network in such a way that logical links are formed between peers that are geographically close, thus avoiding distant connections. Such a technique, called *distributed binning* and which falls into the Network Awareness paradigm, was developed in 2002 by Ratnasamy *et al.* [RHKS02]

The concept is fairly simple: each peer measures the distance separating it from N stations that have known constant "positions". Those stations are referred to as *landmarks*. The "distance" metric considered here is the round trip time (RTT), which is the time a peer takes to receive an ICMP ping reply after issuing a request. Then, the peer sorts the landmarks in order of increasing RTT. The resulting ordered vector of landmarks represents the coordinate of the peer. A *bin* is a set containing peers that have the same ordering. Peers that are closer to each other are likely to have the same ordering, and will

thus belong to the same bin.

However, that ordering only takes into account relative distance information. Absolute distance information is accounted for in a level vector, of size N. The level vector uses a partition of the real axis into M intervals, numbered from 0 to $M-1$. For the ith element in the level vector, we look at the i-th element in the ordering described above, which corresponds to a landmark. Since the RTT to that landmark has already been computed, we check in which interval this RTT falls (we are thus performing an RTT quantisation) and store the index of that interval in the level vector. The bin of a node is thus the level vector appended to the ordering of the landmarks (*e.g.*, $[l_3\ l_1\ l_2 : 0\ 0\ 2]$ where l_i represents landmark i).

The advantage of such a technique is that it is completely distributed, since a peer does not need to know the distance between landmarks, nor the distance of other peers to the different landmarks.

In the context of the Cedar algorithm, a node can compute its bin and send it to a central server. Once the server receives the bins, it can compute the distances among the different peers; it thus selects the m closest neighbours for each peer.

In the initial state of the system, where peers are only beginning to join, the server might not find enough close neighbours for a given peer and may therefore be forced to select distant ones. To avoid this, we can establish an initial identification phase, where peers only submit their bins to the server and a neighbour selection phase, where the server has sufficient peers (or bins) in the system to select the closest neighbours for each one of them.

A.4 Experimental results

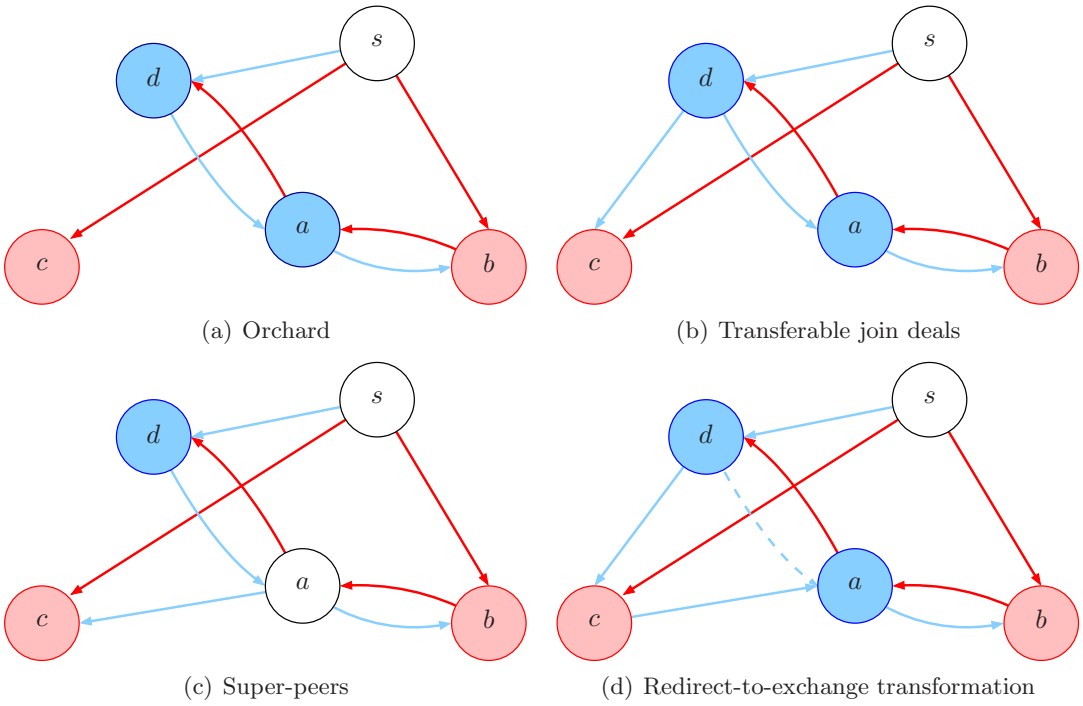
In this section, we evaluate each proposed functionality and compare it with the original Orchard algorithm.

Both the Orchard algorithm and the Cedar algorithm have been simulated using *ns-2* (see Chapters 3 and 4). Peer arrivals in all our tests have been modelled as Poisson random variables with an average of 2 peer arrivals per second. The video stream is encoded in two descriptions using the technique presented in Chapter 2. For each test, the results presented here are averaged over 10 repetitions.

A.4.1 Deadlock resolution

We analyse now the possibilities offered by the new functionalities introduced in Cedar. In order to do that, first of all we study the cases where the Orchard algorithm is unable to deliver both descriptions to all users, and what solutions are available in Cedar in these situations.

In Fig. 6.4, we show an example of stable topology (in the sense that no change in



(a) Orchard (b) Transferable join deals

(c) Super-peers (d) Redirect-to-exchange transformation

Figure 6.4: Example of deadlock resolution using Cedar features.

the overlay occur once this topology is reached) resulting from one of our simulations, focusing on a set of four peers located in proximity of the source. Notice that the server is saturated, *i.e.*, it does not have available slots to accept a new join deal.

In Fig. 6.4(a), we see that, running Orchard, peer c is receiving colour red, but is unable to obtain colour blue. In this scenario, no viable option exists, since peers b and d have already redirected their exchanges through a.

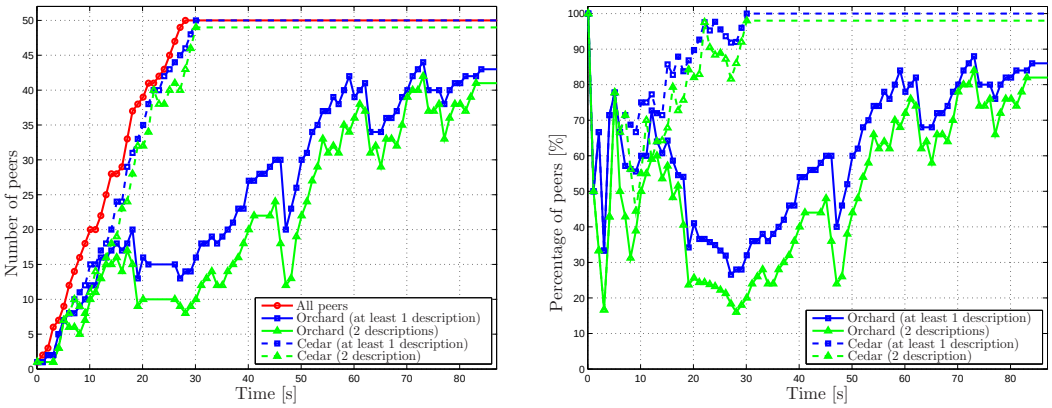
On the other hand, running Cedar, this deadlock can be resolved by using any of the several newly introduced features. In order to see how each one works, we have tested them separately over the same topology.

In Fig. 6.4(b)), we show a resolution of the deadlock using the transferable join deal, where peer c requests d to transfer its join deal for colour blue. Peer d is compelled to accept the deal.

Another possible solution to the deadlock is provided in Fig. 6.4(c) using the super-peers. In this case, peer c strikes a join deal with super-peer a as if it were the source, in order to get colour blue.

Finally, in Fig. 6.4(d), we see how thanks to redirect-to-exchange transformation, peer c is able to obtain colour blue by redirecting the exchange between a and d.

In a full implementation of Cedar, *i.e.*, when all four solutions are available, the latter is chosen, as, like in Orchard, redirection deals are always preferred to join deals (both to the source and to super-peers).



(a) Absolute number of peers.

(b) Percentage of connected peers.

Figure 6.5: Peers receiving at least one and two descriptions over time.

A.4.2 Received descriptions

The combination of all these features allows Cedar to maintain a much higher number of received descriptions than Orchard.

As shown in Figure 6.5, at the beginning of the simulations the performances of the two protocols are comparable, since no situation of deadlock arises.

However, in the long run, the Cedar features create more opportunities for other peers to get their colours, resulting in a non-negligible performance gap with respect to the original Orchard protocol.

A.4.3 Connection time

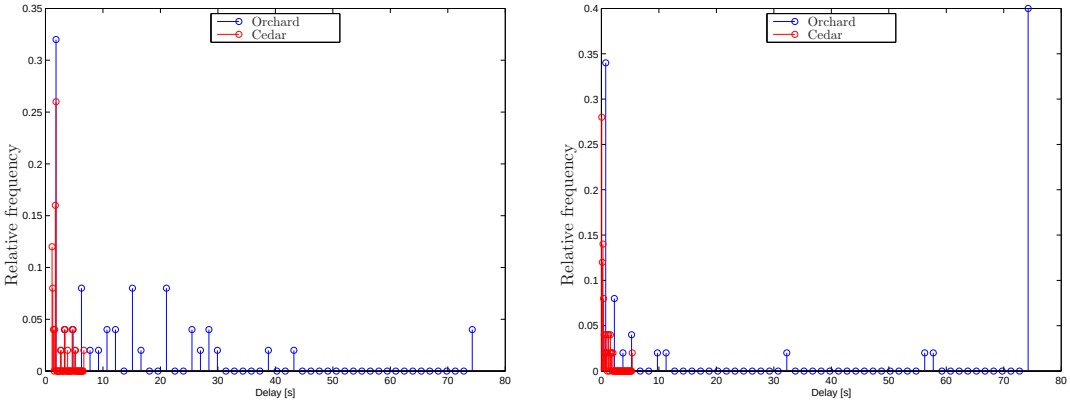
The features introduced in Cedar also significantly reduce the average time that it takes to a generic peer in order to obtain its descriptions, as shown in Figure 6.6.

