HAL
archives-ouvertes.fr

# Le code à effacement Mojette : Applications dans les réseaux et dans le Cloud

Benoît Parrein

▶ **To cite this version:**

Benoît Parrein. Le code à effacement Mojette : Applications dans les réseaux et dans le Cloud. Réseaux et télécommunications [cs.NI]. Université de Nantes, 2015. <tel-01169400>

**HAL Id: tel-01169400**

**https://hal.archives-ouvertes.fr/tel-01169400**

Submitted on 1 Jul 2015

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Le code à effacement Mojette:

## Applications dans les réseaux et dans le Cloud

Benoît Parrein

Habilitation à diriger des recherches

soutenue le 16 juin 2015 à Polytech Nantes
devant le jury composé de:

| | | |
|---|---|---|
| François Taïani | Université de Rennes 1 | Président |
| Christophe Cérin | Université Paris XIII | Rapporteur |
| Vincent Roca | Inria Grenoble | Rapporteur |
| Rodolphe Vauzelle | Université de Poitiers | Rapporteur |
| Laurent Toutain | Télécom Bretagne | Examinateur |
| Jeanpierre Guédon | Université de Nantes | Directeur |

# Table des matières


**1 Curriculum Vitæ** **1**

**2 Introduction** **20**

2.1 Positionnement de ce travail . . . . . 20

2.1.1 Théorie des codes correcteurs . . . . . 20

2.1.2 Traitement d'images . . . . . 21

2.1.3 Réseaux auto-organisés . . . . . 21

2.1.4 Stockage et système distribué . . . . . 22

2.2 Contributions personnelles . . . . . 22

2.2.1 Codes à effacement géométriques . . . . . 22

2.2.2 Schéma de codage conjoint source-canal . . . . . 22

2.2.3 Protocole sécurisé de routage à chemins multiples pour réseaux MANET . 23

2.2.4 Architecture hybride client/serveur et P2P . . . . . 23

2.2.5 RozoFS : un système de fichiers distribué pour le Cloud . . . . . 23

2.3 Plan du manuscrit . . . . . 24

**3 Codes à effacement par transformation Mojette** **25**

3.1 Introduction . . . . . 25

3.2 Rappels sur la transformation Mojette . . . . . 26

3.2.1 Les origines de la transformée . . . . . 26

3.2.2 Transformée Mojette-Dirac . . . . . 27

3.2.3 Une transformation redondante . . . . . 28

3.2.4 Critères de reconstruction d'une forme convexe . . . . . 29

3.2.5 Choix des angles de projections . . . . . 30

3.2.6 Algorithmes de transformation Mojette inverse . . . . . 31

3.3 Codes à effacement par transformation Mojette . . . . . 35

3.3.1 Définitions . . . . . 35

3.3.2 Code Mojette non systématique . . . . . 36

3.3.3 Code Mojette systématique . . . 38
3.3.4 Code Mojette non systématique sur support hexagonal . . . 40
3.3.5 Protection inégale . . . 43
3.4 Conclusion . . . 44

**4 Codes à effacement pour le Cloud (le projet FEC4Cloud) 46**
4.1 Introduction . . . 46
4.2 Tolérance aux pannes . . . 47
4.2.1 Comparaison avec l'approche par réplication . . . 47
4.2.2 Performances des codes Mojette . . . 48
4.3 Intégration au sein du système de fichiers distribué RozoFS . . . 52
4.3.1 Introduction à RozoFS . . . 53
4.3.2 Application au montage vidéo en ligne . . . 53
4.3.3 Expérimentations sur GRID5000 . . . 56
4.4 Code à effacement et Big Data . . . 61
4.4.1 Éléments de contexte . . . 61
4.4.2 Matériels et méthodes . . . 63
4.4.3 Résultats . . . 64
4.5 Conclusion . . . 64

**5 Un protocole hybride P2P et Client/Serveur (le projet P2PWeb) 66**
5.1 Introduction . . . 66
5.2 Le protocole P2PWeb . . . 67
5.2.1 Spécifications générales . . . 67
5.2.2 Matériels et méthodes . . . 68
5.2.3 Résultats . . . 69
5.2.4 Optimisation du protocole . . . 73
5.3 Conclusions et perspectives . . . 74

**6 Réseaux auto-organisés de type ad hoc (le projet SEREADMO) 76**
6.1 Introduction . . . 76
6.2 MP-OLSR . . . 77
6.2.1 Spécifications . . . 77
6.2.2 Mises en œuvre . . . 80
6.2.3 Résultats de simulations . . . 81
6.2.4 Expérimentations SEREADMO . . . 85
6.3 Applications pour le transfert de vidéos scalables . . . 89
6.3.1 Présentation de l'expérimentation . . . 89
6.3.2 Résultats QoS/QoE . . . 91
6.4 Conclusions . . . 93

**7 Conclusion** **95**
7.1 Résumé du manuscrit . . . . . 95
7.2 Perspectives de recherche . . . . . 96

**8 Sélection d'articles** **98**
8.1 Annals of Telecommunications (2003) . . . . . 99
8.2 Elsevier Journal of Ad Hoc Networks (2011) . . . . . 115
8.3 CLOSER Conference (2014) . . . . . 135
8.4 Usenix HostStorage 2015 (soumission/dépôt ArXiv) . . . . . 141

**Table des figures** **146**

**Index** **150**

**Bibliographie** **152**

# Chapitre 1

# Curriculum Vitæ

# Curriculum Vitæ de Benoît Parrein

## 1. Identification

**Nom** Parrein
**Prénom** Benoît
**Date de naissance** 22 février 1975
**Situation** Marié, 2 enfants
**Grade** Classe normale, 6 e échelon
**Établissement** Université de Nantes (Polytech Nantes, département Informatique)
**Section** CNU 27
**Unité de recherche**
Laboratoire IRCCyN (UMR 6597), équipe Image Vidéo Communication (IVC)
**Thèse de doctorat**
**«** Description multiple de l'information par transformation Mojette », Université de Nantes, Novembre 2001, jury composé de : Christine Guillemot (rapporteure), Alain Duhamel (rapporteur), Christian Olivier (président), Jeanpierre Guédon (directeur), Nicolas Normand (co-encadrant).

**Autres diplômes**
DEA Signaux et images en biologie et médecine, Université d'Angers, 1998
Maîtrise de Physique, Université Paris VII, 1997

## 2. Déroulement de carrière

| | |
|---|---|
| *Janvier – Juin 2015* | **Congés pour Recherches (6 mois)** |
| *Avril – Mai 2010* | **Chercheur invité Monash University** (Australie) |
| *Sep. 2004 – aujourd'hui* | **Maître de Conférences**<br>Université de Nantes, Polytech Nantes<br>Département Informatique/Laboratoire IRCCyN |
| *Déc. 2002 – Juin 2004* | **Ingénieur de Recherche**<br>Sud Paris Tech (ex-INT), Évry |
| *Oct. 1998 – Nov. 2002* | **Doctorant puis postdoctorant**<br>Laboratoire IRCCyN, UMR 6597, équipe IVC |
| *Fév. 1998 – Août 1998* | **Stage D.E.A**<br>INSERM, U466, CHU Nantes, Médecine Nucléaire |

## 3. Activités pédagogiques

### 3.1 Enseignements

Vous trouverez ci-dessous la répartition des enseignements par établissement et département. Les volumes horaires et le nombre d'étudiants concernés par la formation sont indiqués sous la forme : (CM TD TP Projet) (Nbre d'étudiants) e.g.(0 3 9 0) (24) .

**Université de Nantes, Polytech Nantes, département Informatique** (~200h)
**Année 1** : micro projet Parcours Élève Ingénieur Polytech (0 0 0 3) (9)
**Année 2** : projet PEIP (0 0 0 22) (12)
**Année 3 :** Principe des réseaux (6 3 9 0) (70), Conception de protocoles (0 12 0 0) (24), Théorie de l'information (0 4.5 0 0) (24), Géométrie algorithmique (0 4.5 0 0) (24).
**Année 4 :** Réseaux 1 (0 0 15 0) (24), Internet et interopérabilité (0 0 9 0) (24), Traitement d'images et Multimédias (0 0 9 0) (24), Cryptographie (9 4.5 0 0) (70), Réseaux 2 (12 0 6 9) (48), Sécurité des réseaux (0 0 6 0) (24).
**Année 5 :** Qualité de Service (0 0 0 6) (16), Internet Multimédia (6.25 6 0 0) (16), Projet R&D (0 0 0 0 20) (4)

**Université de Nantes, Polytech Nantes, master MDM (Multimedia and Data Management)** (~20h)
**Semestre 1 :** Computer Networks, Distributed Systems, Mobile Ad hoc Networks

**Université de Nantes, IUT de Nantes, département Informatique** (~25h)
**Année 1 :** Architecture des réseaux (0 9 0 0) (26)
**Année 2 :** Introduction sécurité réseau, système et logiciel (0 16 0 0) (26)

### 3.2 Responsabilités pédagogiques et administratives

**Membre des deux derniers conseil de perfectionnement du département Informatique de Polytech Nantes** (réuni en janvier 2010 et juin 2012). Composition : 10 industriels, 2 anciens élèves, 2 académiques extérieurs, 10 internes dont la direction de l'école.

**Responsable de l'option « Réseaux, Systèmes et Cloud » (depuis 2011).** Option de dernière année du cursus ingénieur « Systèmes Informatique, Logiciels et Réseaux » du département Informatique de Polytech Nantes. 10 à 20 élèves par an. 150 heures de spécialité. 15 intervenants industriels. Visites sur sites industriels. Conception d'un catalogue de formation continue.

**Responsable de la formation dans le cadre de l'école doctorale STIM (ED-503)** entre 2008 et 2012. Constitution du catalogue des formations (10 modules scientifiques et 12 modules professionnels), validations des modules, équivalences formations extérieures. Membre du bureau de l'ED sur cette période. Participation à l'évaluation AERES (décembre 2011, notation A+).

**Responsable des stages (2008 – 2011).** Stage de fin d'étude d'ingénieur Polytech Nantes, département Informatique. **65 étudiants par an**.

**Chargé de mission « service informatique » (à partir de mars 2015)** nommé par le directeur de Polytech Nantes : diagnostic technique, relations humaines évolution du SI à destination de l'enseignement et de la recherche.

**Membre élu du conseil de l'école** (Collège B)

## 4. Activités de recherche

J'effectue ma recherche au sein de l'équipe Image Vidéo Communication (IVC) du laboratoire IRCCyN (UMR 6597). Mon activité de recherche est fortement trans-disciplinaire. Elle se situe à l'interface des **réseaux et systèmes informatiques** et de **la théorie de l'information et du codage**.
Ma contribution en réseau concerne les réseaux auto-organisés (de type **ad hoc et P2P**) et, plus récemment, le domaine des systèmes de fichiers distribués dédiés au stockage externalisé en mode **Cloud**. Au niveau théorique, j'ai proposé un schéma de **codage conjoint source-canal** appliqué au transport d'images fixes et de la vidéo. J'ai proposé également, en collaboration avec Nicolas Normand, **des codes à effacement géométriques** s'appuyant sur la transformation Mojette et la transformation de Radon finie (FRT).
Depuis 2 quadriennaux, je suis responsable du thème **Réseaux & Services Multimédias** au sein de l'équipe IVC. Je suis l'auteur d'une cinquantaine de publications dans les différents domaines pré-cités (voir section 4.5).

### 4.1 Encadrement doctoral, postdoctoral et master

**Fadi Boulos,** « Trafic audiovisuel à qualité d'usage garantie : caractérisation d'erreurs, de la gêne associée et mécanismes de protection », Université de Nantes, bourse MESR, **10/2006 – 02/2010, taux 50 %** (aujourd'hui responsable avant-vente chez Setelia, Liban).

**Jiazi Yi,** « Protocole de routage à chemins multiples pour réseaux auto-organisés », Université de Nantes, financement Région, **10/2007-11/2010, taux 60 %** (aujourd'hui post-doc au laboratoire LiX, Palaiseau).

**Dimitri Pertin,** « Codes à effacements géométriques pour le stockage distribué en mode cloud », Université de Nantes, thèse CIFRE, **4/2012 - . , taux 50 %**

J'ai également participé à l'encadrement de thèse d'**Eddy Cizeron** entre 2007 et 2010 sur le thème de la description multiple et des réseaux ad hoc (production commune : 2 conférences internationales) et à l'encadrement de thèse de **Dan Radu**, Université de Cluj-Napoca, Roumanie qui a effectué 2 visites dans l'équipe de 6 mois chacune (production commune: 3 conférences

internationales). J'ai fait parti de son jury de thèse à Cluj, Roumanie en septembre 2012.
Au niveau postdoctoral, j'ai recruté et encadré pendant 24 mois **Asmaa Adnane** et **Majd Ghareeb** respectivement sur les projets collaboratifs Sereadmo et P2PWeb entre 2008 et 2010 (pour Asmaa) et 2010 et 2012 (pour Majd). **Alexandre Van Kempen**, docteur de l'Université de Rennes 1, est actuellement sous ma direction dans le cadre du projet FEC4Cloud (contrat post-doctoral de 9 mois).

Depuis 2004 (année de mon recrutement en tant que MCF), j'ai pu encadré un total de plus de **20 stages masters** ou élèves ingénieurs de dernière année soit environ 2 stagiaires par an. La liste ci-dessous concerne la période 2008 – 2014, avec le diplôme préparé, le taux d'encadrement, le thème abordé et leur devenir (si disponible):

(1) **Xiang Zhu** (2008), Polytech Nantes, Master SEGE, 100%, routage ad hoc, aujourd'hui Ingénieur HP China (2) **Sylvain David (**2009), Polytech Nantes, Ing. Informatique, 100%, routage ad hoc et code FEC, Ingénieur R&D Fizians (3) **Wissam Hamdach** (2010), Polytech Nantes, Master SEGE, 100%, routage ad hoc, ingénieur Télécom à Dubaï (4) **Thibaut Perret** (2011), Polytech Nantes, Ing. Informatique, 100%, réseau P2P, thèse au Canada (5) **Jean-Daniel Fischer** (2011), Polytech Nantes, Ing. Informatique, 50%, network time protocol (NTP) and RadClock, ingénieur Bouygues Télécom (6) **Giulio d'Ippolito** (2011), Université Roma Tre, Master, 50%, code à effacement, Ingénieur Système en Italie (7) **Dimitri Pertin** (2012), Polytech Nantes, Ing. Informatique, 50%, code à effacement, thèse CIFRE avec moi et l'entreprise Fizians (8) **Nassima Bouzakaria** (2012), Master Réseaux et Télécom, Université Paul Sabatier Toulouse, 50%, transport vidéo et P2P (9) **Pauline Folz** (2013), Université de Nantes, Master ALMA, 100 %, traitement distribué dans le Cloud, thèse CIFRE à Nantes Métropole (10) **Vikram Runthla** (2014), Polytech Nantes, Master international MDM, 100 %, optimisation de services sur réseaux à faible bande-passante (11) **Chen Chao** (2015), Polytech Nantes, Master international MDM, 50 %, Internet of Thing and Cloud infrastructure, collaboration avec Prof. Jinlong Hu, South China University of Technology (SCUT).

### 4.2 Projets de recherche (avec responsabilités scientifiques)

***Sep. 2006- Nov. 2009 (36 mois) :*** **SEREADMO**, SEcurité et REseaux AD hoc par transformation Mojette. Appel ANR RNRT 2005. Partenaires : Thalès (resp.), IRCCyN, Université de Poitiers. Ce projet est mon premier projet d'envergure. J'ai eu la responsabilité scientifique pour l'IRCCyN dès le début. Suite au départ du chef de projet en 2008 (Éric Grall), la responsabilité de l'ensemble du projet SEREADMO m'a été confiée jusqu'à sa revue finale en novembre 2009 au siège de l'ANR.

***Mars 2010 – Déc. 2012 (24+6 mois):*** **P2PWeb**, Architecture hybride Client/Serveur et P2P pour la navigation Web. Projet PME du pôle Images&Réseaux. Financement Région Pays de la Loire. Partenaires : TMG (resp.), LINA, IRCCyN. J'ai participé activement à la rédaction du projet en proposant le LINA comme partenaire. J'ai été responsable scientifique pour l'IRCCyN. J'ai assisté TMG (Soufiane Rouibia) dans les grandes orientations scientifiques du projet.

***Jan. 2013 – Déc. 2014 (24 mois)*****: FEC4CLOUD,** Codes à effacement pour des infrastructures de stockage en mode Cloud. Appel ANR Émergence 2011. Partenaires : IRCCyN (resp.), ISAE, SATT Ouest Valorisation. Prestataires : Fizians SAS. J'ai monté ce projet de bout-en-bout : choix des partenaires et de l'appel, rédaction du projet scientifique et du budget. Je suis responsable du projet et responsable scientifique pour l'IRCCyN.

**Autres projets** : Mattador (projet CPER, 2006-2009), PrivateP2PImage (projet Fédération Atlanstic, 2009). Soumission de deux projets ANR en décembre 2014 (Privacy4cloud et PROSTO).

### 4.3 Collaborations industrielles et académiques

#### 4.3.1 Collaborations industrielles

La *spin-off* **Fizians** (basée à Polytech, contact Pierre Evenou) spécialisée dans le stockage distribué: accompagnement R&D depuis 2010 (date de la création de la société), projet FEC4Cloud en 2013 ayant vocation à valoriser le projet *open source* RozoFS, participations communes à 3 actions de valorisation en 2013 et 2014[1], co-publications [14][18][49], 1 thèse CIFRE en 2013 (Dimitri Pertin), projets R&D étudiants. La société compte aujourd'hui 7 collaborateurs.

La PME **Trident Media Guard** (**TMG)** (Saint-Sébastien-sur-Loire, contact Soufiane Rouibia) spécialisée dans la sécurité et la protection des droits d'auteur sur les réseaux P2P : montage du projet P2PWeb (voir section 4.2), co-publications [19][44][45], projets R&D étudiants.

Ainsi qu'auprès de grands groupes :

***(i)*** **Thalès** (site de Cholet, contact Éric Grall) : participation projet ANR Sereadmo (voir section 4.2), co-publication d'un chapitre d'ouvrage [8]. ***(ii)*** **British Telecom** (site Ipswich (UK), contact : David Hands) : séjour de 6 mois de Fadi Boulos à Ipswich, co-publication [39]. ***(iii)*** **Orange Labs R&D** (site de Lannion, contact Laurent Reynaud) : échanges informels autour du protocole MP-OLSR, visites techniques, Laurent a été membre du jury de thèse de Jiazi Yi. ***(iv)*** **Alcatel-Lucent** (site Orvault, contact Hervé Briand) : projets R&D étudiants, co-encadrement master recherche Diana Allam.

**Autres contacts** : Technicolor, Orange Labs (Issy-les-Moulineaux), Cloudwatt, OVH, Keosys (Saint Herblain), AccepTV (*spin off* IVC).

#### 4.3.2 Collaborations académiques internationales

Trois collaborations avec l'Australie (sur la transformation de Radon fini FRT, les codes MDS et le protocole RADClock) :

---

1 NEM SUMMIT 2013 (Stand SATT Ouest Valorisation), JRES 2013 (Stand Total Linux) et FOSDEM 2014 (DevRoom)

- **Imants Svalbe**, Senior Lecturer, Faculty of Sciences, **Monash University**, Melbourne, invité à plusieurs reprises entre 2008 et 2015. J'ai bénéficié d'une invitation de 2 mois par l'Université de Monash (avril et mai 2010).
- **Andrew Kingston, Australian National University (ANU)**, Canberra, invité en 2009 pour un mois (post-doc dans l'équipe en 2007). Publication : 2 chapitres d'ouvrage [7][8] et 2 conférences [23][25].
- **Julien Ridoux, Melbourne University**, invité 1 mois en 2011. Co-encadrement du projet de fin d'étude de Jean-Daniel Fischer (élève ingénieur Polytech) portant sur l'évaluation du protocole de synchronisation RADClock.

Collaboration avec la Roumanie (**Université Technique de Cluj-Napoca**) : **Professeur Adina Astilean** et son doctorant **Dan Radu** que j'ai encadré (voir section 4.1 encadrement doctoral). Création d'un accord Erasmus.. Visite d'1 semaine en septembre 2012 et participation au jury de thèse de Dan.

Collaboration avec l'Italie (**Université de Roma Tre**). **Prof. Marco Carli, Federica Battisti, Daniele Bibbo**. Nombreuses visites de mes collègues italiens (Accord Erasmus). Encadrement stage master puis embauche sur le projet P2PWeb de Giulio D'Ippolito en 2012. Publication [20][21].

Collaboration avec deux universités chinoises de Gangzhou (**GDUT et SCUT)** :
Professeur Xu (Vice Dean SCUT). Visite officielle en Chine en 2009.
Collaboration scientifique avec Prof. Jinlong Hu (encadrement d'un master).

Au sein du **MANET WG de l'IETF** (Université d'Ottawa, Funkfeuer Vienna, Navy Research Lab, Poznan University of Technology, Polytechnique…). Cinq meetings entre 2010 et 2012. Reprise des activités au sein de l'IETF depuis mars 2014 (Internet Draft MP-OLSR).

### 4.3.3 Collaborations académiques nationales

Réseaux ad hoc et de capteurs : l'équipe Hipercom de LiX (**Polytechnique, Palaiseau**) : Jiazi Yi (aujourd'hui en post-doc là-bas), Thomas Clausen, Philippe Jacquet, Emmanuel Baccelli. Rédaction d'un draft IETF avec Emmanuel en 2010 en collaboration avec l'Université Technique de Poznan (Andrzej Szwabe).

Sur les aspects de codages : *(i)* Jérôme Lacan (**ISAE, Toulouse**) et Vincent Roca (**INRIA, Privatics, Grenoble**). Initiée au sein du **FECFrame WG de l'IETF** en

2011 (meeting 80, Prague), cette collaboration a abouti à un projet ANR accepté en 2012 (FEC4Cloud avec Jérôme) et un projet ANR soumis en 2014 et 2015 (Privacy4Cloud). *(ii)* Olivier Déforges (**IETR**) et Marie Babel (**IRISA**), 1 revue et 1 conférence co-publiée [2][29] portant sur le codage conjoint source-canal.

Systèmes distribués et Cloud : (*i*) Jacques Jorda (IRIT), 1 soumission ANR commune en 2015, (*ii*) Adrien Lebre, INRIA/Ecole des Mines de Nantes, expérimentations sur Grid'5000.

Communications numériques et réseaux radios : *(i)* Rodolphe Vauzelle, Yannis Pousset, Anne-Marie Poussard**,** équipe (IRCOM SIC), **Université de Poitiers**, 1 conférence [24], 2 chapitres d'ouvrage [10][11]. *(ii)* Patrick Corlay et François-Xavier Coudoux, **Université de Valenciennes**, co-organisation d'une session spéciale dans le cadre de la conférence IEEE ISIVC 2012, invitation GdR ISIS [14].

### 4.4 Rayonnement

#### 4.4.1 Membre de comité de programme ou de relecture

**Organisation d'une session spéciale à IEEE ISIVC 2012** portant sur les services multimédias sur réseaux à perte de paquets. Co-organisation avec Patrick Corlay, Université de Valenciennes. Cinq papiers acceptés.
**Responsable publication** IEEE Packet Video 2003 à Nantes.

Membre du comité d'organisation de la journée Mojette (5 février 2015) : 40 participants (voir **http://www.mojette.org/event/mojette-day/**) et du 3ème Sino-French Workshop on Information and Communication Technologies (SIFWICT 2015) qui aura lieu le 12 Juin 2015 à la Cité des Congrès de Nantes (événement Polytech Nantes, IETR, IRCCyN, LINA).

**Comité de lecture** pour des revues et conférences internationales : Journal of Zhejiang University (2 relecture), SCPA Workshop dans le cadre de IEEE International Conference on Communications (ICC) (2 relectures), *European Future Technology Conference* FET (2 relectures), *International Conference on Computer and Communication Technology* (ICCCT) (2 relectures), *IEEE International Conference on Connected Vehicles and Expo* (TPC pour ICCVE 2013 et 2014) (1 relecture), *International Journal on Communication Systems 2013* (1

relecture), *Journal of Signal, Images and Video Processing* by Springer (Ed. Murat Kunt) (1 relecture en 2013), *Elsevier Ad hoc Journal* (2 relecture en 2014) .

### 4.4.2 Comités de sélection, suivi et jury de thèses

**Membre du comité de sélection** pour le recrutement d'un maître de conférences à l'INSA/IETR de Rennes en 2011.
**Comité de suivi de thèses** de 3 doctorants : Abhijit Sarkar (CIFRE Technicolor), Chen Wei (CIFRE Technicolor), Mounir Tlili (LINA)
**Participations aux jurys de thèse** : (1) Shan Wang, 2007, Université de Poitiers (examinateur), (2) Inès Slama, 2008, Telecom SudParis (examinateur), (3) Dan Radu, 2012, Université Technique de Cluj-Napoca, Roumanie (rapporteur).

### 4.4.3 Expertises

1 **expertise ANR** sur l'appel CONTINT 2012.
1 **expertise internationale** d'un projet de recherche VANET pour le compte du conseil de la recherche de la **City University of Hong Kong**.
1 expertise pour le compte de la **Région Pays de la Loire en 2013 et 2014** (dans le cadre d'une démarche **Recherche-Formation-Innovation (RFI)** : élaboration des domaines d'innovations stratégiques ou DIS) pour le compte du RFI Numérique.

**Membre du groupe de travail « Innovation et valorisation de la recherche » (depuis décembre 2014)** auprès du Vice-Président Valorisation Transfert de l'Université de Nantes dans le cadre de la rédaction du prochain contrat quinquennal (2017-2021).

## 4.5 Publications et production scientifique (extraction HAL)

| | |
|---|---|
| *H Index* | 11 |
| *I10* | 14 (Avril 2015, source Google Scholar) |
| *Publications* | 55 |
| *Articles revues internationales* | 4 |
| *Articles revues nationales* | 1 |
| *Chapitres d'ouvrage* | 5 |
| *Conférences invitées internationales* | 2 |
| *Conférences invitées nationales* | 3 |
| *Conférences internationales* | 20 |
| *Workshops internationaux* | 8 |
| *Conférences nationales* | 5 |
| *Brevets* | 3 |
| *Drafts IETF* | 4 |

## Articles revues internationales avec comité de lecture

[1] «Multipath Optimized Link State Routing for Mobile ad hoc Networks», Yi, Jiazi; Adnane, Hassiba Asmaa; David, Sylvain; **Parrein, Benoît**, 10.1016/j.adhoc.2010.04.007 *Ad Hoc Networks*, (2011-01-01) 9 1 28-47 (**89 citations** : source Google Scholar).

[2] «Joint source-channel coding: secured and progressive transmission of compressed medical images on the Internet», Babel, Marie; **Parrein, Benoît**; Déforges, Olivier; Normand, Nicolas; Guédon, Jean-Pierre; Coat, Véronique, 10.1016/j.compmedimag.2008.01.002 *Computerized Medical Imaging and Graphics*, (2008-06-00) 32 4 258-269. (**12 citations**)

[3] «Internet Distributed Image Information System», Guédon, Jean-Pierre; **Parrein, Benoît**; Normand, Nicolas, *Integrated Computer-Aided Engineering*, (2001-08-00) 8 3 205-214 (**41 citations**).

[4] «Post-synchronized scintigraphic data to estimate antral motility», Le Rest, Catherine; **Parrein, Benoît**; Morin, V.; Bridji, B.; Bizais, Yves, J.; Couturier, O., *Nuclear Medicine Communications*, (2001-00-00) 291-303

## Articles revues nationales avec comité de lecture

[5] «Multimedia forward error correcting codes for wireless LAN», **Parrein, Benoît**; Normand, Nicolas; Guédon, Jean-Pierre, *Annales des Télécommunications*, (2003-00-00) 58 3-4 448-463. (**17 citations**)

## Chapitres d'ouvrages scientifiques

[6] «Multiresolution Mojette transform (chap 6)», Kingston, Andrew; Autrusseau, Florent; **Parrein, Benoît**, *The Mojette transform: Theory and Applications*, (2009-01-09) 29 pages iste & wiley Jeanpierre Guédon ISBN: 9781848210806

[7] «Mojette based security (chap 10)», Kingston, Andrew; Autrusseau, Florent; Grall, Eric; Hamon, Thierry; **Parrein, Benoît**, *The Mojette transform: Theory and Applications*, (2009-01-09) 25 pages iste & wiley Jeanpierre Guédon ISBN: 9781848210806

[8] «Communication, networks and storage (chap 7)», **Parrein, Benoît**; Boulos, Fadi; Normand, Nicolas; Evenou, Pierre, *The Mojette Transform: Theory and Applications*, (2009-01-09) 29 pages iste & wiley Jeanpierre Guédon ISBN: 9781848210806

[9] «Transmission of compressed medical data on fixed and mobile networks», Olivier, Christian; **Parrein, Benoît**; Vauzelle, Rodolphe, *Compression of biomedical images and signals*, (2008-00-00) 277-300 ISTE, Wiley

[10] «Transmission de données médicales compressées sur réseaux fixe et mobile», Olivier, Christian; **Parrein, Benoît**; Vauzelle, Rodolphe, *Compression des images et des signaux médicaux*, (2007-02-00) 299-322 Hermès Traité IC2

## Conférences invitées internationales

[11] «Region-of-Interest Intra Prediction for H.264/AVC Error Resilience», Boulos, Fadi; Chen, Wei; **Parrein, Benoît**; Le Callet, Patrick, 10.1109/ICIP.2009.5414458 *Proceedings of IEEE International Conference on Image Processing*, (2009-11-00) 3109 - 3112 IEEE International Conference on Image Processing (2009-11-07) Cairo Égypte, session spéciale « Visual Attention », conférence invitée (**9 citations**).

[12] «Multi-Path Optimized Link State Routing (MP-OLSR)», Yi, Jiazi; **Parrein, Benoît**, (2010-07-00) IETF'78 (2010-07), MANET WG, Maastricht, Pays-Bas.

## Conférences invitées nationales

[14] « Code à effacement pour le Cloud : application au montage vidéo en ligne », **Benoît Parrein**, Pierre Evenou, Réunion GdR ISIS, Stratégie optimale de transmission de contenus multimédias, 18/10/2013, Institut Telecom, Paris.

[15] «Transmission de données médicales sur réseaux de paquets», **Parrein, Benoît**, (2007-06-00) journée EEA sur le thème des communications multimédias (2007-06) La Rochelle France.

[16] «Transmission prioritaire d'images fixes pilotée par un critère de qualité visuelle», **Parrein, Benoît**; Normand, Nicolas; Le Callet, Patrick; Guédon, Jeanpierre, (2006-01-00) journée inter GDR ISIS-Onde (2006-01) Paris France.

## Conférences internationales avec comité de lecture

[17] « The Mojette Erasure Code for Distributed File Systems », Dimitri Pertin, Benoît Parrein, Nicolas Normand, ACM EuroSys, 13-16 Avril 2014, Amsterdam, Pays-Bas.

[18] « Distributed File System based on Erasure Coding for I/O Intensive Applications », Dimitri Pertin, Sylvain David, Pierre Evenou, **Benoît Parrein**, Nicolas Normand, 4th International Conference on Cloud Computing and Service Science (CLOSER) 2014, 3-5 Avril 2014, Barcelona, Spain.

[19] «P2PWeb: a Client/Server and P2P Hybrid Architecture for Content Delivery over Internet», Ghareed, Majd; Rouibia, Soufiane; **Parrein, Benoît**; Raad, Mohamad; Thareau, Cedric, *Proceedings of the Third International Conference on Communications and Information Technology ICCIT 2013*, (2013-06-19) pp.1-5, Beirut Liban.

[20] **Parrein, B**.; Normand, N.; Ghareeb, M.; D'Ippolito, G.; Battisti, F., "Finite Radon coding for content delivery over hybrid client-server and P2P architecture," Communications Control and Signal Processing (ISCCSP), 2012 5th International Symposium on , vol., no., pp.1,4, 2-4 May 2012

[21] «Spatial Implementation for Erasure Coding by Finite Radon Transform», Pertin, Dimitri; D'Ippolito, Giulio; Normand, Nicolas; **Parrein, Benoît**, *Proceedings of the International Symposium on signal, Image, Video and Communication 2012*, (2012-00-00) 1-4 International Symposium on signal, Image, Video and Communication 2012 (2012-07-04) Valenciennes France.

[21] «QoE enhancement for H.264/SVC video transmission in MANET using MP-OLSR protocol», Radu, Dan; Yi, Jiazi; **Parrein, Benoît**, *Proceedings of ISIVC 2012, the 6th International Symposium on signal, Image, Video and Communications*, (2012-07-04) 1-4 ISIVC 2012, Valenciennes France

[22] Radu, D.; Avram, C.; Astilean, A.; **Parrein, B.**; Jiazi Yi, "Acoustic noise pollution monitoring in an urban environment using a VANET network," Automation Quality and Testing Robotics (AQTR), 2012 IEEE International Conference on, pp.244,248, 24-27 May 2012, doi: 10.1109/AQTR.2012.623771.

[23] «Erasure Coding with the Finite Radon Transform», Normand, Nicolas; Svalbe, Imants; **Parrein, Benoît**; Kingston, Andrew, 10.1109/WCNC.2010.5506385 *Wireless Communications & Networking Conference*, (2010-04-21) 1-6 Wireless Communications & Networking Conference (2010-04-18) Sydney Australie (**12 citations**).

[24] «Realistic SISO and MIMO Physical Layer implemented in two Routing Protocols for Vehicular Ad hoc Network», Poussard, Anne-Marie; Hamidouche, Wassim; Vauzelle, Rodolphe; Pousset, Yannis; **Parrein, Benoît**, *Electronic proceeding of IEEE ITST*, (2009-10-00) 5 pages IEEE ITST (2009-10) Lille France.

[25] «Redundant Image Representation via Multi-Scale Digital Radon Projection», Kingston, Andrew; **Parrein, Benoît**; Autrusseau, Florent, *Proceedings of the International Conf. of Image Processing*, (2008-10-13) 2069 International Conf. of Image Processing (2008-10-13), San Diego, Californie, États-Unis.

[26] «Simulation and Performance Analysis of MP-OLSR for Mobile Ad hoc Networks», Yi, Jiazi; Cizeron, Eddy; Hamma, Salima; **Parrein, Benoît**, *IEEE WCNC 2008*, (2008-03-31) 2235-2240 IEEE WCNC 2008 (2008-03-31) Las Vegas, États-Unis (**44 citations**).

[27] «Lossless Compression Based on a Discrete and Exact Radon Transform: A Preliminary Study», Autrusseau, Florent; **Parrein, Benoît**; Servieres, Myriam, *International Conference on Acoustics, Speech and Signal Processing*, (2006-00-00) 425-428 International Conference on Acoustics, Speech and Signal Processing (2006) Toulouse France (**8 citations**).

[28] «Flexible Storage of Still Images with a Perceptual Quality Criterion», Ricordel, Vincent; Le Callet, Patrick; Carnec, Mathieu; **Parrein, Benoît**, *Advanced Concepts for Intelligent Vision Systems (ACIVS)*, (2005-09-00) - Advanced Concepts for Intelligent Vision Systems (ACIVS) (2005-09-20) Antwerp Belgique.

[29] «Secured and progressive transmission of compressed images on the Internet: application to telemedicine», Babel, Marie; **Parrein, Benoît**; Déforges, Olivier; Normand, Nicolas; Guédon, Jean-Pierre; Ronsin, Joseph, *Proc. of SPIE Electronic Imaging'05*, (2005-01-00) 126-136 SPIE 17th Annual Symposium / Electronic Imaging - Internet Imaging, San José, États-Unis.

[30] « Demosaicking and JPEG2000 compression of microscopy images », **B. Parrein**, M. Tarin, P. Horain, to be published in the Proceedings of the International Conference on Image Processing (ICIP 2004), Singapore, October 24-27, 2004 (**17 citations**).

[31] «Distributed and compressed multimedia transmission using a discrete backprojection operator», Verbert, Pierre; Guédon, Jean-Pierre; **Parrein, Benoît**, *Proceeding of SPIE ITCOM 2002*, (2002-00-00) 4862 315-325 SPIE ITCOM 2002 (2002-08) Boston États-Unis.

[32] «Load-balancing and scalable multimedia distribution using the Mojette transform», Guédon, Jean-Pierre; Normand, Nicolas; Verbert, Pierre; **Parrein, Benoît**; Autrusseau, Florent, 10.1117/12.434272 *Internet Multimedia Management Systems II, ITCOM*, (2001-08-00) 226-234 Internet Multimedia Management Systems II, ITCOM (2001-08) Denver États-Unis.

[33] Scalable Multiple Descriptions on Packet Networks via the n-dimensional Mojette transform, **B. Parrein**, P. Verbert, N. Normand, J.P. Guédon, SPIE ITCom 2001, Denver, USA, 10 p., August 2001.

[34] «Multiple description coding using exact discrete Radon transform», **Parrein, Benoît**; Normand, Nicolas; Guédon, Jeanpierre, *Proceedings of IEEE Data Compression Conference*, (2001-03-00) 508 IEEE Data Compression Conference (2001-03) Snowbird (UT) États-Unis (**11 citations**).

[35] «Distributed image transmission and storage on Internet system», Guédon, Jeanpierre; Normand, Nicolas; **Parrein, Benoît**; Pouliquen, Christophe, *Proceedings of ACIDCA, vol. VPR*, (2000-03-00) 164-169 ACIDCA (2000-03) Monastir Tunisie.

**Workshops internationaux**

[36] «MULTIPATH ROUTING PROTOCOL FOR MANET: APPLICATION TO H.264/SVC VIDEO CONTENT DELIVERY», Yi, Jiazi; Parrein, Benoît; Radu, Dan, *Processing of the WPMC*, (2011-10-00) pp.1-5 WPMC 2011 (2011-10) Brest France

[37] «Multipath OLSR: Simulation and Testbed», Yi, Jiazi; David, Sylvain; Adnane, Hassiba Asmaa; **Parrein, Benoît;** Lecourtier, Xavier, (2009-10-02) 5th OLSR Interop/Workshop (2009-10-02) Vienna Autriche **(7 citations)**.

[38] «A New H.264/AVC Error Resilience Model Based on Regions of Interest», Boulos, Fadi; Chen, Wei; **Parrein, Benoît**; Le Callet, Patrick, 10.1109/PACKET.2009.5152159 *17th International Packet Video Workshop, PV 2009*, (2009-06-30) 1-9 Packet Video (2009-05-11) Seattle, Washington, États-Unis (**14 citations**).

[39] «Perceptual Effects of Packet Loss on H.264/AVC Encoded Videos», Boulos, Fadi; **Parrein, Benoît**; Le Callet, Patrick; Hands, David, S., *Perceptual Effects of Packet Loss on H.264/AVC Encoded Videos*, (2009-00-00) - Fourth International Workshop on Video Processing and Quality Metrics for Consumer Electronics VPQM-09 (2009-01-15) Scottsdale, Arizona, États-Unis (**19 citations**).

[40] «Implementation of Multipath and Multiple Description Coding in OLSR», Yi, Jiazi; Cizeron, Eddy; Hamma, Salima; **Parrein, Benoît**; Lesage, Pascal, *4th OLSR Interop/Work Shop*, (2008-10-14) 19-25 4th OLSR Interop/Work Shop (2008-10-14) Ottawa Canada (**10 citations**).

[41] «Priority image and video encoding transmission based on a discrete Radon transform», **Parrein, Benoît**; Boulos, Fadi; Le Callet, Patrick; Guédon, Jean-Pierre, *Proceedings of IEEE Packet Video Workshop 2007*, (2007-11-00) 6 pages IEEE Packet Video 2007 (2007-11) Lausanne Suisse.

[42] «Multimedia packet transport: multiple layers or descriptions?», Guédon, Jean-Pierre; Normand, Nicolas; **Parrein, Benoît**, *Proceedings of IEEE Packet Video workshop*, (2003-00-00) 7 p. IEEE Packet Video workshop (2003) Nantes France.