While in Orchard, the possible connection time is widespread over the range of 42 s for the first descriptions (Fig. 6.6(a)) and 58 for the second (Fig. 6.6(b)), in Cedar this time is always bounded within the 5 s from the beginning of the simulation.

Notice that a peer may also experience an infinite connection time, in the sense that it is unable to obtain its first or second description. In order to make this relevant information visible, those peers are represented in Fig. 6.6(a) and 6.6(b) as having a connection time of 75s. We observe that while in Orchard a significant number of peers ($\sim 40\,\%$) are unable to receive their second description, and a small number of peer ($\sim 5\,\%$) is even unable to receive their first one, neither of these situations ever occurs in Cedar.

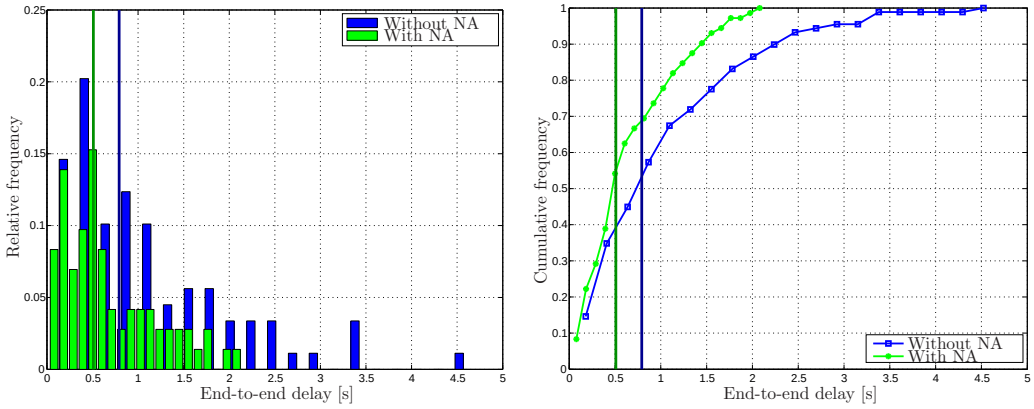
A.4.4 End-to-end delay

Using Network Awareness, a designated server assigns a requesting peer geographically close neighbours, thus reducing the end-to-end delay to get the descriptions distributed



(a) First description. (b) Second description.

Figure 6.6: Probability mass function of the connection time.



(a) Probability mass function. (b) Cumulative distribution function.

Figure 6.7: Distribution of end-to-end delay. Vertical bars represent the average delay.

by the source.

In this experiment, we have considered a *flash crowd* scenario, *i.e.*, all nodes connect almost at the same time at the beginning of the simulation. Moreover, we have considered that peer connection times is distributed as an exponential random variable with an average of 60 seconds.

In Fig 6.7 we compare the distribution of the end-to-end delay over the users of the system with or without network awareness of the neighbouring server.

We observe that, using network awareness, the end-to-end delays are always lower than 2.5 seconds, with an average of 0.5 s, while the maximum delay can be as high as ~ 4.5 seconds and the average delay 0.75 if network awareness is not used, *i.e.*, our technique allows a reduction of more than 30 % of average delay, and almost 45 % of maximum delay.

A.5 Conclusions

In this annex, we discussed to topic of cooperative video multicast in large scale networks.

After an introduction on multimedia content distribution, we have focused our attention the well-known Orchard algorithm, designed to provide a cooperative video multicast service without free-riding.

We have carried out an analysis of Orchard, identified its limitations, and proposed a set of new features, collectively refered to as Cedar, aimed at avoiding deadlocks, reducing the free-riding and, most importatly, reducing the overlay construction time and the end-to-end delay. By also integrating network awareness into the Cedar algorith, we were able to further reduce the end-to-end delay, especially in presence of flash crowds.

Possible future work includes a reputation management system, *i.e.*, based on tokens, that incentivate peers to act in the best interest of the system, in order to gain the reputation needed to strike deals that are, in the short term, unfavourable to their counterpart.

Publications

Journal articles

1. Claudio Greco and Marco Cagnazzo, "A cross-layer protocol for cooperative content delivery over mobile ad-hoc networks", *Inderscience International Journal of Communication Networks and Distributed Systems*, vol. 7, no. 1–2, pp. 49–63, June 2011.

2. Claudio Greco, Marco Cagnazzo, and Béatrice Pesquet-Popescu, "Low-latency video streaming with congestion control in mobile ad-hoc networks", To appear in *IEEE Transactions on Multimedia*, vol. 14, no. 4, August 2012.

Conference papers

1. Claudio Greco, Marco Cagnazzo, and Béatrice Pesquet-Popescu, "H.264-based multiple description coding using motion compensated temporal interpolation", *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, Saint-Malo, France, October 2010.

2. Claudio Greco, Marco Cagnazzo, and Béatrice Pesquet-Popescu, "ABCD : Un protocole cross-layer pour la diffusion vidéo dans des réseaux sans fil ad-hoc", *Proceedings of the Groupe d'Études du Traitement du Signal et des Images*, Bordeaux, France, September 2011.

3. Claudio Greco, Giovanni Petrazzuoli, Marco Cagnazzo, and Béatrice Pesquet-Popescu, "An MDC-based video streaming architecture for mobile networks", *Proceedings of the IEEE Workshop on Multimedia Signal Processing*, Hangzhou, People's Republic of China, October 2011.

4. Irina Delia Nemoianu, Claudio Greco, Marco Cagnazzo, and Béatrice Pesquet-Popescu, "A framework for joint multiple description coding and network coding over wireless ad-hoc networks", *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Kyoto, Japan March 2012.

5. Elie Gabriel Mora, Claudio Greco, Béatrice Pesquet-Popescu, Marco Cagnazzo, and Joumana Farah, "Cedar: an optimized network-aware solution for P2P video multicast", *Proceedings of IEEE International Conference on Telecommunications*, Jounieh, Lebanon, April 2012.

6. Claudio Greco, Irina Delia Nemoianu, Marco Cagnazzo, and Béatrice Pesquet-Popescu,

"A network coding scheduling for multiple description video streaming over wireless networks", *Proceedings of the European Signal Processing Conference* (Accepted for publication), Bucharest, Romania, August 2012.

7. Irina Delia Nemoianu, Claudio Greco, Marco Cagnazzo, Béatrice Pesquet-Popescu, [Title omitted due to double-blind reviewing], *Proceedings of Visual Communications and Image Processing* (Submitted), San Diego, CA, USA, November 2012.

Book Chapters

1. Claudio Greco, Irina Delia Nemoianu, Marco Cagnazzo, Jean Lefeuvre, Frédéric Dufaux, Béatrice Pesquet-Popescu, "Multimedia Streaming", *Elsevier E-Reference Signal Processing* (In preparation).

Project reports

1. Project RNRT DITEMOI, "État de l'art et spécifications des techniques de diversité", Project deliverable D2.4, July 2009.

2. Project RNRT DITEMOI, "Chaine optimisée : Modules logiciels correspondants et rapport de performance", Project deliverable D3.2, May 2010.

3. Project SWAN, "Source aware network coding", Project presentation at the Digiteo Annual Forum (Poster 51), October 2011.

Bibliography

[AAD+07] X. Artigas, J. Ascenso, M. Dalai, S. Klomp, D. Kubasov, and M. Ouaret, "The DISCOVER codec: architecture, techniques and evaluation", in *Proceedings of Picture Coding Symposium*, Lisbon, Portugal, Nov. 2007. *Cited in Sec.* 2.2.1

[ACLY00] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow". *IEEE Transactions on Information Theory*, vol. 46 (4), pp. 1204–1216, Jul. 2000. *Cited in Sec.* 5, 5.1, 5.1.1

[AH81] R. Axelrod and W. Hamilton, "The evolution of cooperation". *Science*, vol. 211 (4489), pp. 1390–1396, Mar. 1981. *Cited in Sec.* A.2

[AMB+04] Y. Andreopoulos, A. Munteanu, J. Barbarien, M. Van der Schaar, J. Cornelis, and P. Schelkens, "In-band motion compensated temporal filtering". *Signal Processing: Image Communication (Elsevier Science)*, vol. 19 (7), pp. 653–673, Aug. 2004. *Cited in Sec.* 1.2.7

[AML+05] N. Andrade, M. Mowbray, A. Lima, G. Wagner, and M. Ripeanu, "Influences on cooperation in BitTorrent communities", in *Proceedings of ACM/SigComm Workshop on Economics of Peer-to-Peer Systems*, Philadelphia, PA, USA, Aug. 2005. *Cited in Sec.* A.2

[ARK+04] A. Al, B. Rao, S. Kudva, S. Babu, D. Sumam, and A. Rao, "Quality and complexity comparison of H.264 Intra mode with JPEG2000 and JPEG", in *Proceedings of IEEE Internantional Conference on Image Processing*, Singapore, Oct. 2004. *Cited in Sec.* 1.2.2

[ASPEF01] M. Alasti, K. Sayrafian-Pour, A. Ephremides, and N. Farvardin, "Multiple description coding in networks with congestion problem". *IEEE Transactions on Information Theory*, vol. 47 (3), pp. 891–902, Mar. 2001. *Cited in Sec.* 4.3

[AWTW02] J. Apostolopoulos, T. Wong, W.-T. Tan, and S. Wee, "On multiple description streaming with content delivery networks", in *Proceedings of IEEE International Conference on Computer Communications*, New York, NY, USA, Jun. 2002. *Cited in Sec.* 4.3

[AZG02] A. Aaron, R. Zhang, and B. Girod, "Wyner-Ziv coding of motion video", in *Proceedings of Asilomar Conference on Signals, Systems and Computers*, Pacific Grove, CA, USA, Nov. 2002. *Cited in Sec.* 2.1.4

[BAA08] H. Badis and K. Al Agha, "Optimal path selection in ad-hoc networks based on multiple metrics: Bandwidth and delay", in *Advances in Wireless Ad-Hoc and Sensor Networks*, Chap. 2, Springer, 2008. *Cited in Sec.* 3.2

[BCG05] R. Bruno, M. Conti, and E. Gregori, "Mesh networks: commodity multi-hop ad-hoc networks". *IEEE Communications Magazine*, vol. 43 (3), pp. 123–131, Mar. 2005. *Cited in Sec.* 3.2

[BDV00] R. Balan, I. Daubechies, and V. Vaishampayan, "The analysis and design of windowed Fourier frame based multiple description source coding schemes". *IEEE Transactions on Information Theory*, vol. 46 (7), pp. 2491–2536, Nov. 2000. *Cited in Sec.* 2.1.3

[Bel58] R. Bellman, "On a routing problem". *AMS Quarterly of Applied Mathematics*, vol. 16, pp. 87–90, 1958. *Cited in Sec.* 3.2

[Bjø01] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves", in *ITU-T VCEG Meeting*, Austin, TX, USA, Apr. 2001. *Cited in Sec.* 2.2.3

[BM08] J. Bondy and U. Murty, *Graph theory*, Springer, 2008. *Cited in Sec.* 5.1.1

[BMAA02] M. Benzaid, P. Minet, and K. Al Agha, "Integrating fast mobility in the OLSR routing protocol", in *Proceedings of IEEE International Conference on Mobile and Wireless Communications Networks*, Stockholm, Sweden, Sep. 2002. *Cited in Sec.* 3.2

[BMAAP04] H. Badis, A. Munaretto, K. Al Aghal, and G. Pujolle, "Optimal path selection in a link state QoS routing protocol", in *Proceedings of IEEE Vehicular Technology Conference*, Milan, Italy, May 2004. *Cited in Sec.* 3.2

[BMJ+98] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad-hoc network routing protocols", in *Proceedings of ACM International Conference on Mobile Computing and Networking*, Dallas, TX, USA, Oct. 1998. *Cited in Sec.* 3.2

[Bov00] A. Bovik, *Handbook of Image & Video Processing*, Elsevier Academic Press, 2000. *Cited in Sec.* 1, 1.1, 1.2.4

[BTB+03] I. Bajić, O. Tickoo, A. Balan, S. Kalyanaraman, and J. Woods, "Integrated end-to-end buffer management and congestion control for scalable video communications", in *Proceedings of IEEE Internantional Conference on Image Processing*, Barcelona, Spain, Sep. 2003. *Cited in Sec.* 4.1

[CAB10] M. Cagnazzo, M. Antonini, and M. Barlaud, "Mutual information-based context quantization". *Signal Processing: Image Communication (Elsevier Science)*, vol. 25 (1), pp. 64–74, Jan. 2010. *Cited in Sec.* 2.2.2

[CBD02] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ad-hoc network research". *Wireless Communications and Mobile Computing (John Wiley & Sons)*, vol. 2 (5), pp. 483–502, Aug. 2002. *Cited in Sec.* 3.5.1

[CBM08] R. Cunningham, B. Biskupski, and R. Meier, "Managing peer-to-peer live streaming applications", in *Proceedings of Springer-Verlag International Conference on Distributed Applications and Interoperable Systems*, Oslo, Norway, Jun. 2008. *Cited in Sec.* 5.2

[CCM08] O. Campana, R. Contiero, and G. Mian, "An H.264/AVC video coder based on a multiple description scalar quantizer". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18 (2), pp. 268–272, Feb. 2008. *Cited in Sec.* 2.1.4

[CFP07] M. Caleffi, G. Ferraiuolo, and L. Paura, "Augmented tree-based routing protocol for scalable ad-hoc networks", in *Proceedings of IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, Pisa, Italy, Oct. 2007. *Cited in Sec.* 3.2

[CGG+78] J. Candy, D. Gloge, D. Goodman, W. Hubbard, and K. Ogawa, "Protection by diversity at reduced quality", Tech. Rep., Bell Labs Memo for Record (not archived), Jun. 1978. *Cited in Sec.* 2.1.2

[CGPP08] O. Crave, C. Guillemot, and B. Pesquet-Popescu, "Multiple description source coding with side information", in *Proceedings of European Signal Processing Conference*, Lausanne, Switzerland, Aug. 2008. *Cited in Sec.* 2.1.4

[CGPPT08] O. Crave, C. Guillemot, B. Pesquet-Popescu, and C. Tillier, "Distributed temporal multiple description coding for robust video transmission". *EURASIP Journal on Wireless Communications and Networking*, vol. 2008, pp. 1–12, Jan. 2008. *Cited in Sec.* 2.1.4

[CGT+10] P. Cataldi, M. Grangetto, T. Tillo, E. Magli, and G. Olmo, "Sliding-window raptor codes for efficient scalable wireless video broadcasting with unequal loss protection". *IEEE Transactions on Image Processing*, vol. 19 (6), pp. 1491–1503, Jun. 2010. *Cited in Sec.* 3.3

[Chi95] L. Chiariglione, "The development of an integrated audiovisual coding standard: MPEG". *Proceedings of the IEEE*, vol. 83 (2), pp. 151–157, Feb. 1995, Invited Paper. *Cited in Sec.* 1.3.1

[CLRS09] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms (3rd edition)*, McGraw-Hill, 2009. *Cited in Sec.* 5.1.3

[CM04] L. Cappellari and G. Mian, "Analysis of joint predictive-transform coding for still image compression". *Signal Processing (Elsevier Science)*, vol. 84 (11), pp. 2097–2114, Nov. 2004. *Cited in Sec.* 1.2.2

[CM06] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media". *IEEE Transactions on Multimedia*, vol. 8 (2), pp. 390–404, Apr. 2006. *Cited in Sec.* 3.1

[CMW99] P. Chou, S. Mehrotra, and A. Wang, "Multiple description decoding of overcomplete expansions using projections onto convex sets", in *Proceedings of Data Compression Conference*, Snowbird, UT, USA, Mar. 1999. *Cited in Sec.* 2.1.3

[Coh03] B. Cohen, "Incentives build robustness in BitTorrent", in *Proceedings of ACM/SigComm Workshop on Economics of Peer-to-Peer Systems*, Berkley, CA, USA, Jun. 2003. *Cited in Sec.* A.2

[CPPG10] O. Crave, B. Pesquet-Popescu, and C. Guillemot, "Robust video coding based on multiple description scalar quantization with side information". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20 (6), pp. 769 –779, Jun. 2010. *Cited in Sec.* 2.1.4

[CS82] D. Comer and D. Stevens, *Internetworking with TCP/IP*, Prentice-Hall, 1982. *Cited in Sec.* 5.1

[CT91] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991. *Cited in Sec.* 1.2.4, 1.2.5

[CWJ03] P. Chou, Y. Wu, and K. Jain, "Practical network coding", in *Proceedings of Allerton Conference on Communication Control and Computing*, Monticello, IL, USA, Oct. 2003. *Cited in Sec.* 5.1.4, 5.3.1, 5.4.1

[CWKS97] B. Crow, I. Widjaja, L. Kim, and P. Sakai, "IEEE 802.11 wireless local area networks". *IEEE Communications Magazine*, vol. 35 (9), pp. 116–126, Sep. 1997. *Cited in Sec.* 4.2

[DB97] B. Das and V. Bharghavan, "Routing in ad-hoc networks using minimum connected dominating sets", in *Proceedings of IEEE International Conference on Communications*, Montréal, QC, Canada, Jun. 1997. *Cited in Sec.* 3.4.2

[DEH+05] S. Deb, M. Effros, T. Ho, D. Karger, R. Koetter, D. Lun, M. Médard, and N. Ratnakar, "Network coding for wireless applications: a brief tutorial", in *Proceedings of IEEE International Workshop on Wireless Ad-hoc Networks*, London, England, UK, May 2005, Invited Paper. *Cited in Sec.* 5.1

[DLL+00] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture". *IEEE Network*, vol. 14 (1), pp. 78–88, Jan. 2000. *Cited in Sec.* A.1

[DPT08] A. De Paola and A. Tamburo, "Reputation management in distributed systems", in *Proceedings of IEEE International Symposium on Communications, Control and Signal Processing*, San Ġiljan, Malta, Mar. 2008. *Cited in Sec.* A.3.2

[DRIK+10] M. Di Renzo, L. Iwaza, M. Kieffer, P. Duhamel, and K. Al Agha, "Robust wireless network coding — An overview", in *Proceedings of ICST International Conference on Mobile Lightweight Wireless Systems*, Barcelona, Spain, May 2010. *Cited in Sec.* 5.2

[DS52] R. Duffin and A. Schaeffer, "A class of non-harmonic Fourier series". *Transactions of the AMS*, vol. 72, pp. 341–366, 1952. *Cited in Sec.* 2.1.3

[DSK99] S. Dogan, A. Sadka, and A. Kondoz, "Efficient MPEG-4/H.263 video transcoder for interoperability of heterogeneous multimedia networks". *IEEE Electronics Letters*, vol. 35 (11), pp. 863–864, May 1999. *Cited in Sec.* 1.3.1

[EFLBS07] A. El Fawal, J. Le Boudec, and K. Salamatian, "Multi-hop broadcast from theory to reality: practical design for ad-hoc networks", in *Proceedings of IEEE/ACM International Conference on Autonomic Computing and Communication Systems*, Rome, Italy, Oct. 2007. *Cited in Sec.* 3.4.1

[EGC82] A. El Gamal and T. Cover, "Achievable rates for multiple descriptions". *IEEE Transactions on Information Theory*, vol. 28 (6), pp. 851–857, Nov. 1982. *Cited in Sec.* 2.1.1

[FAW01] L. Feeney, B. Ahlgren, and A. Westerlund, "Spontaneous networking: an application oriented approach to ad-hoc networking". *IEEE Communications Magazine*, vol. 39 (6), pp. 176–181, Jun. 2001. *Cited in Sec.* 3.2

[FF56] L. Ford and D. Fulkerson, "Maximal flow through a network". *Canadian Journal of Mathematics*, vol. 8, pp. 399–404, 1956. *Cited in Sec.* 5.1.2