[43] «FEC comparison for multimedia wireless distribution», **Parrein, Benoît**; Chatelier, Lionel; Normand, Nicolas; Guédon, Jean-Pierre, *Proceedings of IEEE ASWN'02*, (2002-07-00) 140-144 IEEE ASWN'02 (2002-07) Paris France.

**Conférences nationales avec comité de lecture**

[44] «Une architecture hybride Client/Serveur et Pair-à-Pair pour le streaming vidéo sur l'Internet», Bouzakaria, Nassima; Ghareed, Majd; **Parrein, Benoît**; Rouibia, Soufiane, *Proceedings of the CFIP/NOTERE 2012*, (2012-10-01) pp.1 CFIP/NOTERE 2012 (2012-10-29) Bayonne France.

[45] «Towards a Hybrid Client/Server and P2P Architecture for Content Delivery over the Internet», Rouibia, Soufiane; Ghareed, Majd; **Parrein, Benoît**; Biazzini, Marco; Carvajal-Gomez, Raziel; Perez-Espinosa, Adriana; Serrano-Alvarado, Patricia, *Proceeding of the CFIP/NOTERE*, (2012-10-01) pp.1 CFIP/NOTERE (2012-10-29) Bayonne France.

[46] «SEREADMO : protocole de routage sécurisé pour réseaux ad hoc mobiles», **Parrein, Benoît**; Yi, Jiazi, *CFIP 2009*, (2009-10-12) Colloque Francophone sur l'Ingénierie des Protocoles 2009 (2009-10-12) Strasbourg France.

[47] Transmission prioritaire JPEG2000 sur lien sans fil, **Benoît Parrein**, N. Normand, P.LeCallet, GRETSI 2005, Louvain-La-Neuve, septembre 6-9, 2005.

[48] «Le LAR aux Mojettes», Déforges, Olivier; Babel, Marie; Normand, Nicolas; **Parrein, Benoît**; Ronsin, Joseph; Guédon, Jean-Pierre; Bédat, Laurent, *Proc. of CORESA'04*, (2004-00-00) 165-168 CORESA CORESA 04 - COmpression et REprésentation des Signaux Audiovisuels (2004) France.

## Brevets

[49] « Erasure coding and decoding using Mojette projections », Sylvain David, Pierre Evenou, Jeanpierre Guédon, Nicolas Normand, **Benoît Parrein**, Brevet EP 13306435.2 (2013-10-18).

[50] «Procédé de dématriçage d'image», Klossa, Jacques; Tarin, Marc; Horain, P.; Parrein, Benoît, (2004-05-15) EP04291269

[51] « Procédé sécurisé de fourniture de documents payants via un réseau de communication », Olivier Jacquier, Jeanpierre Guédon, Nicolas Normand, Simon Colombié, **Benoît Parrein**, FR2858497, Juillet 2003

**Drafts IETF**

[52] «Multi-path for Optimized Link State Routing Protocol version 2», Szwabe, Andrzej; Nowak, Adam; Baccelli, Emmanuel; Yi, Jiazi; **Parrein, Benoît,** draft-szwabe-manet-multipath-olsrv2-01.txt, (2010-11-08)

[53] «Multi-path for Optimized Link State Routing Protocol version 2», Szwabe, Andrzej; Nowak, Adam; Baccelli, Emmanuel; Yi, Jiazi; **Parrein, Benoît,** draft-szwabe-manet-multipath-olsrv2-02.txt, (2011-05-2)

[54] Multi-path Extension for the Optimized Link State Routing Protocol version 2 (OLSRv2) , Yi, Jiazi; **Parrein, Benoît,** draft-yi-manet-olsrv2-multipath-00, (2014-04-24).

[55] Multi-path Extension for the Optimized Link State Routing Protocol version 2 (OLSRv2) , Yi, Jiazi; **Parrein, Benoît,** draft-yi-manet-olsrv2-multipath-02, (2014-10-27). Accepté comme WG Document (MANET WG).

**En préparation**

CFIP 2015, IEEE Communication Letters

# Chapitre 2

# Introduction

Ce chapitre d'introduction a pour but de positionner mon travail et d'énumérer mes contributions personnelles qui sont néanmoins le fruit d'un travail collectif réalisé en grande partie au sein de l'équipe IVC du laboratoire IRCCyN, UMR 6597.

## 2.1 Positionnement de ce travail

Mon travail de recherche concerne de manière très large les domaines de l'informatique, des réseaux, du traitement des images et de la théorie des codes. En prémices de mes contributions, je positionne ce travail tant sur les aspects problématiques et outils spécifiques que vis-à-vis des communautés scientifiques existantes.

### 2.1.1 Théorie des codes correcteurs

C'est la théorie qui m'accompagne depuis mes travaux de thèse. Les articles de Rabin [75] et Albanese *et al.* [2] ont été extrêmement fondateurs dans mon activité de recherche. Un focus naturel au vue de l'historique de mon équipe de recherche s'est effectué autour des codes robustes aux effacements de données (ou codes à effacement) particulièrement pertinents dans un contexte de transmission par paquet. Pour ce type de codes, on s'intéresse uniquement à la compensation d'erreur par des mots de code (des paquets) redondants. Par hypothèse, la localisation et la correction intra-paquet sont assurées par des mécanismes extérieurs (nombreux dans les protocoles de communication).

Le sujet des codes correcteurs fait souvent appel à des techniques mathématiques très sophistiquées (interpolation, transformation de Fourier dans des corps finis, courbes elliptiques, géométrie projective...). Utiliser le formalisme Mojette dans ce contexte simplifie énormément l'approche en convenant d'un choix d'angles de projection idoines et en développant des algorithmes performants au sein d'une arithmétique modulaire mais non contrainte.

Mon travail concerne également le codage de source. La transformation Mojette répond en effet très bien à un problème bien connu en théorie de l'information depuis près de 50 ans à savoir la description multiple. Cette bizarrerie théorique fascinante est à la jonction du codage de source (qui vise la compression) et du codage canal (qui vise la protection) [30].

Je suis membre du GDR-IM (Informatique Mathématique) depuis 2012. J'ai participé en octobre de la même année aux journées Codage et Cryptographie (Journées C2) à Dinard (Ille-et-Vilaine). En juin 2014 à l'Université de Bordeaux, j'ai fait la promotion des codes à effacement (algébriques et géométriques) dans des architectures de stockage distribuées lors du Workshop ACN (Algebra, Codes and Networks), identifiée comme action COST-IC1104 [1]. Je vais refaire cette promotion dans le cadre des Rendez-Vous de la Recherche et de l'Enseignement de la Sécurité des Systèmes d'Information (RESSI), édition 2015 à l'Université Technologique de Troyes (UTT).

### 2.1.2 Traitement d'images

La transformation Mojette est une opération tomographique qui a vocation à reconstruire des images à partir de projections. La motivation première concerne l'imagerie médicale mais très vite l'attention s'est portée sur le transfert d'images naturelles [35] comme *Baboon* ou *Lena*. Cette origine explique les références multiples à des outils fortement utilisés en traitement d'images (décorrélation, quantification, codage, filtrage, convolution, morphologie mathématique, ...). Introduire la transformation Mojette en théorie de l'information et en réseaux permet d'accéder à cet arsenal mathématique classique en traitement du signal et des images.

Je suis membre du GDR-ISIS (Information Signal Image viSion) depuis 1998 et j'ai publié à de nombreuses reprises à Coresa ([62, 53, 23]) et Gretsi ([64, 60]) qui sont parmi les 2 conférences nationales de références de ce GDR. Le Thème D du GDR porte sur les aspects de transmission et de codes correcteurs (présentation invitée en octobre 2013 sur les codes à effacements pour le Cloud). En novembre dernier, j'ai présenté mes travaux à la journée inter-GDR (ISIS et SoCSiP) sur les codes correcteurs du point de vue architectures matérielles (Télécom Bretagne).

### 2.1.3 Réseaux auto-organisés

Les réseaux ne sont pas (encore) une science. Louis Pouzin définit la recherche en réseaux aujourd'hui comme de l'artisanat [2]. Je suis assez d'accord avec cette idée. En réseaux, on expérimente, on observe, on utilise des théories transposables (graphes, jeux, décision, ...), on modélise. L'avènement des réseaux virtuels et du *Sofware Defined Network* (SDN) fait exploser les concepts et amène les réseaux à proximité notamment de la science du logiciel. La science des réseaux est en marche. Elle arrive.

Les réseaux auto-organisés regroupent de manière très large : les réseaux de capteurs, les réseaux mobiles ad hoc et les réseaux P2P. Chaque type de réseaux possède sa problématique mais ils sont tous par définition autonomes. L'entrée ou la sortie d'un nœud ne suppose ni une intervention humaine ni une interruption de fonctionnement (sauf topologie extrême). En outre, ils ont une approche distribuée. Les fonctions de pérennité du réseaux, routage ou de partage sont assurées de manière décentralisées. Mon travail a porté en particulier sur la spécification d'un nouveau protocole de routage en réseau ad hoc et sur une redéfinition du modèle client-serveur et du modèle P2P en un modèle hybride.

Je suis membre du GDR-ASR (Architecture Systèmes et Réseaux) depuis 2004. Je participe régulièrement aux meetings de l'IETF depuis 2010 (action de normalisation en cours).

1. http://www.network-coding.eu/
2. Émission *La tête au Carré* sur France Inter le 18/02/2015 : des débuts de l'Internet au piratage informatique.

### 2.1.4 Stockage et système distribué

Ce dernier item possède la singularité d'avoir au départ une forte volonté industrielle en voulant valoriser notamment l'idée de 2001 portant sur le stockage distribué de projections [34]. La création en 2010 de la *spin-off* Fizians (rebaptisé récemment Rozo Systems) illustre parfaitement le processus complet de l'idée à sa valorisation. Depuis ses débuts, l'entreprise axe sa recherche et son développement autour d'un système de fichier distribué dénommé RozoFS (pour Rozo *File System*). Ce projet *open source* a permis (et permet encore) d'expérimenter des solutions dans le domaine de la tolérance aux pannes, de la réparation de nœuds, de systèmes large échelle, de calcul distribué ou encore de respect de la vie privée. RozoFS a donné lieu à de nombreuses comparaisons notamment avec le système de fichier Ceph ou encore HDFS (*Hadoop Distributed File System*). Le développement du produit s'est accompagné d'un grand nombre d'optimisations autour du code Mojette qui pourraient avoir aujourd'hui de nouvelles répercussions sur les schémas de transmission.

Ce travail m'a permis d'appréhender les systèmes large échelle notamment au travers l'usage de la grille GRID5000. Au niveau international, des contacts avec la communauté américaine sur le stockage (très fermée) progressent (lentement) via l'association Usenix principalement. Le fort potentiel économique [3] du sujet peut expliquer la frilosité des échanges.

## 2.2 Contributions personnelles

Cette section présente mes contributions principales depuis mes travaux de thèse soutenus en novembre 2001 jusqu'au projet ANR FEC4Cloud qui se termine en juin prochain.

### 2.2.1 Codes à effacement géométriques

Ces codes sont basés sur la transformation Mojette. Au cours de ma thèse, j'ai défini la notion de *buffer géométrique* qui est un support 2D semi-infini. Associés à des angles Mojette de type $(p_i, 1)$, ces buffers définissent un ensemble de codes à effacement Mojette. En 2001, je présente cette famille de code comme des codes $(1+\varepsilon)MDS$ [63]. Un peu plus tard, une reconstruction régulière et déterministe sera présentée [52].

En 2010, ces codes (en version non systématiques) sont intégrés au système de fichier distribué RozoFS. Lors du projet Sereadmo, une forme systématique simple est découverte. Ce code systématique a fait l'objet d'un dépôt de brevet lors du projet FEC4Cloud [21].

Dans le cadre de la thèse de Dimitri Pertin et du contrat post-doctoral d'Alexandre Van Kempen, ces codes sont actuellement en compétition avec les implémentations Reed-Solomon de la librairie Jerasure (présentation des résultats à ACM Eurosys en 2014 [69]) et de la librairie ISA-L d'Intel (soumission à Usenix HotStorage 2015). Dans un contexte de transmission, un comparatif avec les codes Raptor et LDPC est à l'étude.

### 2.2.2 Schéma de codage conjoint source-canal

Inspiré des travaux Albanese *et al.* [2], j'inscris le code Mojette dans un schéma de transmission prioritaire en 2001 avec deux publications en revue [34, 59]. Ce modèle générique évoluera

3. Pour exemple, la société Intank, principal contributeur du projet Ceph, a été rachetée 175 M$ par RedHat en avril 2014.

ensuite par la nature des interactions avec le codage de source ou de canal. Plusieurs sources sont ainsi testées dont le codeur rennais LAR (Local Adaptive Resolution) [4] ou encore de sources de type vidéo H.264/AVC [58]. Le schéma de transmission a été piloté pendant un temps par un critère psycho-visuel [60]. De manière indirecte, ce travail a abouti à des contributions pour des codages vidéos résilients à la perte de paquet dans le cadre de la thèse de Fadi Boulos [8, 7, 9].

Dans un contexte d'imagerie médicale, j'ai contribué lors de mon post-doc à Telecom Paris-Sud (ex-INT) sur le traitement (*demosaicking*), la compression et l'interaction d'images en biomédecine (cytologie, hématologie et anatomo-pathologie) avec une publication principale [61] et un brevet [40]. Fort de cette expérience, j'ai pu participer à l'élaboration d'un codage d'image sans perte basé sur la Mojette sous la conduite de Florent Autrusseau [3].

### 2.2.3 Protocole sécurisé de routage à chemins multiples pour réseaux MANET

La participation au projet Sereadmo (RNRT 2005) en tant que partenaire puis en tant que leader (sur la dernière année) a provoqué une immersion totale dans le domaine des réseaux et de la sécurité. Dans ce projet, l'attention est portée sur la conception d'un protocole de routage sécurisé pour réseaux ad hoc. Très vite, l'idée d'un protocole de routage multi-chemins est proposée considérant la densité des nœuds mobiles, la diversité des canaux radios et la difficulté d'une attaque coordonnée sur des routes distinctes. Trois thèses que j'ai accompagné sont issues de ce projet : Eddy Cizeron sur les aspects algorithmiques, Jiazi Yi pour le protocole et la validation et Dan Radu pour la partie service (service vidéo notamment). Une publication dans la revue *Elsevier Ad Hoc Journal* [100] (95 citations au 22/04/2015, IF : 1.943) et deux actions de normalisation auprès de l'IETF [93, 101] valorisent cette période.

Du point de vue de la sécurité, un algorithme de chiffrement à clé secrète est développé conjointement avec Éric Grall de Thalès Communication. Inspiré du standard DES, il utilise la transformation Mojette dans l'étape de substitution (on parle de $\mathcal{M}$-box). Cet algorithme est spécifié entièrement dans le Chapitre 10 du livre Mojette [32]. Un protocole de partage de secret est également proposé lors de ce projet de recherche.

### 2.2.4 Architecture hybride client/serveur et P2P

Dans un contexte régional, le projet P2PWeb porté par la société TMG (en la personne de Soufiane Rouibia) propose l'idée d'une architecture mixte client-serveur et P2P. Dans ce protocole hybride, l'idée est qu'un réseau P2P vienne suppléer un serveur de contenu lorsque celui-ci est saturé de requêtes. On démontre par une analyse QoS et QoE l'apport du modèle mixte par rapport à une approche centralisée [85, 10, 29]. Récemment, ce protocole hybride a été proposé dans le cadre de la transmission de contenus vidéos de type *live* (diffusion Internet temps réel de chaînes TNT). Ce travail fait l'objet d'une soumission au Colloque Francophone sur l'Ingénierie des Protocoles (CFIP) 2015.

### 2.2.5 RozoFS : un système de fichiers distribué pour le Cloud

RozoFS est un système de fichier distribué tolérant aux pannes dédiés au architecture de *Cloud storage* extensible. C'est une solution dite SDS (pour *Software Defined Storage*). L'idée générale provient de l'architecture distribuée proposée en 2001 dans [34]. Le code à effacement utilisé est celui de ma thèse (Chap.4 page 79 [57]). L'algorithme de reconstruction est celui

proposé dans [52]. Ce produit *open source* a nécessité ensuite plus de 3 années d'ingénierie (25 hommes-an) pour devenir mature de la part de la société Rozo Systems (ex-Fizians). Des fortes compétences en réseaux et systèmes Unix ont été nécessaires.

Ma contribution a porté sur l'accompagnement et le transfert de cette technologie jusqu'à la solution industrielle au travers du projet ANR Emergence FEC4Cloud dont j'ai la responsabilité scientifique depuis 2012 mais aussi grâce à de nombreux projets étudiants réalisés au sein de l'Université de Nantes et que j'ai pu encadré. L'entreprise Rozo Systems emploie aujourd'hui 8 personnes à plein temps, a fait une levée de fonds de 700K€ en décembre 2014 et s'est installée à San Francisco en ce début d'année 2015.

## 2.3 Plan du manuscrit

Ce manuscrit est articulé en 6 chapitres incluant ce chapitre d'introduction. Le chapitre suivant donne les notions de la transformation Mojette nécessaires ainsi que les définitions et propriétés des différents codes à effacement utilisés dans les chapitres d'applications.

Le chapitre 3 porte sur l'usage des codes à effacement dans des architectures distribuées de type stockage Cloud. L'approche est comparée avec la méthode par réplication de manière théorique et pratique avec des mesures de débits (en Go/s). Ce chapitre définit plus clairement l'intégration du code au sein du système de fichier distribué RozoFS. La solution est comparée avec le projet *open source* Ceph [4]. Il se termine par une incursion dans le domaine du *Big Data* en vérifiant la compatibilité des codes à effacement avec l'algorithme distribué MapReduce. Ce chapitre porte sur les résultats principaux du projet ANR FEC4Cloud.

Le chapitre 4 présente un protocole hybride client-serveur et distribué de type P2P : le protocole P2PWeb. Outre les spécifications du protocole, une analyse QoS (Qualité de Service) et QoE (Qualité d'Usage) est appliquée dans le cadre d'un partage d'un fichier image. Des optimisations sont également proposées et évaluées. Ce chapitre porte sur les résultats principaux du projet Région (Pays de la Loire) P2PWeb.

Le chapitre 5 présente le protocole de routage MP-OLSR qui est un protocole à chemins multiples pour réseaux mobiles ad hoc (MANET) proposé dans le cadre de la thèse de Jiazi Yi. Des résultats de simulations et d'expérimentations réalisées sur le campus de Polytech Nantes (site de la Chantrerie) sont présentés dans le cadre d'un transfert de fichier et d'un service de vidéo à la demande qui utilise le code à effacement Mojette. Ce chapitre porte sur une partie des résultats du projet Sereadmo.

Le dernier chapitre de conclusion résume le document et apporte les perspectives de recherche.

Une sélection de quatre articles (dont une soumission 2015) est donnée en fin de manuscrit (page 98).

Ce manuscrit comporte également une table des figures et un index.

4. http ://ceph.com/

# Chapitre 3

# Codes à effacement par transformation Mojette

## 3.1 Introduction

La transformation Mojette est une transformation de Radon discrète exacte. L'usage de la transformation Mojette en télécommunication est apparu dès 1996 [50]. L'idée est alors de contrôler la redondance en fonction de l'importance des données d'une image (hiérarchie fournie par la transformation en cosinus discrète). L'adéquation "projection Mojette" et "unité de transport" est proposée pour la première fois. On parle alors de codage conjoint source-canal pour sa relation intrinsèque avec le codage d'image. Le schéma de transmission est appliqué à un réseau à qualité de service niveau 2 de type ATM (*Asynchronous Transfert Mode*).

Dans [5], l'auteur proposait d'associer également la théorie des codes avec une transformation de Radon finie. Mais ce rapprochement a pour but uniquement de ramener le problème de la reconstruction de Radon a un problème bien posé en utilisant notamment des algorithmes éprouvés en code correcteur comme le décodage par maximum de vraisemblance. L'usage d'une transformation de Radon dans un schéma de transmission demeure une idée originale depuis près de 20 ans.

Les codes robustes à l'effacement (*erasure resilient codes* en anglais ou codes à effacement) s'appliquent à des erreurs en rafale. Le canal de Gilbert-Elliott des années 1960 modélise grossièrement ce comportement par un processus de Markov à deux états [27]. La correction concerne ici des erreurs symboles consécutives, localisables classiquement par des méta-données de type numéro de séquence ou de segment. Les codes à effacement sont donc bien adaptés à la transmission par paquet. Cette extension aux réseaux de paquet suppose des mécanismes locaux de correction intra-paquet (type CRC ou autre). On suppose en effet que les codes (ou paquets) sont reçus sans erreur.

Les codes de Reed-Solomon [81] sont les codes à effacement les plus connus. Ils sont proposés initialement pour des erreurs symboles en rafale dans des alphabets autre que $GF[2]$ ($GF[2^8]$ principalement). Ces codes sont optimaux au sens où il faut $k$ symboles parmi $n$ mots de codes pour pouvoir décoder $k$ symboles de message. Ces codes sont appelés MDS pour *Maximum Distance Separable*. MacWilliams et Sloane décrivent dans leur ouvrage [45] au Chapitre 11 les codes MDS comme les plus fascinants de la théorie des codes. Le terme *Maximum Distance*

provient de la distance de Hamming maximale qui assure l'optimalité du code. Le terme *Separable* provient de la séparation entre les symboles de message et les symboles de contrôle. MacWilliams et Sloane ajoutent que n'importe quels $k$ symboles (parmi $n$) peuvent être des symboles de message dans le cas particulier des codes MDS. Les termes *systématique* et *non systématique* peuvent donc s'appliquer aussi aux codes MDS pour spécifier la structure du code.

Il faut attendre 1989 pour que Rabin [75] inscrive les codes MDS dans un contexte de transmission et de stockage distribué (même s'il ne fait pas explicitement allusion aux codes MDS). Dans ce papier de référence, un code (déterministe) qui s'appuie sur une matrice de Cauchy non systématique est proposé. La propriété de cette matrice est que toutes les sous-matrices sont inversibles. Un autre code (probabiliste), basé sur des coefficients aléatoires, est également proposé. Ce formalisme est réutilisé dans [2] dans le cadre d'une transmission prioritaire avec des matrices de Cauchy systématiques. Un peu plus tard, Rizzo [84] puis [42] fourniront un codage Reed-Solomon basé sur des matrices de Vandermonde qui possèdent les même propriétés que les matrices de Cauchy mais qui étaient plus rapides à inverser au moment de leur parution. Plus récemment, les travaux de Planck ont remis les matrices de Cauchy au premier plan [72].

Dans son formalisme actuel, le code à effacement Mojette arrive en 2001 [57]. L'adéquation entre projection et mot de code est l'idée élémentaire pour construire ces codes géométriques. La section suivante porte sur un rappel des principaux résultats de la transformation Mojette.

## 3.2 Rappels sur la transformation Mojette

Ces rappels sont extraits principalement des premiers chapitres théoriques du livre Mojette [32]. Ils ont pour but de faciliter la compréhension des codes à effacement Mojette proposés dans la section suivante.

### 3.2.1 Les origines de la transformée

La transformée Mojette a 20 ans. Nous avons célébré cet anniversaire le 5 février dernier à Polytech Nantes [1]. La transformation Mojette est née en 1995 avec la publication de Jeanpierre Guédon *et al.* [33]. Elle est définie comme une transformation de Radon discrète et exacte. Contrairement à sa grande sœur de 1917 [76] (réédité récemment dans [77]) qui est continue, la transformation Mojette nécessite un nombre fini de projections pour reconstruire une image de manière exacte.

L'origine du nom de la transformée fait référence aux haricots de Vendée (*beans* en anglais) et au fait que l'on appelle de manière homonymique *bins* les éléments d'une projection. Accessoirement aussi, on utilisait ces fameux haricots dans cette région du sud de la Loire pour compter ou pour faire des mises aux cartes (distribuées régulièrement autour d'une table ronde par exemple). Jeanpierre Guédon est intarissable sur ces questions d'origine de la Mojette.

Outre ses fonctions tomographiques, utilisées en imagerie médicale, la transformation Mojette possède de multiples applications aujourd'hui. Citons en particulier les applications en codage de source, en chiffrement et tatouage d'image, en analyse (code-barres et QR-codes), en stockage distribué, en réseaux de paquets et en codage canal. Le jeu Mojette est également disponible en ligne sur l'Internet [2].

1. voir le programme à l'adresse http://www.mojette.org/event/mojette-day/
2. http://www.mojette.net

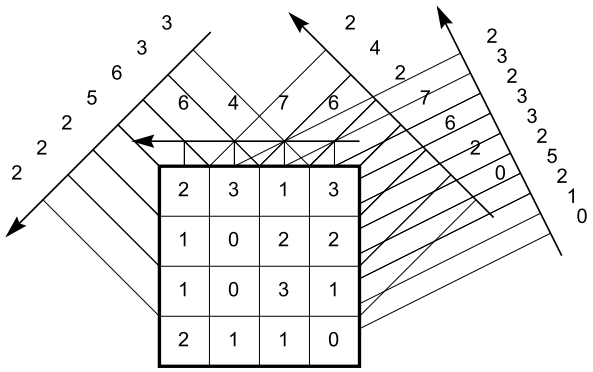



FIGURE 3.1 – Transformation Mojette d'une image $f$ avec une série de projections discrètes $S_4 = \{(0,1),(1,1),(2,1),(-1,1)\}$ (extrait du Chapitre 3 du livre Mojette [32]).

Plusieurs transformations ont été définies en fonction de la nature du pixel. La transformée Mojette-Dirac (définie ci-après) est la plus connue pour son application sur des pixels de type fonction de Dirac pondérée. Mais il existe aussi des modèles de pixels alternatifs comme des fonctions spline d'ordre 0 (la Mojette-Haar) à $n$, ou comme des transformations Mojette 3D (appliquées sur des voxels) ou $nD$ (voir le Chapitre 3 du livre Mojette [32] pour plus de détails à ce sujet).

### 3.2.2 Transformée Mojette-Dirac

La forme générale de la transformation Mojette-Dirac est donnée à l'équation 3.1.

$$[\mathcal{M}f](b,p_i,q_i) = \mathrm{proj}_{p_i,q_i}(b) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} f(k,l)\,\Delta(b+kq_i-lp_i)\,, \tag{3.1}$$

où $\Delta(m)$ est la fonction de Kronecker avec $\Delta(0)=1$ et $\Delta(m)=0$ sinon.

L'équation $b+kq-lp$ représente la ligne de projection à savoir la série des pixels projetés. $\Delta(b+kq-lp)$ est par conséquent égal à 1 pour ces pixels. La Figure 3.1 illustre le résultat de la transformation Mojette sur une image $4\times 4$ pour 4 directions de projections qui (dans le sens horaire) sont respectivement $(0,1),(1,1),(2,1)$ et $(-1,1)$ soit l'ensemble

$$S_4 = \{(0,1),(1,1),(2,1),(-1,1)\}.$$

Les projections peuvent être obtenues non seulement par addition mais par n'importe quelle opération linéaire. L'opération modulo peut être également utilisée pour calculer les projections. Les Mojette XOR seront particulièrement prisées dans le cas d'application des codes correcteurs.

Au sujet de la forme générale Mojette de l'équation 3.1, les remarques suivantes peuvent être faites :

1. les entiers $p_i$ et $q_i$ sont premiers entre eux ;

**Algorithm 1:** Mojette-Dirac d'une image $f(k,l)$ avec un support rectangulaire $P \times Q$ (extrait du Chapitre 3 du livre Mojette [32])

**Input**: image $f(k,l)$, size $P \times Q$ of $f$ support, set of projection angles $S_I = \{(p_i, q_i), 1 \leq i \leq N\}$
**Output**: projection values $\text{proj}(b, p_i, q_i)$

```
Set the (0,0) position of f(k,l) at the bottom left corner;
foreach projection index i of S_I do
    B(i) ← (Q−1)|p_i| + (P−1)q_i + 1;
    initialize vector proj of length B(i);
    foreach pixel (k,l) do
        proj(−q_i k + p_i l) ← proj(−q_i k + p_i l) + f(k,l);
```

2. la transformation est linéaire ;
3. le vecteur $\text{proj}_{p_i,q_i}$ possède un nombre de bins correspondant à $B(i)$ où :

$$B(i) = (Q-1)|p_i| + (P-1)|q_i| + 1 \ , \tag{3.2}$$

pour une image de dimensions $P \times Q$ ;

4. l'ordre de complexité de l'algorithme 1 est $O(I)$ pour chaque calcul d'une projection. Pour une série de $N$ projections, la complexité totale est de $O(IN)$ ;
5. l'algorithme 1 est toujours valable même si le support initial est non rectangulaire.

### 3.2.3 Une transformation redondante

Le nombre de bins total $\sum_{i=1}^{N} B(i)$ est classiquement supérieur au nombre de pixels $I = P \times Q$ pour pouvoir reconstruire l'image d'origine. Ce taux de redondance, *Red*, est calculé comme suit :

$$\text{Red} = \frac{\sum_{i=1}^{N} B(i)}{I} - 1. \tag{3.3}$$

**Quand la redondance est négative,** l'ensemble de projections correspondantes n'est manifestement pas suffisant pour permettre la reconstruction.

**Une redondance égale à zéro** survient quand le nombre de bins égale le nombre de pixels. On parle dans ce cas de projections dites dégénérées (avec un pixel par *bin*) qui sont du type $(1, P)$ ou $(Q, 1)$ pour un support rectangulaire de dimensions $(P \times Q)$. C'est en fait une sérialisation.

**La redondance est positive** lorsqu'un ensemble reconstructible de projections non dégénérées est choisi. Une redondance positive ne garantit pas nécessairement la reconstruction (voir section suivante sur les conditions de reconstruction).

Outre le dénombrement des bins par rapport au nombre de pixels de l'image, les projections entre elles possèdent une très grande corrélation. Un élément commun est par exemple la somme

de l'image qui se trouve dans la somme des bins de chacune des projections. Cette forte corrélation est exploitée dans le cadre de la compression sans perte d'image proposée notamment dans [3] où un codage prédictif intra et inter projection est réalisé.

Dans un schéma de transmission, la transformation Mojette permet toujours de calculer une projection supplémentaire unique à l'image des codes à débits variables (ou *rateless code* [44]). L'accroissement linéaire de la taille provoqué par l'équation 3.2 peut être contourné de manière théorique par l'usage d'une plus grande dimension (par un changement de corps fini ou par l'usage de la Mojette $3D$ ou $nD$) ou de manière pratique en supprimant des bins (redondants) qui ne sont pas utilisés lors de la reconstruction. Ce calcul de projections supplémentaires va permettre d'adapter la redondance en fonction de l'importance de l'information et des caractéristiques du canal. Il est possible de calculer ces projections sans passer par la reconstruction du support 2D en appliquant des séries de convolutions 1D sur les projections existantes [68].

### 3.2.4 Critères de reconstruction d'une forme convexe

On distingue deux critères en fonction de la forme du support. Le critère de Katz [39] est dévolu au support rectangulaire. Le critère de reconstruction complète de Nicolas Normand [51] porte de manière plus générale sur les formes convexes. Ces deux critères sont détaillés dans ce qui suit.

#### Critère de Katz

Le problème mal-posé de la transformation de Radon [76] a été résolu géométriquement par Myron Katz [39]. La condition de reconstructibilité, au sens de Katz, s'exprime comme suit :

Soit une série de pixels dans un tableau rectangulaire $(P \times Q)$ et une série de projection $S_N$ de $N$ directions de projection données par $S_N = \{(p_i, q_i), 1 \leq i \leq N\}$ avec $(|q_i| > 0)$, on définit $P_N$ comme $P_N = \sum_{i=0}^{N-1} |p_i|$ et $Q_N$ comme $Q_N = \sum_{i=0}^{N-1} |q_i|$.

**Lemme 3.1** *Si* $(P_N \geq P)$ *ou* $(Q_N \geq Q)$ *alors une image unique de dimension* $(P \times Q)$ *est reconstructible par la série de projections* $S_N$.

La preuve de ce lemme est donnée dans [39].

#### Critère de reconstruction complète de Normand

Les conditions de reconstruction de Normand [51] ont été démontrées pour n'importe quelle image convexe en associant un vecteur de projection $(p, q)$ avec un Élément Structurant à 2 Pixels (ES2P) comme couple de points $\{O, (p, q)\}$. Il a été démontré que la rétro-projection correspond en fait à l'ouverture morphologique (érosion puis dilatation) du support, avec l'ES2P du vecteur de projection. Par extension, ceci signifie aussi que la reconstruction d'image est associable à une série d'ouvertures morphologiques définies par les ES2P de l'ensemble des directions de projection. C'est ainsi que l'on peut étendre le critère de Katz énoncé plus haut et le généraliser à n'importe quel support convexe en utilisant des dilatations. Une image convexe est reconstructible si et seulement si le résultat des dilatations successives par les ES2P n'est pas inclus dans le support de l'image (même si ceci concerne un seul pixel). Le résultat général peut être exprimé par le théorème suivant avec équivalence des 2 énoncés.

**Théorème 3.2 Reconstructibilité complète**

**Énoncé 1.** *Les 2 propositions sont équivalentes :*
*i) $f(k,l)$ défini sur le convexe $G$ est reconstructible par $\{\text{proj}_{p_i,q_i}, 1 \leq i \leq N\}$ ;*
*ii) $R$ construit par une série de $N$ dilatations $\{O,(p_i,q_i), 1 \leq i \leq N\}$ n'est pas inclus dans $G$.*

**Énoncé 2.** *Les 2 propositions sont équivalentes :*
*i) $G$ est reconstructible par $\{\text{proj}_{p_i,q_i}, 1 \leq i \leq N\}$ ;*
*ii) l'érosion de $G$ par $R$ est nulle.*

La preuve de ce théorème est donné en détails dans [51].

Il faut noter que malgré le non respect des deux critères énoncés plus haut, il est possible d'envisager une reconstruction partielle dans des cas de corrélation spatiale entre les pixels voisins au sein d'une image (voir travaux d'Olivier Phillipé [70]). Cette reconstruction partielle ne s'applique plus lorsqu'on évoque des éléments d'information sans corrélation spatiale.

### 3.2.5 Choix des angles de projections

Outre, le fait que la série de projections doit satisfaire les critères de reconstruction énoncés à la section précédente, il est naturellement légitime de vouloir connaître quelle série est optimale au vue du nombre de combinaisons possibles.

La réponse dépend bien de l'application et de la fonction affectée à une projection Mojette. On détaille ici deux applications : en reconstruction tomographique, et en codage canal.

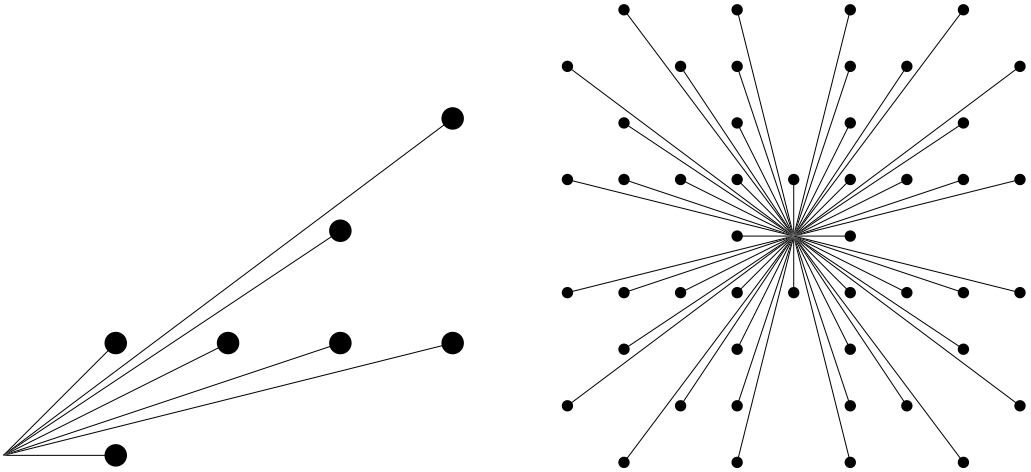
**Pour la reconstruction tomographique**

En reconstruction tomographique (traitée notamment dans les travaux de thèse de Myriam Servières[86]), il est intéressant de chercher à maximiser cette redondance pour décrire une surface ou un volume. C'est le cas notamment de la reconstruction par rétroprojection filtrée exacte (algorithme exact FBP). Dans ce contexte, les séries (ou suites) de Farey fournissent tous les angles discrets possibles sur un espace compact. La Figure 3.2 illustre une série de Farey d'ordre 4 au sein du premier octant $[0, \frac{\pi}{4}[$ ainsi que sur l'intervalle $[0, \frac{\pi}{4}[$ sous la forme d'angle $\frac{q}{p}$. Les séries de Farey d'ordre $N$ contiennent toutes les fractions irréductibles arrangées en ordre croissant entre 0 et 1 dont le dénominateur n'excède pas $N$. Par exemple, $F_4$ est la série $\{\frac{0}{1}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{1}{1}\}$.

**Pour le codage canal**

Dans un contexte de codage canal déterministe i.e non probabiliste, il est intéressant de veiller à ce que toutes les projections aient le même pouvoir de reconstruction. En effet, dans un code Mojette déterministe, un nombre reçu de $K$ projections permettra toujours de reconstruire le support initial quel que soit le motif de perte et l'ordre de réception. L'idée du paramètre d'angle $q$ constant répond à ce besoin d'équivalence des projections.

Dans un contexte de télécommunication, il peut être intéressant de chercher à minimiser la redondance. Nous allons satisfaire ici une des deux conditions du critère de Katz (voir lemme 3.1 plus haut). Avec $q = 1$, la redondance est minimisée car dans ce cas, on sélectionne de fait l'ensemble des entiers relatifs consécutifs autour de la valeur nulle pour le paramètre $p$ (qui



(a) Série de Farey d'ordre 4 ($F_4 = \{\frac{0}{1}, \frac{1}{4}, \frac{1}{3}, \frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{1}{1}\}$).

(b) Par symétrie, tous les angles discrets $(p, q)$ au sein d'une image $5 \times 5$.

FIGURE 3.2 – Représentations graphiques de la série de Farey $F_4$ (extraites du Chapitre 1 du livre Mojette [32]).

fait malheureusement augmenter linéairement la taille des projections comme l'indique la relation 3.2).

Avec $q = 1$, le critère de Katz se réduit à une égalité pour $K$ projections reçues à $\sum_{i=1}^{K} q_i = K$. On peut donc réécrire le lemme 3.1 (de la page 29) pour ce cas particulier.

**Lemme 3.3** *Un support de hauteur $Q = K$ est reconstructible par exactement $K$ projections d'angles Mojette $(p_i, 1)$ quelle que soit la largeur $P$ du support.*

Il est possible naturellement de satisfaire l'autre condition de Katz (à savoir $P_N \geq P$). Dans ce cas, le choix des angles devient $p = 1$ et $q \in \mathbb{Z}$ avec des valeurs consécutives en fonction de la largeur du support. La transposition est immédiate.

Cette proposition de $q = 1$ est encore aujourd'hui utilisée dans beaucoup d'applications de stockage et de transmission utilisant la Mojette. L'usage de la transformation Mojette dans un contexte de codage de source (voir [3] notamment) ou encore dans un schéma de chiffrement symétrique (voir la section 10.4.2 du Chapitre 10 du livre Mojette [32]) fera ressortir un choix d'angles différent.

### 3.2.6 Algorithmes de transformation Mojette inverse

La conception d'algorithmes de Mojette inverse a été très prolifique au cours des vingt dernières années. En effet, on ne compte pas moins d'une dizaine d'algorithmes répertoriés dans le livre Mojette [31]. Cette section se limite à la description de deux d'entre eux à savoir la reconstruction itérative standard et la reconstruction géométrique déterministe. Cette dernière version est particulièrement adaptée au contexte des codes à effacement de la section 3.3. Dans les deux cas, on supposera l'usage de projections non-bruitées à savoir que les projections sont reçues sans erreur.