[FFLT05] N. Franchi, M. Fumagalli, R. Lancini, and S. Tubaro, "Multiple description video coding for scalable and robust transmission over IP". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15 (3), pp. 321–334, Mar. 2005. *Cited in Sec.* 2.1.2, 2.2

[FJL00] M. Frodigh, P. Johansson, and P. Larsson, "Wireless ad-hoc networking: the art of networking without a network". *Ericsson Review*, vol. 4, pp. 248–263, 2000. *Cited in Sec.* 3.2

[FKM+07] C. Fragouli, D. Katabi, A. Markopoulou, M. Medard, and H. Rahul, "Wireless network coding: opportunities & challenges", in *Proceedings of IEEE Military Communications Conference*, Orlando, FL, USA, Oct. 2007. *Cited in Sec.* 5.2

[FLB10] R. Fracchia and C. Lamy-Bergot, "P2ProxyLite: effective video streaming in wireless ad-hoc networks", in *Proceedings of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Istanbul, Turkey, Sep. 2010. *Cited in Sec.* 3

[FPP07] J. Fowler and B. Pesquet-Popescu, "Wavelets in source coding, communications, and networks: an overview". *EURASIP Journal of Image and Video Processing*, vol. 1, p. 27 pp., Jan. 2007, Article ID 60539. *Cited in Sec.* 1.2.7

[FWLB06] C. Fragouli, J. Widmer, and J.-Y. Le Boudec, "A network coding approach to energy efficient broadcasting: from theory to practice", in *Proceedings of IEEE International Conference on Computer Communications*, Barcelona, Spain, Apr. 2006. *Cited in Sec.* 5.2

[FWLB08] ———, "Efficient broadcasting using network coding". *IEEE/ACM Transactions on Networking*, vol. 16 (2), pp. 450–463, Apr. 2008. *Cited in Sec.* 5.2

[GARRM05] B. Girod, A. Aaron, S. Rane, and D. Rebollo-Monedero, "Distributed video coding". *Proceedings of the IEEE*, vol. 93 (1), pp. 71–83, Jan. 2005. *Cited in Sec.* 2.2.1

[GB81] E. Gafni and D. Bertsekas, "Distributed algorithms for generating loop-free routes in networks with frequently changing topology". *IEEE Transactions on Communications*, vol. 29 (1), pp. 11–18, Jan. 1981. *Cited in Sec.* 3.2

[GC11] C. Greco and M. Cagnazzo, "A cross-layer protocol for cooperative content delivery over mobile ad-hoc networks". *Inderscience International Journal of Communication Networks and Distributed Systems*, vol. 7 (1–2), pp. 49–63, Jun. 2011. *Cited in Sec.* 3.4, 4.3.2, 4.4.1, 5.4.1

[GCPP10] C. Greco, M. Cagnazzo, and B. Pesquet-Popescu, "H.264-based multiple description coding using motion compensated temporal interpolation", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Saint-Malo, France, Oct. 2010. *Cited in Sec.* 2.2.2, 4.4.3

[GCPP11] ———, "ABCD : un protocole cross-layer pour la diffusion vidéo dans des réseaux sans fil ad-hoc", in *Proceedings of the Groupe d'Études du Traitement du Signal et des Images*, Bordeaux, France, Sep. 2011. *Cited in Sec.* 3.4.2, 3.4.4

[GCPP12] ———, "Low-latency video streaming with congestion control in mobile ad-hoc networks". *IEEE Transactions on Multimedia*, vol. 14 (4), p. 14pp, 2012, to appear. *Cited in Sec.* 4.4.1

[GCPW02] N. Gogate, D.-M. Chung, S. Panwar, and Y. Wang, "Supporting image and video applications in a multi-hop radio environment using path diversity and multiple description coding". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12 (9), pp. 777–792, Sep. 2002. *Cited in Sec.* 3.3

[Ger79] A. Gersho, "The channel splitting problem and modulo-PCM coding.", Tech. Rep., Bell Labs Memo for Record (not archived), Oct. 1979. *Cited in Sec.* 2.1.2

[Ger05] M. Gerla, "From battlefields to urban grids: new research challenges in ad-hoc wireless networks". *Elsevier Journal on Pervasive and Mobile Computing*, vol. 1 (1), pp. 77–93, 2005. *Cited in Sec.* 3.2

[GGP01] T. Guionnet, C. Guillemot, and S. Pateux, "Embedded multiple description coding for progressive image transmission over unreliable channels", in *Proceedings of IEEE Internantional Conference on Image Processing*, Thessaloniki, Greece, Oct. 2001. *Cited in Sec.* 2.1.4

[GK98] V. Goyal and J. Kovačević, "Optimal multiple description transform coding of Gaussian vectors", in *Proceedings of Data Compression Conference*, Snowbird, UT, USA, Mar. 1998. *Cited in Sec.* 2.1, 2.1.3

[GK01] V. K. Goyal and J. Kovačević, "Generalized multiple description coding with correlating transforms". *IEEE Transactions on Information Theory*, vol. 47 (6), pp. 2199–2224, Sep. 2001. *Cited in Sec.* 2.1.3

[GKAV98] V. Goyal, J. Kovačević, R. Arean, and M. Vetterli, "Multiple description transform coding of images", in *Proceedings of IEEE Internantional Conference on Image Processing*, Chicago, IL, USA, Oct. 1998. *Cited in Sec.* 2.1.3

[GKV99] V. Goyal, J. Kovačević, and M. Vetterli, "Quantized frame expansions as source channel codes for erasure channels", in *Proceedings of Data Compression Conference*, Snowbird, UT, USA, Mar. 1999. *Cited in Sec.* 2.1.3

[GMR06] C. Gkantsidis, J. Miller, and P. Rodriguez, "Anatomy of a P2P content distribution system with network coding", in *Proceedings of International Workshop on Peer-to-Peer Systems*, 2006. *Cited in Sec.* 5.2

[GNCPP12] C. Greco, I. D. Nemoianu, M. Cagnazzo, and B. Pesquet-Popescu, "A network coding scheduling for multiple description video streaming over wireless networks", in *Proceedings of European Signal Processing Conference*, Bucharest, Romania, Aug. 2012. *Cited in Sec.* 5.4.2

[Goy01] V. K. Goyal, "Multiple description coding: compression meets the network". *IEEE Signal Processing Magazine*, vol. 18 (5), pp. 74–93, Sep. 2001. *Cited in Sec.* 2.1, 2.1.1, 2.1.4, 4.3

[GPCPP11] C. Greco, G. Petrazzuoli, M. Cagnazzo, and B. Pesquet-Popescu, "An MDC-based video streaming architecture for mobile networks", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Hangzhou, PRC, Oct. 2011. *Cited in Sec.* 2.2.2, 2.2.3, 2.12, 2.13, 2.14, 3.5.6

[GPP11] R. Gaetano and B. Pesquet-Popescu, "OpenCL implementation of motion estimation for cloud video processing", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Hangzhou, PRC, Oct. 2011. *Cited in Sec.* 5.5

[GPT+07] C. Guillemot, F. Pereira, L. Torres, T. Ebrahimi, R. Leonardi, and J. Ostermann, "Distributed monoview and multiview video coding". *IEEE Signal Processing Magazine*, vol. 24 (5), pp. 67–76, Sep. 2007. *Cited in Sec.* 2.2.2

[GR05] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution", in *Proceedings of IEEE International Conference on Computer Communications*, Mar. 2005. *Cited in Sec.* 5.2

[GV02] M. Gastpar and M. Vetterli, "On the capacity of wireless networks: the relay case", in *Proceedings of IEEE International Conference on Computer Communications*, Jun. 2002. *Cited in Sec.* 5.1.2

[GVK98] V. Goyal, M. Vetterli, and J. Kovačević, "Multiple description transform coding: robustness to erasures using tight frame expansions", in *Proceedings of IEEE International Symposium on Information Theory*, Cambridge, MA, USA, Aug. 1998. *Cited in Sec.* 2.1.3

[GVM+07] A. I. Gavrilescu, F. Verdicchio, A. Munteanu, I. Moerman, J. Cornelis, and P. Schelkens, "Scalable multiple description image coding based on embedded quantization". *EURASIP Journal on Image and Video Processing*, vol. 1, pp. 20–30, Jan. 2007. *Cited in Sec.* 2.1.4

[GVT98] V. Goyal, M. Vetterli, and N. Thao, "Quantized overcomplete expansions in R^N: analysis, synthesis, and algorithms". *IEEE Transactions on Information Theory*, vol. 44 (1), pp. 16–31, Jan. 1998. *Cited in Sec.* 2.1.3

[H2690] "H.261: Video codec for audiovisual services at $p\times 64$ kbits", ITU-T Recommendation, 1990. *Cited in Sec.* 1.3.1

[H2695] "H.262: Generic coding of moving pictures and associated audio information", ITU-T Recommendation, 1995. *Cited in Sec.* 1.3.1

[H2696] "H.263: Video coding for low bit rate communication", ITU-T Recommendation, 1996. *Cited in Sec.* 1.3.1

[H2603] "H.264: Advanced Video Coding for generic audiovisual services", ITU-T Recommendation, 2003. *Cited in Sec.* 1.3.2

[HKM+03] T. Ho, R. Koetter, M. Médard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting", in *Proceedings of IEEE International Symposium on Information Theory*, Kanagawa, Japan, Jun. 2003. *Cited in Sec.* 5.1.4

[HMK+06] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast". *IEEE Transactions on Information Theory*, vol. 52 (10), pp. 4413–4430, Oct. 2006. *Cited in Sec.* 5.1.4

[HMS+03] T. Ho, M. Médard, J. Shi, M. Effros, and D. Karger, "On randomized network coding", in *Proceedings of IEEE International Symposium on Information Theory*, Kanagawa, Japan, Jun. 2003. *Cited in Sec.* 5.1.4

[HOTV99] C. Ho, K. Obraczka, G. Tsudik, and K. Viswanath, "Flooding for reliable multicast in multi-hop ad-hoc networks", in *Proceedings of ACM Iternational Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, Seattle, WA, USA, Aug. 1999. *Cited in Sec.* 3.2, 3.5.6

[HT09] L. Hanzo and R. Tafazolli, "Admission control schemes for 802.11-based multi-hop mobile ad-hoc networks: a survey". *IEEE Communications Surveys & Tutorials*, vol. 11 (4), pp. 78–108, Oct. 2009. *Cited in Sec.* 3.2

[Huf52] D. Huffman, "A method for the construction of minimum-redundancy codes". *Proceedings of the IRE*, vol. 40 (9), pp. 1098–1101, Sep. 1952. *Cited in Sec.* 1.2.5

[IKLAA11] L. Iwaza, M. Kieffer, L. Liberti, and K. Al Agha, "Joint decoding of multiple-description network-coded data", in *Proceedings of IEEE International Symposium on Network Coding*, Beijing, PRC, Jul. 2011. *Cited in Sec.* 5.3

[JCWF07] D. Jurca, J. Chakareski, J.-P. Wagner, and P. Frossard, "Enabling adaptive video streaming in P2P systems". *IEEE Communications Magazine*, vol. 45 (6), pp. 108–114, Jun. 2007. *Cited in Sec.* 1, A.1

[JHMJ01] J. Jetcheva, Y. Hu, D. Maltz, and D. Johnson, "A simple protocol for multicast and broadcast in mobile ad-hoc networks", IETF Internet Draft, Jul. 2001. *Cited in Sec.* 3.2

[JMB01] D. Johnson, D. Maltz, and J. Broch, "DSR: the dynamic source routing protocol for multi-hop wireless ad-hoc networks". *Ad-hoc Networking (ed. T. Imielinski and H. Korth), Kluwer Academic Publishers*, vol. 5, pp. 139–172, 2001. *Cited in Sec.* 3.2

[JMC+01] P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized link state routing protocol for ad-hoc networks", in *Proceedings of IEEE International Multitopic Conference*, Lahore, Pakistan, Dec. 2001. *Cited in Sec.* 3.2

[JO99] W. Jiang and A. Ortega, "Multiple description coding via polyphase transform and selective quantization", in *Proceedings of SPIE International Symposium on Visual Communications and Image Processing*, San Jose, CA, USA, Jan. 1999. *Cited in Sec.* 2.2.2

[JSC+05] S. Jaggi, P. Sanders, P. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction". *IEEE Transactions on Information Theory*, vol. 51 (6), pp. 1973–1982, Jun. 2005. *Cited in Sec.* 5.1.3

[Kar90] P. Karn, "MACA: a new channel access method for packet radio", in *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, London, ON, Canada, Sep. 1990. *Cited in Sec.* 3.4.1

[KDG02] J. Kovačević, P. Dragotti, and V. Goyal, "Filter bank frame expansions with erasures". *IEEE Transactions on Information Theory*, vol. 48 (6), pp. 1439–1450, Jun. 2002, Invited Paper. *Cited in Sec.* 2.1.3

[KK05] V. Kawadia and P. Kumar, "A cautionary perspective on cross-layer design". *IEEE Wireless Communications Magazine*, vol. 12 (1), pp. 3–11, Feb. 2005. *Cited in Sec.* 3

[KK08] R. Koetter and F. Kschischang, "Coding for errors and erasures in random network coding". *IEEE Transactions on Information Theory*, vol. 54 (8), pp. 3579–3591, Aug. 2008. *Cited in Sec.* 5.1

[KKH+05] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard, "The importance of being opportunistic: practical network coding for wireless environments", in *Proceedings of Allerton Conference on Communication Control and Computing*, Monticello, IL, USA, Sep. 2005, Invited Paper. *Cited in Sec.* 5.2

[KM03] R. Koetter and M. Médard, "An algebraic approach to network coding". *IEEE/ACM Transactions on Networking*, vol. 11 (5), pp. 782–795, Oct. 2003. *Cited in Sec.* 5.1.3

[KR04] J. F. Kurose and K. W. Ross, *Computer networking: a top-down approach featuring the Internet*, Pearson Education, 2004. *Cited in Sec.* 5.1

[KRH+06] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless network coding". *ACM SigComm Computer Communication Review*, vol. 36 (4), pp. 243–254, Aug. 2006. *Cited in Sec.* 5.2

[KRH+08] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "XORs in the air: practical wireless network coding". *IEEE/ACM Transactions on Networking*, vol. 16 (3), pp. 497–510, Jun. 2008. *Cited in Sec.* 5.2

[KS78] L. Kleinrock and J. Silvester, "Optimum transmission radii for packet radio networks or why six is a magic number", in *Proceedings of National Telecommunications Conference*, Birmingham, AL, USA, Dec. 1978. *Cited in Sec.* 3.4.1

[KSG04] M. Kalman, E. Steinbach, and B. Girod, "Adaptive media playout for low-delay video streaming over error-prone channels". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14 (6), pp. 841–851, Jun. 2004. *Cited in Sec.* 3.1

[KSGM03] S. Kamvar, M. Schlosser, and H. Garcia-Molina, "The Eigentrust algorithm for reputation management in P2P networks", in *Proceedings of ACM International Conference on World Wide Web*, Budapest, Hungary, May 2003. *Cited in Sec.* A.3.2

[KVMAA09] G. Karbaschi, A. C. Viana, S. Martin, and K. Al Agha, "On using network coding in multi hop wireless networks", in *Proceedings of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Tokyo, Japan, Sep. 2009. *Cited in Sec.* 5.2

[KWZ01] B. Krishnamurthy, C. Wills, and Y. Zhang, "On the use and performance of content distribution networks", in *Proceedings of ACM SigComm Workshop on Internet Measurement*, San Francisco, CA, USA, Nov. 2001. *Cited in Sec.* A.1

[LBFM+11] C. Lamy-Bergot, R. Fracchia, M. Mazzotti, S. Moretti, E. Piri, T. Sutinen, J. Zuo, J. Vehkaperä, G. Feher, G. Jeney, G. Panza, and P. Amon, "Optimisation of multimedia over wireless IP links via X-layer design: an end-to-end transmission chain simulator". *Multimedia Tools and Applications (Springer Netherlands)*, vol. 55 (2), pp. 261–288, Nov. 2011. *Cited in Sec.* 3

[LBFV+09] C. Lamy-Bergot, R. Fracchia, J. Vehkaperä, T. Sutinen, E. Piri, M. Mazzotti, G. Panza, G. Feher, G. Jeney, and P. Amon, "Optimisation of multimedia over wireless IP links via X-layer design: an end-to-end transmission chain simulator", in *Proceedings of ICST International Mobile Multimedia Communications Conference*, London, England, UK, Sep. 2009. *Cited in Sec.* 3

[LG00] E. Lam and J. Goodman, "A mathematical analysis of the DCT coefficient distributions for images". *IEEE Transactions on Image Processing*, vol. 9 (10), pp. 1661–1666, Oct. 2000. *Cited in Sec.* 1.2.3

[Li01] W. Li, "Overview of fine granularity scalability in MPEG-4 video standard". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11 (3), pp. 301–317, Mar. 2001. *Cited in Sec.* 1.3.1

[Llo82] S. Lloyd, "Least squares quantization in PCM". *IEEE Transactions on Information Theory*, vol. 28 (2), pp. 129–137, Mar. 1982. *Cited in Sec.* 1.2.4

[LPFA00] W. S. Lee, M. Pickering, M. Frater, and J. Arnold, "A robust codec for transmission of very low bit-rate video over channels with bursty errors". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10 (8), pp. 1403–1412, Dec. 2000. *Cited in Sec.* 2.1.1

[LR82] G. Langdon and J. Rissanen, "A simple general binary source code". *IEEE Transactions on Information Theory*, vol. 28 (5), pp. 800–803, Sep. 1982, correspondence. *Cited in Sec.* 1.2.5

[LSM07] C. Lochert, B. Scheuermann, and M. Mauve, "A survey on congestion control for mobile ad-hoc networks". *Wireless Communications and Mobile Computing (John Wiley & Sons)*, vol. 7 (5), pp. 655–676, Jun. 2007. *Cited in Sec.* 3.5.1

[Lub02] M. Luby, "LT codes", in *Proceedings of IEEE Symposium on Foundations of Computer Science*, Vancouver, BC, Canada, Nov. 2002. *Cited in Sec.* 3.3

[LWMP02] S. Lin, Y. Wang, S. Mao, and S. Panwar, "Video transport over ad-hoc networks using multiple paths", in *Proceedings of IEEE International Symposium on Circuits and Systems*, Scottsdale, AZ, USA, May 2002. *Cited in Sec.* 3.3

[LWP+07] M. Lu, J. Wu, K. Peng, P. Huang, J. Yao, and H. Chen, "Design and evaluation of a P2P IPTV system for heterogeneous networks". *IEEE Transactions on Multimedia*, vol. 9 (8), pp. 1568–1579, Dec. 2007. *Cited in Sec.* 3.1

[LYC03] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding". *IEEE Transactions on Information Theory*, vol. 49 (2), pp. 371–381, Feb. 2003. *Cited in Sec.* 5.1.3

[LZ07] M. Liu and C. Zhu, "Multiple description video coding using hierarchical B pictures", in *Proceedings of IEEE International Conference on Multimedia and Expo*, Beijing, PRC, Jun. 2007. *Cited in Sec.* 2.2.2, 2.2.3, 5.5

[MAPPF08] T. Maugey, T. André, B. Pesquet-Popescu, and J. Farah, "Analysis of error propagation due to frame losses in a distributed video coding system", in *Proceedings of European Signal Processing Conference*, Lausanne, Switzerland, Aug. 2008. *Cited in Sec.* 4.3

[MBAAP03] A. Munaretto, H. Badis, K. Al Agha, and G. Pujolle, "QoS for ad-hoc networking based on multiple metrics: bandwidth and delay", in *Proceedings of IEEE International Conference on Mobile and Wireless Communications Networks*, Singapore, Oct. 2003. *Cited in Sec.* 3.2

[MCH+07] S. Mao, X. Cheng, Y. T. Hou, H. D. Sherali, and J. H. Reed, "On joint routing and server selection for MD video streaming in ad-hoc networks". *IEEE Transactions on Wireless Communications*, vol. 6 (1), pp. 338–347, Jan. 2007. *Cited in Sec.* 3.1