### Reconstruction locale itérative

L'écriture de la transformation Mojette directe sous forme d'un système linéaire est héritée notamment des travaux d'Olivier Philippé [70] (voir en particulier l'exemple simple de la section 4.3.3 page 96). Inverser les projections Mojette est donc équivalent à inverser le système linéaire suivant :

$$\mathcal{A}X = B$$

où $B$ sont les données projetées, $\mathcal{A}$ est la matrice de projection et $X$ sont les pixels que l'on souhaite reconstruire. Cependant, les dimensions du système sont trop larges pour une inversion efficace. La reconstruction se fera donc sous forme itérative.

L'algorithme de reconstruction (algorithme 2) est présenté ci-dessous. Il exploite le fait que le nombre de pixels contenus dans chaque bin n'est pas constant. Dans certain cas, un bin peut correspondre à un pixel unique. Ce bin particulier peut être directement rétro-projeté sur le support original de manière à initier la reconstruction. Cette même valeur rétro-projetée est ôtée de la valeur des autres bins situés dans les autres projections. Cette mise à jour peut faire apparaître potentiellement d'autres correspondances univoques bin-pixel propageant ainsi la reconstruction des bords de l'image vers le centre. De manière itérative donc, on procédera à trois opérations successives jusqu'à reconstruction complète :

1. trouver les correspondances univoques bins-pixel ;
2. rétro-projeter la valeur du bin ;
3. Mettre à jour l'ensemble de projections de manière à faire apparaître de nouvelles correspondances univoques.

Deux sous-problèmes restent non résolus dans l'algorithme 2. Premièrement, nous devons localiser le bin dans la projection qui peut être rétro-projeté *i.e.* en correspondance univoque. Deuxièmement, nous avons à déterminer lequel des pixels $(k, l)$ sur la ligne de projection $b = q_i k - p_i l$ est à reconstruire. Pour ce faire, il va falloir collecter des *meta*-données liant la forme et la projection pour entreprendre la Mojette inverse.

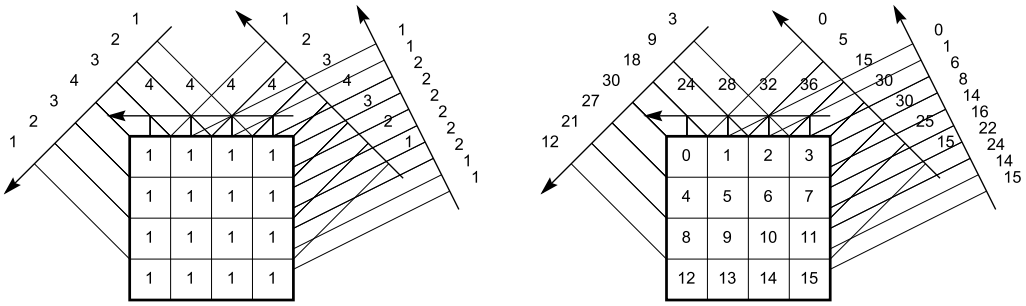
Pour illustrer la phase d'initialisation et le déroulement de l'algorithme 2, nous allons prendre exemple sur les 4 projections Mojette calculées en Figure 3.1 (en page 27). Cet exemple vérifie le critère de Katz. Il est donc reconstructible.

Une méthode simple permet de résoudre les deux problèmes de localisation par l'usage de deux images de compteurs qui vont être projetées puis reconstruites de la même manière que

**Algorithm 2:** Reconstruction Mojette itérative (extrait de [32] page 87).

**Input**: $n$ : pixel count, $N$ : number of projections, $\mathcal{M}_{p_i,q_i} f, \forall i \in [1..N]$ : projections
**Output**: $f(k, l)$ : pixel value of the reconstructed image for each pixel $k, l$

```
while ∃ b, i, a reconstructible bin in M_{p_i,q_i} do
    Find the unique pixel (k, l) projected onto [M_{p_i,q_i} f](b);
    f(k, l) ← [M_{p_i,q_i} f](b);
    Remove the contribution of pixel (x, y) in all projections
    for i ← 1 to N do
        b ← q_i × k − p_i × l;
        [M_{p_i,q_i} f](b) ← [M_{p_i,q_i} f](b) − f(k, l);
```



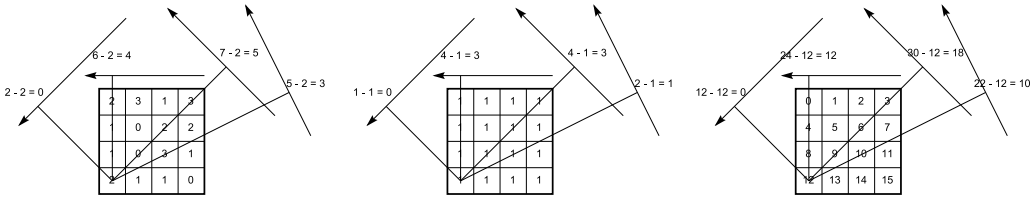

(a) (b)

FIGURE 3.3 – Transformation Mojette de (a) l'image unitaire $f_1$ et de (b) l'image des index uniques $f_i$ (extrait de [32]).

l'image inconnue. La première image possède des pixels de valeur 1. Elle se réfère à l'*image unitaire*. Elle est notée $f_1$ avec $f_1(k,l) = 1, \forall(k,l)$. La seconde image est l'*image des index*, notée $f_i$, dont les pixels sont étiquetés par un index unique, classiquement dans un ordre lexicographique e.g. $f_i(k,l) = k + lP$ où $P$ est la largeur de l'image [26]. On appelle cet indice unique le *Dietmar* en souvenir de son inventeur, Dietmar Eggemann. La projection de ces deux images va permettre respectivement de déterminer les correspondances univoques ainsi que la localité du pixel à rétro-projeter à chaque itération. Un exemple de ces deux images et de leurs projections correspondantes est illustré à la Figure 3.3 dans le cas d'une image $4 \times 4$.

La rétro-projection d'un bin, la mise à jour des projections et des 2 images de compteurs sont illustrées en Figure 3.4. À l'issue de la reconstruction, l'ensemble des projections dévolues aux trois images doivent contenir uniquement des valeurs de bins nulles.

La complexité de l'algorithme de la reconstruction locale itérative est identique à la transformation Mojette directe à savoir elle est en $O(IN)$ où $I$ et $N$ sont respectivement le nombre de pixels et le nombre de projections. Cette complexité ne prend pas en compte l'application dans sa phase d'initialisation des deux transformations Mojette directe pour obtenir les projections des images $f_1$ et $f_i$.




(a) (b) (c)

FIGURE 3.4 – Mise à jour des trois transformations $\mathcal{M}_{-1,1}f$, $\mathcal{M}_{-1,1}f_1$ et $\mathcal{M}_{-1,1}f_i$ après reconstruction du pixel $(0,0)$ situé en bas à droite de l'image $4 \times 4$ (extrait de [32]).

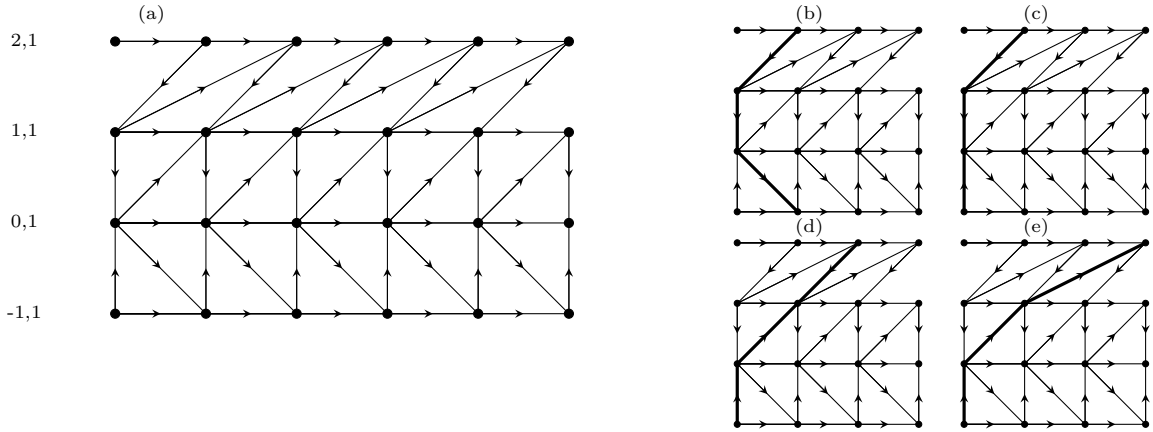



FIGURE 3.5 – (a) Graphe de dépendance inter-pixelique pour une image de hauteur 4 ($Q = 4$). (b-e) Les 4 graphes possibles de reconstruction (les parcours sont représentés en gras) (extrait de [52]).

### Reconstruction géométrique déterministe

Pour cette méthode de reconstruction [52], on suppose que $\sum_{i=1}^{N} q_i = Q$. Selon le critère de Katz défini à la section 3.1, cette condition est suffisante pour assurer la reconstructibilité. En seconde hypothèse, toute autre projection redondante est ignorée. Ces deux hypothèses impliquent que cet algorithme peut reconstruire des images de dimensions infinies à la volée. Seule la hauteur, le paramètre $Q$, doit être finie, alors qu'il n'y a pas de restriction sur la largeur $P$.

Quand une image est reconstruite en utilisant la méthode standard itérative présentée à la section précédente, la reconstruction débute au niveau des coins de l'image pour ensuite se propager vers la droite. Une fois l'étape de reconstruction des coins effectuée, il est à noter que les projections et les lignes de l'image sont intimement liées : la même projection est utilisée pour reconstruire le prochain pixel sur la même ligne. Cet algorithme propose d'exploiter cette régularité qui confère à cette méthode de reconstruction un caractère hautement déterministe lié à la géométrie du support et des projections. La description qui suit se limite au cas où les $q_i$ sont constants à 1.

**Le cas où $q_i = 1$ pour $i \in \mathbb{Z}_Q$.** Si l'ensemble des projections est trié par $p_i$ *i.e.* $p_0 < p_1 < \ldots$ et la reconstruction réalisée de la gauche vers la droite, alors la ligne $r$ de l'image, $f(k, r)$ pour $k \in \mathbb{Z}_P$, est reconstruite par la projection Mojette $\mathcal{M}_{p_r,1} f$ (la preuve de cette propriété est donnée dans [52]).

Donc, seule la ligne $Q - 1$ peut être reconstruite par $\mathcal{M}_{p_{Q-1},1} f$. Puisque la reconstruction nécessite que seulement un pixel reste inconnu sur la ligne de projection, $\mathcal{M}_{p_{Q-1},1} f$ ne peut pas être utilisé pour reconstruire une autre ligne du support. Cette preuve peut être répétée pour montrer que $\mathcal{M}_{p_{Q-2},1} f$ reconstruit la ligne $Q - 2$ et ainsi de suite jusqu'à $\mathcal{M}_{p_0,1} f$.

Puisque chaque projection est affectée à une ligne de l'image, un graphe de dépendance peut être réalisé pour montrer les relations entre projections dans le processus de reconstruction. La Figure 3.5(a) montre ce graphe pour une image de hauteur 4 lignes avec les projections Mojette $(2, 1), (1, 1), (0, 1)$ et $(-1, 1)$. Ici, les nœuds correspondent à des pixels et les arc orientés représentent les dépendances entre chaque pixel dans le processus de reconstruction. Les Figures 3.5(b)

à (e) représentent les 4 chemins possibles de reconstruction en fonction de l'ordre de reconstruction retenu. Un décalage de ces chemins de 1 pixel vers la droite propage la reconstruction de l'image.

C'est sur la base de cet algorithme que les ingénieurs de Rozo Systems (Didier Féron en particulier) vont mettre en œuvre une transformation Mojette directe et inverse ultra-rapide en réduisant drastiquement le nombre de lectures et d'écritures permettant d'atteindre des débits d'encodage supérieurs à 10 Go/s. Cette implémentation très rapide fait l'objet d'un brevet [22]. Les performances seront présentées dans le prochaine chapitre en comparaison avec des implémentations de codes Reed-Solomon dans le cadre du stockage distribué (voir page 51 et page 146).

## 3.3 Codes à effacement par transformation Mojette

Cette section présente les différents codes à effacement Mojette dans un cadre MDS avec une comparaison avec les approches algébriques de type Reed-Solomon.

### 3.3.1 Définitions

Blömer *et al.* [6] fournissent une définition générale des code à effacement **MDS** qui est ici traduite et légèrement adaptée à notre système de notations :

DÉFINITION 3.1.– *Un code à effacement, spécifié par un quadruple $(k, n, b, r)$, est une fonction $E$ qui associe les messages $M = (M_1, \ldots, M_k)$ de $k$ paquets de taille $b$ à un encodage de type $E(M) = (E_1(M), \ldots, E_n(M))$ de $n$ paquets de taille $b$, tel que n'importe quelles séries de $r$ paquets indicés $i_j$ soit $E_{i_1}(M), \ldots, E_{i_r}(M)$ de $E(M)$ déterminent de manière unique le message $M$. Le code est dit Maximum Distance Separable (MDS) si et seulement si $r = k$.*

Cette définition générale couvre les constructions **systématiques** et **non systématiques**. Dans le cas systématique, $k$ termes parmi $n$ de $E(M)$ sont issues d'une matrice génératrice comportant une sous-matrice identité et sont donc une simple copie des $k$ termes du message $M$.

Dans certain cas, il peut être intéressant de relâcher légèrement l'optimalité des codes MDS. En effet, pour le décodage, l'inversion matricielle est classiquement une opération qui possède une complexité cubique. La complexité peut être réduite dès que le nombre de paquet nécessaire est légèrement supérieur à $k$. On parle de code $(1+\varepsilon)$MDS lorsque $r = (1+\varepsilon)k$.

En théorie des codes, les codes sont dits **déterministes** lorsqu'à partir d'un certain seuil de réception de $k$ paquets (ou $(1+\varepsilon)k$ paquets), permettent de décoder le message de manière certaine. Par opposition, il existe des codes dits **probabilistes** qui à partir d'un seuil de $k$ paquets (ou $(1+\varepsilon)k$ paquets) permettent de décoder avec une probabilité (en général élevée) le message d'origine. Ce cadre statistique offre le plus souvent une réduction des complexités algorithmiques. Rabin [75] d'une part et Albanese *et al.* [2] d'autre part utilisent ces deux formes de codes pour construire leurs systèmes. Les codes Reed-Solomon sont par définition déterministes. Les codes *Fountain* [11], *Tornado* [43] ou encore les codes *Raptor* [88] sont des exemples de codes probabilistes.

Le passage de la transformation Mojette au code à effacement est immédiat. Les projections Mojette sont ici les mots de codes. Les projections redondantes permettent de compenser celles qui sont perdues. Pour assurer un caractère déterministe à la reconstruction (au sens du nombre de projections reçues), le code à effacement se restreint à des valeurs de $q$ constantes (le plus souvent égale à 1). De cette manière, la réception d'un nombre suffisant de projections permettra

toujours d'assurer la reconstruction au sens du critère de Katz (lemme 3.1) réécrit dans le cas où $q = 1$ (lemme 3.3).

Pour désigner le support de données à projeter, on parlera de **buffer géométrique**. Ce buffer géométrique de forme variable agit à la manière d'un entrelaceur. Sa dimension aura un impact sur les délais d'encodage et de décodage. Les données ne sont pas exclusivement des données images mais s'appliquent à tout élément d'information numérique. On utilisera parfois le terme d'**ixels** pour désigner ces éléments d'information d'une longueur binaire quelconque.

### 3.3.2 Code Mojette non systématique

Cette forme de code est la plus simple en matières de construction et de décodage puisque c'est l'application directe de la transformation Mojette telle que définie à l'équation 3.1 en page 27. En version non-systématique, l'opérateur Mojette est utilisé pour obtenir la totalité des mots de code. En supposant la forme triviale rectangulaire du buffer géométrique, le critère de Katz (lemme 3.1) est utilisé pour établir les conditions de reconstruction au décodage. On définit ce premier code de la manière suivante :

DÉFINITION 3.2.– *Un code à effacement Mojette non systématique, spécifié par le triplet* $(K, N, B(i))$*, est une fonction* $\mathcal{M}$ *qui transforme une image* $f$ *constituée de* $K$ *lignes de dimension* $P$ *en une série de* $N$ *projections* $proj_{p_1,q_1}, \ldots, proj_{p_N,q_N}$ *de taille variable* $B(i)$*, telles que* $K$ *projections parmi* $N$ *permettent de reconstruire de manière unique et déterministe l'image* $f$*.*

En pratique, on prendra $q = 1$ et $p$ une série d'entiers relatifs autour de la valeur nulle comme défini dans la section 3.2.5 sur le choix des angles. Par la suite, on désignera par $Mojette(K, N)$ un code à effacement Mojette où $K$ projections parmi $N$ sont nécessaires pour reconstruire.

#### Exemple du code Mojette(3,5) non systématique

La construction du code non systématique Mojette est illustrée à la Figure 3.6. L'exemple porte sur un code $Mojette(3, 5)$, où un total de 5 projections sont calculées selon la série d'angles $S = \{(-2, 1), (-1, 1), (0, 1), (1, 1), (2, 1)\}$ qui est centrée autour de la projection $(0, 1)$. Le code est binaire (définit dans $GF(2)$) dans le sens où les additions Mojette s'effectuent *modulo* 2. Dans ce schéma, une ligne sépare le bloc de messages à coder (en haut) du bloc de mots de codes redondants (en bas). Pour rappel, $b$ désigne l'indice du bin. Le critère de Katz est vérifié dès réception de 3 projections parmi 5. Ce code est donc tolérant à la perte de 2 projections parmi 5. Cette tolérance est vérifiée même si la taille du buffer géométrique augmente dans le sens de la longueur (dimension $P$).

#### Décodage

Le décodage peut débuter dès réception d'une première projection. La reconstruction complète est assurée dès lors que le critère de Katz est satisfait. Tout algorithme de Mojette inverse peut être employé ici. L'algorithme proposé dans [52] (et rappelé dans la section 7 page 34) est particulièrement adapté à ce décodage avec $q = 1$.

#### Propriétés

Dans sa formulation, le code Mojette peut paraître MDS au sens de la définition 3.1. En effet :

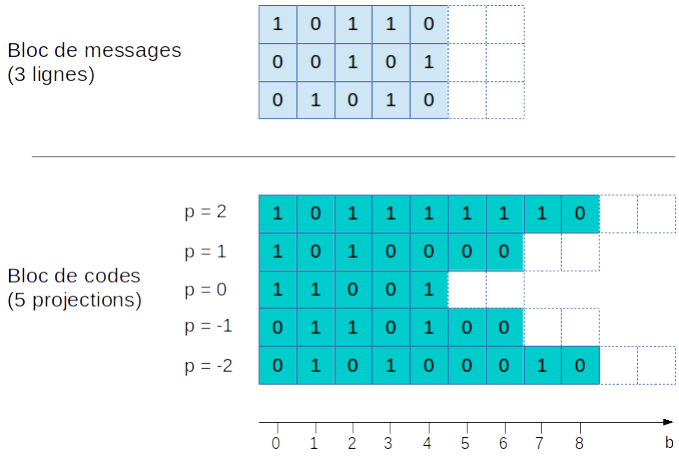



FIGURE 3.6 – Construction du code binaire Mojette(3,5) non systématique : le buffer géométrique d'entrée possède 3 lignes et 5 projections sont calculées en sortie de codage.

- un total de $N$ projections sont calculées ;
- $N - K$ projections sont redondantes ;
- il suffit de $K$ projections parmi $N$ pour pouvoir décoder.

Il convient néanmoins de considérer la taille variable des projections $B(i)$ car celles-ci augmentent linéairement avec l'angle comme l'indique l'équation 3.2. En particulier, lorsque $q = 1$, cette taille devient :

$$B(i) = |p_i| (Q - 1) + P. \tag{3.4}$$

La taille des projections augmente donc ici linéairement avec $|p_i|$.

Aussi, il est nécessaire d'estimer le paramètre $\varepsilon$, défini littéralement au sens Mojette comme le rapport entre le nombre de bins d'un ensemble suffisant sur le nombre de symboles de message moins l'unité. Ce paramètre $\varepsilon$ va donc dépendre des angles de projections et du nombre d'ixels, qui lui même varie en fonction de sa représentation binaire *e.g* à capacité égale, un support binaire aura des dimensions plus importantes qu'un support d'octets.

Dans la forme redondante de la Mojette, on suppose $\varepsilon$ positif (le cas de projections dégénérées induit $\varepsilon = 0$). Un dénombrement du nombre de symboles et du nombre de bins permet d'établir simplement (voir [57]) :

$$\varepsilon = \frac{(K-1)\sum_{i=1}^{K} |p_i|}{KP}, \tag{3.5}$$

où $K$ correspond au nombre de projections nécessaires et, de manière équivalente, au nombre de lignes du buffer géométrique. La relation 3.5 nous permet d'estimer le paramètre $\varepsilon$ en fonction de la forme du buffer géométrique et les paramètres du code. Dans [57], la parité du nombre total de projections $N$ est prise en compte.

| Capacité du buffer (octets) | L (#bins (o)) | $\varepsilon$ |
|---|---|---|
| 65 | {25,21,17,13,17,21,25} | .4308 |
| 605 | {133,129,125,121,125,129,133} | .0463 |
| 1 Ko | {217,213,209,205,209,213,217} | .0283 |
| 2 Ko | {422,418,414,410,414,418,422} | .0146 |
| 5 Ko | {1036,1032,1028,1024,1028,1032,1036} | .0055 |
| 10 Ko | {2060,2056,2052,2048,2052,2056,2060} | .0027 |

TABLE 3.1 – Mesure du paramètre $\varepsilon$ en fonction de la capacité du buffer géométrique rectangulaire pour un code $Mojette(5,7)$ défini modulo 256.

Le tableau 3.1 porte sur le calcul du paramètre $\varepsilon$ pour un code $Mojette(5,7)$ défini ici *modulo* 256. Hormis le cas particulier d'un très petit buffer géométrique (65 octets) où la valeur de ce paramètre est rédhibitoire (près de 70%), on constate que rapidement la valeur du paramètre $\varepsilon$ est négligeable (une valeur de 4 ‰ pour un buffer de capacité 10 Ko[3]). La définition de ce code modulo 2 est bien entendu possible. Elle a pour conséquence une augmentation du nombre de symboles et donc une réduction du paramètre $\varepsilon$. Ces considérations amènent l'énoncé de la propriété suivante :

PROPRIÉTÉ 3.1.– *le code $Mojette(K,N)$ non systématique est (1+$\varepsilon$)MDS.*

Il est possible de localiser les bins qui ne seront pas utilisés dans la reconstruction [94]. Ces bins particuliers peuvent simplement être ôtés des projections (moyennant une description idoine). Cette réduction du nombre de bins a pour effet de se positionner dans un cadre quasi MDS (seul un bin supplémentaire est conservé par projection pour amorcer la reconstruction). La projection est en revanche légèrement déstructurée (projection à trous).

### 3.3.3 Code Mojette systématique

Cette forme diffère de la section précédente dans la mesure où le message doit apparaître cette fois de manière explicite dans le bloc de mots de codes de sortie. Pour ce faire, nous allons associer les lignes du buffer avec des projections Mojette pour constituer les mots de codes. Une modification de l'algorithme de Mojette inverse est nécessaire au niveau de la phase d'initialisation pour prendre en compte la cohabitation des deux espaces image et projeté au sein des mots de codes. L'association ligne du support et projections Mojette s'exprime dans la définition qui suit :

DÉFINITION 3.3.– *Un code à effacement Mojette systématique, spécifié par le triplet $(K,N,B(i))$, est une fonction $\mathcal{M}$ qui transforme une image $f$ constituée de $K$ lignes de dimension $P$ en une série de $N-K$ projections $proj_{p_1,q_1},\ldots,proj_{p_{N-K},q_{N-K}}$ de taille variable $B(i)$ auquel s'ajoute la série des $K$ lignes de dimension $P$, tel qu'un ensemble de $K$ objets (lignes ou projections) parmi $N$ permettent de reconstruire de manière unique et déterministe l'image $f$.*

3. Des buffers de tailles de 8 à 16 Ko vont être utilisés dans l'application de système de fichier distribué tolérant aux pannes.

### Exemple du code Mojette(3,5) systématique

La construction du code systématique Mojette est illustrée à la Figure 3.7. L'exemple est pris sur le code Mojette binaire $(3, 5)$ où 3 projections parmi 5 sont nécessaires pour reconstruire le buffer géométrique initial. Cinq projections sont produites, comprenant des lignes du buffer ainsi que 2 projections supplémentaires. La complexité de l'opération de codage est réduite ici à $O(I(N - K))$ pour $K$ lignes du buffer géométrique.

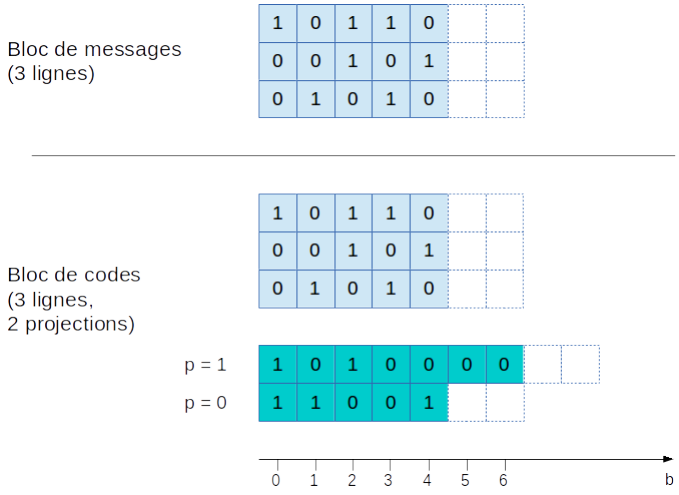



FIGURE 3.7 – Construction du code binaire Mojette (3,5) systématique : le buffer géométrique d'entrée possède 3 lignes et 2 projections sont calculées. Le codage est composé des 3 lignes du buffer initial et des 2 projections calculées.

### Décodage

Le décodage est conditionné par le critère de Katz modifié (lemme 3.3) qui se résume par $K$ mots de codes nécessaires et suffisants pour reconstruire le buffer géométrique initial. Les $K$ mots de code peuvent être de manière indifférente des lignes du buffer ou des projections. La complexité est au plus $O(I(N-K))$ dans le cas où toutes les projections produites sont sollicitées pour la reconstruction.

En pratique, l'algorithme de décodage est similaire à ceux présentés à la section 3.2.6 à la différence d'une phase d'initialisation spécifique illustrée à la Figure 3.8. Avant d'effectuer un décodage standard, il convient de mettre à jour les projections reçues (ainsi que les informations de position et du nombre d'ixels antécédents) à l'aide des projections recalculées en fonction du motif de lignes reçues (avec initialisation à zéro pour l'ensemble des données manquantes). Au terme de cette phase d'initialisation, le décodage est similaire à l'algorithme présenté à la section 7 page 34, relative à la reconstruction géométrique déterministe.

Les tailles des mots de ce code sont plus faibles que dans le cas systématique puisque certains mots de code sont des lignes du support de taille minimale $P$. Le paramètre $\varepsilon$ sera donc plus réduit que précédemment. Par conséquent, on obtient la même propriété que dans le cas non

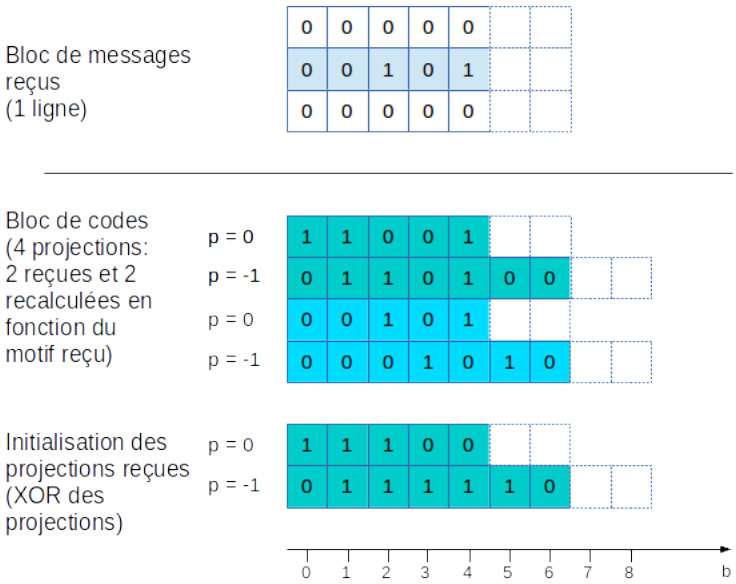



FIGURE 3.8 – Phase d'initialisation des projections pour un décodage binaire Mojette (3,5) systématique en fonction des pertes de lignes du buffer géométrique : les 2 lignes perdues sont exactement compensées par 2 projections. Ces 2 projections sont initialisées par un XOR bin à bin avec les 2 projections recalculées en fonction du motif de lignes reçues (ici la deuxième ligne).

systématique à savoir :

PROPRIÉTÉ 3.2.– *le code Mojette*$(K, N)$ *systématique est (1+*$\varepsilon$*)MDS.*

Ce code a fait l'objet d'un dépôt de brevet en octobre 2013 [21].

### 3.3.4 Code Mojette non systématique sur support hexagonal

Les deux codes précédents peuvent présenter l'inconvénient de délivrer des tailles de mots de code (de projections) différentes en fonction des angles Mojette choisis (et si on exclut l'idée de projection à trous de [94]). Pour obtenir des tailles constantes quel que soit l'angle, il convient de modifier la forme rectangulaire du buffer géométrique au profit d'une forme $pq-$convexe hexagonale. Cette construction est issue des travaux de [57].

#### Définition du support hexagonal

Le principe est d'adjoindre aux supports rectangulaires précédents quelques ixels de manière à rendre la forme hexagonale, en conservant les propriétés de $pq-$convexité. Sous cette contrainte,

les bords gauches du support répondent aux équations de droites discrètes suivantes :

$$m = -k + p_{max}l, \tag{3.6}$$

pour le bord supérieur gauche où $p_{max}$ est la valeur algébrique maximale du paramètre $p$ pour la série de projections considérée. Et :

$$m = -k + p_{min}l, \tag{3.7}$$

pour le bord inférieur droit où $p_{min}$ est la valeur algébrique minimale du paramètre $p$ pour la série de projections considérée.

Il en est de même pour les bords droits avec les équations de droites discrètes 3.6 et 3.7 respectivement pour les bords inférieurs et supérieurs du support hexagonal.

### Exemple du code Mojette(3,5)

Le schéma de la Figure 3.9 représente une construction hexagonale pour un code binaire Mojette non systématique $(3, 5)$ avec la série $S = \{(-2, 1), (-1, 1), (0, 1), (1, 1), (2, 1)\}$. Dans cette série $p_{max} = 2$ et $p_{min} = -2$. Ces deux coefficients directeurs dessinent les pentes des bords supérieurs et inférieurs du support hexagonal. Les projections obtenues sont de taille identique correspondant à la longueur de support (ici de 9 éléments).

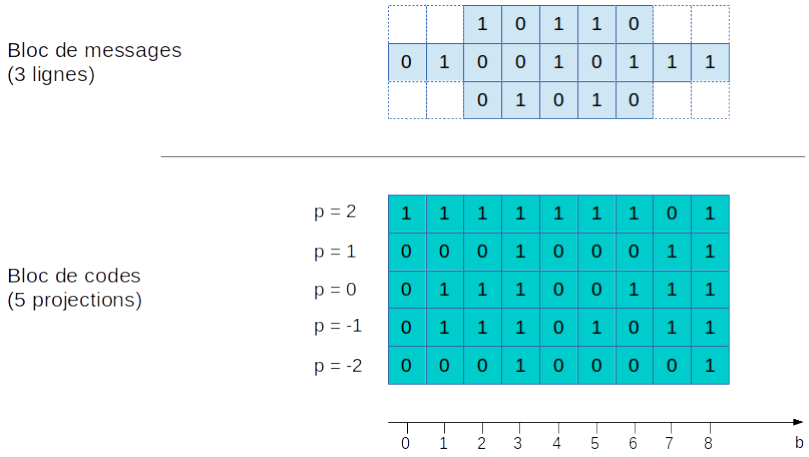



FIGURE 3.9 – Support hexagonal et projections correspondantes pour un code binaire Mojette non-systématique de paramètre $(3, 5)$. Les 5 projections obtenues sont de taille constante (9 bins).

### Propriétés

Les propriétés suivantes sont énoncées au sens de la taille des projections, des critères de reconstruction et de la comparaison par rapport à la solution optimale MDS.

**Taille des projections**. La taille des projections peut être déterminée de manière générale à l'aide des indices *min* et *max* des bins. Ces indices peuvent être positifs ou négatifs en fonction de l'équation de la droite de projection. Aussi, le nombre de bins peut s'écrire de la manière suivante :

$$B(i) = |m_{max} - m_{min}| + 1. \tag{3.8}$$

Pour rappel, l'équation de la droite par laquelle passent tous les ixels projetés est $m = -qk+pl$. Pour ce type de support, les 2 ixels de coordonnées $(k_1, l_1)$ et $(k_I, l_I)$ parmi tous les $I$ ixels se projettent respectivement dans le premier et dernier bin pour l'ensemble des $N$ projections d'une série $S$ définie pour ce type de contexte. Ce nombre de bins devient alors :

$$\begin{aligned} B(i) &= |-q_i k_I + p_i l_I - (q_i k_1 + p_i l_1)| + 1 \\ &= |-q_i \Delta k + p_i \Delta l| + 1 \ \forall i = 1, 2, ...N. \end{aligned} \tag{3.9}$$

En outre, on convient que $\Delta l = 0$ puisque les 2 ixels extrêmes sont choisis sur la même ligne du support hexagonal. Le nombre de bins explicité à l'équation 3.9 devient ainsi :

$$\begin{aligned} B(i) &= |-q_i \Delta k| + 1 \\ &= |q_i| \, \Delta k + 1 \ \forall i = 1, 2, ...N. \end{aligned} \tag{3.10}$$

Enfin, dans le cas particulier où $q = 1$ pour toutes les $N$ projections, cette taille de projection se réduit simplement à :

$$B(i) = \Delta k + 1 \ \forall i = 1, 2, ...N. \tag{3.11}$$

On vérifie bien cette propriété avec l'exemple de la Figure 3.9 où chaque projection possède 9 bins en relation avec les abscisses $k = 0$ et $k = 8$ du support hexagonal. On arrive ainsi à la propriété suivante :

PROPRIÉTÉ 3.3.– *Les projections d'un support hexagonal sont de tailles identiques égales à la largeur $P$ du support.*

**Conditions de reconstruction et inversibilité**. Le critère de reconstruction est identique aux formes rectangulaires présentées plus haut. Pour un support d'épaisseur $K$, il est nécessaire d'avoir au moins $K$ projections pour satisfaire le critère de Katz (on considérera dans ce cas le rectangle qui contient entièrement la forme hexagonale tel que représentée à la Figure 3.9). En considérant la forme $pq-$convexe, on pourra également utiliser le théorème de reconstruction complète [51].

**Surcoût de la représentation hexagonale**. L'accroissement de la taille des projections (au regard d'un faible ajout d'ixels sur la forme) peut se traduire par un surcoût de la représentation, estimé par le paramètre $\epsilon$. Le tableau 3.2 renseigne la valeur du paramètre $\epsilon$ pour différentes

| Capacité du buffer (octets) | L (#bins (o)) | $\varepsilon$ |
|---|---|---|
| 64 | 20 | .5625 |
| 604 | 128 | .0596 |
| 1 Ko | 212 | .0352 |
| 2 Ko | 417 | .0181 |
| 5 Ko | 1032 | .0078 |
| 10 Ko | 2056 | .0039 |

TABLE 3.2 – Mesure du paramètre $\varepsilon$ en fonction de la capacité du support hexagonal (le buffer géométrique).

tailles du buffer hexagonal. Ces valeurs sont à comparer au tableau 3.1 donné pour des buffers de capacités identiques mais de formes rectangulaires. Malgré un léger surcoût par rapport à la forme rectangulaire, on constate que le code obtenu devient rapidement proche de l'optimal MDS. On en déduit donc la propriété suivante :

PROPRIÉTÉ 3.4.– *le code* $Mojette(K, N)$ *non systématique sur support hexagonal est* $(1+\varepsilon)$*MDS.*

Au regard des deux précédents codes présentés, il est naturellement possible de définir un code systématique sur support hexagonal.

### 3.3.5 Protection inégale

La protection inégale est proposée dans le cadre de la transmission d'une source hiérarchique (ou scalable). Cette source se compose de différents sous-flux qui possèdent une priorité propre pour pouvoir être exploitables à la réception. La protection inégale considère la hiérarchie de la source en allouant des codes correcteurs de rendements distincts et ce, de manière à pouvoir décoder les flux prioritaires en premier. L'idée fondamentale a été proposée par Albanese *et al.* [2] à l'aide de codes MDS basés sur des matrices de Cauchy. Cet article a donné lieu à de nombreux travaux dans les années 2000 [49, 1, 4] et à des connexions intéressantes avec le problème de codage à description multiple défini aux Bell Labs dans la fin des années 1970 (voir l'état de l'art très documenté de Vivek Goyal [30] au sujet de ce beau problème en théorie de l'information).

Dans ma thèse, j'ai proposé un schéma de protection inégale basé sur la transformation Mojette. Ce schéma s'appuie sur des régions $pq$-convexes reconstructibles de manière différenciées avec un nombre de projections qui est fonction de l'importance de la donnée. Une région importante sera reconstruite par un faible nombre de projections. À l'inverse, une région moins importante demandera un plus grand nombre, voire la totalité, des projections produites. Une illustration de ce schéma de protection inégale est donnée à la Figure 3.10 où 3 sous-flux de données sources sont protégés de manière inégale : la région 1 (resp. 2 et 3) est reconstructible par 3 projections (rep. 4 et 5). Ce schéma autorise la perte de projections qui va fragiliser en premier l'information la moins importante. Contrairement à l'approche originale de [50] proposée sur des canaux ATM, ce travail permet le transport de sources hiérarchiques sur des réseaux de paquets de type *best effort* comme l'Internet *i.e.* dépourvu de Qualité de Service (QoS).

Les angles de projection du schéma de la Figure 3.10 sont choisis de manière à avoir une équivalence entre les projections en termes de reconstruction. Avec $q = 1$, cette condition est respectée. On raisonne ainsi en nombre de projections nécessaires quel que soit son angle. Le schéma de protection inégale obtenu est déterministe.

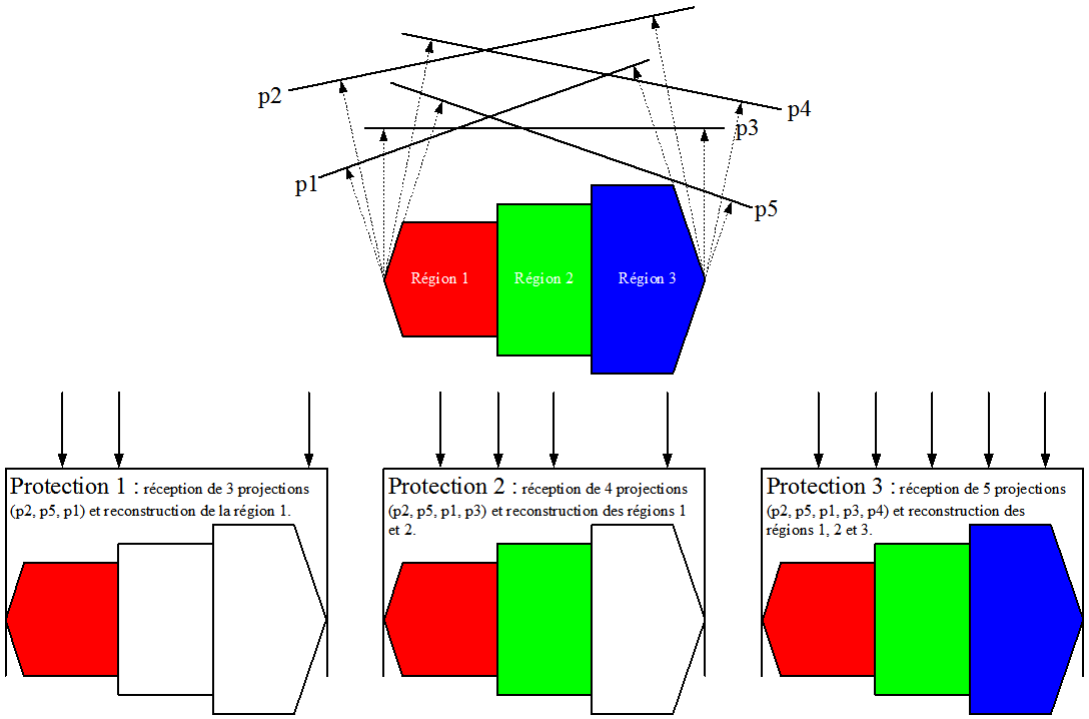



FIGURE 3.10 – Schéma illustrant la protection inégale Mojette. Chaque région du buffer géométrique est reconstructible par un nombre graduel de projections. Dans ce schéma, la région 1 (resp. 2 et 3) est reconstructible par 3 projections (resp. 4 et 5). La région 1, plus facilement reconstructible, est mieux protégée que les régions 2 et 3 vis-à-vis des pertes ou effacement de projections (extrait de [57] Chapitre 5).