[MES07] J. D. Mol, D. H. P. Epema, and H. J. Sips, "The Orchard algorithm: building multicast trees for P2P video multicasting without free-riding". *IEEE Transactions on Multimedia*, vol. 9 (8), pp. 1593–1604, Dec. 2007. *Cited in Sec.* A.2

[MF09] E. Magli and P. Frossard, "An overview of network coding for multimedia streaming", in *Proceedings of IEEE International Conference on Multimedia and Expo*, New York, NY, USA, Jul. 2009. *Cited in Sec.* 5.2

[MGPP+12] E. G. Mora, C. Greco, B. Pesquet-Popescu, M. Cagnazzo, and J. Farah, "Cedar: an optimized network-aware solution for P2P video multicast", in *Proceedings of IEEE International Conference on Telecommunications*, Jounieh, Lebanon, Apr. 2012. *Cited in Sec.* 5.2

[Mil80] S. Miller, "Fail-safe transmission without standby facilities", Tech. Rep. TM80-136-2, Bell Labs, Aug. 1980. *Cited in Sec.* 2.1.2

[MKK01] M. K. Marina, G. D. Kondylis, and U. C. Kozat, "RBRP: a robust broadcast reservation protocol for mobile ad-hoc networks", in *Proceedings of IEEE International Conference on Communications*, Beijing, PRC, May 2001. *Cited in Sec.* 3.4.1, 4.3.2

[MLP+03] S. Mao, S. Lin, S. Panwar, Y. Wang, and E. Celebi, "Video transport over ad-hoc networks: multistream coding with multipath transport". *IEEE Journal on Selected Areas in Communications*, vol. 21 (10), pp. 1721–1737, Dec. 2003. *Cited in Sec.* 3.1, 3.3

[MOKM08] A. Mohammed, M. Ould-Khaoua, and L. Mackenzie, "Improvement to efficient counter-based broadcast scheme through random assessment delay adaptation for MANETs", in *Proceedings of UKSim European Symposium on Computer Modeling and Simulation*, Liverpool, England, UK, Sep. 2008. *Cited in Sec.* 3.2

[MPE93] "MPEG-1: Coding of moving pictures and associated audio for digital storage media at up to about $1,5$ Mbit/s — Part 2: Video", ISO/IEC Standard 11172-2, 1993. *Cited in Sec.* 1.3.1

[MPE95] "MPEG-2: Generic coding of moving pictures and associated audio information — Part 2: Visual", ISO/IEC Standard 13818-2, 1995. *Cited in Sec.* 1.3.1

[MPE99] "MPEG-4: Coding of audio-visual objects — Part 2: Visual", ISO/IEC Standard 14496-2, 1999. *Cited in Sec.* 1.3.1

[MPE03] "MPEG-4: Coding of audio-visual objects — Part 10: Advanced Video Coding", ISO/IEC Standard 14496-10, 2003. *Cited in Sec.* 1.3.2

[MPG85] H. Musmann, P. Pirsch, and H.-J. Grallert, "Advances in picture coding". *Proceedings of the IEEE*, vol. 73 (4), pp. 523–548, Apr. 1985. *Cited in Sec.* 2.2

[MPP08] T. Maugey and B. Pesquet-Popescu, "Side information estimation and new symmetric schemes for multi-view distributed video coding". *Elsevier Journal of Visual Communication and Image Representation*, vol. 19 (8), pp. 589–599, Sep. 2008. *Cited in Sec.* 4.3

[MSK+02] A. Majumda, D. Sachs, I. Kozintsev, K. Ramchandran, and M. Yeung, "Multicast and unicast real-time video streaming over wireless LANs". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12 (6), pp. 524–534, Jun. 2002. *Cited in Sec.* 3.3

[NGCPP12] I. D. Nemoianu, C. Greco, M. Cagnazzo, and B. Pesquet-Popescu, "A framework for joint multiple description coding and network coding over wireless ad-hoc networks", in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Kyoto, Japan, Mar. 2012. *Cited in Sec.* 5.4.1

[NNcC07] K. Nguyen, T. Nguyen, and S. ching Cheung, "Peer-to-peer streaming with hierarchical network coding", in *Proceedings of IEEE International Conference on Multimedia and Expo*, Beijing, PRC, Jul. 2007. *Cited in Sec.* 5.2

[NNGS10] G. Nath-Nayak and S. Ghosh Samaddar, "Different flavours of man-in-the-middle attack, consequences and feasible solutions", in *Proceedings of IEEE International Conference on Computer Science and Information Technology*, Chengdu, PRC, Jul. 2010. *Cited in Sec.* 3.4.1

[ns2] "The Network Simulator — ns-2", Website. *Cited in Sec.* 3.5.1, 4.4.1

[NTCS99] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The broadcast storm problem in a mobile ad-hoc network", in *Proceedings of ACM International Conference on Mobile Computing and Networking*, Seattle, WA, USA, Aug. 1999. *Cited in Sec.* 3.4

[NTNB09] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast using network coding". *IEEE Transactions on Vehicular Technology*, vol. 58 (2), pp. 914–925, Feb. 2009. *Cited in Sec.* 5.2

[OBL+04] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with H.264/AVC: tools, performance, and complexity". *IEEE Circuits and Systems Magazine*, vol. 4 (1), pp. 7–28, Apr. 2004. *Cited in Sec.* 1.1, 1.3.2

[Ohm05] J.-R. Ohm, "Advances in scalable video coding". *Proceedings of the IEEE*, vol. 93 (1), pp. 42–56, Jan. 2005, Invited Paper. *Cited in Sec.* 1.3.1

[Oku95] S. Okubo, "Reference model methodology — A tool for the collaborative creation of video coding standards". *Proceedings of the IEEE*, vol. 83 (2), pp. 139–150, Feb. 1995. *Cited in Sec.* 1.3.1

[OWVR97] M. Orchard, Y. Wang, V. Vaishampayan, and A. Reibman, "Redundancy rate-distortion analysis of multiple description coding using pairwise correlating transforms", in *Proceedings of IEEE Internantional Conference on Image Processing*, Washington, DC, USA, Oct. 1997. *Cited in Sec.* 2.1.3

[Oza80] L. Ozarow, "On a source-coding problem with two channels and three receivers". *Bell Systems Technical Journal*, vol. 59 (10), pp. 1909–1921, May 1980. *Cited in Sec.* 2.1.1

[Par62] E. Parzen, "On estimation of a probability density function and mode". *Annals of Mathematical Statistics*, vol. 33 (3), pp. 1065–1076, 1962. *Cited in Sec.* 3.5.2, 4.4.3

[PB94] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers". *ACM SigComm Computer Communication Review*, vol. 24, pp. 234–244, Oct. 1994. *Cited in Sec.* 3.2

[PC97] V. Park and M. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks", in *Proceedings of IEEE International Conference on Computer Communications*, Kobe, Japan, Apr. 1997. *Cited in Sec.* 3.2

[PCPP10] G. Petrazzuoli, M. Cagnazzo, and B. Pesquet-Popescu, "High order motion interpolation for side information improvement in DVC", in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, Dallas, TX, USA, Mar. 2010. *Cited in Sec.* 2.2.1

[PD11] P. Pathak and R. Dutta, "A survey of network design problems and joint design approaches in wireless mesh networks". *IEEE Communications Surveys & Tutorials*, vol. 13 (3), pp. 396–428, Jul. 2011. *Cited in Sec.* 3

[PL01] W. Peng and X. Lu, "AHBP: an efficient broadcast protocol for mobile ad-hoc networks". *Journal of Computer Science and Technology*, vol. 16, pp. 114–125, Mar. 2001. *Cited in Sec.* 3.2, 3.4

[PR99] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing", in *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, New Orleans, LA, USA, Feb. 1999. *Cited in Sec.* 3.2

[Pro06] "Proxim ORiNOCO 11b/g client PC card specifications", 2006. *Cited in Sec.* 3.5.1

[PS01] J. Proakis and M. Salehi, *Digital communications*, McGraw-Hill, 2001. *Cited in Sec.* 2.1.4

[PS02] V. N. Padmanabhan and K. Sripanidkulchai, "The case for cooperative networking", in *Proceedings of International Workshop on Peer-to-Peer Systems*, Cambridge, MA, USA, Mar. 2002. *Cited in Sec.* 5.2

[PVdS10a] H. Park and M. Van der Schaar, "Evolution of resource reciprocation strategies in P2P networks". *IEEE Transactions on Signal Processing*, vol. 58 (3), pp. 1205–1218, Mar. 2010. *Cited in Sec.* A.3.2

[PVdS10b] J. Park and M. Van der Schaar, "A game theoretic analysis of incentives in content production and sharing over peer-to-peer networks". *IEEE Journal of Selected Topics in Signal Processing*, vol. 4 (4), pp. 704–717, Aug. 2010. *Cited in Sec.* A.3.1, A.3.2

[PWCS02] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking", in *Proceedings of ACM SigComm International Workshop on Network and Operating Systems Support for Digital Audio and Video*, Miami Beach, FL, USA, May 2002. *Cited in Sec.* 5.2, A.1

[QoS94] "E.800: Definitions of terms related to quality of service", ITU-T Recommendation, 1994. *Cited in Sec.* 3.5.4

[QS04] D. Qiu and R. Srikant, "Modeling and performance analysis of BitTorrent-like peer-to-peer networks", in *Proceedings of ACM SigComm Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Portland, OR, USA, Aug. 2004. *Cited in Sec.* 5.2

[Reu80] D. O. Reudink, "The channel splitting problem with interpolative coders", Tech. Rep., Bell Labs, Oct. 1980, tM80-134-1. *Cited in Sec.* 2.1.4

[RF07] I. Radulovic and P. Frossard, "Multiple description image coding with redundant expansions and optimal quantization", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Crete, Greece, Oct. 2007. *Cited in Sec.* 2.1.3