La forme du support, inspirée des supports hexagonaux de la section 3.3.4 à la page 40, garantit des tailles de projections constantes. Si cette contrainte n'est pas nécessaire comme dans beaucoup d'applications communicantes, le supports peuvent reprendre les formes rectangulaire d'origine. Avec, dans les deux cas, des frontières entre région variables en fonction du motif de projections reçues. Ce découpage en région est étudié précisément dans mon manuscrit de thèse [57].

Ce protocole de transport a été proposé dans le cadre du transfert d'image codée en Jpeg progressif [63] et pour la transmission sécurisée issue du codeur source adaptatif LAR dans le cadre d'une application en télémédecine [4]. Il est possible d'asservir le taux de redondance (et donc le niveau de protection) sur un critère psycho-visuelle à l'instar des travaux proposés dans [60] et sur un modèle de perte de paquets de type Gilbert-Elliott [59]. Cette approche de codage conjoint source-canal psycho-visuel a été ensuite étendue à la vidéo de type H.264/AVC (voir [8], [7] et [9]).

## 3.4 Conclusion

Ce chapitre présente les codes à effacement Mojette. Le code Mojette non systématique sur support rectangulaire, le code Mojette systématique sur support rectangulaire et le code Mojette

non systématique sur support hexagonal sont successivement définis. Ces codes sont déterministes dans la mesure où un nombre fini de $K$ projections prises parmi un ensemble $N$ est suffisant pour reconstruire de manière certaine le support initial (appelé buffer géométrique dans ce contexte). Ces codes sont $(1+\varepsilon)MDS$ (voire quasi MDS) dans la mesure où ils nécessitent $k+\varepsilon$ pour pouvoir décoder les $k$ symboles de message d'origine. Un schéma de protection inégale Mojette est également présenté. Il intègre un codage Mojette non systématique à redondance variable en fonction de l'importance des flux sources.

Cousine de la transformation Mojette, la transformation de Radon finie (FRT) [46] permet également de fournir un ensemble de codes à effacement géométriques dotés de propriétés intéressantes. Ils ne seront pas traités dans ce manuscrit (voir [54] pour plus de détails).

Par définition, les codes à effacement permettent de tolérer des pertes de paquets ou encore la perte de nœuds de stockage, comme nous allons le voir dans le chapitre suivant qui traite du stockage distribué en mode Cloud. Dans ce contexte, les codes à effacement géométriques Mojette ont l'avantage de la simplicité algorithmique vis-à-vis des approches algébriques de type Reed-Solomon.

# Chapitre 4

# Codes à effacement pour le Cloud (le projet FEC4Cloud)

## 4.1 Introduction

Comme l'indique James Plank dans son article de 2013 [71], les codes à effacement constituent la technologie élémentaire pour les systèmes tolérants aux pannes. Le nombre de pannes peut être très variable d'un système à l'autre. Un système RAID-6 peut tolérer par exemple la panne de deux disques avec des implémentations originales proches dans l'esprit des codes Mojette e.g les codes RDP [19]. Les systèmes de type Cloud tolèrent eux un plus grand nombre de pannes [12, 36]. Enfin, l'archivage numérique peut donner lieu à des situations réellement catastrophiques [83, 90, 91].

Cependant, il n'existe pas aujourd'hui de codes algébriques suffisamment efficaces pour satisfaire aux exigences du stockage distribué avec des entrées et sorties (E/S) intensives. Ces codes à effacement sont le plus souvent relégués à l'accès à des données dites "froides" pour lesquelles les temps d'accès ne sont pas critiques.

En matière de Cloud, il convient de distinguer les systèmes à entrées/sorties intensives des systèmes dédiés au Big Data. Même si la frontière tend à s'estomper aujourd'hui, elle demeure bien réelle par la distinction des solutions proposées. Si HDFS (*Hadoop Distributed File System*) est la référence en matière de calcul massif de données, Ceph, Gluster et Lustre (dédié HPC pour ce dernier) sont désignés comme *Distributed File System* (DFS) pour un nombre intensif de lectures et écritures. Si le point commun est la tolérance aux pannes, une différence majeure est la taille des objets manipulés. Pour HDFS, la taille par défaut des unités de données est de 64 Mo. Ceph utilise des objets de taille 8 fois plus petite à savoir de 8 Mo par défaut. En matière de taille, il faut descendre encore si on requiert une qualité de service similaire à des *File System* locaux comme `ext4`. Pour ce faire, des tailles de l'ordre de 8 ou 4 Ko semblent plus adaptées. À cette granularité, la performance des disques et du réseau d'interconnexion est un facteur important.

Ces solutions logiciels (*Software Defined Storage* ou SDS) doivent garantir une très faible indisponibilité des données (de la minute à la dizaine d'heures par an). Compte tenu de la relative faible espérance de vie du matériel informatique (un disque a une durée de vie de 4 ans en moyenne), il convient de mettre en place des mécanismes de tolérance aux pannes comme

la réplication ou des codes à effacement. Contrairement au contexte des réseaux de paquet, le nombre de pertes est relativement réduit (4 effacements en simultané pour un seul fichier est un événement assez rare en stockage). Ces pertes concernent ici la disparition d'un nœud de stockage (temporaire ou définitive). Les mécanismes de tolérance aux pannes se doivent néanmoins de pouvoir fournir une capacité de correction plus importante sans pénalité supplémentaire sur les temps de codage et décodage.

Aujourd'hui, à notre connaissance, seul RozoFS fournit un système de fichier distribué tolérant aux pannes basé sur un code à effacement pour des lectures et écritures intensives. Les débits et le nombre d'entrées/sorties par seconde (IOPS) autorise des applications critiques telles que le montage vidéo en ligne, l'exécution de machines virtuelles ou encore les échanges transactionnels pour des bases de données distribuées.

## 4.2 Tolérance aux pannes

### 4.2.1 Comparaison avec l'approche par réplication

On peut bien entendu convenir que l'approche par réplication est, elle aussi, une approche par code (utilisant un code à répétition MDS). On distinguera néanmoins dans ce qui suit les deux approches pour mieux les mettre en comparaison.

Il est aisé de démontrer que pour un même taux de disponibilité, la réplication nécessite plus d'espace de stockage (deux fois plus dans des systèmes classiques tolérants jusqu'à deux pannes [95, 55]). Cette préférence pour l'approche par code persiste lorsque la probabilité de faute d'un nœud est introduite. On définit ici la faute d'un nœud par son indisponibilité temporaire ou permanente (qui peut avoir différentes origines : redémarrage, alimentation électrique, fin de vie d'un disque, coupure câble...).

En effet, on considère dans cette étude que chaque effacement d'un nœud est i.i.d (soit indépendant et identiquement distribué). Soit $f$ la probabilité de faute d'un nœud individuel (la probabilité de disponibilité de ce même nœud est alors $1-f$). Le nombre réel de fautes d'une série de nœuds indépendants suit une loi binomiale. La probabilité de disponibilité d'un objet stocké à l'aide d'un code MDS $(k,n)$, soit $D_{MDS}$ égale à la probabilité qu'au moins $k$ nœuds soient disponibles, est

$$D_{MDS} = \sum_{j=k}^{n} \binom{n}{j} f^{n-j}(1-f)^{j} \tag{4.1}$$

La disponibilité au sens de la réplication, avec $r$ répliques, s'écrit de la manière suivante :

$$D_{rep} = 1 - f^{r}. \tag{4.2}$$

**Application numérique 1 :** avec 1 million de nœuds, dont 10% qui sont inaccessibles, le stockage de deux répliques fournit 0,99 de disponibilité. Pour un facteur de rendement de $\frac{1}{2}$, un document découpé en 32 fragments (soit 64 blocs au total) fournit 0,999999998 de disponibilité (soit 8 neuf après la virgule) (extrait de [95]).

**Application numérique 2 :** si la probabilité de faute d'un nœud est $f = 0,1$, alors pour le même surplus de stockage de 3, correspondant à $r = 3$ pour la réplication et à un code MDS de type $(9, 3)$, les probabilités de perdre un objet sont respectivement

$$1 - D_{rep} = 10^{-3} \tag{4.3}$$

pour la réplication, et

$$1 - D_{MDS} \sim 3.10^{-6} \tag{4.4}$$

pour une approche par code à effacement (extrait de [55]).

Ces simples applications numériques extraites de la littérature exprime bien l'amélioration de la disponibilité de la donnée en utilisant une approche par code.

Cette amélioration a bien entendu un coût en complexité et en nombre d'entrées/sorties que l'on peut appréhender sur la Figure 4.1 (extraite de [28]). En effet, que ce soit dans un cadre de lecture dégradée (avec une panne par exemple sur un bloc de données) ou même pour de simples écritures (en l'absence de panne), l'approche par code à effacement et l'usage de bloc de parité (P1 et P2 sur la figure) nécessite significativement un plus grand nombre d'entrées/sorties (voir figure 4.1(a) en comparaison de la figure 4.1(b)).

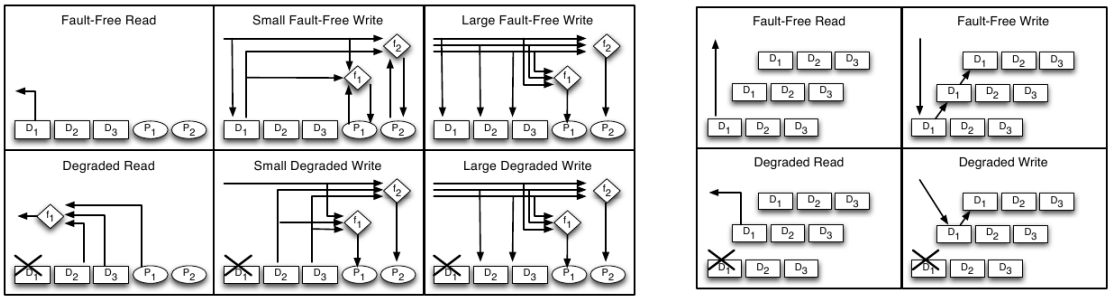



FIGURE 4.1 – Illustration de la complexité des opérations de lecture et écriture pour l'approche par codes à effacement (a) comparée à l'approche par réplication (b) pour différents niveaux de pannes (extrait de [28]).

Fort de ce constat, une première solution consiste à optimiser les codes à effacement pour ne pas pénaliser les opérations d'entrées/sorties (E/S). Ce faisant, il est possible d'imaginer l'usage des codes à effacement même dans le cas où les données sont qualifiées "chaudes" à savoir des données très fréquemment utilisées avec des contraintes fortes en matières de vitesses de lecture et écriture.

### 4.2.2 Performances des codes Mojette

Cette section porte sur la comparaison des performances (du point de vue du débit d'encodage et de décodage) entre les trois implémentations de codes algébriques (`Cauchy-origin`[6],

`Cauchy-good`[74] et la librairie ISA-L d'Intel® basée sur les travaux de [71]) et les deux codes géométriques (`Mojette`[57] et `Mojette-good`[22]). L'ensemble de ces 5 codes à effacement sont présentés rapidement dans les paragraphes suivants.

### Présentation des codes testés

**Code de Cauchy-Reed-Solomon (CRS) basé sur des opérations XOR.** En 1995, Blömer *et al.* [6] ont développé la première implémentation d'un code de Cauchy-Reed-Solomon (CRS) qui évite l'usage de multiplications au sein d'un corps de Galois (GF). Le principe est d'étendre la matrice génératrice original de Cauchy en une représentation binaire. Une telle modification change les additions en simple XOR et les multiplications en ET logique. Cette modification a clairement redonné un second souffle au code original proposé par Irving S. Reed et Gustave Solomon [81] dans les années 60 avec un impact majeur dans les systèmes de transmission [2] et de stockage [82]. Une mise en œuvre de ce code est disponible au sein de la librairie Jerasure 1.2 [73]. Dans la suite, la dénomination "`Cauchy-origin`" fera référence à cette implémentation.

**Code de Cauchy 'Good'.** En 2006, Plank *et al.* [74] ont initié une nouvelle implémentation avec l'observation suivante : toutes les matrices de Cauchy ne sont pas équivalentes en nombre de XOR exécutés par mot de code. L'optimisation concerne ainsi la recherche de "bonnes" matrices qui sont particulièrement creuses (avec très peu de '1' à l'intérieur). Un gain d'un facteur 2 est obtenu à la fois en codage et en décodage [72]. Ce code est disponible également au sein de la librairie Jerasure 1.2 [73]. Cette mise en œuvre, appelée "`Cauchy-good`" dans la suite du document, a été retenue dans un premier temps dans une version du système de fichier distribué Ceph [96].

**Implementation ISA-L.** Outre l'optimisation précédente, l'*Intelligent Storage Acceleration Library* (ISA-L) de la société Intel® s'appuie sur les travaux de [71] qui visent à accélérer significativement les opérations arithmétiques des corps de Galois à l'aide d'instructions SIMD (ou SSE) sur des mots de 128 bits. En pratique, un gain d'un facteur 3 est avancé par rapport aux approches XOR de [6] (soit un gain d'un facteur $\frac{3}{2}$ par rapport aux `Cauchy-good` de [74]). C'est l'implémentation la plus rapide (à ce jour) des codes de Reed-Solomon. Une intégration est en cours au sein d'HDFS (*Hadoop Distributed File System*) pour le Big Data [1] ainsi que pour Ceph.

**Code Mojette** $(1+\varepsilon)MDS$**.** Ce code proposé en 2001 [57] (et redétaillé dans la section 3.3.2 de ce document) est un code Mojette particulier où $q = 1$. Ce paramètre confère aux projections une même capacité de reconstruction. Le code obtenu est $(1 + \varepsilon)MDS$. Au niveau décodage, il bénéficie d'une optimisation géométrique proposée en 2006 [52]. Ce code est implémenté au sein du système de fichier distribué RozoFS [66]. L'implémentation `Mojette` fera référence à cette mise en œuvre dans ce chapitre.

**Code Mojette 'Good'.** Cette optimisation du code Mojette réduit de manière drastique le nombre d'écritures sur les projections en concentrant cette opération pour la seule fonction de rétro-projection. Cette mise en œuvre, dénommée ici `Mojette-good`, a fait l'objet d'un dépôt de brevet en mars 2014 [22]. Le terme `good` est un clin d'œil à la mise en œuvre de James Planck.

1. https://issues.apache.org/jira/browse/HDFS-7285

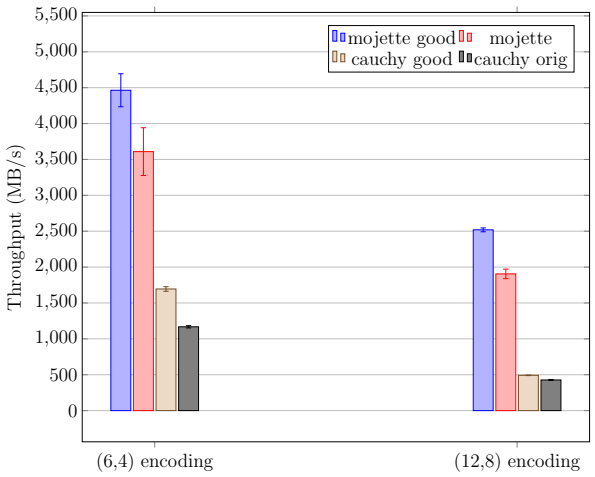



FIGURE 4.2 – Débit d'encodage (en Mo/s) pour le code $(4, 6)$ et le code $(8, 12)$ (première campagne de tests).

### Matériels et méthodes

La comparaison entre les codes pré-cités s'est effectuée en deux temps (en fonction des disponibilités des librairies). La première campagne de tests (mars 2014) a consisté à comparer l'approche Mojette avec la librairie Jerasure. La seconde campagne (septembre 2014) vise à comparer les codes Mojette avec la librairie ISA-L qui fait référence aujourd'hui. L'ensemble des résultats de ces deux campagnes de mesures est présenté dans la suite.

Pour la première campagne (mars 2014), la mesure concerne les débits d'encodage et de décodage en Mo/s. Elle résulte directement des délais d'encodage et de décodage pour un bloc de données (incluant les écritures sur le disque dur). Ces délais sont mesurés en nombre de cycles CPU. Ils sont donc indépendants de l'activité de la machine. Le fichier test concerne un fichier vidéo de 704 Mo découpé en blocs de 4 Ko. L'ensemble des codes testés concerne des codes MDS ou $(1+\epsilon)MDS$ pour la Mojette de type $(4, 6)$ et $(8, 12)$ permettant de compenser respectivement 2 ou 4 effacements. Pour les implémentations de Cauchy, la taille du mot, $w$, est égale à 4 ce qui satisfait la condition $2^w \geq N$ ($w = 8$ affiche à peu près les même performances).

Ces premiers tests sont conduits sur une machine Linux munie d'une distribution Debian (noyau 3.2) pour architecture x86-64 avec un processeur Intel Xeon de 3,10 GHz, 4 Go de RAM et une mémoire cache de 8 Mo.

Pour la seconde campagne (septembre 2014), la mesure concerne plus finement le nombre de cycles CPU avec une comparaison avec l'instruction élémentaire `memcpy` qui consiste à faire une simple copie en mémoire d'un bloc. Deux tailles de blocs sont testées : 8192 et 4096 octets. Cette campagne une implémentation Mojette plus performante que lors de la première campagne.

Ces seconds tests sont conduits sur des machines moins puissantes que précédemment : un processeur Intel Xeon (E5-2403) cadencé à 1,8 GHz, 16 Go de RAM et 8 Mo de cache (une des machines du cluster FEC4Cloud).

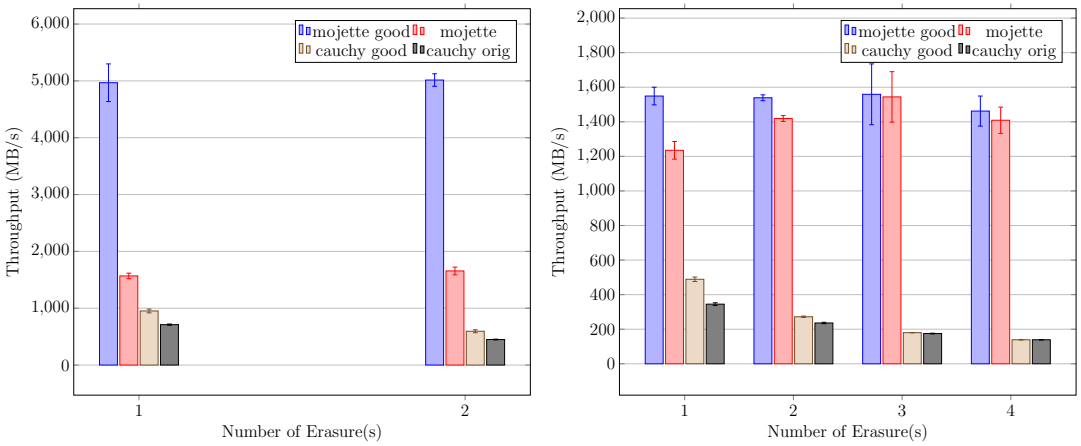



FIGURE 4.3 – Débit de décodage (en Mo/s) sur des blocs de 4Ko pour les codes $(4, 6)$ et $(8, 12)$ respectivement à gauche et à droite de la figure (première campagne de tests).

### Comparaison avec les implémentations de Blömer [6] et Plank [74] (première campagne de tests)

La Figure 4.2 montre les débits d'encodage pour les quatre codes testés dans le cadre de la première campagne de mesures. L'implémentation `Mojette-good` enregistre un débit de 4,5 Go/s en $(4, 6)$ et 2,5 Go/s en $(8, 12)$. On note au passage la linéarité d'exécution (2 fois plus de projections produites dans le cas de la $Mojette(8, 12)$. Elle surpasse l'implémentation `Cauchy-good` respectivement d'un facteur 3 et 5. L'implémentation Mojette standard présente de bonnes performances en affichant respectivement 3,5 Go/s et un peu moins de 2 Go/s. Elle surpasse nettement l'implémentation de Cauchy. Ces bonnes performances des codes Mojette sont obtenues malgré un volume de données produit 3 fois plus important. Pour rappel, les deux codes Mojette utilisés sont en effet non systématiques, à la différence des implémentations de Cauchy.

La Figure 4.3 présente les débits de décodage pour les 4 codes testés en $(4, 6)$ et $(8, 12)$ (soit une capacité de correction d'au plus 2 et 4 effacements respectivement). Dans le cas des implémentations de Cauchy sous forme systématique, ces effacements concernent des blocs explicites du message (et non des blocs de parité) afin de solliciter de la même manière le décodeur (qu'il soit systématique ou non). Le débit de décodage pour l'implémentation `Mojette-good` s'élève à 5 Go/s indépendamment du nombre d'effacements. On constate le même phénomène avec l'implémentation standard de la Mojette avec des débits cependant nettement moins importants. On voit ici la forte optimisation de l'implémentation `Mojette-good` qui bénéfice d'un pré-calcul de la Mojette avec une réduction drastique du nombre d'écritures en mémoire et sur le disque dur. L'implémentation `Mojette-good` délivre un débit 5 à 10 fois plus élevé que la meilleure implémentation de Cauchy testée. Contrairement aux implémentations Mojette, les performances des codes Cauchy sont dépendants du nombre d'effacements en raison de leur forme systématique. Ces résultats ont été publiés sous forme d'un poster à Eurosys 2014 [69].

### Comparaison avec la librairie ISA-L (deuxième campagne de tests)

La deuxième campagne compare le code Mojette (`Mojette-good` uniquement) avec l'implémentation RS par matrice de Cauchy de la librairie ISA-L. Les résultats sont donnés Figure 4.4

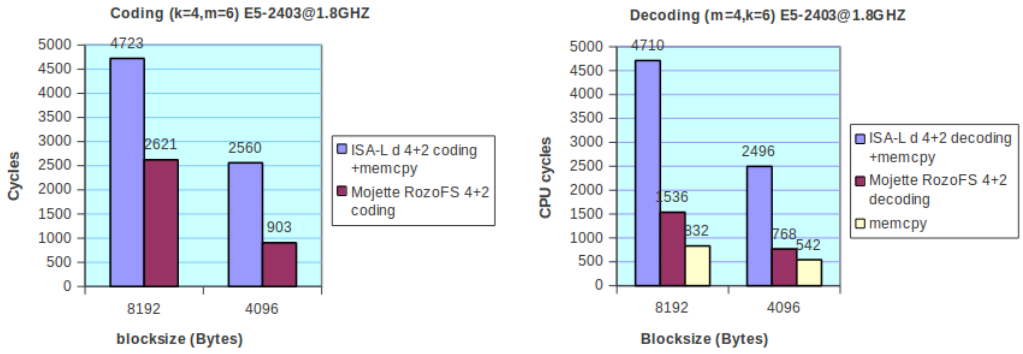



FIGURE 4.4 – Nombre de cycles CPU pour le codage (figure de gauche) et décodage (figure de droite) pour la Mojette et pour la librairie ISA-L. L'instruction `memcopy` est donnée à titre de référence pour le codage et le décodage (deuxième campagne de tests).

en nombre de cycles CPU. Pour les deux tailles de blocs, le code Mojette surpasse très nettement les performances de la librairie ISA-L. Le facteur est d'autant plus important que la taille du bloc est petite. Pour 4096 octets, le décodage Mojette est très proche de l'optimal à savoir la commande `memcpy` qui est une simple copie en mémoire. S'agissant des même volumes de données, les mesures effectuées pour l'instruction `memcpy` sont valables pour le codage. Fonction des tailles de blocs de la cadence du processeur utilisé, c'est en substance des débits de 8 Go/s (pour des blocs de 4 Ko) en encodage et 9,6 Go/s en décodage qui sont mesurées pour cette implémentation du code Mojette.

C'est un gain d'un facteur 2 en codage et d'un facteur 3 en décodage en faveur de l'approche Mojette qui est enregistré malgré un volume de données encodées supérieur (code non systématique (Mojette) contre code systématique (ISA-L)). Ainsi, dans le cas d'un codage $(4, 6)$, c'est 3 fois plus de données encodées et 2 fois plus de données décodées pour la Mojette. C'est donc potentiellement un gain d'un facteur 6 que nous pouvons attendre dans le cas d'une Mojette systématique par rapport à cette même librairie ISA-L. La régularité de la reconstruction géométrique, la proximité des zones mémoires sollicitées (pixels ou bins connexes) et l'affranchissement de l'arithmétique des corps de Galois sont les principales raisons théoriques de ce gain considérable. Ces résultats ont été présentés lors d'une journée Inter GDR ISIS et SoCSiP en novembre 2014 à Télécom Brest dédiée à l'architecture des codes correcteurs. Ils font l'objet d'une soumission à HotStorage 2015 dont l'article est présent en fin de cet ouvrage.

## 4.3 Intégration au sein du système de fichiers distribué RozoFS

Cette section porte sur la description générale du système de fichier distribué RozoFS ainsi que sur la description de deux cas de déploiement autour d'une application de montage vidéo en ligne d'une part et sur le cluster GRID5000 d'autre part.

### 4.3.1 Introduction à RozoFS

RozoFS est une solution de type *Software Defined Storage* (SDS). Ce logiciel s'est bâti autour du code à effacement Mojette. C'est une différence majeure avec ses concurrents qui sont tous au départ basés sur la réplication. L'idée élémentaire est donnée en 2001 dans [34] : stocker des ensembles de projections Mojette au sein d'un système de stockage distribué composé d'une grappe de serveurs distants sur l'Internet. La Figure 4.5 illustre l'architecture client-serveur envisagée il y a 15 ans. Trois serveurs disposent de trois ensembles de projections potentiellement reconstructibles (variante possible). Des clients distants (de $D1$ à $D6$) effectuent des requêtes distribuées auprès des serveurs de manière à rassembler une série de projections reconstructibles (ici une image médicale). Les projections sont acheminées à travers un réseau cœur. L'intérêt réside dans la répartition de charge au sein du réseau cœur, ainsi que le transit de codes (les projections) peu exploitables par une interception.

Pour RozoFS, le schéma de 2001 a été modifié. Le schéma de la Figure 4.6 présente de manière très simplifiée le réseau de stockage où chaque nœud (étiqueté de $N1$ à $N6$) héberge une projection d'un fichier (étiquetées de $p1$ à $p6$). Le schéma est ici tolérant à deux pannes i.e. 4 projections parmi 6 sont nécessaires pour reconstruire le fichier. Deux autres niveaux de protection et de répartition sont disponibles de manière standard à savoir 2 projections parmi 3, et 8 projections parmi 12, permettant de tolérer respectivement 1 et 4 pannes. Un serveur de méta-données localise les projections dans le cluster. Afin d'éviter tout point unique de panne (le *Single POint of Failure*), celui-ci est répliqué à l'aide du logiciel DRBD [2](*Distributed Replicated Block Device*) comme illustré à la Figure 4.7. L'ajout de nœud peut se faire à chaud, ce qui confère au réseau de stockage un caractère de type *scale-out* [3]. Le serveur de méta-données garantit la répartition de charge. Les clients (qui ne sont pas représentés sur ce schéma de la Figure 4.6) effectuent des requêtes sur chacun des serveurs et s'occupent du codage (pour l'écriture) et du décodage (pour la lecture) à l'instar du schéma proposé en 2001 (voir Figure 4.5). Ces clients sont le plus souvent co-localisés sur les nœuds de stockage, ce qui suppose des liaisons inter-nœuds très performantes. Ces client locaux sont le point d'entrée pour les clients distants du cluster. Le code Mojette utilisé est non systématique. Il y a donc même en mode non dégradé (sans panne) sollicitation du code Mojette (en lecture et en écriture).

En pratique, bien entendu, l'architecture est un plus complexe comme le témoigne le schéma de la Figure 4.7. En effet, il existe deux réseaux distincts pour cloisonner les clients des serveurs (ici co-localisés sur un même nœud). Les nœuds de stockage sont interconnectés via un réseau à forte bande passante (plusieurs liens de 10 GE peuvent être agrégés entre eux pour éviter tout point de congestion). L'agrégation peut être également réalisée sur les voies montantes à destination des clients. Le démon `storaged` unifie les ressources de stockage issues potentiellement de différentes technologies (ici SAS et SSD). Le démon `exportd` est responsable de la gestion des méta-données. Le serveur de méta-données est répliqué ici à l'aide de l'outil DRBD (*Distributed Replicated Block Device*). L'accès aux données s'effectue en mode bloc. Le système de fichiers distribué obtenu est compatible POSIX. Un complément technique est disponible à `https://github.com/rozofs/rozofs/wiki/AboutRozoFS`.

### 4.3.2 Application au montage vidéo en ligne

RozoFS (comme tout DFS) peut être considéré comme un "système complexe". Le comportement du système global va dépendre de beaucoup de paramètres, dont les caractéristiques des

2. http ://doc.ubuntu-fr.org/drbd
3. qui s'oppose à l'approche de type *scale-up* où le principe est d'empiler des disques sur un même nœud.

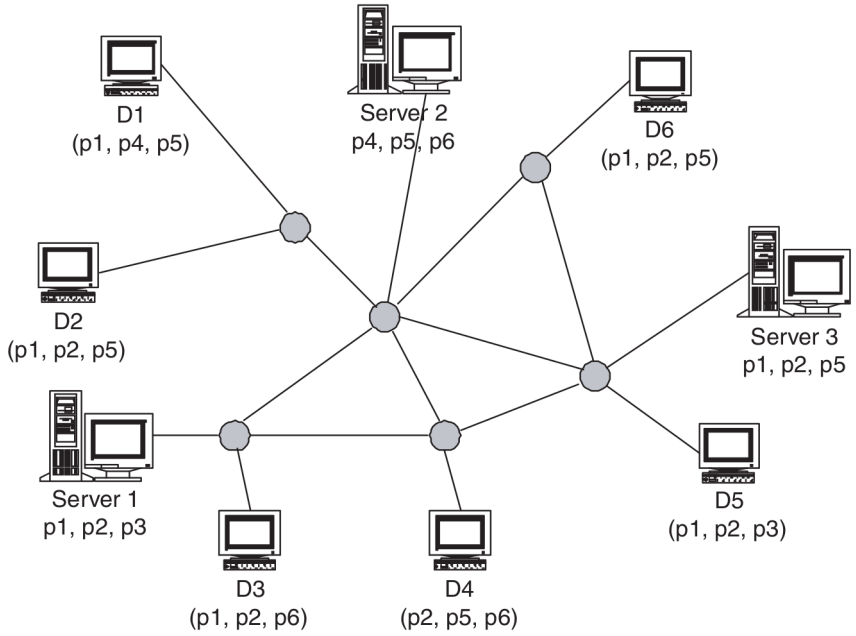



FIGURE 4.5 – Stockage distribué Mojette original de 2001 [34].

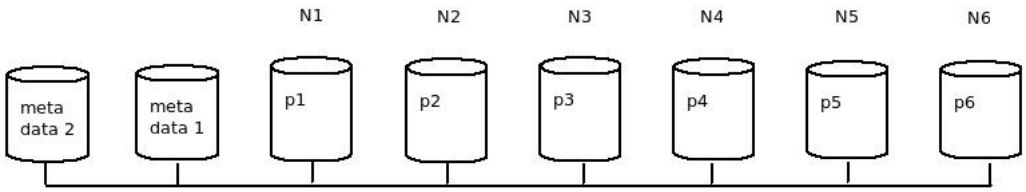



FIGURE 4.6 – Stockage distribué RozoFS (schéma simplifié).

disques et la qualité de service (QoS) réseau. Il est très difficile de prédire ces performances. On ne peut avoir dans ce contexte qu'une approche empirique dans un premier temps, qui vise à démontrer la possibilité d'utiliser des codes à effacement Mojette pour des applications E/S intensives.

Un premier déploiement vise à faire cette démonstration sur une application de type montage vidéo en ligne. Cette application est en production depuis juin 2013 dans le cadre de préparation de contenus vidéos pour de la formation à distance auprès d'un grand organisme national. L'architecture de la plateforme de montage est présentée à la Figure 4.8. On retrouve des similitudes avec la plateforme de la Figure 4.7. Le cluster est composé de 6 serveurs (Intel Xeon@2.40 GHz, 12 GB RAM) pourvus chacun de 10 disques (SAS 7200) de 2 TB. Un contrôleur RAID-5 permet d'assurer la fiabilité locale. Un serveur de méta-données actif est co-localisé sur un des nœuds de stockage (un serveur passif est répliqué sur un autre à l'aide de DRBD). Au niveau réseau, 4 ports Ethernet Gigabit agrégés relient les nœuds de stockage entre eux. Cette forte réservation de bande passante résulte du fait des fortes communications inter-nœuds par lesquels transitent les projections. Les clients distants (FCP pour le logiciel de montage Final Cut Pro d'Apple) ont accès à la plateforme via un réseau Gigabit Ethernet séparé.

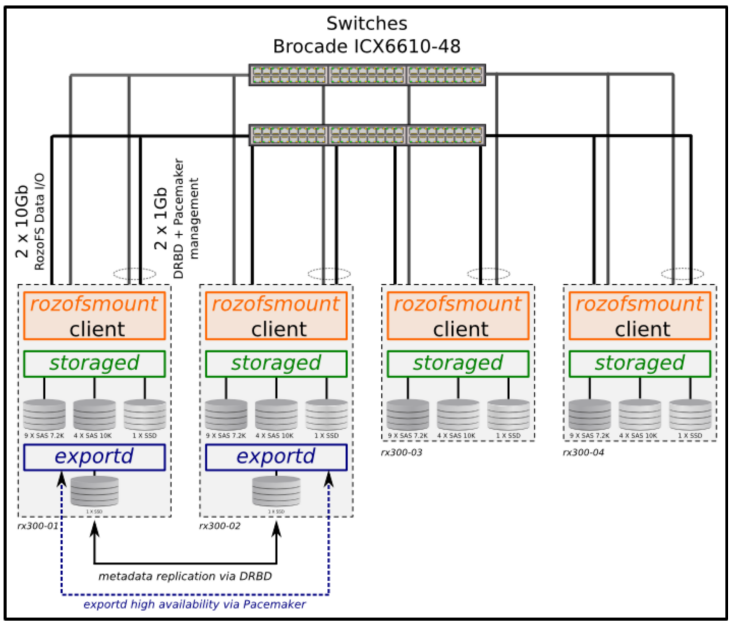



FIGURE 4.7 – Schéma général d'une plateforme stockage distribué RozoFS incluant les *daemons* `exportd` et `storaged` ainsi que le point de montage client `rozofsmount`.

Le montage vidéo en ligne suppose ici l'intervention de 5 opérateurs(trices) en parallèle qui travaillent chacun(e) sur environ 5 contenus vidéo brutes distincts en lecture pour une vidéo en écriture (format propriétaire Apple ProRes 422). Chaque vidéo requiert un débit minimum de 200 Mb/s (soit 25 Mo/s) de manière à éviter des artefacts sur l'image (suppression, gel, gigue,...). Pour ce type d'application, les lectures intensives avec accès séquentiels et aléatoires sont cruciales.

Au delà du fait que la plateforme est en production depuis près deux ans, nous avons voulu la caractériser à l'aide de l'outil IOZone [4] pour vérifier si elle répondait correctement aux exigences de l'application en établissant ses limites. IOZone est un logiciel de *benchmark* pour système de fichier. Le fichier élémentaire de charge a une taille de 1 Go avec un découpage en blocs de 32 Ko qui correspond à l'unité d'échange du logiciel de montage. Pour émuler un comportement multi-sources, 5 tâches (*threads*) par client sont instanciées. Chaque client induit donc l'écriture de 5 Go utile, ce qui se traduit par une écriture physique sur les disques de 7,5 Go étant donnée la redondance (2 projections sont nécessaires parmi 3).

Les résultats de ce test de charge sont donnés Figure 4.9. La mesure concerne le débit cumulé (en MB/s) en fonction du nombre de clients distants. Ce débit est un débit moyen sur 5 tests indépendants (l'écart-type est peu significatif ici). Un débit (en lecture) de 25 Mo/s par client est défini pour valider la plateforme, pour le montage vidéo en ligne. L'écriture est moins critique ici dans la mesure où elle peut effectuée de manière asynchrone sans dégrader la qualité d'expérience.

En mode séquentiel, les débits en lecture croissent linéairement avec le nombre de clients. La charge maximale à 5 tâches pour les 5 clients voit le provisionnement d'un débit de 72 Mo/s par client (soit 1800 $Mo.s^{-1}/25$) qui est bien au delà du seuil de 25 Mo/s défini.

4. http://www.iozone.org

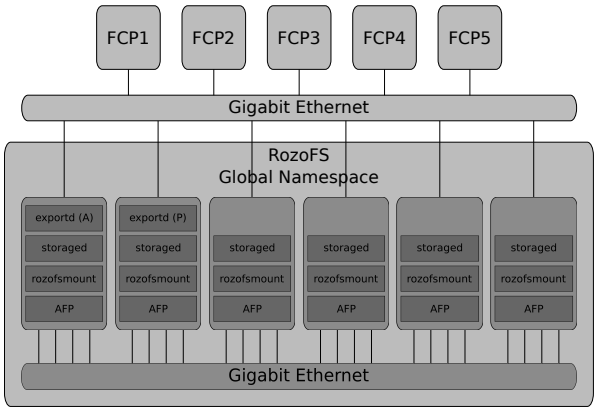



FIGURE 4.8 – Cluster de machines dédiées au montage vidéo en ligne.

En lecture aléatoire, on observe le même comportement (de type *scale out*) en fonction du nombre de clients. Même si la pente est moindre, du fait de ce mode d'accès, le débit provisionné par client avoisine les 30 Mo/s.

La plateforme de montage vidéo est donc validée par l'usage et la mesure.

Ce travail expérimental a fait l'objet d'une publication [66].

Plusieurs critiques peuvent être adressées à cette expérimentation. L'ensemble des tests réalisés n'apportent pas d'élément de comparaison avec des solutions existantes, notamment celles qui utilisent la réplication. La mesure du débit (en Mo/s) n'est en outre pas adaptée pour des accès aléatoires.

### 4.3.3 Expérimentations sur GRID5000

Une autre expérimentation a été réalisée au sein de la grille de calcul GRID5000. L'objectif de cette étude est :

1. comparaison avec un DFS existant i.e CephFS ;
2. conduite des tests en mode "boite noire" (sans maîtrise de l'infrastructure sous-jacente) ;
3. établir le comportement de RozoFS dans un contexte fortement distribué.

Ce travail a été effectué dans le cadre du projet de recherche et développement de Bastien Confais, élève ingénieur 5ème année, Polytech Nantes, département informatique (semestre 9 de l'année 2014-2015) qui a donné lieu à une publication.

#### Différences fonctionnelles avec Ceph

Le tableau 4.3.3 répertorie l'ensemble des différences fonctionnelles. Sur ce point, Ceph dispose d'un net avantage par rapport à RozoFS en proposant un algorithme de placement performant au sein des nœuds (voir [96]) et une gestion fine des versions.

| **Fonctions** | **RozoFS** | **CephFS** |
|---|---|---|
| Versions | 1.5.2 | 0.56.1 |
| Tolérance aux pannes | par codes à effacement Mojette (1 projection par nœud de stockage) | par réplication totale du fichier |
| Cohérence des données | Cohérence des projections garantie | L'accès à la dernière version du fichier est garantie |
| Écriture des données et de ses réplicats | Les clients écrivent en parallèle sur les serveurs | Les clients n'écrivent que sur un serveur (primary OSD). Ce dernier déploie ensuite les réplicats (à la manière de la Figure 4.10 |
| Méta-données | Un serveur de métadonnées (répliqué) | Les méta-données sont placées sur les serveurs de stockage (application de l'algorithme CRUSH[96] par un serveur central) |
| Journalisation | Le serveur de méta-donnée sert de référence (logs système) | Le système de fichier est journalisé. Possibilité de rejouer les transactions |
| Sélection des serveurs | les serveurs les moins chargés du cluster sont choisis | algorithme de placement multi-critères CRUSH[96] |
| Mode *scale up* (ajout d'un serveur à la volée) | Pas de rééquilibrage de charge | Rééquilibrage entre tous les serveurs |

TABLE 4.1 – Différences fonctionnelles entre CephFS et RozoFS (extrait de [18]).

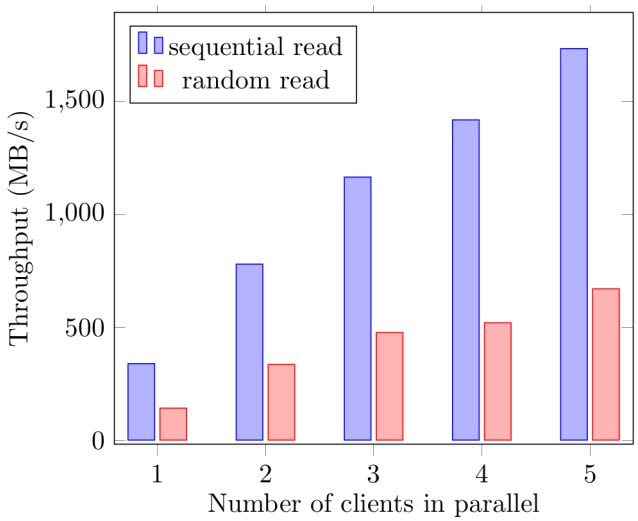



FIGURE 4.9 – Débit cumulé (en Mo/s) enregistré en fonction du nombre de clients pour la lecture (séquentielle et aléatoire).

### Matériels et méthodes

L'expérimentation est un test de charge similaire à la section précédente avec une tolérance aux pannes de deux serveurs de stockage (mode dégradé non testé). Ceci se traduit pour RozoFS par une distribution de type 6 projections dont 4 sont nécessaires et pour Ceph, l'usage de la triplication pour garantir une disponibilité même en cas de panne. Des tailles de fichiers distincts sont utilisés en fonction du mode d'accès. En mode séquentiel, des fichiers de 1 Go sont utilisées alors que des fichiers de 128 Mo sont utilisés pour le mode aléatoire. Les fonctions de lecture et d'écriture sont qualifiées. Dix itérations par test sont effectuées (donc pour un total de 10 Go utile stocké). On utilisera le nombre d'entrées/sorties par seconde i.e. les IOPS pour les mesures en accès aléatoire. Des tailles de blocs de 64 Ko pour l'accès séquentiel et 8 Ko pour l'accès aléatoire sont utilisées. Un nombre de 10 clients max est réservé. Le cluster de stockage comporte 8 nœuds.

Au niveau de Grid5000, nous avons principalement utilisé le cluster Sagittaire (basé à Lyon) pour conduire nos tests. Ce cluster est composé de processeurs de type AMD Opteron 250

| | RAM | Débit disque (écriture) | | Débit disque (lecture) | |
|---|---|---|---|---|---|
| | (Mo) | $\bar{x}$ (Mo/s) | $\sigma$ (Mo/s) | $\bar{x}$ (Mo/s) | $\sigma$ (Mo/s) |
| Serveurs | 2012 | 65.118 | 7.514 | 67.736 | 9.845 |
| Clients | $7 \times 2012$<br>$3 \times 16024$ | non utilisé | non utilisé | non utilisé | non utilisé |

| | | Débit réseau | | Latence réseau | |
|---|---|---|---|---|---|
| | | $\bar{x}$ (Mbits/s) | $\sigma$ (Mbits/s) | $\bar{x}$ (ms) | $\sigma$ (ms) |
| Serveurs | | 678.453 | 285.633 | 0.105 | 0.008 |
| Clients | | 466.667 | 202.441 | 0.104 | 0.009 |

TABLE 4.2 – État (au sens moyenne et écart-type) des disques et du réseau au cours du test des machines client/serveur réservées au sein du cluster Sagittaire (basé à Lyon).

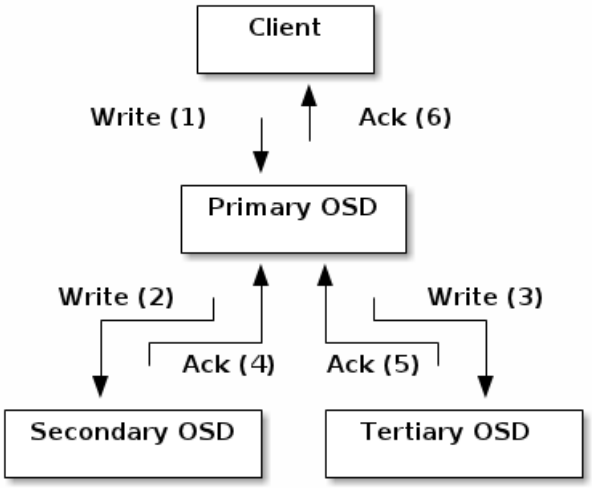


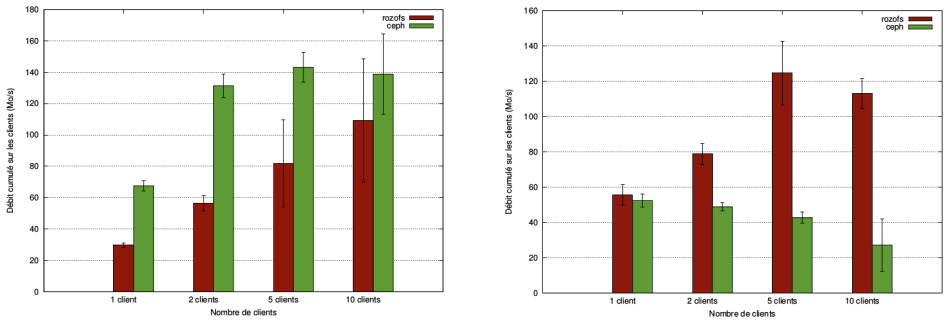

FIGURE 4.10 – Déploiement des réplicats (ici au nombre de 2) au sein du système de fichier CephFS (voir http://ceph.com/docs/master/architecture/).

(cadencés à 2.4GHz) avec un réseau d'interconnexion en Gigabit Ethernet. Le tableau 4.3.3 résume l'état des 8 serveurs de stockage et des 10 clients en matières mémoire, accès disques et réseaux.

**Résultats**

Les Figures 4.11 et 4.12 illustrent les performances des deux DFS en lecture et écriture respectivement pour l'accès séquentiel et pour l'accès aléatoire. Sur ces deux figures, on retrouve le comportement de type *scale up* de Rozo en lecture (séquentiel et aléatoire) de l'expérimentation précédente. En séquentiel (mesure en débit en Mo/s), Ceph présente des meilleures performances que Rozo en lecture. Le constat est inversé pour l'écriture. En aléatoire (voir Figure 4.12), c'est RozoFS qui délivre les meilleurs performances en lecture (avec $14k$ IOPS pour 10 clients). En écriture, la conclusion n'est pas évidente si on considère les écart-types mesurées. Ceph semble délivrer néanmoins le plus grand nombre d'entrées-sorties par seconde (de l'ordre de $6k$ IOPS pour 2 clients). D'autres tests sont nécessaires pour conclure plus finement sur cette fonction d'écriture aléatoire.

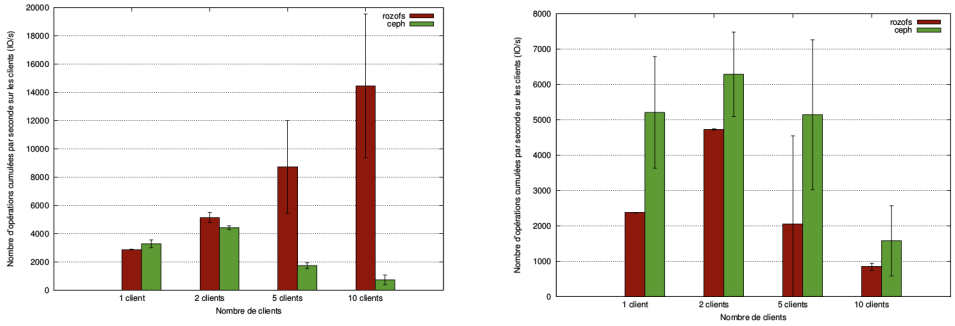
Le tableau de la Figure 4.13 indique le remplissage du cluster pour la charge de 10 Go utile (10 fichiers de 1Go) pour RozoFS et CephFS. La charge de RozoFS v2 est estimée (cette version est pourvue d'une optimisation sur la longueur des projections). On constate ici la charge de la triplication pour Ceph qui multiplie par 3 la charge utile pour assurer une tolérance à deux pannes. L'apport du code à effacement (Mojette) est visible ici sur Rozo qui sollicite la moitié de la charge pour le même taux de disponibilité. En outre, la répartition de la charge pour Rozo sur les 8 nœuds de stockage semble plus équilibrée que pour Ceph qui affiche des différences d'un facteur 3 entre le nœud 1 et le nœud 8.




(a) Lecture séquentielle

(b) Écriture séquentielle

FIGURE 4.11 – Débit cumulé moyen (en Mo/s) pour CephFS et RozoFS pour la lecture et l'écriture séquentielle.




(a) Lecture aléatoire

(b) Écriture aléatoire

FIGURE 4.12 – Nombre moyen d'entrées-sorties cumulées par seconde (IOPS) pour CephFS et RozoFS pour la lecture et l'écriture aléatoire.

### Conclusion de l'expérimentation sur GRID5000

L'analyse des performances sur les fonctions de lecture et d'écriture tant séquentielles qu'aléatoires ne permet pas de départager clairement les deux DFS pour le cluster considéré. En revanche, le volume de données stocké est deux fois moindre pour RozoFS pour une même tolérance aux pannes. On retrouve, par l'expérience, l'intérêt des codes à effacement en matière de réduction de l'espace de stockage global. D'autres résultats (en cours de consolidation) ont été obtenus sur le cluster de Nantes qui est beaucoup plus performant que le cluster Sagittaire. Les gains en débits de RozoFS sont plus nets par rapport à Ceph dans ce contexte pour l'ensemble des fonctions d'entrées/sorties testées ce qui laisse penser que Rozo s'adapte mieux que Ceph à des évolutions matérielles du système de stockage.

| | Serveurs de stockage (Go) | | | | | | | | Total (Go) |
|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | |
| rozoFS | 2.4 | 2.4 | 2.1 | 2.1 | 2.1 | 2.1 | 2.1 | 2.1 | 17.4 |
| rozoFS v2 | 2.0 | 2.0 | 2.0 | 2.0 | 2.0 | 1.8 | 1.8 | 1.5 | 15.3 |
| ceph | 6.0 | 5.3 | 5.0 | 5.0 | 4.7 | 4.3 | 2.2 | 2.0 | 34.5 |

FIGURE 4.13 – Répartition de la charge et capacité totale du cluster pour le stockage de 10 Go de fichiers utiles pour CephFS et RozoFS (estimation pour RozoFS v2).

## 4.4 Code à effacement et Big Data

Ce travail a été effectuée dans le cadre du stage master de Pauline Folz, Master 2 Recherche ALMA (Architecture Logicielle et Systèmes Distribués), Faculté des Sciences et Techniques, Université de Nantes (soutenu en septembre 2013).

### 4.4.1 Éléments de contexte

Cette section vise à positionner le travail en présentant l'algorithme MapReduce, référence dans le traitement distribué de données massives ainsi, que deux travaux expérimentaux représentatifs évaluant les performances de MapReduce sur des données encodées.

#### MapReduce par l'exemple

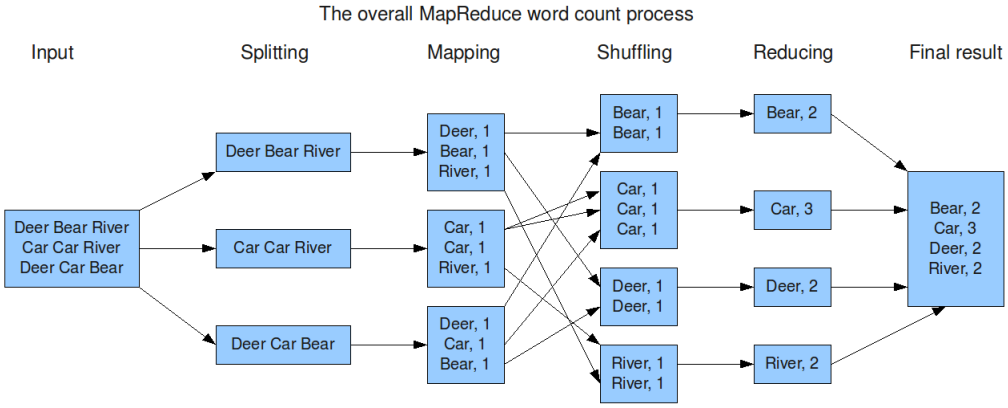



FIGURE 4.14 – Exemple d'un traitement analytique de type *wordcount* (ou comptage de nombre d'occurrences d'un mot) au travers de l'algorithme MapReduce - Extrait de http://blog.trifork.com/2009/08/04/introduction-to-hadoop/.

Le principe de MapReduce consiste à fragmenter un jeu de données massif en entrée pour l'affecter au sein d'unités de traitement et de stockage distribuées. La philosophie de l'approche consiste à déplacer les traitements et non les données. Le traitement va se faire en effet à l'endroit où est stockée la donnée.

L'exemple de la Figure 4.14 permet de comprendre les phases de l'algorithme au travers d'un exemple simple de comptage du nombre d'occurrences de mots (le traitement analytique *wordcount*). La première phase consiste à produire 3 fragments du jeu de données initiales. Sur chacun des fragments, le traitement est effectué de manière indépendante. Une phase de *shuffle* collecte les résultats au sein des différentes unités de traitement de manière exhaustive. Une dernière phase (*reduce*) vise à résumer sous la forme d'un couple clé-valeur le résultat (en supprimant les doublons éventuels) pour présentation finale.

L'objectif visé est bien entendu la rapidité des traitements au regard de la masse de données en présence. Pour remplir cet objectif, un traitement centralisé aurait supposé soit une ressource spectaculaire (de type supercalculateur) soit un temps infini pour parvenir à la solution finale. Il convient de noter que cette approche suppose que le jeu de données est fragmentable. Les données images et vidéos le sont difficilement.

HDFS (*Hadoop Distributed File System*) fournit l'environnement ad hoc pour l'exécution de l'algorithme MapReduce avec un système de fichier tolérant aux pannes (par réplication en natif) et des blocs de données par défaut de 64 MB.

#### Travaux relatifs

La première étude concerne un travail mené par Microsoft Research en 2010 [103]. Les traitements qui vont suivre concernent :

- la fonction de tri (*sort*) ;
- la fonction de comptage *wordcount* ;
- un calcul intensif de type séquençage ADN (jeu de données *CloudBurst*).

Pour cette expérience, le cluster est composé de 8 nœuds interconnectés par un réseau Ethernet 1 Gbps. Le jeu de données relatif aux traitements *wordcount* et *sort* est de taille 6 Go. *Cloudburst* est de taille 4 Go. Au niveau code à effacement, la librairie Jerasure est utilisée (implémentation `Cauchy-good` de Reed-Solomon en mode systématique [72]). L'application du code se fait au niveau de la granularité du bloc HDFS de 64 MB i.e l'association de $k$ blocs produit $n - k$ blocs de parité.

La Figure 4.15 montre les temps d'exécution des traitements *sort*, *wordcount* et *CloudBurst* en fonction du mode de répartition i.e réplications ou codes à effacement (3 parmi 5 ou 5 parmi 7). Le temps d'exécution est décomposé en phase de *map*, de *shuffle* et de *reduce*.

Le tri se caractérise par un nombre d'écritures importantes. Le temps d'exécution croit linéairement avec le nombre de réplication alors que l'approche par code délivre un temps d'exécution constant quel que soit le mode de répartition. Le volume de données écrites sur le réseau et sur le disque (réduit dans le cas des codes) explique ces performances.

Pour le traitement analytique *wordcount*, les données de sortie sont assez réduites. Les approches par réplication et par codes ne sont pas réellement discriminantes.

Pour *CloudBurst*, la sortie est également réduite en taille comme avec la fonction *wordcount*. Cependant, les données intermédiaires sont importantes. En conséquence, les auteurs constatent le même comportement qu'avec la fonction de tri avec un réel avantage pour les approches par code.

En substance donc, les codes à effacement (ici Reed-Solomon systématique) ont tendance à réduire les temps d'exécution de par la réduction du volume de données qu'ils réalisent. Avec un cluster et des jeux de données sensiblement différents (61 machines interconnectées en 10 GbE avec des jeux de 120 Go), les auteurs de [28] parviennent aux même conclusions.

Cependant, ces deux études ([28, 103]) font un usage des codes au niveau bloc HDFS. En utilisant de surcroît une forme systématique, il paraît normal de ne pas voir d'incompatibilité entre les traitements distribués et l'approche par code, voire une amélioration des performances, au vu de la réduction du nombre d'écritures sur le système distribué, pour une tolérance aux pannes identique. Qu'en est-il lorsque ce découpage concerne de plus petits blocs comme il est nécessaire de manipuler dans le cadre de système de fichiers avec entrées/sorties intensives comme CephFS ou RozoFS ? Est-ce que le paradigme qui consiste à déplacer les traitements sur les données reste valable dans ce contexte ?

### 4.4.2 Matériels et méthodes

Dans cette nouvelle expérimentation, nous avons pour objectif la comparaison d'HDFS et de RozoFS. Ces deux DFS ont deux approches différentes dans la mesure où le premier prône le déplacement des traitements sur les données alors que le second nécessite de déplacer les données sur les unités de calcul.

L'approche HDFS est bien illustrée à la Figure 4.16 où l'on voit des *job* MapReduce à l'intérieur des unités de stockage (des *DataNodes* en langage *Hadoop*). Ceci est possible dans la mesure où le fichier de départ est fragmenté en 4 blocs (de 64 MB par défaut) qui vont être répliqués dans les unités de stockage. Pour RozoFS, cette approche n'est pas possible puisque les blocs sont codés Mojette de manière non systématique. Il faut donc, au préalable du job MapReduce, rapatrier le nombre de projections nécessaires. On peut donc s'attendre à une légère augmentation du temps d'exécution. L'objectif de cette étude est d'avoir une première estimation. L'avantage demeure la réduction du volume de données par rapport à l'approche par réplication. Une API est spécialement développée pour faire fonctionner MapReduce sur un cluster RozoFS.

Le cluster utilisé est celui du projet FEC4Cloud composé de 8 machines (Dell R320 : disque dur SATA 7200 et processeur Intel Xeon 4 cœurs à 1.8 GHz) interconnectées par un réseau Gigabit. Le jeu de données est un fichier de textes Twitter de 700 Mo. Les traitements expérimentés sont les fonctions *wordcount* et *grep*. On mesure les temps d'exécution (en minutes) pour RozoFS et HDFS moyennés sur 10 essais pour chaque DFS. Les écarts-types ne sont pas suffisamment significatifs pour être relevés.

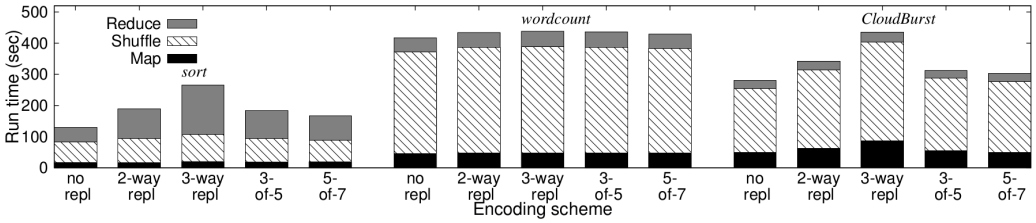



FIGURE 4.15 – Temps exécution (sec.) des traitements *sort*, *wordcount* et *CloudBurst* en mode MapReduce (extrait de [103]).

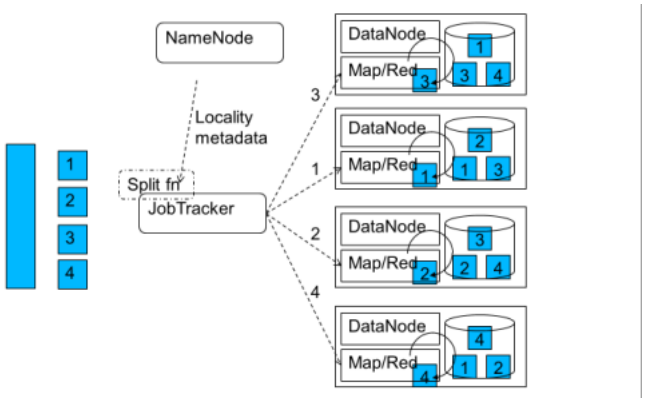



FIGURE 4.16 – Interactions entre l'algorithme MapReduce et le système de fichier distribué HDFS (source : http://www.datafactz.com/blog/2014/11/12/).

### 4.4.3 Résultats

Les résultats de l'expérimentation sont donnés à la Figure 4.17 en matière de temps d'exécution pour chaque traitement et chaque DFS. Le temps CPU est environ 4 fois plus que le temps d'exécution pour la simple raison que nous sommes en présence d'une architecture 4 cœurs. Ils fournissent la répartition des phases *Map* et *Reduce* de l'algorithme.

Les résultats obtenus ne nous permettent pas de discriminer les deux approches. On peut néanmoins conclure que l'approche proposée par RozoFS à savoir le déplacement des données sur les unités de traitement n'est pas rédhibitoire dans ce contexte d'expérimentation.

## 4.5 Conclusion

Ce chapitre porte sur les principaux résultats du projet FEC4Cloud (ANR-12-EMMA-0031). Outre l'état de l'art établi en faveur des codes à effacement, il décrit les très bonnes performances d'un code Mojette non systématique par rapport aux librairies de références ISA-L d'Intel et Jerasure de James Plank. Les expérimentations du systèmes de fichier RozoFS (qui intègre le code Mojette non systématique) sur la plateforme du projet d'une part, et sur la grille de calcul GRID5000 d'autre part démontrent la pertinence d'utiliser ce type de tolérance aux pannes pour des applications E/S intensives tel que le montage vidéo en ligne. Les volumes stockés sont deux fois moindre avec l'approche par code.

La dernière expérimentation vise à vérifier la compatibilité d'une approche par code avec des traitements distribués pour le BigData. Le but poursuivi notamment dans [28] et [103] est classiquement de réduire le volume de données stockées et écrites, sans pénaliser les temps d'exécution en appliquant les codes au niveau de blocs HDFS, soit des tailles de 64 MB. Les gains en temps d'exécution sont réels mais ils ne constituent pas en soi une rupture avec le paradigme qui prône le déplacement des traitements sur les données. Dans RozoFS, le code Mojette est non systématique. La distribution des traitements va nécessairement solliciter un déplacement de données. Cette approche suppose un code à effacement ultra-rapide et un réseau performant. L'expérience conduite sur la plateforme FEC4Cloud n'apporte pas de contre-indication à cette façon de faire. Il convient d'envisager des expériences complémentaire notamment sur un cluster beaucoup plus

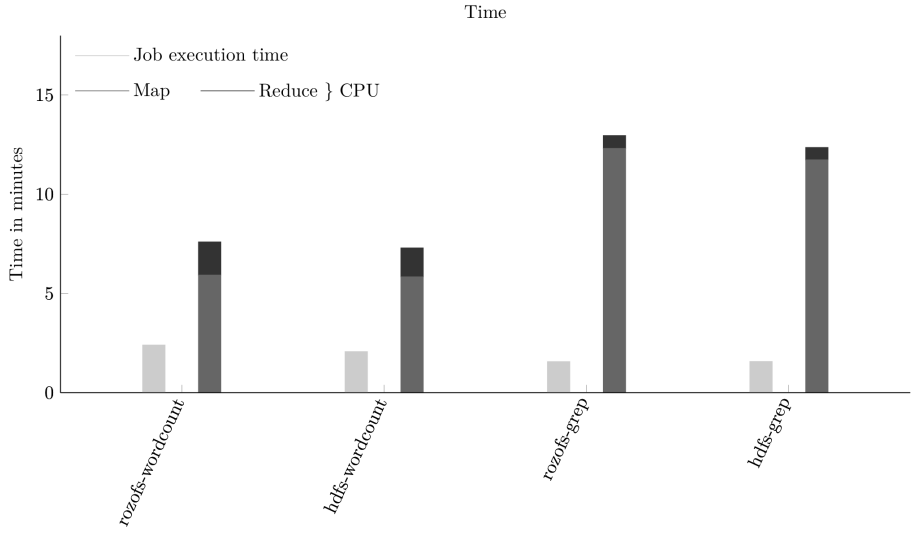



FIGURE 4.17 – Temps d'exécution moyen pour les deux traitements analytiques *wordcount* et *grep* effectués sur RozoFS et sur HDFS.

important (type Grid5000) et avec des jeux de données et des traitements discriminants. L'enjeu est de taille puisqu'il concerne l'usage d'un seul DFS pour des applications de type E/S intensifs et pour des applications de type BigData.

# Chapitre 5

# Un protocole hybride P2P et Client/Serveur (le projet P2PWeb)

## 5.1 Introduction

La différence entre le *Cloud Storage* et les réseaux pair-à-pair (P2P) est assez ténue. La mauvaise presse des réseaux de partage de fichiers est certainement la principale raison de cette ligne de démarcation arbitraire. On peut néanmoins évoquer le fait que dans le cas des réseaux P2P, l'utilisateur final fournit une partie de l'infrastructure en rendant accessible une partie de son espace disque à tous. Même s'il est très populaire, ce cas d'usage particulier n'est cependant pas le seul possible pour la pléthore de protocoles P2P. Et il est vraisemblable que nous n'ayons cesse de nous inspirer des technologies P2P dans le monde du Cloud (avec un nécessaire habillage pour le rendre politiquement correct). La récente solution (propriétaire) proposée par la société Scality va dans ce sens. En revisitant des protocoles ancestraux comme Chord [89] en mode Cloud en y ajoutant un code à effacement de type *Information Dispersal Algorithm* [75]), les concepteurs du "Scality Ring" affichent des performances compatibles à des cluster de l'ordre du *Peta Bytes*.

Ce chapitre tente également de réhabiliter dans la mesure du possible les réseaux P2P. L'idée fondatrice présentée ici est l'association des deux modèles majeurs qui régissent les applications et services sur l'Internet à savoir le modèle client/serveur et le modèle pair-à-pair. L'idée n'est pas de moi mais du Dr Soufiane Rouibia, responsable de la R&D au sein de la société Trident Média Guard (TMG). Cette idée a par ailleurs germé à différents endroits de la planète donnant lieu aujourd'hui à des déploiements d'applications très connues tels que Wikipédia, Spotify ou plus récemment StreamRoot pour la fourniture de contenus multimédias. Soufiane s'est présenté à moi avec la ferme intention de qualifier objectivement son idée de protocole notamment en matière de Qualité de Service (QoS) et de Qualité d'expérience (QoE). J'y ai ajouté naturellement l'idée de tolérance aux pannes et de haute disponibilité par l'association du modèle d'architecture hybride et du code à effacement. Le docteur Patricia Serrano-Alvarado, membre du Laboratoire d'Informatique Nantes Atlantique (LINA), spécialiste de l'anonymisation et de la recommandation au sein des systèmes distribués a rejoint rapidement notre groupe pour former le projet P2PWeb (projet OSEO/Région d'une durée de 24 mois entre septembre 2010 et 2012). Cet aspect du

projet porté par Patricia ne sera pas développé dans ce document. Le lecteur intéressé est invité à se rendre autour des travaux du projet PriServ [37] visant à supporter le respect de la vie privée au sein des réseaux P2P. Nous allons dans ce chapitre nous concentrer sur les spécifications du protocole P2PWeb ainsi que sur l'analyse des performances du point de vue QoS et QoE lorsque les deux modèles client-serveur et pair-à-pairs sont associés. Des optimisations sont également présentés en fin de chapitre.

## 5.2 Le protocole P2PWeb

Cette section présente les spécifications du protocole, la plateforme de test composée d'une centaine de machine virtuelles et les performances de l'architecture hybride en termes de QoS/QoE.

### 5.2.1 Spécifications générales

Le schéma de la Figure 5.1 résume les composants du protocole. Il met en jeu un serveur de contenu ainsi qu'une population de clients (spécifiques P2PWeb) naturellement connectés au travers d'un navigateur `http` standard. Passé un seuil de saturation, un ensemble de pairs s'auto-organise (avec l'aide du serveur) pour récupérer le contenu par l'intermédiaire des clients déjà connectés. Des mesures de QoS/QoE établissent une voie de retour à destination du serveur qui peut modifier à la demande sa liste de clients ou de pairs contributeurs (des *seeders* en langage BitTorrent).

En matière d'événements, nous pouvons résumer le protocole de la manière suivante :

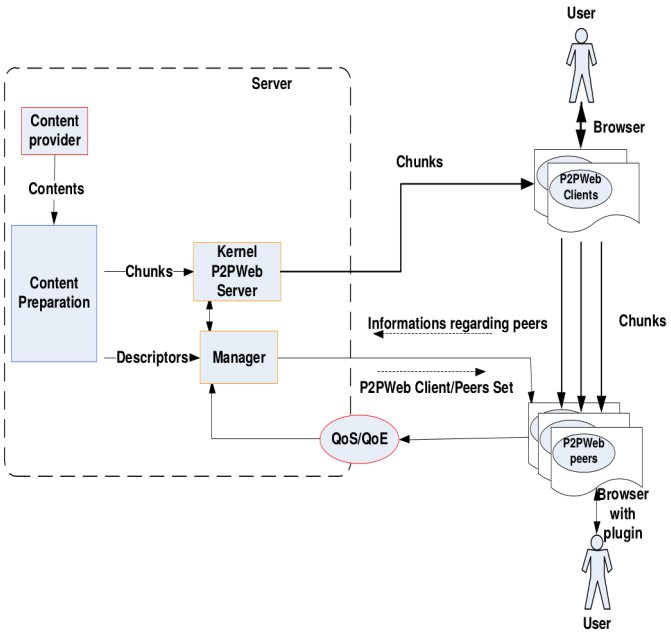



FIGURE 5.1 – Schéma général du protocole P2PWeb incluant de manière hybride un serveur de contenu (à gauche) et une population hétérogène de clients et de pairs (à droite).

- le serveur reçoit des demandes de connexions de la part de clients distants ;
- le nombre important de demandes de connexions oblige le serveur à passer en mode surchargé et à déléguer la fourniture du contenu (principalement audio-visuel) via ses clients ;
- les nouveaux arrivants reçoivent du serveur une liste de clients susceptibles de leur fournir le contenu requis formant ainsi un réseau de pairs ;
- les clients retransfèrent les morceaux de fichiers déjà reçus à destination des nouveaux arrivants en allégeant la charge du serveur central.

En procédant ainsi, nous évitons simplement la duplication d'un serveur de contenus qui reste une opération très coûteuse quand elle est souscrite auprès d'un opérateur CDN (*Content Delivery Networks*) [1]. Les utilisateurs unifiés par un contenu s'auto-organisent au sein d'un réseau pair-à-pair complémentaire à l'architecture client/serveur initiale.

Pour la réalisation d'un prototype, il est possible de s'appuyer sur des protocoles élémentaires type `http` (pour la partie Client/Serveur) et le protocole `BitTorrent` (pour la partie P2P).

Outre une implémentation standard, il est intéressant de répondre aux problématiques suivantes : *(i)* Comment mesurer en termes de QoS l'apport du réseau P2P à une architecture Client/Serveur ? *(ii)* N'est-il pas possible d'améliorer la réactivité du réseau P2P en permettant plus rapidement l'intégration des nouveaux arrivants aux réseaux P2P ? *(iii)* Est-ce que ce protocole hybride est bien adapté aux contenus à contraintes temporelles comme des vidéos ?

### 5.2.2 Matériels et méthodes

Un premier prototype de *plugin* P2PWeb a été développé par la société TMG pour des navigateurs `http` standard. Ce *plugin* se base sur une mise en œuvre *open source* du protocole BitTorrent en langage Python.

Pour valider l'approche P2PWeb et constituer une preuve de concept, une plateforme de test est réalisée spécialement. Celle-ci est représentée à la Figure 5.2. Elle se compose d'un serveur de contenu et de deux clusters respectivement de 23 et 50 machines virtuelles. Cette partition artificielle résulte de l'usage de deux machines hôtes distinctes (deux serveurs XEN). Chaque machine possède une adresse IP public de manière à décloisonner le trafic du cluster vers l'extérieur (l'Internet public). En matière de contenu, un simple fichier JPEG de 39 Mo va l'être l'objet du partage. Cette taille correspond à la taille standard d'un buffer HD en streaming (en prévision de travaux futurs sur le transfert de contenu de type *live*). Le pair de mesure va être situé derrière un pare-feux comme c'est le cas pour de nombreux terminaux connectés à l'Internet [25]. En outre, les mesures sont en général limitées à 60 secondes. Cette minute correspond au TTL (*Time-To-Live*) de la fourniture d'un fragment de fichier (un *chunk* dans le jargon) pour beaucoup d'applications P2P [13]. La taille du chunk est fixée, de manière classique pour des réseaux de type BitTorrent, à 512 octets.

Au niveau de la topologie du réseau, les liens montants des clients et des pairs sont fixées à 300 Ko/s. Aucune limite sur les débits descendants n'est fixée. Hormis la première expérimentation, les clients sont considérés comme détenteurs de l'intégralité du contenu au même titre que le serveur central. Bien entendu, la répartition clients-pairs va jouer un rôle sur les mesures, de

1. 1 million de dollars par jour dépensés en bande passante pour YouTube en 2009 [14]

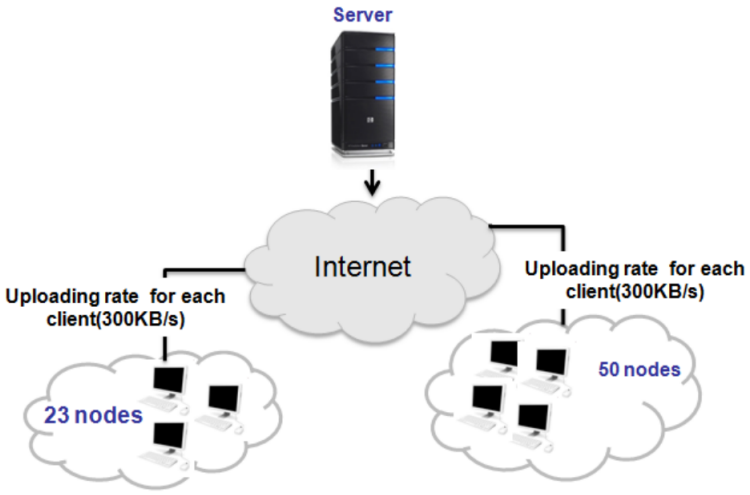



FIGURE 5.2 – Plateforme de tests du projet P2PWeb composé de deux clusters de 23 et 50 nœuds.

même que le nombre de pairs sollicitant le contenu générique, et le débit montant accordé par les clients/pairs pourvoyeurs.

Nous allons mesurer classiquement le temps de téléchargement d'un contenu pour le pair de mesure ainsi que les débits pour les différentes stratégies et topologies réseaux. L'état des connexions (vers des clients ou vers d'autres pairs) va être monitoré. Le temps nécessaire au pair de mesure pour débuter son téléchargement sera examiné aussi comme un critère de performance (classée ici comme critère de QoE).

Chaque nœud est introduit au sein du réseau à période fixe de 0,2 secondes. Le pair de mesure est introduit en dernier. Il faut donc environ 15 secondes pour lancer l'ensemble des 73 machines virtuelles. Lorsqu'il s'agit de mesurer le comportement du pair de mesure, le "T0" correspond au moment où la machine est entièrement démarrée.

### 5.2.3 Résultats

**Première expérimentation : comparaison avec une architecture client/serveur simple**

Cette première expérimentation vise à montrer l'intérêt d'une architecture hybride vis-à-vis d'une architecture uniquement client serveur. Nous sollicitons pour cela le première cluster de 23 machines. Dans un cas, les 23 machines virtuelles agissent toutes comme des clients avec une requête de type `http` sur le contenu générique hébergé par le serveur. Dans le scénario P2PWeb, les 8 premiers nœuds lancés se comportent comme des clients classiques. Le 9ème nœud et les suivants ne reçoivent que du serveur central une liste de pairs composée de ses 8 clients connectés. Ils s'auto-organisent ensuite autour d'un réseau P2P de substitution.

La Figure 5.3 montre les temps nécessaires de téléchargement (complet) du contenu générique en fonction de l'architecture et du nombre de nœuds qui composent le réseau. Si le modèle client-serveur délivre un temps de téléchargement linéaire avec le nombre de clients connectés, celui-ci décroit drastiquement en présence de l'architecture mixte P2PWeb (courbe en pointillés). À

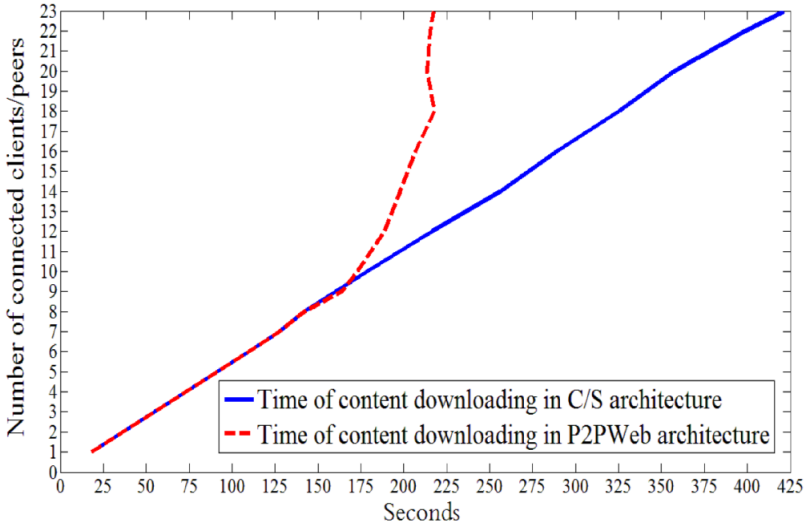



FIGURE 5.3 – Comparaison des temps de téléchargement entre une architecture client/serveur simple et une architecture hybride P2PWeb.

partir de l'introduction du 9ème nœud, nous tendons vers un temps de téléchargement constant quel que soit le nombre de clients en présence.

Ce premier résultat illustre bien l'intérêt d'une architecture hybride par rapport à une architecture client/serveur standard. Dans la suite, nous considérerons uniquement une population hybride en faisant varier le nombre de clients par rapport au nombre de pairs et en sollicitant le deuxième cluster de 50 nœuds portant ainsi le réseau expérimental à 73 nœuds au total.