[RFW+10] I. Radulovic, P. Frossard, Y.-K. Wang, M. Hannuksela, and A. Hallapuro, "Multiple description video coding with H.264/AVC redundant pictures". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20 (1), pp. 144–148, Jan. 2010. *Cited in Sec.* 2.2.2

[RHKS02] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection", in *Proceedings of IEEE International Conference on Computer Communications*, New York, NY, USA, Jun. 2002. *Cited in Sec.* A.3.3

[Ric03] I. Richardson, *H.264 and MPEG-4 video compression*, Wiley Online Library, 2003. *Cited in Sec.* 1.1, 1.2.2, 1.3.1

[Ric10] ———, *The H.264 Advenced Video Compression standards*, Wiley Online Library, 2010. *Cited in Sec.* 1.1, 1.2.6, 1.3.2

[Rij96] K. Rijkse, "H.263: video coding for low-bitrate communication". *IEEE Communications Magazine*, vol. 34 (12), pp. 42–45, Dec. 1996. *Cited in Sec.* 1.3.1

[RJW+99] A. Reibman, H. Jafarkhani, Y. Wang, M. Orchard, and R. Puri, "Multiple description coding for video using motion compensated prediction", in *Proceedings of IEEE Internantional Conference on Image Processing*, Kobe, Japan, Oct. 1999. *Cited in Sec.* 2.1.1

[RM94] A. Rodriguez and K. Morse, "Evaluating video codecs". *IEEE Multimedia*, vol. 1 (3), pp. 25–33, Sep. 1994. *Cited in Sec.* 1.1

[RMSM01] E. Royer, P. Melliar-Smith, and L. Moser, "An analysis of the optimum node density for ad-hoc mobile networks", in *Proceedings of IEEE International Conference on Communications*, Helsinki, Finland, Jun. 2001. *Cited in Sec.* 3.6

[RT99] E. M. Royer and C.-K. Toh, "A review of current routing protocols for ad-hoc mobile wireless networks". *IEEE Personal Communications Magazine*, vol. 6 (2), pp. 46–55, Apr. 1999. *Cited in Sec.* 3.2

[RVdSC01] H. M. Radha, M. Van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP". *IEEE Transactions on Multimedia*, vol. 3 (1), pp. 53–68, Mar. 2001. *Cited in Sec.* 3.1

[RW10] A. Ramasubramonian and J. Woods, "Multiple description coding and practical network coding for video multicast". *IEEE Signal Processing Letters*, vol. 17 (3), pp. 265–268, Mar. 2010. *Cited in Sec.* 5.2

[RWW+07] I. Radulovic, Y. Wang, S. Wenger, A. Hallapuro, M. Hannuksela, and P. Frossard, "Multiple description H.264 video coding with redundant pictures", in *Proceedings of ACM International Workshop on Mobile Video*, Augsburg, Germany, Sep. 2007. *Cited in Sec.* 2.2.2

[SBG08] E. Setton, P. Baccichet, and B. Girod, "Peer-to-peer live multicast: a video perspective". *Proceedings of the IEEE*, vol. 96 (1), pp. 25–38, Jan. 2008, Invited Paper. *Cited in Sec.* 5.2

[Sch98] R. Schäfer, "MPEG-4: a multimedia compression standard for interactive applications and services". *IEEE Electronics & Communication Engineering Journal*, vol. 10 (6), pp. 253–262, Dec. 1998. *Cited in Sec.* 1.3.1

[SG04] E. Setton and B. Girod, "Congestion-distortion optimized scheduling of video over a bottleneck link", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Siena, Italy, Sep. 2004. *Cited in Sec.* 4.1

[SGF02] R. Schollmeier, I. Gruber, and M. Finkenzeller, "Routing in mobile ad-hoc and peer-to-peer networks: a comparison", in *Proceedings of IEEE International Conference on Peer-to-Peer Computing*, Linköping, Sweden, Sep. 2002. *Cited in Sec.* 3.2

[Sha48] C. E. Shannon, "A mathematical theory of communication". *Bell Systems Technical Journal*, vol. 27, pp. 379–423,623–656, Oct. 1948. *Cited in Sec.* 1.1

[Sha59] ———, "Coding theorems for a discrete source with a fidelity criterion", in *IRE National Convention Record*, vol. 4, 1959. *Cited in Sec.* 1.1

[Sho06] A. Shokrollahi, "Raptor codes". *IEEE Transactions on Information Theory*, vol. 52 (6), pp. 2551–2567, Jun. 2006. *Cited in Sec.* 3.3

[SHW03] T. Stockhammer, M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13 (7), pp. 657–673, Jul. 2003. *Cited in Sec.* 3.3

[Sik97a] T. Sikora, "The MPEG-4 video standard verification model". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 7 (1), pp. 19–31, Feb. 1997. *Cited in Sec.* 1.3.1

[Sik97b] ———, "MPEG digital video-coding standards". *IEEE Signal Processing Magazine*, vol. 14 (5), pp. 82–100, Sep. 1997. *Cited in Sec.* 1.3.1

[SM00] J. Sucec and I. Marsic, "An efficient distributed network-wide broadcast algorithm for mobile ad-hoc networks", Tech. Rep., Rutgers University, Sep. 2000. *Cited in Sec.* 3.4.1

[SM07] H. Seferoglu and A. Markopoulou, "Opportunistic network coding for video streaming over wireless", in *Proceedings of IEEE Packet Video Conference*, Lausanne, Switzerland, Nov. 2007. *Cited in Sec.* 5.2

[SMW06] H. Schwarz, D. Marpe, and T. Wiegand, "Analysis of hierarchical B pictures and MCTF", in *Proceedings of IEEE International Conference on Multimedia and Expo*, Toronto, ON, Canada, Jul. 2006. *Cited in Sec.* 1.1

[SMW07] ———, "Overview of the scalable video coding extension of the H.264/AVC standard". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17 (9), pp. 1103–1120, Sep. 2007, Invited Paper. *Cited in Sec.* 1.3.2

[SO10] G. Sullivan and J. Ohm, "Recent developments in standardization of high efficiency video coding (HEVC)", in *Proceedings of SPIE Conference on Applications of Digital Image Processing*, San Diego, CA, USA, Aug. 2010. *Cited in Sec.* 1.2.6

[SR05] D. Stutzbach and R. Rejaie, "Characterizing churn in peer-to-peer networks", Tech. Rep., University of Oregon, Jun. 2005. *Cited in Sec.* A.1

[SRL08] R. Santos, C. Rocha, B. Rezende, and A. Loureiro, "Characterizing the YouTube video-sharing community", White Paper, 2008. *Cited in Sec.* A.1

[STL04] G. Sullivan, P. Topiwala, and A. Luthra, "The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions", in *Proceedings of SPIE Conference on Applications of Digital Image Processing*, San Diego, CA, USA, Aug. 2004. *Cited in Sec.* 1.2.2

[Süh11] K. Sühring, "JM reference software release 17.0", Source Code, Jan. 2011. *Cited in Sec.* 2.2.3, 5.4.2

[SW73] D. Slepian and J. Wolf, "Noiseless coding of correlated information sources". *IEEE Transactions on Information Theory*, vol. 19 (4), pp. 471–480, Jul. 1973. *Cited in Sec.* 2.2.1

[SW98] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression". *IEEE Signal Processing Magazine*, vol. 15 (6), pp. 74–90, Nov. 1998. *Cited in Sec.* 1.1

[SW05] G. Sullivan and T. Wiegand, "Video compression — From concepts to the H.264/AVC standard". *Proceedings of the IEEE*, vol. 93 (1), pp. 18–31, Jan. 2005, Invited Paper. *Cited in Sec.* 1.1

[SYZ+05] E. Setton, T. Yoo, X. Zhu, A. Goldsmith, and B. Girod, "Cross-layer design of ad-hoc networks for real-time video streaming". *IEEE Transactions on Wireless Communications*, vol. 12 (4), pp. 59–65, Aug. 2005. *Cited in Sec.* 4.1

[Tan03] A. S. Tanenbaum, *Computer Networks (4th edition)*, Prentice Hall, 2003. *Cited in Sec.* 3.2, 5.1

[TCF11] N. Thomos, J. Chakareski, and P. Frossard, "Prioritized distributed video delivery with randomized network coding". *IEEE Transactions on Multimedia*, vol. 13 (4), pp. 776–787, Aug. 2011. *Cited in Sec.* 5.2, 5.3.2

[Tou98] J. Tourrilhes, "Robust broadcast: improving the reliability of broadcast transmissions on CSMA/CA", in *Proceedings of IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, Boston, MA, USA, Sep. 1998. *Cited in Sec.* 3.4.1

[TPP09] N. Tizon and B. Pesquet-Popescu, "Adaptive video streaming with a recursive distortion model", in *Proceedings of the Groupe d'Études du Traitement du Signal et des Images*, Dijon, France, Sep. 2009. *Cited in Sec.* 4.3

[TPPP07] C. Tillier, C. T. Petrişor, and B. Pesquet-Popescu, "A motion-compensated overcomplete temporal decomposition for multiple description scalable video coding". *EURASIP Journal on Image and Video Processing*, vol. 1, pp. 1–12, 2007. *Cited in Sec.* 2.1.3

[TPPVdS04] C. Tillier, B. Pesquet-Popescu, and M. Van der Schaar, "Multiple descriptions scalable video coding", in *Proceedings of European Signal Processing Conference*, Vienna, Austria, Sep. 2004. *Cited in Sec.* 2.1.3

[TvdSPP05] D. Turaga, M. van der Schaar, and B. Pesquet-Popescu, "Complexity scalable motion compensated wavelet video encoding". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15 (8), pp. 982–993, Aug. 2005. *Cited in Sec.* 1.2.7

[UAF+10] K. Ugur, K. Andersson, A. Fuldseth, G. Bjøntegaard, L. Endresen, J. Lainema, A. Hallapuro, J. Ridge, D. Rusanovskyy, C. Zhang, A. Norkin, C. Priddle, T. Rusert, J. Samuelsson, R. Sjoberg, and Z. Wu, "High performance, low complexity video coding and the emerging HEVC standard". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20 (12), pp. 1688–1697, Dec. 2010. *Cited in Sec.* 1.2.6