#### Scénario avec 10 clients et 63 pairs

Pour ce scénario, les 73 nœuds sont répartis comme suit. Les 10 premiers arrivant se comportent comme des clients. Les 63 autres (dont le pair de mesure) s'auto-organisent autour du réseau P2P. À la manière du réseau BitTorrent, le serveur délivre une liste de 50 nœuds P2PWeb aux nouveaux nœuds arrivants (dont le pair de mesure). En examinant le type de connexion, on constate que le pair de mesure sollicite les 10 clients déjà en place plus une petite vingtaine de pairs (18 pour être exact) qui ont déjà reçu une partie du contenu. La Figure 5.4 témoigne de l'état (statique) des connexions du pair de mesure lors des 60 secondes nominales. Plus de changements topologiques (en matières de type de connexions) sont à attendre dans le scénario suivant (voir de manière anticipée la Figure 5.6 page 72).

Les débits de téléchargement enregistrés au niveau du pair de mesure sont représentés à la Figure 5.5 pour une période de 60 sec. Le débit va croissant linéairement (jusqu'à 350 Kb/s) au fur et à mesure que les pairs précédents ont téléchargé l'intégralité du contenu, libérant ainsi de la bande passant pour le pair de mesure. Outre le débit de téléchargement, la courbe renseigne également le débit montant en provenance des autres pairs qui est dans ce cas très faible. La majeure partie du trafic est donc issue des clients redistributeurs et non des autres pairs. Les nouveaux pairs arrivant (comme le pair de mesure) sont en fait assez mal accueillis

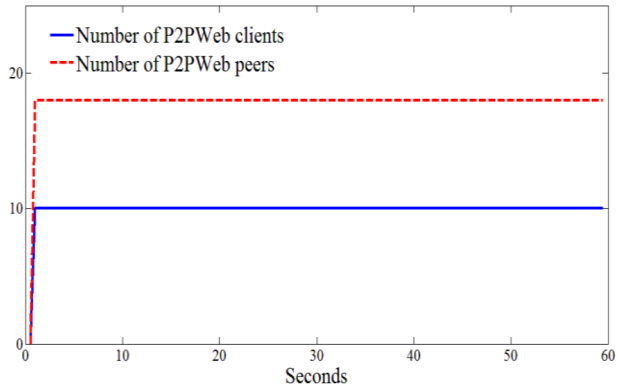



FIGURE 5.4 – Répartition des connexions du pair de mesure pour 10 clients et 63 pairs.

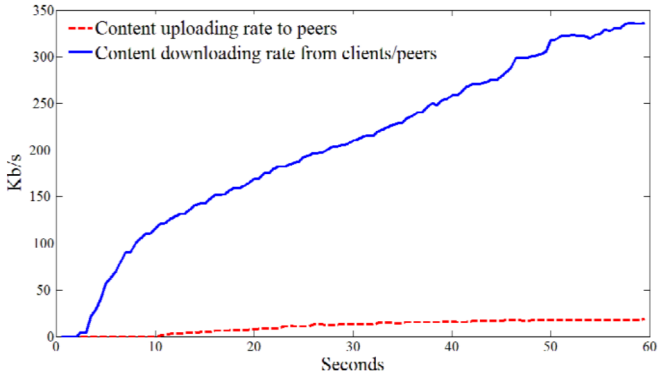



FIGURE 5.5 – Débits descendants et montants enregistrés pour le scénario avec 10 clients et 63 pairs.

par les autres pairs pour la simple raison qu'ils n'ont aucun contenu à échanger. Le principe du "donnant-donnant" (*tit for tat* en anglais) cher aux réseaux P2P s'applique ici.

En termes de pourcentage de contenu téléchargé, on enregistre au niveau du pair de mesure seulement 45% de contenu téléchargé à l'issue des 60 sec avec la topologie de 10 clients pour 63 pairs [2]. Ce résultat n'est pas satisfaisant et oblige à revoir la répartition du nombre de clients et du nombre de pairs.

### Scénario avec 25 clients et 48 pairs

Dans ce scénario, la capacité du serveur à satisfaire un plus grand nombre de clients est augmentée. Cette fois, 25 clients sont connectés directement au serveur pour 48 pairs qui vont requérir le contenu sur la base d'une liste de pairs/clients.

La Figure 5.6 fournit l'état des connexions avec le pair de mesure. Des changements topologiques sont notables par rapport aux connexions établies dans le scénario précédent. Les

2. les résultats complets en matières de taux de téléchargement sont disponibles dans [85]

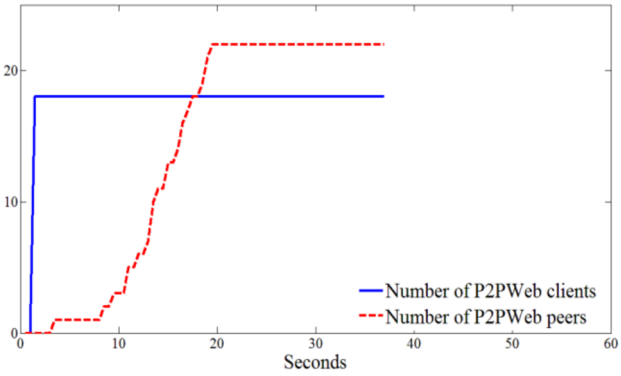



FIGURE 5.6 – Répartition des connexions du pair de mesure pour 25 clients et 48 pairs.

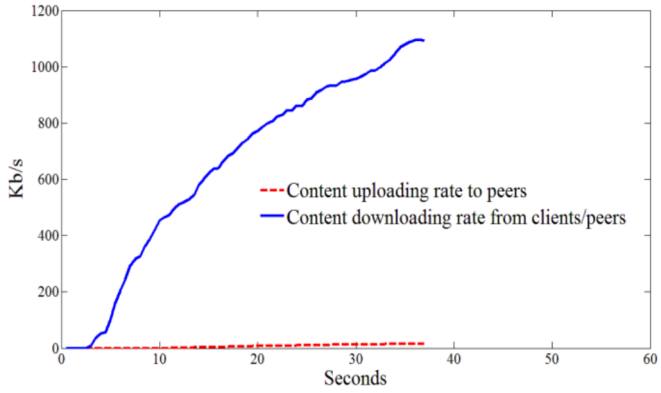



FIGURE 5.7 – Débits descendants et montants enregistrés pour le scénario avec 25 clients et 48 pairs.

clients sont naturellement sollicités en premier. Ils possèdent en effet l'intégralité du contenu. Les connexions avec les autres pairs s'établissent progressivement pour atteindre un nombre total de 50 nœuds qui correspond à la liste des pairs fournie par le serveur.

Pour cette répartition client/pairs, les débits de téléchargement ont significativement augmenté à l'image des résultats obtenus en Figure 5.7 pour atteindre près de 1200 Kb/s. Rien de surprenant. On dispose en effet d'un plus grand nombre de sources relais qui demeurent néanmoins des clients qui partagent (à hauteur de 300 Ko/s de leur débit montant) et non des serveurs dupliqués comme dans une approche de type CDN.

Cette fois, contrairement au scénario précédent, l'image JPEG désigné comme contenu de référence est entièrement téléchargée et ce en moins de 40 secondes. Dans ce scénario, l'apport des clients au sein du réseaux de partage P2P est mis en évidence. L'hybridation des nœuds à la fois client et pairs (*seeders*) a évité l'ajout coûteux d'un deuxième serveur de contenu. Cette expérimentation a été présentée lors du Colloque Francophone pour l'Ingénierie des Protocoles (CFIP) de 2012 [85].

La participation des pairs reste cependant extrêmement basse à l'image des débits montants mesurés en Figure 5.7 (courbe en pointillés). Ce constat nous amène à proposer des optimisations

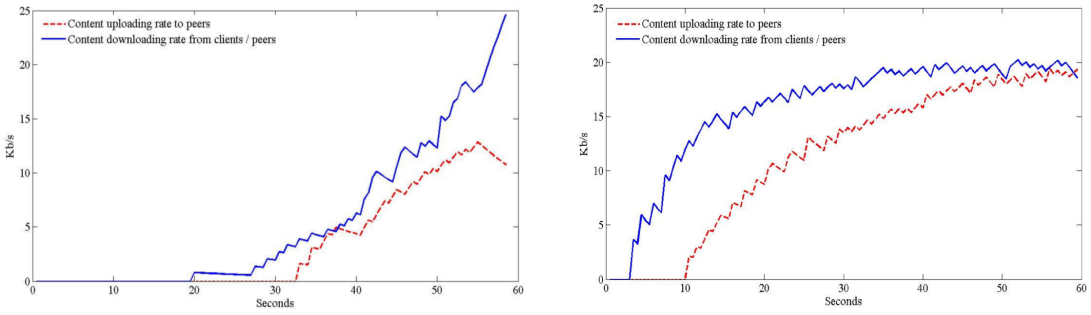



FIGURE 5.8 – Débits descendant et montant enregistrés dans le cas d'un *tracker* passif (figure de gauche) et d'un *tracker* actif (figure de droite).

qui vont favoriser l'intégration des nouveaux pairs jusqu'alors pénalisés par le principe *tit for tat* des réseaux P2P.

### 5.2.4 Optimisation du protocole

Dans les deux scénarios précédents, on a pu constater la faible participation des pairs à relayer le contenu. Le principe du "donnant-donnant" en est la cause. Dans cette section, nous proposons deux optimisations simples qui visent à faciliter les échanges pairs à pairs.

#### Modification de l'activité du tracker

La première concerne l'activité du *tracker* centralisé au niveau du serveur. Celui-ci fournit en plus de la liste de clients/pairs un morceau du fichier partagé (le *chunk*) de manière à rendre les nouveaux venus plus attractifs. Ce c*chunk* est choisi dans un premier temps de manière aléatoire par le *tracker*. L'expérience qui suit vise à montrer l'intérêt d'un *tracker* actif (qui fournit un chunk aléatoire en plus de sa liste) vis-à-vis d'un *tracker* inactif (qui fournit seulement sa liste de clients/pairs). Pour rappel, la taille du *chunk* est fixée à 512 octets (soit une part infime du contenu de référence de taille 39 MB).

Pour ce faire, les débits montants sont réduits à 60 Ko/s et seuls 5 clients sont supposés être en connexion avec le serveur (et donc, détenteurs du contenu de référence dans son intégralité). Les 68 autres nœuds sont des pairs.

La Figure 5.8 montre les débits descendants et montants enregistrés sur les 60 premières secondes de test en fonction de l'activité du *tracker*. La figure de gauche témoigne des débits et du temps nécessaire à initier les échanges lorsque le *tracker* est inactif. On enregistre près de 20 secondes nécessaires (soit le tiers du temps de référence) pour débuter le transfert des données vers le pair de mesure. En revanche, lorsque le *tracker* est actif (figure de droite) en délivrant un *chunk* d'amorce, ce temps de démarrage est réduit à une poignée de secondes et les débits moyens enregistrés sont significativement supérieurs.

L'ajout d'un chunk (ici aléatoire) a significativement amélioré la qualité de service pour les nouveaux arrivants en réduisant la latence induite au démarrage des échanges ainsi que les débits de téléchargement.

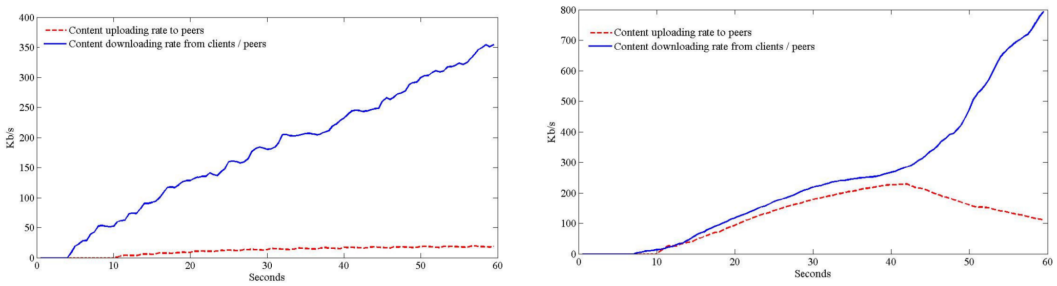



FIGURE 5.9 – Débits descendant et montant enregistrés dans le cas de la fourniture d'un chunk aléatoire (figure de gauche) et d'un chunk rare (figure de droite).

#### Fourniture d'un *chunk* rare

La seconde optimisation concerne le transfert à des nouveaux pairs d'un *chunk* rare (qui est peu présent dans le reste du réseau P2P) de manière à être encore plus rapidement accepté par la communauté de pairs déjà en place (le *swarm*).

Pour cette expérience, nous remontons les débits montant des clients et des pairs aux 300 Ko/s d'origine (le débit précédent à 60 Ko/s nous ayant permis de récupérer seulement 20 % du contenu du fichier image). La répartition clients/pairs est respectivement de 10 et 63.

La Figure 5.9 montre les débits descendants et montants enregistrés sur les 60 premières secondes de test selon le mode de sélection du *chunk* à envoyer par le *tracker*. La figure de gauche concerne une sélection aléatoire du chunk envoyé. On constate une faible activité des pairs qui reçoivent principalement le contenu de la part des clients connectés au serveur. En revanche, lorsque ce mode de sélection change pour l'envoi d'un *chunk* rare au sein du *swarm*, le comportement des pairs se modifie comme l'illustre la figure de gauche. Dans ce cas, les pairs participent plus activement au retransfert des morceaux de fichiers en augmentant significativement le débit moyen de téléchargement. Le débit a en effet quasiment doublé, portant le taux de transfert à 80% contre 45% lorsque la sélection du chunk à envoyer est faite de manière aléatoire. Ce comportement est transitoire puisque rapidement le *chunk* devient commun au sein du *swarm* (observation au delà des 40 secondes).

La sélection d'un *chunk* rare au sein de la communauté de pairs a facilité significativement l'intégration des nouveaux arrivants en améliorant le critère de taux de téléchargement effectué.

## 5.3 Conclusions et perspectives

Dans ce chapitre, nous avons présenté les résultats principaux du projet P2PWeb qui a donné naissance un protocole du même nom. Le concept du protocole est de fournir un modèle mixte client/serveur et pair-à-pair lorsqu'un contenu web (vidéo en particulier) est fortement demandé. Dans ce modèle hybride, les clients prennent le relais d'un serveur de contenu qui est saturé en s'auto-organisant avec des nouveaux arrivants autour d'un réseau P2P spontané qui va permettre le partage du contenu. Ce principe évite la coûteuse duplication du serveur de type CDN (*Content Delivery Networks*).

Plusieurs expérimentations ont été conduites au travers d'une plateforme de 73 machines virtuelles adressées au sein de l'Internet public. Les résultats de la section 5.2.3 page 69 illustrent

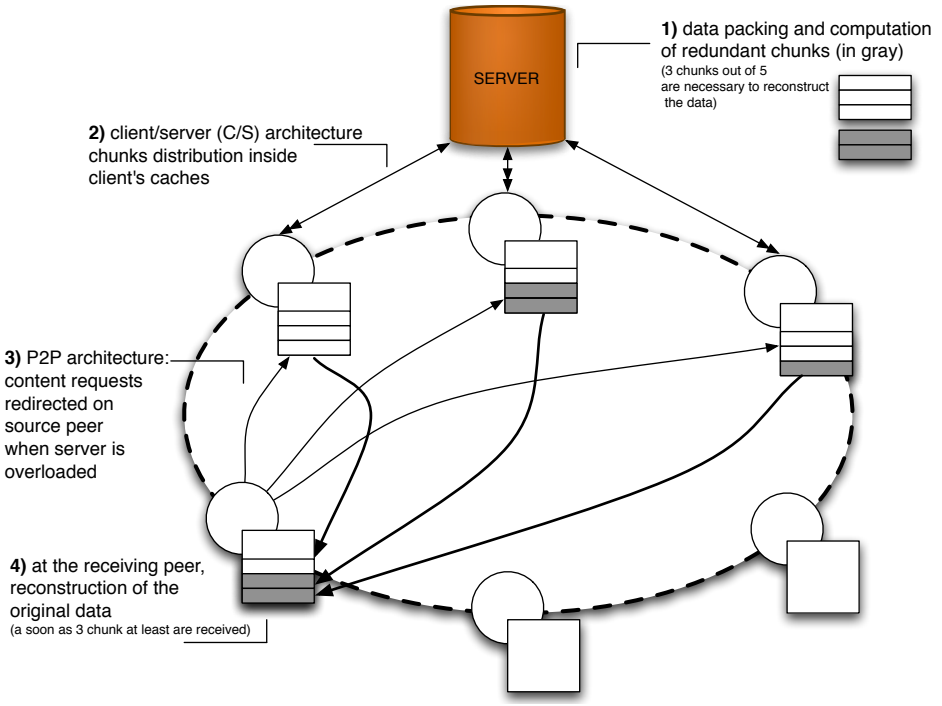



FIGURE 5.10 – Schéma général du protocole P2PWeb incluant un code à effacement systématique.

clairement l'intérêt de l'approche mixte par rapport à une approche client/serveur simple. Les temps de téléchargement peuvent être significativement réduits à l'image de la Figure 5.3 de la page 70. Des scénarios faisant varier le nombre de clients et de pairs ont permis d'analyser le protocole P2PWeb plus précisément du point de vue topologique ainsi que du point de vue de la QoS avec observation des débits de téléchargement montants et descendants. Naturellement, plus le contenu est disponible au niveau des clients, plus les pairs vont bénéficier d'un taux de téléchargement élevé. Quelques optimisations permettent de doper le trafic P2P notamment lorsque le serveur fournit un *chunk* (morceau du contenu requis) en plus d'une liste de pairs. Cette optimisation est d'autant plus efficace lorsque le *chunk* en question est rare dans la communauté de pairs (le *swarm*).

En termes de perspectives, nous pouvons évoquer l'idée d'adjoindre à ce modèle des codes à effacement tels que présentés au Chapitre 2. Cette idée est illustrée à la Figure 5.10. Dans cette figure, en plus des *chunk* P2P standards, des *chunk* codés sont ajoutés pour augmenter la disponibilité du contenu. Dans ce contexte, la notion de *chunk* rare disparaît puisque tous sont équivalents en termes de décodage. Adjoindre un code MDS au sein de réseaux P2P a déjà été proposé notamment dans [20]. Mais, l'usage de codes géométriques performants tel que ceux construits par la transformée Mojette permettrait de garantir de manière originale une haute disponibilité sur des services temps réel tel que le P2P *Live*. La diffusion de contenu *live* au sein du réseau P2PWeb a récemment été étudié. Ces travaux ont été soumis au prochain Colloque Francophone pour l'Ingénierie des Protocoles (CFIP) 2015 qui aura lieu à Télécom Paris en juillet prochain.

# Chapitre 6

# Réseaux auto-organisés de type ad hoc (le projet SEREADMO)

## 6.1 Introduction

Les réseaux mobiles ad hoc ou MANET (*Mobile Ad hoc NETworks*) sont des réseaux sans fil et sans infrastructure *i.e* sans point d'accès. Dans ce type de topologie décentralisée, tous les nœuds peuvent jouer le rôle de routeur. La particularité ici est que ces nœuds sont mobiles et possèdent des liens radios qui sont par définition instables. La problématique de ce type de réseau est donc d'assurer les fonctions de routages dans un contexte de changement fréquent de topologie. Ils sont très utilisés à l'origine par les militaires (infanterie, drones,...) mais ont de plus en plus d'applications civiles (catastrophe naturelle, jeux en réseaux, événement spontané,...).

Par l'absence d'infrastructure, les réseaux ad hoc appartiennent à la famille des réseaux auto-organisés au même titre que les réseaux capteurs et les réseaux P2P du chapitre précédent. La différence avec les réseaux de capteurs tient au fait que les équipements visés sont en général plus autonomes en matière d'énergie pour les réseaux MANET. Pour cette raison, les réseaux de capteurs s'intéressent généralement à des problématiques de couche 2 (liaison). Les réseaux P2P sont des réseaux large échelle avec potentiellement des millions de nœuds en présence. Les réseaux P2P sont des réseaux de type *overlay* avec des problématiques au niveau application (couche 7 du modèle OSI).

Le routage pour réseaux ad hoc fait l'objet d'une littérature très abondante. Parmi la myriade de protocoles proposés depuis une vingtaine d'années, on distingue deux approches : l'approche réactive (ou à la demande) et l'approche pro-active. L'approche réactive consiste à calculer une route à la demande dès lors qu'un nœud mobile a du trafic à transmettre. DSR [38], AODV [65] sont des protocoles réactifs. *A contrario*, l'approche pro-active maintient les routes entre les nœuds mobiles de manière périodique même en l'absence de trafic. OLSR [17] est un protocole pro-actif à états de lien. Il se base sur l'algorithme de Dijsktra pour le calcul du plus court chemin. C'est aujourd'hui le seul protocole maintenu au sein du groupe travail MANET de l'IETF (*Internet Engineering Task Force*).

Dans ce chapitre, nous proposons une version multi-chemins du protocole OLSR intitulé Multi-Path OLSR (ou MP-OLSR). La répartition spatiale des chemins multiples permettent un couplage intéressant avec les codes à effacement Mojette en allouant une projection par route.

Des versions MP de OLSR ont déjà été proposées dans la littérature. Kun *et al.*[41] en 2005 [41] proposent par exemple des routes strictement nœuds disjoints. Zhou *et al.*[104], la même année, propose un routage source pur en retirant les nœuds retenus pour la prochaine itération de l'algorithme de Dijkstra. Cette approche présente l'inconvénient de provoquer rapidement une pénurie de nœuds lorsque le réseau est peu dense. L'originalité du protocole MP-OLSR réside dans une modification de l'algorithme de Dijkstra de manière à supporter les différentes types de routes à savoir nœuds disjoints, liens disjoints et voire des routes non disjointe en fonction de la topologie du réseau.

Ce travail est le résultat du projet collaboratif SEREADMO (Securité des Réseaux Ad Hoc par transformation MOjette) qui était un projet RNRT porté par Thales Communication (site de Cholet) qui s'est déroulé entre 2006 et 2009. Depuis la fin du projet, le protocole MP-OLSR a donné lieu à de nombreuses collaborations dont la plus active concerne l'équipe de Thomas Clausen au Laboratoire d'informatique de l'École Polytechnique à Palaiseau (LIX). MP-OLSR est actuellement Internet Draft [101] à l'IETF en version *Working Group Document* pour le groupe de travail MANET.

## 6.2 MP-OLSR

MP-OLSR est un routage à chemins multiples basé sur OLSR [17, 16]. Contrairement à OLSR, il s'appuie sur un routage réactif dans le sens où les routes multiples sont calculées à la demande. L'algorithme Dijkstra a été modifié ici pour pouvoir obtenir des routes strictement disjointes ou partiellement disjointes en fonction de la topologie en présence. Cette modification algorithmique est la contribution essentielle de la thèse d'Eddy Cizeron [15]. Des fonctions auxiliaires comme le recalcul de routes par des nœuds médians ou encore la détection de boucles de routage ont été ajoutées pour améliorer les performances en termes de qualité de service. Cet ajout de fonctions ainsi que la spécification et le test du protocole MP-OLSR ont fait l'objet du travail de thèse de Jiazi Yi [99]. Les spécifications du protocole sont détaillées dans la section suivante.

### 6.2.1 Spécifications

Cette section contient l'énoncé de l'algorithme K-Dijkstra portant sur le calcul des chemins multiples, l'élaboration des fonctions auxiliaires comme le recalcul de routes par les nœuds intermédiaires ou l'adjonction d'un code à effacement Mojette, ainsi que des précisions en matière protocolaire.

#### K-Dijkstra

L'algorithme proposé dans [15] reprend l'idée de recherche successive du plus court chemin calculé dans l'algorithme Dijkstra de 1959 [24]. Néanmoins, on va ici non pas interdire les portions déjà utilisées (comme dans [104]) mais simplement les défavoriser par deux fonctions de coût distinctes $f_p$ et $f_e$. Lorsqu'une nouvelle route est définie, ces deux fonctions pénalisent respectivement les liens qui la composent ainsi que les liens qui pointent vers elle de manière à rendre l'ensemble moins intéressant pour la recherche suivante. La valeur de cette pénalité va avoir bien entendu un impact sur la nature des routes obtenues qui pourront être de fait **strictement disjointes**, ou de **liens disjoints** ou encore **partiellement disjointes** en fonction de la topologie du réseau.

Soit $f_p : \mathcal{R}^{*+} \to \mathcal{R}^{*+}$ et $f_e : \mathcal{R}^{*+} \to \mathcal{R}^{*+}$ deux fonctions de coût telles que $id < fp$ et $id \leq fe < fp$ (avec $id$ la fonction identité). L'algorithme 3 prend en paramètre le graphe initial représentant la topologie du réseau $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$, le couple source destination $(s, d) \in \mathcal{E}^2$ et l'entier $K$ correspondant au nombre de routes recherchées. Il retourne un $K$-uplet $(P_1, P_2, ..., P_K)$ de routes entre S et D.

**Algorithm 3:** MultiPathDijkstra(s,d,$\mathcal{G}$,K)

```
1  begin
2      c_1 ⟵ c;
3      G_1 ⟵ G;
4      for i ← 1 to N do
5          SourceTree_i ⟵ Dijkstra(G_i, s);
6          P_i ⟵ GetPath(SourceTree_i, d);
7          forall the arcs e in E do
8              if e OR Reverse(e) is in P_i then
9                  c_{i+1}(e) ⟵ f_p(c_i(e))
10             else if the vertex Head(e) is in P_i then
11                 c_{i+1}(e) ⟵ f_e(c_i(e))
12             else
13                 c_{i+1}(e) ⟵ c_i(e)
14         G_{i+1} ← (V, E, c_{i+1})
15     return (P_1, P_2, ..., P_K)
```

La fonction $f_p$ est utilisée pour incrémenter le coût des arcs qui appartiennent à la route calculée. La fonction $f_e$ est utilisée pour incrémenter le coût des arcs qui pointent vers les nœuds retenus. Le rôle de ces fonctions est de moduler le compromis entre les routes disjointes par les nœuds et les liens. Selon les valeurs des fonctions $f_p$ et $f_e$, le comportement sera le suivant :

- si $id < f_e = f_p$, les routes seront plutôt disjointes par les nœuds ;
- si $id = f_e < f_p$, les routes seront plutôt disjointes par les liens ;
- si $id < f_e < f_p$, les routes seront de préférence disjointes par les nœuds et dans le cas contraire disjointes par les liens.

On verra par la suite que ce dernier cas (où $id < f_e < f_p$) apporte le plus de souplesse dans la détection de routes multiples en exploitant au mieux la topologie du réseau. Sauf cas particulier où le caractère disjoint des routes est strictement nécessaire, on aura généralement $f_e = 2$ et $f_p = 3$ visant à pénaliser plus fortement les liens qui composent une route déjà retenue (fonction $f_p$).

### Les protocoles OLSR et MP-OLSR

Le protocole OLSR (*Optimized Link State Routing*) [17] est un protocole de routage proactif à états de lien. Les nœuds reçoivent des messages de routage de manière périodique (via les paquets HELLO et de *Topology Control* (TC)). Cette réception entraîne une mise à jour de la table de routage et un calcul régulier des routes par l'algorithme d'origine de Dijkstra. La Figure 6.1 illustre le format d'un paquet de donnée pour le protocole OLSR. Le transport s'effectue *via* le protocole UDP. L'en-tête IP précise les adresses source et destination. Quand

un paquet atteint un nœud intermédiaire, celui-ci consulte sa table de routage et retransmet le paquet au nœud suivant. Le routage s'effectue donc saut-par-saut.

FIGURE 6.1 – Format d'un paquet de données OLSR (extrait de [99]).

Le protocole MP-OLSR effectue un routage par la source. Il est donc nécessaire d'inclure la route complète à emprunter de la source à la destination dans les paquets émis. La Figure 6.2 montre le format d'un paquet où la liste de nœuds est insérée dans le champs optionnel de l'entête IP spécifique au routage source. Si la mise à jour des informations topologiques se fait périodiquement (de la même manière qu'OLSR à l'aide des paquets HELLO et TC), le calcul des routes multiples se fait à la demande. Un ordonnanceur de type *Round Robin* est classiquement utilisé pour répartir les paquets au sein des différentes routes.

L'usage des informations topologiques OLSR ainsi que le routage source IP (v4 en particulier) font que les deux protocoles OLSR et MP-OLSR sont compatibles. Un réseau peut donc être composé de ces 2 types de nœuds sans qu'il y ait perturbation dans la fonction de routage.

FIGURE 6.2 – Format d'un paquet MP-OLSR (extrait de [99]).

### Fonctions auxiliaires de MP-OLSR

Le routage par la source, dans le cas où la topologie est susceptible de changer fréquemment, nécessite la mise en place de fonctions supplémentaires pour garantir la qualité de service du réseau mobile ad hoc.

**Recalcul de routes par les nœuds médians.** Le principe est très simple. Il suppose la participation des nœuds médians à l'effort de routage. Avant qu'un nœud médian procède à la retransmission d'un paquet MP-OLSR, ce même nœud vérifie la présence du prochain saut dans son voisinage (via sa table des voisins). S'il est toujours présent, le paquet est retransmis immédiatement. Sinon, le nœud médian recalcule une route simple et retransmet ensuite le paquet. Cette précaution permet de maintenir des routes multiples même dans le cas où les nœuds sont très mobiles.

**Détection de boucles de routage.** En théorie, les chemins générés par l'algorithme de Dijkstra sont dépourvus de boucles de routage. Cependant, l'ajout de la fonction de recalcul de routes (ainsi que la fonction LLN (*Link Layer Notification*) de OLSR) peut engendrer de manière temporaire de telles boucles. Il convient alors de vérifier à la volée si la nouvelle route n'emprunte pas des nœuds précédents. Cette vérification est facilitée par l'usage du routage par la source (liste des nœuds traversés contenue dans le paquet).

**Ajout du code à effacement Mojette.** Cette fonction n'est pas détaillée dans les multiples drafts déposés à l'IETF [92, 93, 101]. Elle apporte néanmoins une réelle amélioration en matière de Qualité de Service (QoS) et de Qualité d'Usage (QoE) comme le témoigne la section 6.3 de ce chapitre portant sur le transport de flux vidéo sur un réseau MP-OLSR. Le principe est d'adjoindre à l'approche chemins multiples un code à effacement (ici Mojette) de manière à autoriser la rupture d'une (ou plusieurs) route(s) sans perturber la reconstruction du flux vidéo encodé. Ce schéma est détaillé dans la section 6.3 et est adaptable à tout type de flux de paquets. La présentation du code Mojette est à l'ordre du jour de la prochaine édition de l'IRTF (la branche recherche de l'IETF) au sein du groupe de travail *Network Coding*.

### 6.2.2 Mises en œuvre

Plusieurs mises en œuvre du protocole MP-OLSR ont été réalisées à des fins de simulations ou à des fins d'expérimentation sur un réseau MANET réel.

#### Mise en œuvre pour le simulateur Qualnet

Le simulateur Qualnet de *Scalable Networks Technologies*[1] est un outil puissant pour développer et expérimenter des nouveaux protocoles. Le temps de développement peut être fortement réduit en comparaison d'autres simulateurs plus populaires comme NS2. Dans notre cas, ce temps a été réduit d'un facteur 3 (6 mois pour NS2 contre 2 mois avec Qualnet). La forte modularité du simulateur en adéquation avec la modélisation OSI (comme l'illustre la Figure 6.3) ainsi que la clarté du code sont les principales raisons de cette efficacité. Qualnet est la version payante du simulateur Glomosim. Le noyau de ces 2 simulateurs est plus adapté aux communications mobiles que son concurrent NS2. Qualnet possède en outre une interface graphique utilisateur efficace pour visualiser les scénarios et les résultats statistiques. Les récentes amélioration de NS3 tend à réduire ces différences.

À l'image d'OLSR, MP-OLSR est un protocole de routage de la couche Application. Il utilise UDP comme protocole de transport pour ces paquets HELLO et TC qui mettent à jour les tables de routage IP. La retransmission par les nœuds médians est une fonction également de la couche IP. Dans sa version Qualnet, MP-OLSR se base sur la version 2 d'OLSR de l'Université de Niigata, Japon [98] développée pour le simulateur. Une version NS2 de MP-OLSR est également disponible (voir `http://www.jiaziyi.com/index.php/research-projects`).

#### Implémentation NetFilter

Cette implémentation a été principalement développé par Pascal Lesage de la société Keosys dans le cadre du projet SEREADMO avec des contributions significatives de Jiazi Yi et de Sylvain David. Une distribution Linux de type *Backtrack*[2] est utilisée en perspective d'une expérimentation réelle. Après étude des différentes implémentations OLSR, il a été choisi d'utiliser la mise en œuvre open-source OLSRd (version v0.6.0) comme base de développement. L'extension multi-chemins utilise un module de type *NetFilter* pour détecter les changements topologiques au sein du démon *olsrd*, pour calculer les chemins multiples à l'aide de l'algorithme K-Dijkstra (algorithme 3) ainsi que de filtrer les paquets émanant de la source, de nœuds médians ou de la destination. Plus de détails sont disponibles en annexe B.2 de [99]. Un projet élève ingénieur

1. http ://web.scalable-networks.com/content/qualnet
2. http ://www.backtrack-linux.org/

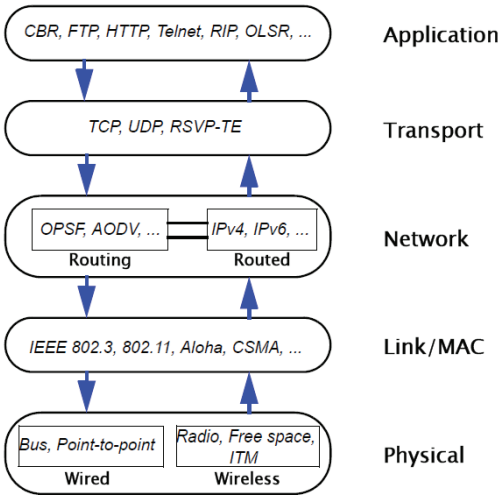



FIGURE 6.3 – Description modulaire du simulateur Qualnet avec des exemples de protocoles mis en œuvre.

Polytech (en collaboration avec l'Université de Paraná, Brésil) à récemment fait le portage vers des noyaux linux plus récent avec un déploiement vers des nœuds de type RaspBerry Pi.

### 6.2.3 Résultats de simulations

Dans ce qui suit, sont présentés les performances du protocole MP-OLSR vis-à-vis de l'approche à chemin unique proposé par OLSR. Une comparaison est également effectuée (en fin de section) avec deux autres protocoles réactifs DSR (*Dynamic Source Routing*) et AODV (*Ad hoc On demand Distance Vector*). On détaille dans un premier temps les paramètres de simulation.

#### Paramètres de simulation

La topologie comporte 81 nœuds mobiles répartis sur une surface de $1500m \times 1500m$, reliés par des liens radios de type IEEE802.11b. Ces nœuds sont initialement positionnés de manière uniforme sur une grille $9 \times 9$. La mobilité (de type urbaine) ne dépasse pas les 10 m/s (soit 36 km/h). Le trafic généré est de type CBR (*Constant Bit Rate*) à 40 Kb/s. Pour éviter des effets de bord indésirables, les sources de trafic sont démarrées après 15 s de simulation et stoppées une dizaine de secondes avant la fin. L'encapsulation couche 3 et 4 est de type UDP/IPv4. Chaque simulation a une durée de 100 s pour 100 réalisations distinctes. Le tableau 6.1 résume ces principaux paramètres.

Les paramètres spécifiques de OLSR et MP-OLSR sont listés au tableau 6.2. Ils font état d'un paramétrage standard pour OLSR (période de 5 s et de 2 s respectivement pour l'envoi des paquets TC et Hello). Pour MP-OLSR, le nombre de routes calculées est de 3 au regard du nombre de nœuds au sein du réseau mobile. Les fonctions de coût $f_e$ et $f_p$ sont égales respectivement à $2c$ et $3c$ avec comme métrique de routage le nombre de sauts (soit $c = 1$).

| Principaux Paramètres | Valeurs |
|---|---|
| Simulateur | Qualnet 5.0 |
| Protocole de routage | OLSRv2, MP-OLSR et DSR |
| Aire de simulation | $1500m \times 1500m$ |
| Nombre de nœuds | 81 |
| Mobilité | RWP, vitesses max 0-10 m/s |
| Temps de simulation | 100 secondes |
| Nombre de simulations | 100 |
| Trafic | 4 à 10 sources CBR |
| Débit source | 40 Kbit/s |
| Encapsulation couche 3 et 4 | UDP/IPv4 |
| Encapsulation couche 1 et 2 | IEEE 802.11b |
| Wireless Channel Frequency | 2.4 GHz |
| Modèle Pathloss | Two Ray Ground |
| Modèle Shadowing | Constant |
| Zone de couverture | 270 m |
| Bande passante PHY | 11Mbps |

TABLE 6.1 – Principaux paramètres des simulations Qualnet 5.0

| Paramètres | Valeurs |
|---|---|
| Période messages TC | 5 s |
| Période messages HELLO | 2 s |
| Fréquence Rafraîchissement Timeout | 2 s |
| Neighbor Hold Time | 6 s |
| Topology Hold Time | 15 s |
| Duplicate Hold Time | 30 s |
| Link Layer Notification (LLN) | Oui |
| Nb. de chemins dans MP-OLSR | 3 |
| MP-OLSR $f_e$ | $f_e(c) = 2c$ |
| MP-OLSR $f_p$ | $f_p(c) = 3c$ |

TABLE 6.2 – Paramètres OLSR et MP-OLSR

**Taux de paquets délivrés**

On présente ici le taux de paquets délivrés (PDR) mesuré au travers de deux scénarios représentatifs. L'un met en jeu 4 sources de type CBR (de 40 Kb/s chacune) alors que le second scénario implique 10 sources CBR (pour un débit total de 400 Kb/s).

Les résultats du scénario à 4 sources sont donnés à la figure 6.4. Ils indiquent le PDR pour les protocoles MP-OLSR et OLSR en fonction de la vitesse max des 81 nœuds en mouvement dans le réseau. L'augmentation de la vitesse max a pour conséquence des changements topologiques plus fréquents (par des ruptures fréquentes de liaisons radio). Globalement, l'approche à chemins multiples apporte un meilleur taux comparé à l'approche à chemin unique (environ 5% de paquets en plus sont délivrés). Les bons résultats d'OLSR à vitesse très faible semblent provenir du placement uniforme initial en forme de grille $9 \times 9$. Les deux protocoles assurent jusqu'à 80% de paquets délivrés même en cas de vitesse max à 10 m/s.

Les résultats du scénario à 4 sources sont donnés à la Figure 6.5. L'ajout de sources CBR a

pour conséquence directe d'augmenter le trafic global (de 160 Kb/s à 400 Kb/s). Statistiquement, plus de nœuds mobiles sont concernés par le routage et le transfert de paquets. L'apport de chemins multiples est plus significatif que dans le scénario précédent. On mesure jusqu'à 30% de paquets délivrés en plus avec le protocole MP-OLSR qu'avec le protocole OLSR.

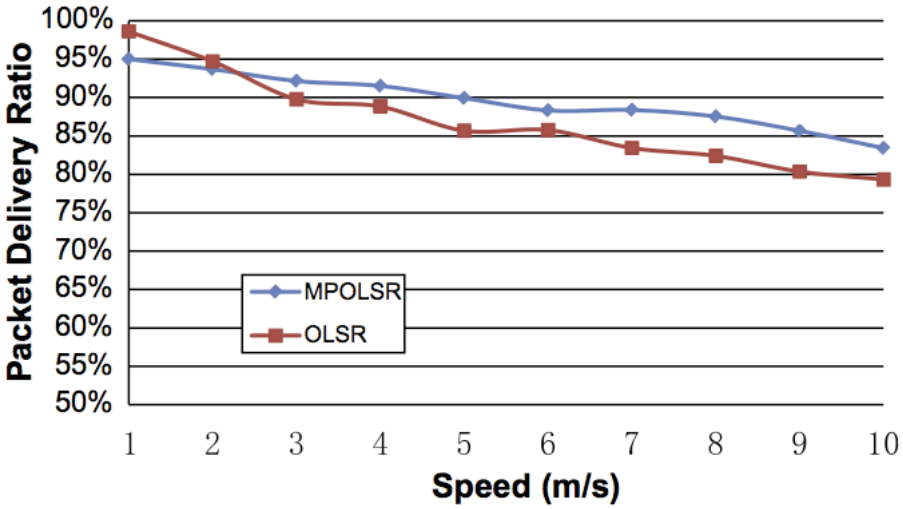



FIGURE 6.4 – Taux de paquets délivrés en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR, enregistrés pour les protocoles MP-OLSR et OLSR (100 simulations de 100 s).

### Délai de bout-en-bout

La mesure du délai est donnée à la Figure 6.6. Elle porte sur le scénario à 4 sources CBR uniquement. Les écarts sont significatifs entre OLSR et MP-OLSR. On mesure jusqu'à 4 fois plus de délai entres les 2 approches (avec 0,04 s pour MP-OLSR et 0,16 s pour OLSR). Les oscillations sur les mesures d'OLSR témoignent en outre d'une forte variation entre les résultats (même après 100 simulations différentes). Les résultats de MP-OLSR sont en revanche beaucoup plus stables.

### Comparaison avec les protocoles DSR et AODV

Aux résultats de taux de paquets délivrés et du délai de bout-en-bout entre OLSR et MP-OLSR sur le scénario à 4 source, on ajoute l'observation des protocoles DSR et AODV (Figures 6.7 et 6.8). Les résultats sont sans appel dans ce contexte. Les approches par états de liens (OLSR et MP-OLSR) résistent beaucoup mieux à la mobilité des nœuds radios. Le protocole DSR fournit une latence particulièrement importante (jusqu'à 4 s de délai de bout-en-bout). Ce dernier ne dispose pas d'un recalcul de route par les nœuds médians. Un paquet de type *RERR* fait office simplement d'alerte qui doit remonter jusqu'à la source.

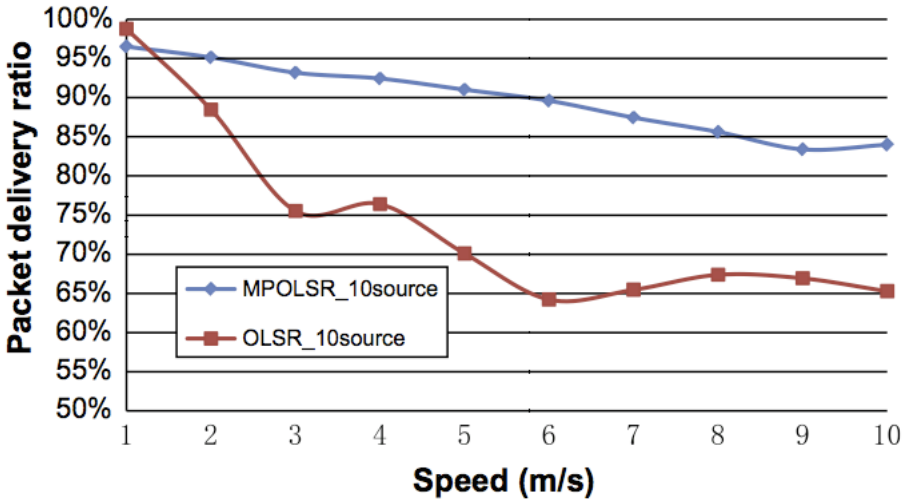



FIGURE 6.5 – Taux de paquets délivrés en fonction de la mobilité pour le scénario à 81 nœuds et 10 sources de type CBR enregistrés pour les protocoles MP-OLSR et OLSR (100 simulations de 100 s).

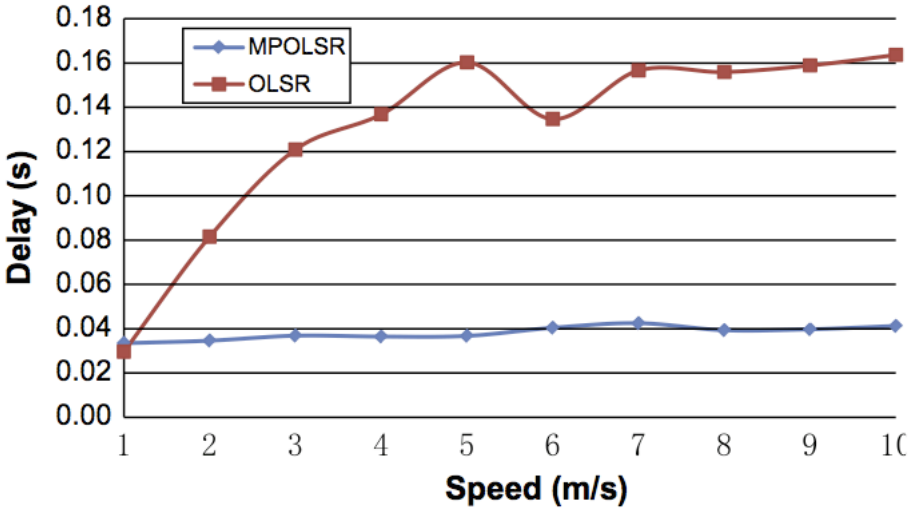



FIGURE 6.6 – Délais de bout-en-bout en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR mesurés (en secondes) pour les protocoles MP-OLSR et OLSR (100 simulations de 100 s).

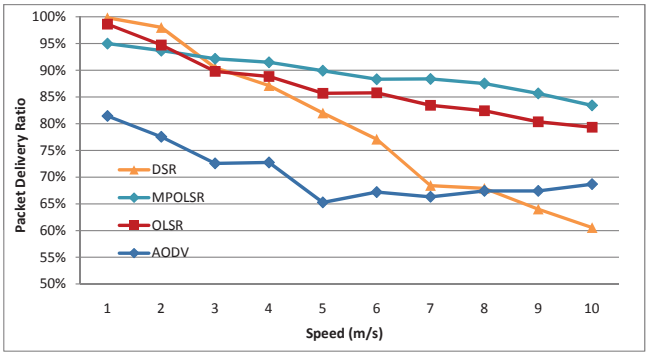



FIGURE 6.7 – Comparaison des taux de paquets délivrés pour les protocoles DSR, AODV, OLSR et MP-OLSR en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR.

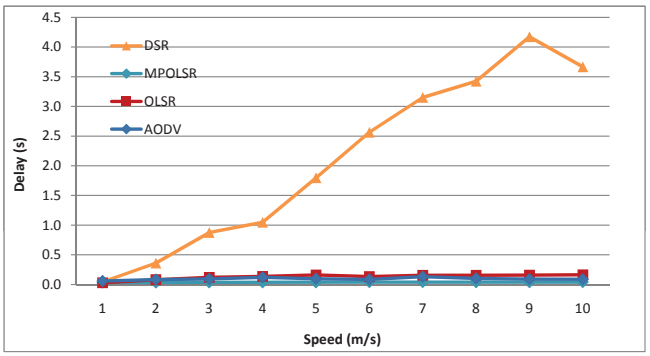



FIGURE 6.8 – Comparaison des délais de bout-en-bout mesurés (en secondes) pour les protocoles DSR, AODV, OLSR et MP-OLSR en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR.

### 6.2.4 Expérimentations SEREADMO

Après plusieurs mois de préparation, l'ensemble des membres du projet SEREADMO s'est réuni la journée du 2 juin 2009 pour mener une campagne de mesures réelles autour du protocole MP-OLSR. Pour cette journée, une dizaine de personnes a été mobilisée venant des 4 partenaires du projet : Université de Poitiers (laboratoire IRCOM-SIC), Thalès Communications (site de Cholet), Keosys SARL et l'Université de Nantes (laboratoire IRCCyN-équipe IVC). Quelques photos de cette journée sont disponibles à la Figure 6.9.

#### Présentation

Les machines retenues pour l'expérimentation étaient de type NoteBook (Asus eeePC 901 muni d'un processeur Intel Atom 1,6 GHz, 1024 MB RAM et 20 GB HDD avec une interface radio de type 802.11b/g). Nous avons disposé de 15 machines pour réaliser plusieurs topologies dont celle décrite à la Figure 6.10.

FIGURE 6.9 – Photos de la journée d'expérimentation SEREADMO (2 juin 2009). Du haut à droite en bas à gauche : Sylvain David (Université de Nantes), Matthieu Fournier (Université de Poitiers), Jiazi Yi (Université de Nantes), moi-même (Université de Nantes) expliquant en salle le déroulant de la journée, Xavier Lecourtier (société Keosys) et Sylvain David.

Chaque nœud disposait d'une adresse LAN et WLAN ainsi que les différents modules et services : *olsrd*, NetFilter, Wireshark pour l'analyse, *ssh* pour le contrôle à distance et NTP pour la synchronisation de l'ensemble des nœuds. Le trafic a été généré par l'intermédiaire du service *uftp* (ftp sur udp) ainsi qu'un service de streaming audio via l'application client/serveur *vlc*. Cette dernière application avait l'intérêt de qualifier subjectivement la qualité des liens radios composant les routes calculées.

Dans ce contexte, les protocoles OLSR et MP-OLSR ont été respectivement testés.

### Topologies

Différentes topologies, de complexités graduelles, ont été testées durant cette journée d'expérimentation. La topologie de la Figure 6.10 est la plus complète. Elle met en jeu 9 nœuds radios (sans mobilité) disposés à la fois à l'intérieur et à l'extérieur du bâtiment IRESTE de l'école Polytech Nantes. Au regard de la topologie, trois routes étaient à disposition entre la source 10.0.0.100 et la destination 10.0.0.98 avec chacune 3 ou 4 sauts. Des règles *iptables* ont été mises en place pour faciliter l'application de cette topologie.

Pour faciliter l'expérimentation et la qualité des liens radios, chaque nœud était à vue de

son saut suivant (espacés d'une centaine de mètres pour les plus éloignés). Si les liens intérieurs (adresses IP : 10.0.0.108, 10.0.0.90 et 10.0.0.102) étaient relativement stables (car disposés au sein d'une grande avenue intérieure), les nœuds situés au niveau des parkings (adresses IP : 10.0.0.98, 10.0.0.93, 10.0.0.105, 10.0.0.95) possédaient en revanche des liaisons radios beaucoup moins stables comme le témoigne la Figure 6.11. Cette instabilité peut s'expliquer à cet endroit par le passage régulier de véhicules perturbant le lien radio.

### Principaux résultats et conclusions

Pour la topologie considérée, les résultats sont indiqués au Tableau 6.3. Ils concernent une session UFTP visant à transférer un fichier d'exactement 17,8 Mo de la source à la destination (débit de 62 Ko/s requis à l'ouverture de la session).

Pour OLSR, le transfert est perturbé en fin de session. Même si le fichier est inutilisable en l'état, on relève néanmoins les caractéristiques du transfert (31,90% de paquets perdus, un débit de 34 Ko/s, une connexion perdue après 8 minutes et 43 secondes).

Pour MP-OLSR, en revanche, le transfert est complet. Il aboutit au bout de 9 minutes et 40 secondes (pour un débit effectif de 31,42 Ko/s). Le taux de paquet perdu est de 37,05% sur toute la durée de la transmission. En pratique, MP-OLSR use de 1 à 2 routes au regard des fortes instabilités radios. La faible diversité suffit néanmoins à assurer le service de transfert de fichiers.

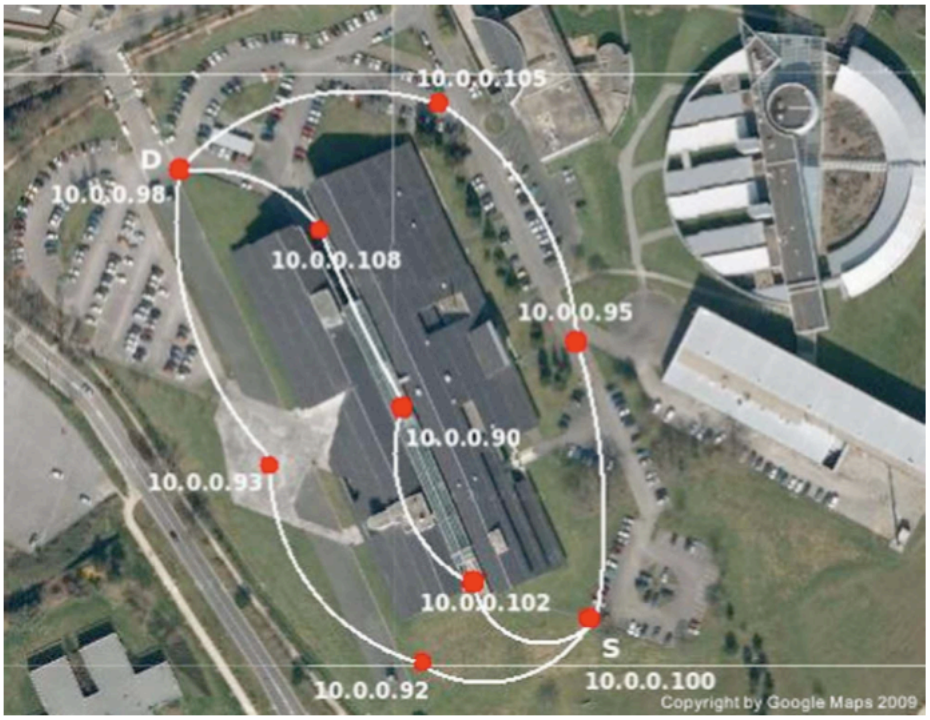



FIGURE 6.10 – Vue aérienne de l'expérimentation du protocole MP-OLSR autour du bâtiment IRESTE sur le campus de la Chantrerie à Polytech Nantes. Exemple de topologie à 9 nœuds situés à l'intérieur et à l'extérieur du bâtiment.

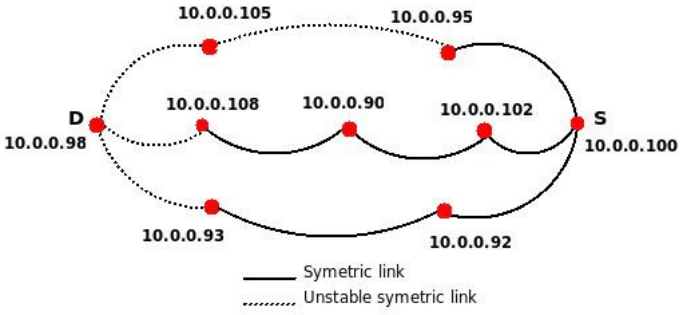



FIGURE 6.11 – Qualités des liens radios (entre stables et instables) lors de l'expérimentation du scénarios de la figure 6.10.

| Protocol | MP-OLSR | OLSR |
|---|---|---|
| Duration of the transmission | 9m40s | 8m43s (Connection lost) |
| Test duration | 14m6s | 9m6s |
| Size of the sent file | 17.8 MB | 17.8 MB |
| Requested rate | 62KB/s | 62KB/s |
| Average rate | 31.42 KB/s | 34.85KB/s |
| Packets sent by 10.0.0.100 | 14528 | 12548 |
| Packets received by 10.0.0.98 | 9145 | 8544 |
| Rate of lost packets | 37.05% | 31.90% (Connection lost) |

TABLE 6.3 – Résultats d'une session UFTP portant sur le transfert d'un fichier de 17.8 Mo sur la topologie de la figure 6.10 pour les protocoles OLSR et MP-OLSR (à 3 chemins).

La thèse de Jiazi Yi [99] contient les résultats d'un plus grand nombre de scénarios de tests. Tous concluent sur l'instabilité du protocole à chemin unique OLSR avec rarement l'aboutissement du transfert total du fichier considéré, contrairement à l'approche par chemins multiples MP-OLSR. La section suivante témoigne de la pertinence de MP-OSLR dans d'autres contextes applicatifs.

## 6.3 Applications pour le transfert de vidéos scalables

Cette section présente l'usage du protocole MP-OLSR dans le contexte du transfert de vidéo de type H.264/SVC. Pour cette application, un code FEC MDS de type FRT [54] est mis en œuvre dans un schéma de protection inégale. Les expérimentations ont été menées à l'aide de l'outil de simulation Qualnet. Ce travail a fait l'objet de deux publications en conférences [80, 102].

### 6.3.1 Présentation de l'expérimentation

Cette présentation détaille le choix des contenus vidéos pour l'expérimentation, les paramètres de protection inégale ainsi que la mise en place du dispositif i.e. l'intégration du flux vidéo codé source et codé canal au sein du simulateur de paquets Qualnet.

#### Description des contenus vidéos

Nous avons sélectionné 4 séquences vidéo parmi de nombreuses séquences du Video Quality Expert Group (VQEG) [3]. Les séquences retenues sont illustrées à la figure 6.12. En matière de contenu, elles couvrent largement l'espace de représentation SI/TI défini dans la recommandation de l'ITU P.910 [56] qui vise à quantifier la complexité spatiale (*Spatial Information)*) et temporelle (*Temporal Information)*. Pour quantifier le contenu spatial, l'index SI a recours à un filtrage passe-haut (de type Sobel) pour dénombrer la part de contours au sein de la séquence (pour la seule composante luminance). L'index TI procède de manière similaire en mesurant la quantité

3. http ://www.vqeg.org

FIGURE 6.12 – Séquences vidéos utilisées pour l'expérimentation. Du coin haut-gauche au coin bas-droit, une image des séquences *Skatefar*, *Burzoom*, *Aspen* et *Shadowboxing*.

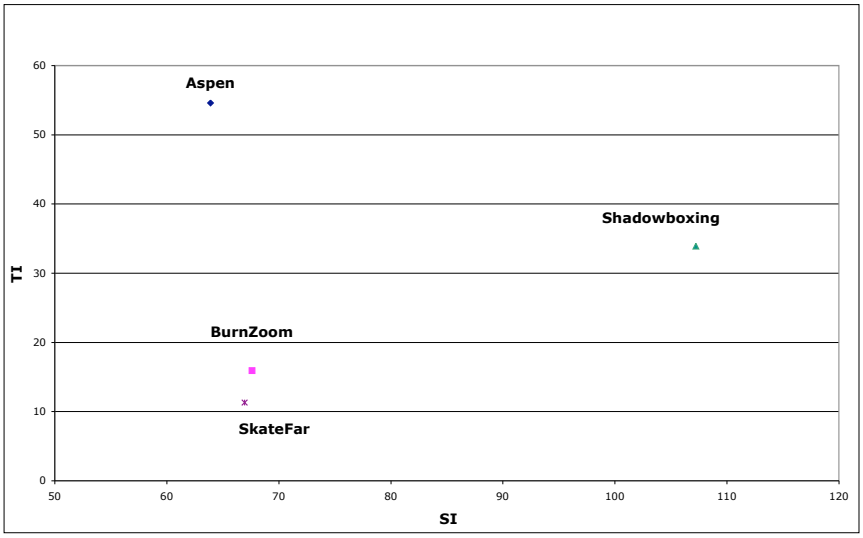



FIGURE 6.13 – Information spatiale et temporelle (SI/TI) des quatre séquences sélectionnées pour l'expérimentation.

de mouvement maximale entre deux images consécutives. Le graphe de la Figure 6.13 représente les index SI et TI pour les quatre séquences sélectionnées qui couvrent assez bien l'espace de représentation des complexités spatiales et temporelles. La séquence *Aspen* est qualifiée par une variabilité temporelle forte en matières de contenus (nombreux changements de plan). À l'inverse, *Shadowboxing* comporte une complexité spatiale forte (avec une variabilité temporelle assez élevée) par une image de fond texturée (avec un mouvement significatif du boxeur). Les deux séquences *Skatefar* et *BurnZoom* sont des plans fixes avec des index SI/TI relativement faibles. La séquence *BurnZoom* fait néanmoins exception puisqu'il s'agit d'un zoom sur une texture dynamique délicate (le feu). Cette dernière séquence illustre les limites de la représentation SI/TI à saisir les complexités spatiales et temporelles d'une séquence vidéo.

Cette pré-analyse du contenu vise à vérifier, aux termes de l'expérimentation, la sensibilité des séquences vidéos en fonction de la complexité de leur contenu.

### Protection inégale du flux vidéo scalable

Le format vidéo utilisé est H.264/SVC. Ce format permet de transporter au sein d'un même flux différentes versions (sous forme de couches vidéos) qui se raffinent en résolution spatiale et temporelle ainsi qu'en qualité. On parle de scalabilité respectivement spatiale, temporelle et qualité pour désigner ce phénomène de raffinement progressif.

Dans cette expérimentation, on utilise la scalabilité spatiale et temporelle. Le flux transmis est composé de 2 couches spatiales (une de base de résolution QCIF et une autre de raffinement

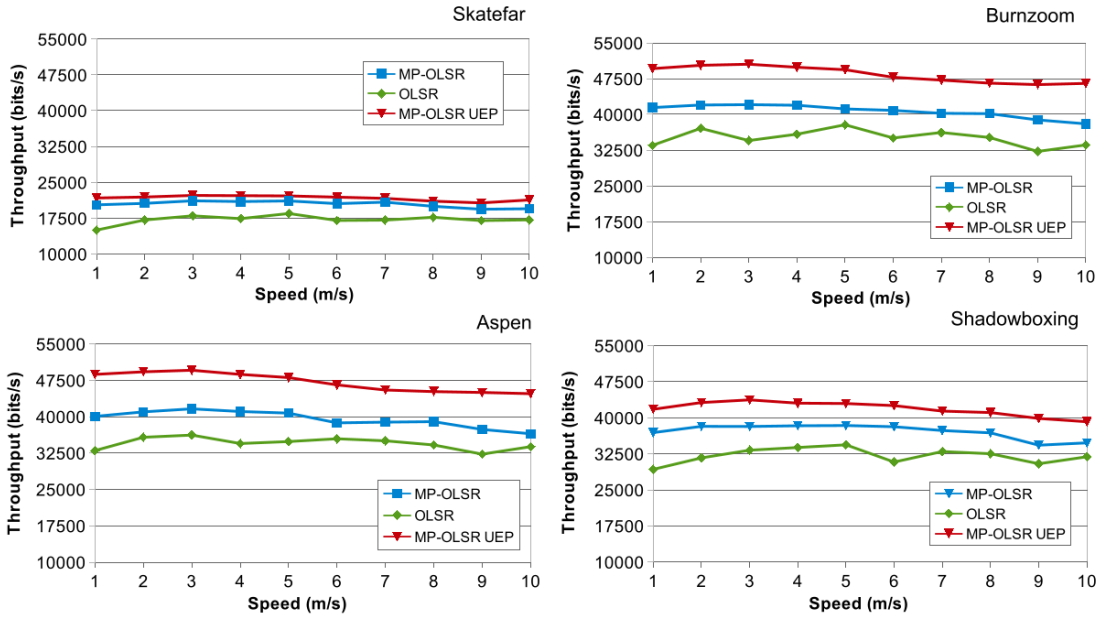



FIGURE 6.14 – Débit applicatif (vidéo et protection) mesuré pour les 4 séquences vidéos sélectionnées et les 3 protocoles de routages OLSR, MP-OLSR et MP-OLSR UEP (Unequal Error Potection).

de résolution CIF) ainsi que de 5 couches de raffinement temporelles. Ce sont ces couches de raffinement temporelles que nous souhaitons protéger à l'aide du code FEC systématique géométrique de type FRT. Ce type de code est retenu ici pour ses qualités strictement MDS et pour les tailles constantes des différents mots de code obtenus. En pratique, ces paquets seront identifiés et tracés dans le simulateur.

Seules les 2 premières couches temporelles sont protégées par le code FRT. La redondance est de type $(2, 3)$ qui autorise la perte d'1 paquet parmi 3. Les 3 autres couches ne sont pas protégées. Elles possèdent une sensibilité moindre à la perte de paquet. La protection inégale ainsi définie introduit un très faible coût vis-à-vis d'un schéma sans protection. Ce coût va être mesuré objectivement dans la section suivante à l'aide du débit du trafic reçu (données vidéo et protection).

### 6.3.2 Résultats QoS/QoE

Les résultats sont fournis en matières de Qualité de Service (ou QoS) et de Qualité d'Usage (ou QoE).

#### Qualité de Service

La qualité de service est ici quantifiée sous la forme de débit du trafic applicatif reçu (i.e. vidéo et protection). Les résultats en matières de taux de paquets délivrés, de délais et de gigue confirment les résultats de la section 6.2.3. Ils sont omis de cette section par souci de concision.

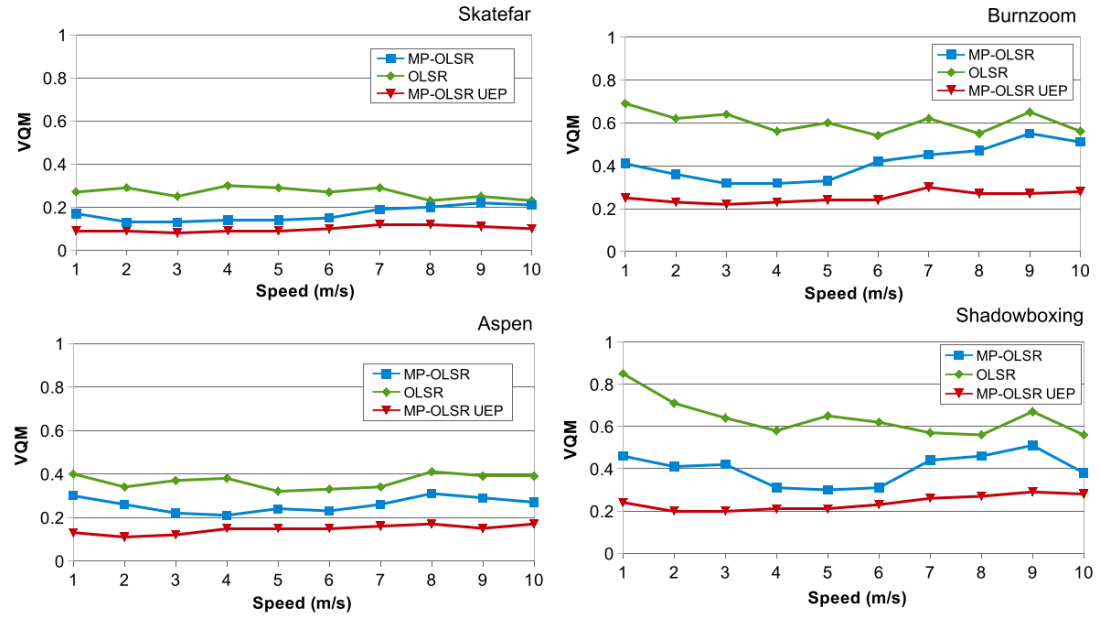



FIGURE 6.15 – Résultats en matière de qualité d'usage VQM (Video Quality Metric) pour les protocoles OLSR, MP-OLSR et MP-OLSR utilisant de la protection inégale (MP-OLSR UEP) sur les séquences de référence : Skatefar, Burnzoom, Aspen, Shadowboxing. Plus le critère VQM est faible, plus la vidéo est de meilleure qualité.

En matière de débit applicatif (*throughput*), les résultats, pour chaque séquence sélectionnée, sont donnés à la Figure 6.14. Les débit enregistrés sont volontairement bas (entre 15 et 50 Kb/s environ) pour satisfaire les contraintes de simulations. Néanmoins, par cette mesure, on voit clairement le coût en débit de la protection inégale appliquée. Exceptée pour la séquence *Skatefar*, la protection UEP engendre un sur-débit de 10 à 15 % pour les 3 séquences *Burnzoom, Aspen* et *Shadowboxing*. La séquence *Skatefar* possède un débit très bas en regard du contenu spatial et temporel évalué par les index SI et TI. La séquence *Burnzoom* fait office d'exception dans la représentation SI/TI. On n'observe pas de distinction en débits purs entre les deux autres séquences (*Aspen* et *Shadowboxing*) supposées plus complexes au niveau du critère SI/TI.

### Qualité d'Usage

Pour mesurer la qualité d'usage (après décodage des flux vidéos), 2 métriques sont utilisées pour évaluer les schémas de transmissions. Le PSNR sur la composante luminance est utilisé dans [102]. Connue pour sa plus forte corrélation avec le jugement humain (évalué notamment par le *Mean Opinion Score* ou MOS), la métrique *Video Quality Metric* ou VQM [97] est préférée dans une seconde campagne de mesures. VQM possède une échelle entre 0 et 1. Une valeur à 0 signifie aucune dégradation perceptible vis-à-vis de la vidéo de référence (métrique *full reference*). Une valeur à 1 porte sur des clips vidéos fortement dégradés. Dans cette étude et par observation des vidéos décodées, on établit un seuil de 0,3 en VQM pour différencier les vidéos acceptables des vidéos à rejeter.

Les résultats du point de vue VQM sont donnés à la Figure 6.15. L'apport de la protection

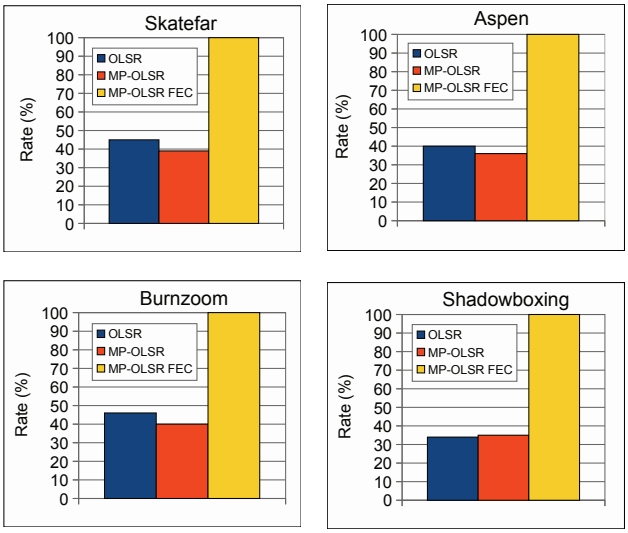



FIGURE 6.16 – Taux de décodage de chacune des séquences vidéos en fonction du protocole de routage et de la protection utilisés.

inégale est manifeste puisqu'aucune des 4 séquences ne dépasse nettement ce seuil empirique de 0,3 aux différentes vitesses observées. MP-OLSR sans protection fait apparaître des contenus non acceptables au sens de ce seuil notamment pour les séquences *Burnzoom* et *Shadowboxing*. L'approche chemin unique OLSR fournit le plus souvent des vidéos au delà de ce seuil (1 porte sur des vidéos fortement dégradées). Il semble que la mesure VQM soit plus sensible aux dégradations temporelles au regard des valeurs mesurées pour les séquences *Burnzoom* et *Shadowboxing*. Les séquences *Skatefar* et *Aspen* sont visiblement moins sensibles à l'usage des différents protocoles (même si le seuil de 0,3 permet encore de distinguer les approches entre elles).

Outre la mesure objective de la qualité de la vidéo perçue, il apparaît intéressant de mesurer également le taux de décodage réalisé pour chacune des approches. En effet, la perte de paquets est fortement préjudiciable à cette opération et à fortiori lorsque ces mêmes paquets ne sont pas traités de manière inégale. La Figure 6.16 témoigne de l'intérêt de l'approche par protection inégale dans cette mesure de qualité d'usage binaire. Avec l'approche Unequal FEC (qui vise à protéger en priorité les couches temporelles de base), l'ensemble des vidéos sont décodables, là où les 2 autres approches fournissent grossièrement une vidéo dans un cas sur deux (voire moins dans le cas de la séquence *Shadowboxing*).

## 6.4 Conclusions

Ce chapitre résume les travaux effectués au sein du projet RNRT SEREADMO entre 2006 et 2009 ainsi que les travaux de thèses d'Eddy Cizeron, Jiazi Yi et Dan Radu que j'ai pu encadrer entre 2007 et septembre 2012 (date de la soutenance de Dan).

Ce dernier chapitre détaille les contributions faites sur cette période autour du protocole MP-OLSR qui est un protocole de routage à chemins multiples pour réseaux ad hoc mobiles basé

sur OLSR [17, 16]. Ces contributions portent sur l'adaptation de l'algorithme Dijkstra au calcul de chemins multiples, la spécification de fonctions auxiliaires de type recalcul de route par des nœuds médians et détection de boucles de routage et aussi l'adjonction d'un codage à effacement en parfaite adéquation avec les routes redondantes obtenues. Ce protocole de routage a fait l'objet d'une évaluation de performances dans un contexte de simulation (à l'aide de l'outil Qualnet) mais également au travers d'une implémentation réelle du protocole sur des équipements mobiles de type PC Notebook Linux. En comparaison de l'approche à chemin unique type OLSR, Les résultats en termes de Qualité de Service sont significativement en faveur du protocole MP-OLSR notamment en termes de taux de paquets délivrés et en termes de délais de bout-en-bout, et ce dans les deux contextes d'expérimentation simulation et implémentation réelle.

Le protocole de routage MP-OLSR a été appliqué au transfert de vidéos scalables de type H.264/SVC. Dans cette application, la fonction code à effacement est particulièrement sollicitée. Elle est mise en œuvre sous la forme de protection inégale en relation avec les flux applicatifs hiérarchique délivrés par le codeur vidéo. Au prix d'une faible redondance (seules 2 couches SVC sur 5 sont protégées), la qualité de la vidéo perçue (mesurée objectivement à l'aide de la métrique perceptuelle VQM) est significativement améliorée par l'action combinée du routage à chemins multiples et la protection par code à effacement.

Ce protocole a été également appliqué dans un contexte de supervision de la pollution sonore en milieu urbain par téléphone cellulaire. Le niveau sonore est en fait mesuré au cours d'une simple communication par téléphonie mobile qui fait office de capteurs. La mesure est rapatriée par l'usage de MP-OLSR. L'application et le déploiement en milieu urbain sont détaillés dans la thèse Dan Radu [79] ainsi que dans la conférence AQTR 2012 [78]. MP-OLSR a également fait l'objet d'une étude de déploiement au sein du réseau sans fil *Funkfeuer Free Net*[4] autour de la métropole de Vienne en Autriche (400 nœuds radios impliqués) et également au sein d'un campus d'une université brésilienne. Récemment, le protocole a été porté sur des nœuds de type RaspBerry Pi.

MP-OLSR est actuellement Internet Draft [101] à l'IETF en version *Working Group Document* pour le groupe de travail MANET. Plusieurs expérimentations sont actuellement en cours dont une en collaboration avec l'Université Fédérale de Paraná, Brésil et du LINA (Laboratoire d'Informatique de Nantes Atlantique).

4. voir http ://www.funkfeuer.at/

# Chapitre 7

# Conclusion

## 7.1 Résumé du manuscrit

Ce document porte sur mes travaux de recherche de 2001 (année de ma soutenance de thèse de doctorat) à aujourd'hui (soit environ 15 années). Cette recherche s'est faite en tant que membre de l'équipe IVC du laboratoire IRCCyN, UMR 6597. Les contributions concernent les domaines de la théorie des codes correcteurs, le stockage distribué, les réseaux pair-à-pairs et les réseaux ad hoc.

Au sujet des codes correcteurs, de nouveaux codes robustes aux effacements de paquets sont définies sous la forme systématique et non systématique. Ces codes reposent sur la transformée Mojette et sont $(1+\varepsilon)MDS$. L'approche géométrique discrète est comparée à l'approche algébrique utilisée notamment pour les codes linéaires de type Reed-Solomon (RS). Les débits enregistrés dépassent les 10 Go/s. Des gains d'un facteur 2 en codage et d'un facteur 3 en décodage sont observées par rapport à l'implémentation RS de référence d'Intel® (librairie ISA-L). Une comparaison avec la réplication donne le codage Mojette non systématique à quelques centaines de cycles CPU de l'instruction `memcopy`.

Ces bonnes performances en débit nous ont permis d'imaginer des architectures de stockage distribué pour des applications avec des entrées/sorties intensives. Un exemple de déploiement est présenté autour du montage vidéo en ligne. Le débit en lecture séquentielle atteint 1,5 Go/s au niveau du système de fichier distribué. Sur la plateforme de montage considérée, l'application permet de manipuler jusqu'à 25 vidéos en parallèle pour un débit réservé de 5 Mo/s par vidéo. L'expérimentation est étendue à l'exécution de machines virtuelles et à des bases de données distribuées. Ce système de fichier distribué, nommé RozoFS, est le seul à notre connaissance à pouvoir manipuler des données chaudes par l'intermédiaire d'un code à effacement (de surcroît non systématique) jusque là cantonné aux applications d'archivage de données froides. Ces travaux se sont effectués dans le cadre du projet ANR Émergence FEC4Cloud.

Dans le protocole P2PWeb, nous proposons de réunir les deux modèles client-serveur et pair-à-pair en un seul modèle hybride où les clients s'auto-organisent autour d'un serveur de contenu et d'un réseau P2P. Les nœuds du réseaux sont alternativement client et fournisseur de contenu à l'intérieur d'un réseau de partage. L'approche évite la duplication coûteuse du serveur sans perte de qualité de service (QoS) ni de qualité d'usage (QoE). Un démonstrateur est réalisé au travers d'un contenu de type image fixe et d'un contenu *live*. Ces travaux ont été réalisés dans le cadre du projet Région (Pays de la Loire) P2PWeb entre mars 2010 et décembre 2012.

La contribution au sein des réseaux ad hoc mobiles (MANET) porte un protocole de routage à chemins multiples. Ce protocole repose sur le protocole OLSR (*Optimized Link State Routing*) [17]. La modification concerne l'algorithme Dijkstra avec l'usage de deux fonctions de coûts ainsi que l'ajout de fonctions auxiliaires comme le recalcul de routes par un nœud médian et la détection de boucles de routage. Un code à effacement de type $MDS$ est ajouté dans le cadre d'une communication vidéo. Les résultats en matières de QoS et de QoE sont nettement en faveur du protocole multi-chemins dans un contexte de simulations et d'une implémentation réelle menée sur le campus Chantrerie de Polytech Nantes en juin 2009. Ce protocole, intitulé MP-OLSR, est actuellement en phase de normalisation auprès de l'Internet Engineering Task Force (IETF). Ce travail a été réalisé au sein du projet RNRT Sereadmo entre 2006 et 2009.

## 7.2 Perspectives de recherche

### Cryptosystème à base de codes correcteurs linéaires

Les codes à effacement de type Reed-Solomon sont à la base d'un algorithme de partage du secret de MacEliece [48] qui généralise la proposition d'origine de Shamir [87]. Ce cryptosystème est considéré comme théoriquement incassable au sens de la théorie de l'information même avec des ressources de calcul phénoménales de type ordinateur quantique. Un algorithme de chiffrement asymétrique est également proposé par McEliece [47] basé sur des codes Goppa. La rapidité des algorithmes de décodage permettent d'utiliser des clés plus longues. Elle garantit donc la confidentialité du système. Il est naturel d'envisager ces schémas en continuité de nos travaux sur la tolérance aux pannes avec une fonction supplémentaire de secret partagé par codes à effacement permettant de satisfaire le respect de la vie privée sans occulter les contraintes en matière de sécurité intérieure. Ce terrain de jeu cryptographique est envisagé dans le projet Privacy4Cloud (soumission ANR 2015 en phase 2).

### Métrologie d'un système de fichier distribué

Un système de fichier distribué (DFS) peut être considéré comme un système complexe à bien des égards notamment lorsqu'il est multi-sites et large échelle. Nous avons dans ce travail observé à plusieurs reprises le comportement des DFS comme Rozo ou encore Ceph. Il reste à caractériser ce comportement voire à le modéliser quand cela est possible. Dans ce contexte, la programmation multi-agents pourrait apporter quelques éléments de réponses. Il est nécessaire néanmoins d'élargir les scénarios pour améliorer notre prédiction en apportant plus de réalité dans nos tests notamment autour de l'écriture concurrentielle, des cas de pannes ou encore de la latence inter-sites. Cette métrologie peut également concerner le traitement distribué sur des architectures distribuées par codes avec une reprise des travaux conduits au Chapitre 3 (section 4.4, page 61) sur l'application de l'algorithme MapReduce sur RozoFS d'une part et sur HDFS d'autre part. Dans ce contexte, des grilles de calcul type Grid5000 permettent d'évaluer les performances en contrôlant la répartition de charge et la consommation électrique totale. Un sujet de thèse est proposé dans cette direction pour la rentrée universitaire 2015.