[Vai93] V. Vaishampayan, "Design of multiple description scalar quantizers". *IEEE Transactions on Information Theory*, vol. 39 (3), pp. 821–834, May 1993. *Cited in Sec.* 2.1.4

[VD94] V. Vaishampayan and J. Domaszewicz, "Design of entropy constrained multiple description scalar quantizers". *IEEE Transactions on Information Theory*, vol. 40 (1), pp. 245–250, Jan. 1994. *Cited in Sec.* 2.1.4

[VDJ+12] F. Verbist, N. Deligiannis, M. Jacobs, J. Barbarien, P. Schelkens, A. Munteanu, and J. Cornelis, "Probabilistic motion-compensated prediction in distributed video coding". *Multimedia Tools and Applications (Springer Netherlands)*, vol. 1, pp. 1–26, Feb. 2012, accepted for publication. *Cited in Sec.* 2.2.1

[VdSC07] M. Van der Schaar and P. Chou, *Multimedia over IP and wireless networks: compression, networking, and systems*, Academic Press, 2007. *Cited in Sec.* 1.2.7

[VJ99] V. Vaishampayan and S. John, "Balanced interframe multiple description video compression", in *Proceedings of IEEE Internantional Conference on Image Processing*, Kobe, Japan, Oct. 1999. *Cited in Sec.* 2.1.1

[VKG03] R. Venkataramani, G. Kramer, and V. Goyal, "Multiple description coding with many channels". *IEEE Transactions on Information Theory*, vol. 49 (9), pp. 2106–2114, Sep. 2003. *Cited in Sec.* 2.1.1

[VMG+06] F. Verdicchio, A. Munteanu, A. Gavrilescu, J. Cornelis, and P. Schelkens, "Embedded multiple description coding of video". *IEEE Transactions on Image Processing*, vol. 15 (10), pp. 3114–3130, Oct. 2006. *Cited in Sec.* 2.1.4

[VS10] D. Vukobratović and V. Stanković, "Unequal error protection random linear coding for multimedia communications", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Saint-Malo, France, Oct. 2010. *Cited in Sec.* 5.2, 5.3.2

[WA95] C. Wong and O. Au, "Fast motion compensated temporal interpolation for video", in *Proceedings of SPIE International Symposium on Visual Communications and Image Processing*, Taipei, ROC, May 1995. *Cited in Sec.* 2.2

[WAT96] C.-K. Wong, O. Au, and C.-W. Tang, "Motion compensated temporal interpolation with overlapping", in *Proceedings of IEEE International Symposium on Circuits and Systems*, Atlanta, GA, USA, May 1996. *Cited in Sec.* 2.2

[WB09] Z. Wang and A. Bovik, "Mean squared error: love it or leave it? A new look at signal fidelity measures". *IEEE Signal Processing Magazine*, vol. 26 (1), pp. 98–117, Jan. 2009. *Cited in Sec.* 1.1

[WBSS04] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity". *IEEE Transactions on Image Processing*, vol. 13 (4), pp. 600–612, Apr. 2004. *Cited in Sec.* 1.1, 4.3

[WC02] B. Williams and T. Camp, "Comparison of broadcasting techniques for mobile adhoc networks", in *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking & Computing*, Lausanne, Switzerland, Jun. 2002. *Cited in Sec.* 3.2, 3.5.3

[WCG+07] M. Wien, R. Cazoulat, A. Graffunder, A. Hutter, and P. Amon, "Real-time system for adaptive video streaming based on SVC". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17 (9), pp. 1227–1237, Sep. 2007, Invited Paper. *Cited in Sec.* 1.3.2

[WCWK10] D. Wu, S. Ci, H. Wang, and A. Katsaggelos, "Application-centric routing for video streaming over multihop wireless networks". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20 (12), pp. 1721–1734, Dec. 2010. *Cited in Sec.* 3.3

[WCX00] X. Wu, P. Chou, and X. Xue, "Minimum conditional entropy context quantization", in *Proceedings of IEEE International Symposium on Information Theory*, Sorrento, Italy, Jun. 2000. *Cited in Sec.* 2.2.2

[WFLB05] J. Widmer, C. Fragouli, and J. Le Boudec, "Low-complexity energy-efficient broadcasting in wireless ad-hoc networks using network coding", in *Proceedings of IEEE Workshop on Network Coding, Theory and Applications*, Riva del Garda, Italy, Apr. 2005. *Cited in Sec.* 5.2

[WL02] Y. Wang and S. Lin, "Error-resilient video coding using multiple description motion compensation". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12 (6), pp. 438–452, Jun. 2002. *Cited in Sec.* 2.1.1

[WL07a] M. Wang and B. Li, "Lava: a reality check of network coding in peer-to-peer live streaming", in *Proceedings of IEEE International Conference on Computer Communications*, Anchorage, AK, USA, May 2007. *Cited in Sec.* 5.2

[WL07b] ———, "Network coding in live peer-to-peer streaming". *IEEE Transactions on Multimedia*, vol. 9 (8), pp. 1554–1567, Dec. 2007. *Cited in Sec.* 5.2

[WL07c] ———, "R^2: random push with random network coding in live peer-to-peer streaming". *IEEE Journal on Selected Areas in Communications*, vol. 25 (9), pp. 1655–1666, Dec. 2007. *Cited in Sec.* 5.2

[WOR97] Y. Wang, M. Orchard, and A. Reibman, "Multiple description image coding for noisy channels by pairing transform coefficients", in *Proceedings of IEEE Workshop on Multimedia Signal Processing*, Princeton, NJ, USA, Jun. 1997. *Cited in Sec.* 2.1.3

[WOVR01] Y. Wang, M. Orchard, V. Vaishampayan, and A. Reibman, "Multiple description coding using pairwise correlating transforms". *IEEE Transactions on Image Processing*, vol. 10 (3), pp. 351–366, Mar. 2001. *Cited in Sec.* 2.1.3

[WRL05] Y. Wang, A. Reibman, and S. Lin, "Multiple description coding for video delivery". *Proceedings of the IEEE*, vol. 93 (1), pp. 57–70, Jan. 2005, Invited Paper. *Cited in Sec.* 2.1.1, 2.1.2, 2.2, 4.3

[WSBL03] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13 (7), pp. 560–576, Jul. 2003. *Cited in Sec.* 1.2.2, 1.3.2, 3.1

[WSJ+03] T. Wiegand, H. Schwarz, A. Joch, F. Kossentini, and G. Sullivan, "Rate-constrained coder control and comparison of video coding standards". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13 (7), pp. 688–703, Jul. 2003. *Cited in Sec.* 1.1, 3.3

[WXL07] F. Wang, Y. Xiong, and J. Liu, "mTreebone: a hybrid tree/mesh overlay for application-layer live video multicast", in *Proceedings of IEEE International Conference on Distributed Computing Systems*, Toronto, ON, Canada, Jun. 2007. *Cited in Sec.* 5.2

[WXL10] ———, "mTreebone: a hybrid tree/mesh overlay for application-layer live video multicast". *IEEE Transactions on Parallel and Distributed Systems*, vol. 21 (3), pp. 379 –392, march 2010. *Cited in Sec.* 5.2

[WZ76] A. Wyner and J. Ziv, "The rate-distortion function for source coding with side information at the decoder". *IEEE Transactions on Information Theory*, vol. 22 (1), pp. 1–10, Jan. 1976. *Cited in Sec.* 2.1.4, 2.2.1

[WZ98] Y. Wang and Q.-F. Zhu, "Error control and concealment for video communication: a review". *Proceedings of the IEEE*, vol. 86 (5), pp. 974–997, May 1998. *Cited in Sec.* 3.1

[WZ07] W. Wei and A. Zakhor, "Multiple tree video multicast over wireless ad-hoc networks". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17 (1), pp. 2–15, Jan. 2007. *Cited in Sec.* 3.3

[XGB02] K. Xu, M. Gerla, and S. Bae, "How effective is the IEEE 802.11 RTS/CTS handshake in ad-hoc networks?", in *Proceedings of IEEE Global Telecommunications Conference*, Taipei, ROC, Oct. 2002. *Cited in Sec.* 3.4.1

[XKSY08] H. Xie, A. Krishnamurthy, A. Silberschatz, and R. Y. Yang, "P4P: explicit communications for cooperative control between P2P and network providers", White Paper, 2008. *Cited in Sec.* A.3.2

[YLHX06] J. Yu, M. Li, F. Hong, and G. Xue, "Free-riding analysis of BitTorrent-like peer-to-peer networks", in *Proceedings of IEEE Asia-Pacific Conference on Services Computing*, Guangzhou, PRC, Dec. 2006. *Cited in Sec.* 5.2, A.1

[YYRZ11] F. Ye, R. Yim, S. Roy, and J. Zhang, "Efficiency and reliability of one-hop broadcasting in vehicular ad-hoc networks". *IEEE Journal on Selected Areas in Communications*, vol. 29 (1), pp. 151–160, Jan. 2011. *Cited in Sec.* 3.4.1, 4.3.2

[ZL09] C. Zhu and M. Liu, "Multiple description video coding based on hierarchical B pictures". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19 (4), pp. 511–521, Apr. 2009. *Cited in Sec.* 2.2.2, 2.2.3, 2.12, 2.13, 2.14, 5.5

[ZLC02] C. Zhu, X. Lin, and L.-P. Chau, "Hexagon-based search pattern for fast block motion estimation". *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12 (5), pp. 349–355, May 2002. *Cited in Sec.* 1.3.2

[ZM00] S. Zhu and K.-K. Ma, "A new diamond search algorithm for fast block-matching motion estimation". *IEEE Transactions on Image Processing*, vol. 9 (2), pp. 287–290, Feb. 2000. *Cited in Sec.* 1.3.2

[ZSG05] X. Zhu, E. Setton, and B. Girod, "Congestion-distortion optimized video transmission over ad-hoc networks". *Signal Processing: Image Communication (Elsevier Science)*, vol. 20, pp. 773–783, Sep. 2005, Invited Paper. *Cited in Sec.* 4.1