### Évolutions du protocole P2PWeb

Les codes à effacement ne sont pas aujourd'hui intégrés au protocole P2PWeb. L'apport devrait pourtant être significatif en matières de QoS et de QoE dans la mesure où l'approche par

code fournit des *chunks* équivalents remettant en cause l'intérêt des *chunks* rares distillés par le serveur central à des nouveaux pairs au sein de l'architecture hybride. Le partage de contenus ayant des contraintes temporelles fortes (type *live*) devrait bénéficier d'autant plus de ce mode de distribution. Dans ce même contexte, l'étude du transport "sans couture" d'un modèle client-serveur vers un modèle P2P à l'aide du protocole DASH (*Dynamic Adaptive Streaming over HTTP*), exploré par Nassima Bouzakaria en 2012 [10], mérite d'être consolidée et approfondie. Le recrutement d'un enseignant/chercheur à Polytech Nantes en septembre 2015 pourrait être la ressource nécessaire pour ce sujet.

## MANET et Mobile Cloud Computing

Le protocole MP-OLSR est actuellement en phase d'Internet Draft au sein du groupe de travail MANET de l'IETF [101]. Cette action nécessite un accompagnement en matières d'expérimentation et de caractérisation du protocole. Ce travail s'effectue en étroite collaboration avec Jiazi Yi, chercheur au Laboratoire d'informatique de l'école Polytechnique (LiX). Une expérimentation est actuellement réalisée sur un réseau comportant une cinquantaine de nœuds RaspBerry Pi en collaboration avec l'Université Fédéral de Paranà, Brésil et le LINA (Laboratoire d'Informatique de Nantes Atlantique).

L'interconnexion avec une infrastructure de type Cloud est un sujet émergent notamment au sein de notre région (volet commun de l'actuel RFI Électronique et Numérique en région Pays de la Loire). Il rejoint les problématiques d'Internet des Objets et de mobilité. Un stage master a débuté en avril sur ces thèmes en collaboration avec le Professeur Jinlong Hu, de la South China University of Technology (SCUT) de Canton avec une volonté partagée de répondre à un appel à projet par la suite.

# Chapitre 8

# Sélection d'articles

Quatre articles composent cette sélection :

- page 99, revue *Annals of Telecommunications* de mars 2003 [59] ;
- page 115, revue *Elsevier Journal of Ad Hoc Networks* de janvier 2011 [100] ;
- page 135, conférence *CLOSER 2014* [66] (travaux de thèse en cours de Dimitri Pertin) ;
- page 141, soumission *Usenix HotStorage 2015* (dépôt ArXiv [67]).

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 1



# Multimedia Forward Error Correcting codes for wireless LAN

Benoît PARREIN* — Nicolas NORMAND* — Jeanpierre GUÉDON*


## Abstract

*In this paper the forward error correction (FEC) codes useful for multimedia wireless transmissions are discussed. The class of usable codes are produced by multiple description (packet equivalence during transmission) after an initial hierarchical representation (joint source channel coding) stage. The Mojette transform is used to perform the implementation. A fair comparison with other optimal codes (systematic or not) is provided. JPEG2000 images transmission under a wireless link shows possible strategies.*


**Key words:**

**CODES CORRECTEUR D'ERREURS PAR ANTICIPATION POUR RÉSEAUX LOCAUX MULTIMÉDIAS SANS FIL**


## Resumé

*Dans cet article sont discutés les codes correcteurs par anticipation (FEC) utilisables pour le transport du multimédia sur liens hertziens . La classe de codes utilisée est produite par une description multiple (équivalence des paquets sur le lien) après hiérarchisation des données initiales (codage conjoint source-canal). Pour ce faire, la transformation Mojette est utilisée. Une comparaison avec les autres types de codage systématiques ou non est présentée et l'application au transport d'images JPEG2000 montre les stratégies possibles.*


**Mots clés :**

## Contents


I. *Introduction*
II. *Multimedia Forward Error Correcting codes*
III. *Mojette transform*
IV. *Application to images transmission*
V. *Discussion*
VI. *Conclusions*
*References (17 ref.)*


* Laboratoire IRCCyN, Équipe Image & Vidéo Communications, CNRS UMR 6597, École, polytechnique de l'Université de Nantes , La Chantrerie Rue Christian Pauc, BP 50609, F-44306 Nantes cedex3, {prenom.nom}@irccyn.ec-nantes.fr and {prenom.nom}@polytech.univ-nantes.fr

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 2



## I. INTRODUCTION

For the last ten years, the communications concept of Quality of Service (denoted QoS) has been understood at two distinghinshed levels: network and service by the corresponding communities. The first concerned applications were Internet both for the network (seamless while many different protocols coexist) and for the new services (convergence voice and IP trafic). Wireless communications are now involved in the QoS challenge and face more dramatic constraints for the physical layer. The most important paradigm to be solved for the communication QoS is the mixture between real-time and exact received information. While this trade-off was simply done for voice transmission only (errors at destination were ignored), complex error correcting codes have to be implemented for internet messages that consists of many layers of independant encoding (from the source encoding: transform, quantization, entropic coding; to the multiple network layers). Thus, forward error correcting codes (denoted FEC codes in the following) have been implemented at the application layer to protect against Internet packet losses. A more general mechanism called multiple description (denoted as MD) has been developed for Internet lately while studied in the early eighties for voice transmission (see [5]). For an information theory standpoint, it consists in generating multiple and redundant descriptions of the source and sending these independent descriptions along the network onto differents channels. A reception of a limited number of descriptions allows for the recovery of the initial source. In this paper, this mechanism is adapted onto wireless links while considering each packet of information as emited onto a specific channel. These considerations are detailed and examplified in the following section. The implementation proposed here uses the Mojette transform to perform the multiple description. This transform is very simple as shown in section III, exhibit almost optimal properties from communication theory and a low order of complexity for its true implementation, and can be extended to multiple flows with different priorities in terms of signal recovery. This seems the clue for multimedia applications over wireless networks: the initial flows are prioritized (for instance the low frequencies of a still image or voice is more important than its high frequencies) but the wireless bitstream has to be simply implemented to be efficient. The term multimedia invoked here applies to voice, image, composite documents. The area of video transmission over wireless links is more complicated because of the time order relationships between the video encoding and the produced bitstream. However, some new attempts like the Moving JPEG2000 are concerned with the focus of this paper. The Mojette multiple description applied to JPEG2000 still images in section IV shows this compromise between blind discarded portions of bitstream and image recovery. In this way, it acts as a joint source-channel coding scheme. Related implementation of other optimal FEC codes are discussed in section 5.

## II. MULTIMEDIA FORWARD ERROR CORRECTING CODES

The rapid growth of multimedia real-time applications has led to different strategies for MFEC encoding when packets are used to convey the bit stream. From physical layer to application, this section reviews resilient mechanisms implemented at different stages, each corresponding to a specific source and channel definition.

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 3



### II.1. Physical layer

A first approach, that could be called the historical one, is to protect the stream (the source is here a series of Internet packets) at the physical (PHY) layer. Classical channel coding is employed in this case. While no Forward Error Correcting (FEC) code is implemented in the version b of 802.11, both the version a and its european counterpart HIPERLAN 2 (H/2) use convolutional coding at the PHY layer. For a fixed message length in bits, k = 7, the common FEC rate is 1/2. By puncturing codes, other available rates are 2/3, 3/4 for the 802.11a and 9/16, 3/4 for H/2 [17]. Block codes like Reed-Solomon and also scrambling could be added on the Media Access Control (MAC) layer (802.11e).

A more sophisticated method should support FEC by adapting adequately modulation methods. BPSK, QPSK, 16QAM and 64 QAM are chosen according to the radio link quality or the application prioritisation. For example, the 802.11a standard consists of 8 PHY layer modes (from 6 Mbps to 54 Mbps). In [16], this modes are used to adequately protect both the basement layer and the enhanced layers. More exceptional is when the adaptive modulation takes into account both source and channel properties.

### II.2. At the application level (JPEG2000)

At the application level, the multimedia stream syntax improves resilience in a noisy environment. Audio encoders take profit of the packetization to send in a single packet both a classical piece of voice and a low frequency version of the previous packet [14]. More elaborated, the recent still image standard JPEG-2000 specifies several resilient tools [10] to preserve the source from multiple error types as corrupted packet body, packet header errors, missing bytes.

Basically, JPEG-2000 resilient tools work at two levels. First, an entropy coding box is implemented to protect code-block data. A special symbol sequence called segmentation symbols can be coded with a fixed context at the end of each bitplane. If a wrong sequence is decoded, an error occurs and the following bit planes are considered as corrupted. This method can be improved by a regular predictable algorithm which introduces useful correlations.

The segmentation in packets provides a second stage of protection. The short size of packets reduces losses within a small spatial image area. Furthermore, every packet header can be stamped by a Start Of Packet (denoted as SOP) marker with a sequence index. In the erasure case, the next unaffected packet is searched in the codestream and decoding proceeds from here. For a better source integrity, these packet headers can be moved to the main (or tile) headers and transmitted through a channel with much lower error rate. Despite all these resilient tools, very few missing bytes can lead to impossible decoding. Even in case of possible decoding, resulting pictures can be below service acceptability threshold. A decoding example is given in Figure 1.

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 4

FIG. 1. – JPEG 2000 image reconstruction including resilient mechanisms after a burst error of thirty consecutive bytes starting at byte OXD9C.

*Reconstruction d'une image JPEG-2000 incluant les outils de résistance aux pertes pour une rafale s'étalant sur trente octets consécutifs à partir de l'octet OXD9C.*

## II.3. Priority encoding transmission from MDS codes

Notice that modern multimedia encoding will make use of different sub streams to transmit hierarchical information to the network. Image encoding using JPEG2000 or SPIHT encoders provide different quality streams ; each of them could therefore be encoded with respect to network priorities. Visio or video encoding can also gain from this kind of hierarchy for transmitting images differently according to the specific temporal encoding. When no hierarchy can be transmitted along the network protocol or when for different packets of the same hierarchy level can not all be transmitted in a given time range, a mechanism of priority encoding transmission (called PET) has to be set up. When this is the case, packet are all supposed to have the same probability to cross the network (and thus to be erased in a node or onto a physical link). This situation has been largely examined for different networks protocols during the last twenty years. For instance, the AAL1 layer superimposed on top of the ATM layer was designed for an ATM packet loss (conveying multimedia real-time contents) in

1780-Her/Telecom 58/3-4 20/03/03 15:03 Page 5



the beginning of the eighties. It was implementing a Reed Solomon code RS(128,124). This means that for 124 packets of information four additional packets were generated and then the resulting 128 packets were sent to the network. When a set of any 124 packets arrives at destination the original information is retrieved. In this case, the code is optimal and belongs to the Minimum Distance Separable (MDS) family.

Chapter 11 of the Mac Williams and Sloane [11] describes MDS codes as the most fascinating part of the coding theory. Actually, it was shown that if $n$ packets are computed from $m$ message packets over a Galois field, $GF[2^L]$, any set of $m$ packets among $n$ is sufficient to decode the original message. The property is illustrated in Figure 2. Reed-Solomon represents the most important class belonging to the MDS family.

The construction of MDS codes proposed by [1] can be realized with a generator matrix if and only if each squared sub-matrices are invertible. Cauchy and Vandermonde matrices have both this property but the implementation with a Cauchy matrix [1] are much faster than Vandermonde implementation [15]. However, MDS codes imply a high computational cost.

Low cost, approximated MDS codes called $(1+\varepsilon)$*MDS* are often preferred. These codes need $(1+\varepsilon)m$ packets at the decoding side. With this property, Tornado codes [9] provides a good trade-off between the optimal solution and encoding/decoding times. Redundant packets are computed by linear combinations of message packets with redundant packets. Matrices are very sparse contrary to Cauchy matrices. The average number of variables per equations is small enough to operate by simple substitutions. The reception of a packet number lightly greater than $m$ leads to a tornado of substitutions. Below this number, few substitutions proceed. Tornado codes only use the XOR operation and avoid fields operations and matrix inversions. These codes are used in PET systems. By computing the correct stretch factor according to channel properties, it is possible to satisfy sub streams priorities. However, decoding is probabilistic in the sense that the message can be obtained from $(1+\varepsilon)m$ with a good probability. Moreover, the decoding inefficiency, $\varepsilon$, is 5.4% for video delivery [2] and the granularity of protection *i.e.* allowed redundant rates can be ensured with a high complexity cost. All these reasons leads to propose an other MFEC.

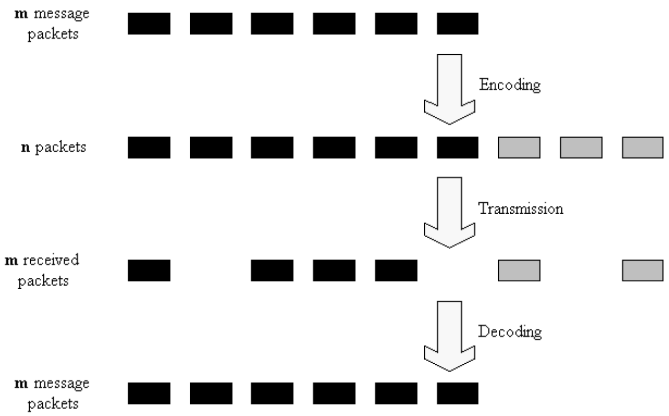



FIG. 2. – How to balance packet loss by FEC codes ? From a set of m message packets, a redundant set is computed into n packets ($n \geq m$). The reception of at least m packets allows to decode the original message in the MDS case.

*Comment compenser la perte de paquets par les codes FEC ? A partir d'une série de m paquets de message, une série redondante est calculée au travers de n paquets ($n \geq m$). La réception d'au moins m paquets permet de décoder le message original dans le cas MDS.*

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 6



## III. MOJETTE TRANSFORM

### III.1. From a 2-D support to multiple 1-D supports

The Mojette transform is based on the Radon transform that allows to describe a 2-D signal by a projected 1-D signals set [12]. To realize the transform, additions are simply used in projection directions which is determined by a couple of integers ($p$, $q$) prime to each other. In the case of binary elements, the XOR operation is used. Each projection is composed with a set of sum elements called bins. The definition of a projection ($p$, $q$) is established from direct Mojette transform of an image $f(k,l)$ and defined as follows :

$$M_{p,q}f(k,1) = \sum_{k} \sum_{1} f(k,1)\ \Delta\ (m + kq - 1p), \tag{1}$$

with the Krönecker function, $\Delta(m) = \{1$ if $m = 0, 0$ if $m \neq 0\}$.

Equation E.1 is the definition of the Mojette-Dirac transform : the summation in the plane corresponds to only take on each line crossed points of the 2-D grid as depicted Figure 2. Other interpolation kernels may be used but are not described in this paper (see [7]).

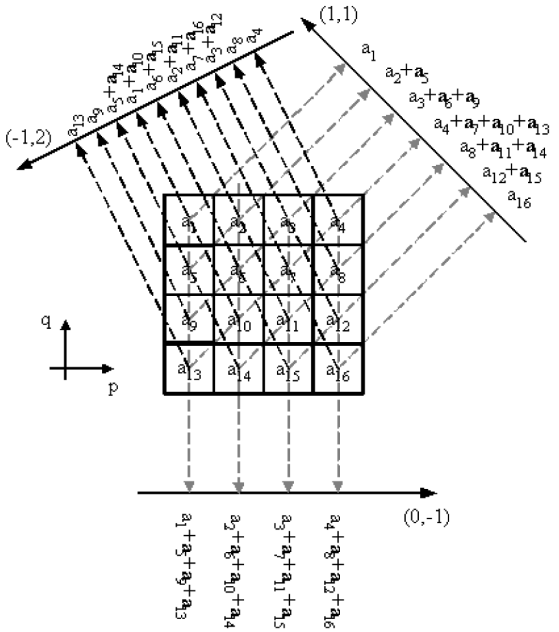



FIG. 3. – 4×4 square Mojette transform with three projections.

*Transformation Mojette sur un carrée 4 ×4 pour trois projections.*

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 7



By extension, the direct Mojette transform of an image $f(k,l)$ of rectangular support P×Q is defined by $MF$ that represent a set of $N$ projections $M_{p,q} f(k,l)$ such as

$$Mf = \{\text{proj}_{pi,qi},\ i = 1,2...N\}. \quad (2)$$

The transform is illustrated Figure 3 for three classical projections. Since each pixel contributes to a bin, complexity is necessarily of $O(PQ = I)$ for any projection. For a set of $N$ projections, complexity is simply $O(IN)$, linear both in the pixels number I and in the projections number. The construction and the reconstruction order is not a priori fixed.

## III.2. Reconstruction conditions

### III.2.1. Rectangular support

Let $A$, a rectangular support composed of $P \times Q$ pixels. Let B be the number of bins of the projection defined from its angle $(p, q)$ :

$$B = (Q - 1)|p| + (P - 1)\ |q| + 1. \quad (3)$$

A lemma found in [8] and corresponding to this situation gives the following result. Let $S$, a set of $N$ projections $\{(p_i, q_i),\ i \in N\}$

$$S \text{ reconstructs } A \Leftrightarrow P \le P_I = \sum_{i=1}^{I} |p_i| \text{ or } Q \le Q_I = \sum_{i=1}^{I} |q_i|. \quad (4)$$

### III.2.2. Generalization by mathematical morphology

For a non rectangular support, equations E.3 and E.4 are not applicable. In order to compute the number of bins, sides of any polytope viewed from angle $(p, q)$ can not be ever written so easily. This problem can be simplified by considering bins number as the difference between indices $m_1$ and $m_{max}$ of first and last bin :

$$B = |m_{max} - m_1| + 1 \quad (5)$$

In order to generalize the reconstruction condition, a projection angle (p,q) is associated with 2 pixels structuring elements (2PSE) as a point couple {O,(p, q)}. It is demonstrated in [12] that a back projection corresponds in fact to a morphological opening (erosion and dilation) of the support with 2PSE defined by the direction.

It means that a image reconstruction is likely a set of morphological openings by 2PSE defined by projection direction. The reconstructibility criteria of Katz find here an another expression with just dilations. A convex image is not reconstructible if and only if the dilation result by 2PSE is not included in the image support (even if one pixel is concerned). This general result is expressed in the following theorem by the equivalence of two statements.

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 8



**Reconstructibility theorem** [12]
**Statement 1 : both propositions are equivalent**
i) $f(k,l)$ defined on the convex $G$ is reconstructible by $\{proj_{pi,qi}, i=1,2...N\}$ ;
ii) $R$ constructed by $N$ dilations set $\{O,(p_i,q_i),i=1,2...N\}$ is not included in $G$.
**Statement 2 : both propositions are equivalent**
i) $G$ is reconstructible by $\{proj_{pi,qi}, i=1,2...N\}$ ;
ii) the erosion of $G$ by $R$ gives the empty set.

To compute the reconstruction, the reverse algorithm needs three kinds of data : the bin value, the sum of crossed pixels and the sum of pixels positions (the last two informations are computed at the decoding side without transmission and were added to speed-up the algorithm). The first step consists in finding explicit bins that can be directly back projected. Following the location of the pixels, all projections are then updated in corresponding bins. Next steps find other explicit bins up to the final reconstruction. Figure 4 exemplified the beginning of the reverse algorithm.

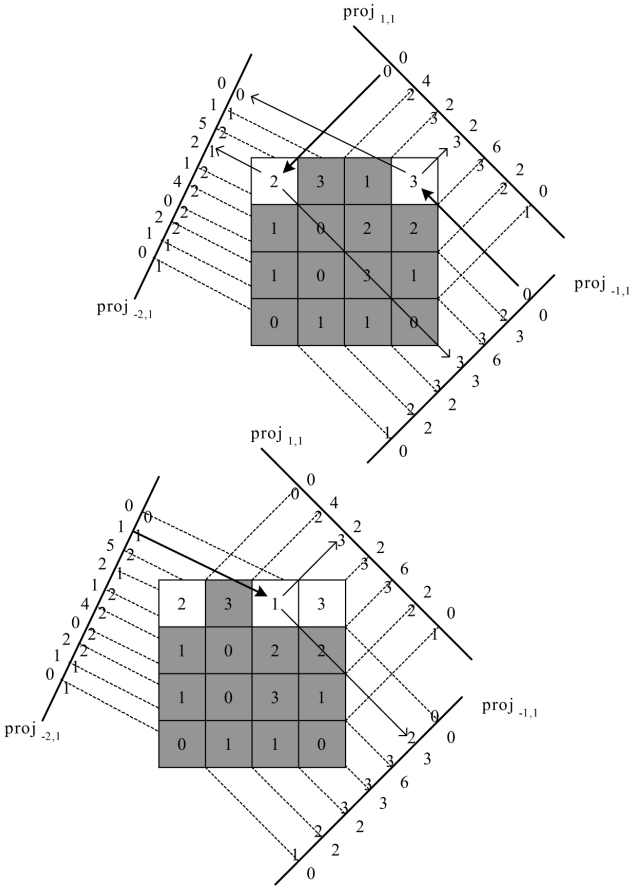



FIG. 4. – Beginning of the reverse Mojette transform. Three pixels are reconstructed. The top value on each line represents the sumation value whereas the underline number gives for each bin the corresponding current number of pixels.

*Début de la transformation Mojette inverse. Trois pixels sont reconstruits. Sur les projections, les valeurs situées vers l'extérieur correspondent aux sommes des pixels alors que les valeurs situées à l'intérieur indiquent le nombre de pixels contribuant à chaque bin.*

1780-Her/Telecom 58/3-4 20/03/03 15:04 Page 9



## IV. APPLICATION TO IMAGES TRANSMISSION

FIG. 5. – Lena test image encoded by JPEG-2000 with a rate of 1 bit per pixel (bpp). A 64×64 Region of Interest (ROI) is specified and retained as the most important information of the source.

*Image test Léna codée par du JPEG-2000 pour un débit de 1 bpp. Une région d'intérêt est spécifiée et retenue comme étant l'information la plus importante de la source.*

This section illustrates the implementation of priority encoding transmission (PET) system using the Mojette transform onto scalable streams featuring JPEG-2000 [6]. In this application to images transmission, the source coding is spread into 32 equivalent descriptions which can be used in any order of reception. Without this MFEC layer, original JPEG 2000 provides non equivalent packets : erasures at the head of the stream lead to important degradations.

A $512 \times 512$ Lena test image is encoded at 1 bit per pixel (1 bpp) including a $64 \times 64$ Region Of Interest (ROI) coding as depicted in Figure 5. This ROI is considered as the most important information. The total initial information volume is 256 Kbits partitioned in four substreams. Its distribution is described in table I. Substreams are allocated in a support shown on Figure 6 that is called a geometrical buffer for its role in this transmission scheme. These kind of supports produces projections with a constant size. Moreover, the reconstruction is really scalable.

The 32 angles are taken such that all projections are equivalent for the reconstruction which only depends on the number of received projections. In [13], it is shown that this condition is checked for the set $S = \{(p, q); p = -15, -14 \ldots 15,16 \text{ and } q = 1\}$.

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 10



Protection levels (*i.e.* the numbers of required projections) were computed for an exponential loss profile with a mean value of 10 %. The computation consists in the maximization of a quality criteria as the Peak Signal to Noise Ratio (PSNR) [13]. Our system protects substream 1 by a possible reconstruction from 22 projections among 32 (erasure packets of .6875 % authorized) whereas substreams 2 and 3 can be reconstructed by 28 and 30 projections respectively among the 32 sent. The last substream requires all projections for decoding. Each projection contains 1047 bins which are byte encoded.

TABLE I.– Description of substreams including sizes, quality at the reconstruction and number of required projections.

*Description des sous-flux incluant la taille, la qualité obtenue après reconstruction et le nombre requis de projections.*

| | **Stream 1** | **Stream 2** | **Stream 3** | **Stream 4** |
|---|---|---|---|---|
| Size (Kbits) | 33 | 43 | 86 | 94 |
| PSNR (dB) | 22.12 | 32.64 | 37.21 | 39.56 |
| #projections | 22 | 28 | 30 | 32 |

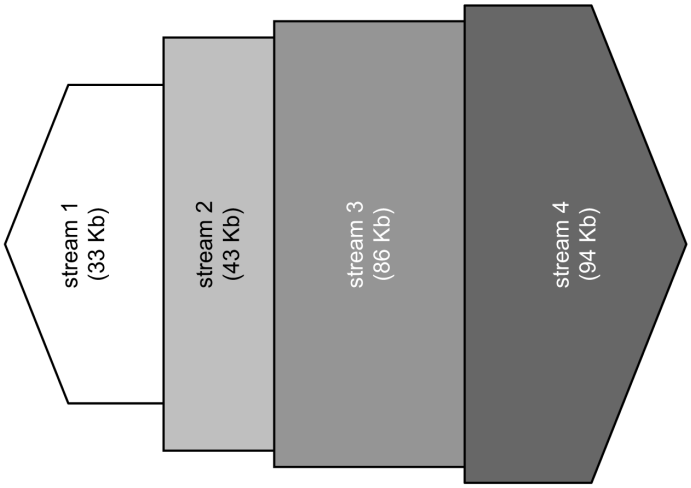



FIG. 6. – Geometrical buffer used for the substreams multiplexing.

*Mémoire tampon géométrique utilisée pour le multiplexage des sous-flux.*

The image is progressively reconstructed with the successive incomings projections. Figure 7 shows the quality criteria *i.e.* PSNR for the image and for the ROI related to the received projections number. Corresponding images are given on Figure 8. An acceptable level (35.08 dB) is quickly obtained for the ROI with just 22 projections. The whole image contains many degradations at this reconstruction step. Six following projections bring more details in

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 11



order to obtain a suitable version (32.64 dB). Refinements are given with the decoding of substreams 3 and 4 respectively with 30 and 32 projections. Both ROI and image qualities are improved with a significant gain for the whole image.

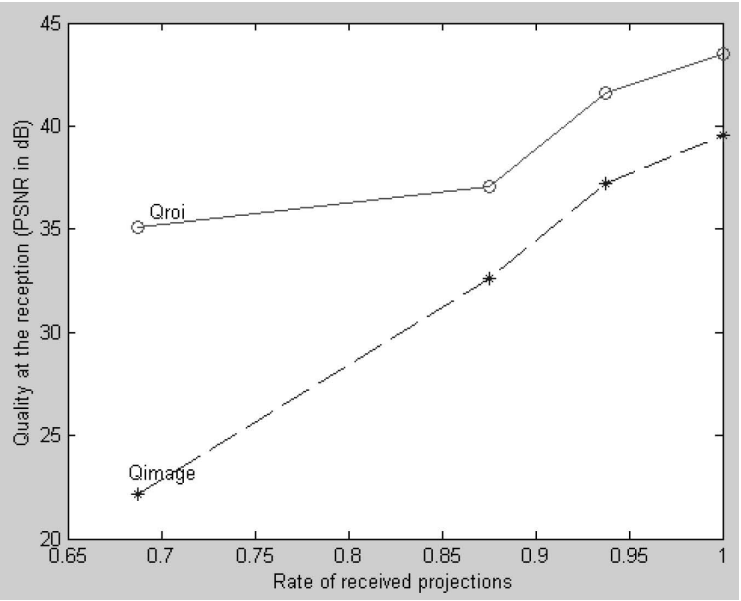



FIG. 7. – Quality levels obtained after the reception of 22, 28, 30 and 32 projections. A good quality for the ROI (in filled line) is quickly obtained as soon as the first 22 projections are available.

*Niveaux de qualité obtenus après réception de 22, 28, 30 et 32 projections. Pour la ROI, un bon niveau de qualité (ligne pleine) est obtenu rapidement dès que les 22 premières projections sont disponibles.*

## V. DISCUSSION

### V.1. Systematic or not systematic codes ?

#### V.1.2. Systematic codes

Implementation of a systematic FEC code for multimedia transmission turns out to be very efficient in terms of real-time characteristics while cumbersome and complicated to manage. The codes belonging to this category are the MDS Reed-Solomon and the (1+ε)MDS Tornado codes. When the source produces the multimedia streaming a first version can be packetized while its copy can be used to compute the redundancy. This point of view allows for a true real-time gain (no lag generated at the encoder) only if the algorithmic complexity is low enough compared to the time range for sending the original information. Of course, this temporal range depends on the possible bitrate on the channel and will be considered as small on

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 12

FIG. 8.– Successive refinements of Lena image. From upper left to lower right, Lena is reconstructed with 22, 28, 30 and 32 projections.

*Raffinements successifs de l'image Léna. De l'image en haut à gauche à celle d' en bas à droite, Léna est respectivement reconstruite par 22, 28, 30 et 32 projections.*

wireless channels. Moreover, the encoder complexity is high and a lot of spatial memory is needed.

At the decoder side, the lag is only present when erasures occur. Thus when erasures probability is high as in the wireless case, the decoding delay can not justify systematic codes implementation. However, it has been very often referred to the use of the systematic codes for multimedia transport as the right way to do because all the resilient errors algorithms are based on this principle (the data is not entirely arrived but the available portion can be used to predict the erased content). Multiple Description paradigm follows the reverse way : a single description must contain as much as possible of the global information which can only be represented by transforming the original information.

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 13



#### V.2.3. Non systematic codes

When a non systematic code is used, each packetized portion of information conveys an information that can be added to the (already) decoded information to enhance the result as in the multiple description scheme. In this respect, RS codes give binary results : either there is not enough at the decoder and nothing is restituted or the whole initial information can be computed. By comparison, the Mojette transform gives progressive decoding according to the number of descriptions present at the decoder site. Another advantage of non systematic codes is the ability to use the inherent redundancy to procure an additional error detector/corrector scheme. However, this supplementary stage can only be implemented when the lower network layer is able to feed the higher layer with noisy data. In such a case, a modified version of the Mojette transform denoted as spline Mojette transform will add a convolution step for each projection with a specific discrete kernel according to the angle of projection. The goal is to correlate the data by spreading it onto the projection when using a known kernel [7].

TABLE II.– Systematic and non-systematic code comparison.

*Comparaison des codes systématiques et non-systèmatiques.*

| | + | – |
|---|---|---|
| Systematic Codes | 1. Null complexity for zero packet loss case<br>2. Easy data pipeline | 1. Reduced protection granularity<br>2. Complexity increasing for high packet losses |
| Non-systematic Codes | 1. Constant complexity whatever the reception scenario<br>2. Fitted granularity to source and channel constraints | 1. Almost MDS<br>2. Constant complexity |

### V.2. Robustness of JPEG-2000

Let try to highlight the resilience lack of a JPEG-2000 stream. The weakness is principally due to the embedded structure of the source. All bytes are not equivalent. An erasure is much more destructive when located at the beginning of the flow.

In the following, an erasure is simulated and included in the JPEG-2000 stream for several coding methods. The length of the error is constant and concerns thirty consecutive bytes. Considering $512 \times 512$ Lena coded at 1 bpp, the bit error rate (BER) is $9.10^{-4}$. This rate correspond to a basic erasure over a short distance wireless link [3]. All missing bytes are set to zero because an entire missing packet does not allow the decoding. The main header is supposed error free as recommended in [4].

First, the describing error is moved regularly from the first JPEG-2000 packet (excluding main header) until it has no more effects. All erasures are tested with error resilient (ER) tools and over a simple source. As depicted in Figure 9, the ER source which includes segmentation symbols can be reconstructed more easily than the basic stream. Without resilient

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 14



mechanism, an erasure at the beginning (near first third) does not allow the decoding. The segmentation symbols improve significantly the qualities reconstruction. Very good versions are obtained when the erasure location is located between 3500th and 10000th byte. However, this reconstruction is random when the burst occurs before. This weak zone concerns at least 10 % of the total volume of information elements.

When the decoding can be done, the reconstructed images are composed of several visual defects which can be unacceptable for certain services. Despite a better resilience than the previous source (better synchronisation of coding words, no bypass of the AC), the image shown in Figure 1 illustrates a burst occurring at the byte OXD9C i.e. 3484. This erasure implies the loss of just one JPEG-2000 packet.

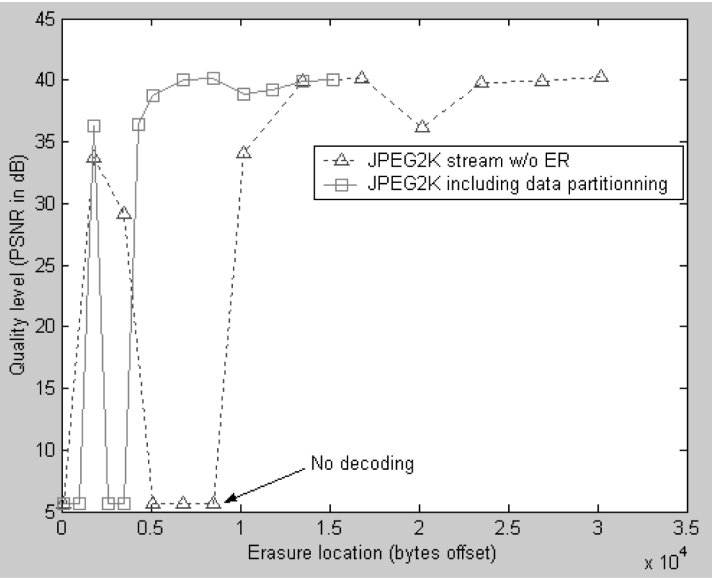



FIG. 9.– Reconstruction qualities (PSNR) in function of the erasure location for an error resilient (filled line) and a basic JPEG 2000 source.

*Qualités de reconstruction (PSNR) en fonction de la position de l'effacement pour une source JPEG-2000 résistante aux erreurs (ligne pleine) et une source JPEG-2000 classique (ligne pointillée).*

## VI. CONCLUSIONS

Network transmissions will always have to deal with packet losses, especially in the case of wireless applications. Fortunately, when integrity is mandatory and delay is not an issue, a full data reception can be ensured at the transport protocol level by retransmitting lost packets. However, multimedia streams do not fit in this scheme. If lost packets of an interactive multimedia stream were retransmitted, they would arrive too late for playback. A better strategy is to protect the emitted data such that the receiver can use the redundancy in order to retrieve partly or totally the lost information.

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 15



This protection can take place at several stages of the transmission, from the physical layer to the application. When resilient tools are used at the application level, we can still observe dramatic quality drops when some specific parts of the data is lost. This relative sensibility has been shown for JPEG 2000 in this paper. Thus, some extra protection is needed but this will have a cost in terms of network bandwidth. The seamless network issue leads also to put this protection at high level in order to be used both for wireless and for IP networks.

To achieve an optimal use of the bandwidth, both network and source characteristics must be taken into account. Priority encoding systems allocate redundancy according to the weight of each information part. Using progressive encoding, they reach the multiple description motto: each new partial description introduces a refinement in the decoding.

Classical PET implementations use either Reed-Solomon codes, which are optimal in terms of coding efficiency because of their MDS properties but have high computational costs, or Tornado codes which are easier to compute but require an extra unusable redundancy $(1+\varepsilon)$MDS and have only probabilistic properties.

As an alternative to both RS and Tornado codes the Mojette transform was proposed. It is a $(1+\varepsilon)$MDS code with deterministic properties and very low computational needs. Furthemore, the decoding cost is constant (only depending on the amount of decoded data). Thus, the Mojette transform constitutes an excellent code for PET systems, especially when many packets are lost as it is the case for wireless communications.

*Manuscrit reçu le 8 octobre 2002*
*Accepté le 17 février 2003*

## Acknowledgements

The authors would like to thank Dr David Guyard from Bouygues Telecom who gave a useful contribution for this work. This study was supported by the state-region Pays de la Loire grant named “multimedia mobile communications” (C2M).

## REFERENCES

[1] BLÖMER (J.), KALFANE (M.), KARP (R.), KARPINSKI (M.), LUBY (M.), ZUCKERMAN (D.), An XOR-Based Erasure-Resilient Coding Scheme, ICSI Tech. Rep. TR-95-048, August 1995.

[2] BYERS (J.W.), LUBY (M.), MITZENMACHER (M.), REGE (A.), A Digital Fountain Approach to Reliable Distribution of Bulk Data, ICSI Tech.Report, TR-98-013, 1998.

[3] COSTAMAGNA (E.), FAVALLI (L.), GAMBA (P.), “Multipath Channel Modeling With Chaotic Attractors”, Proc. of the IEEE, **90**, n° 5, 2002, p. 842-859.

[4] EDWARDS (E.), FUTEMA (S.), ITAKURA (E.), TOMITA (N.), LEUNG (A.), FUKUHARA (T.), RTP Payload Format for JPEG 2000 video streams, Internet Draft, IETF, june 2002.

[5] GOYAL (V.), “Multiple description coding : compression meets the network”, IEEE Signal Processing Mag., **18**, no. 5, pp. 74-93, Sept. 2001.

[6] GROSBOIS (R.), SANTA-CRUZ (D.), EBRAHIMI (T.), “New Approach to JPEG-2000 compliant Region of Interest Coding”, SPIE 46th Applications of Digital Image Processing, San Diego, 10p., 2001.

1780-Her/Telecom 58/3-4 20/03/03 8:29 Page 16



[7] GUÉDON (J.), NORMAND (N.), "Spline Mojette Transform : Applications in Tomography and Communications", Eusipco 2002, **2**, pp. 407-410, 2002.
[8] KATZ (M.), Questions of uniqueness and resolution in reconstruction from projections, Lectures notes in Biomathematics, **26**, *Springer Verlag*, 1979.
[9] LUBY (M.), MITZENMACHER (M.), SHOKROLLAHI (A.), SPIELMAN (D.), STEMANN (V.), "Practical Loss-Resilient Codes", 29th ACM Symposium on Theory of Computing, 150-159, 1997.
[10] MOCCAGATTA (I.), SOUDAGAR (S.), LIANG ( J.), CHEN (H.), " Error resilient coding in JPEG 2000 and MPEG 4", IEEE JSAC, **18**, n° 6, june 2000, pp. 899-914.
[11] MAC WILLIAMS (F.), SLOANE (N.), The Theory of Error Correcting Codes, *North-Holland*, 1977.
[12] NORMAND (N.), GUÉDON (J.), "La transformée Mojette : une représentation redondante pour l'image", Comptes rendus de l'Académie des Sciences de Paris, Section informatique théorique, pp. 123-126, jan 1998.
[13] PARREIN (B.), "Description Multiple de l'Information par transformation Mojette", PhD thesis (in french), Univ. Nantes (2001), 121-125. available at http://www.irccyn.ec-nantes.fr/~parrein.
[14] PERKINS (C.), KOUVELAS (I.), HODSON (O.), HARDMAN (V.), HANDLEY (M.), BOLOT (J.C.), VEGA-GARCIA (A.), FOSSE-PARISIS (S.), RTP Payload for Redundant Audio Data, RFC 2198, IETF, Sept. 1997.
[15] RIZZO (L.), VICISANO (L.), "A Reliable Multicast Data Distribution Protocol based on Software FEC technics", Proc. of the fourth IEEE HPCS'97 workshop, 10 p.
[16] VAN DER SHAAR (M.), MEEHAN (J.), " Robust Fine-Granularity-Scalability for Wireless Video", PV 2002 workshop, 10 p., April 2002.
[17] WALKE (B.), ESSELING (N.), HABETHA (J.), HETTICH (A.), KADELKA (A.), MANGOLD (S.), PEETZ (J.), VORNEFELD (U.), "IP over Wireless Mobile ATM - Guaranteed Wireless QoS by HIPERLAN/2", Proc. of the IEEE, **89**, n° 1, p. 21-40, 2001.

ARTICLE IN PRESS

Ad Hoc Networks xxx (2010) xxx–xxx

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc

# Multipath optimized link state routing for mobile ad hoc networks

Jiazi Yi *, Asmaa Adnane, Sylvain David, Benoît Parrein

*Université de Nantes, Nantes Atlantique Universités IRCCyN, CNRS UMR 6597, Polytech'Nantes, rue Christian Pauc, BP50609, 44306 Nantes cedex 3, France*

ARTICLE INFO

*Article history:*
Received 4 November 2009
Received in revised form 15 March 2010
Accepted 26 April 2010
Available online xxxx

*Keywords:*
MANET
OLSR
Multipath routing
MP-OLSR
Testbed
Backward compatibility

ABSTRACT

Multipath routing protocols for Mobile Ad hoc NETwork (MANET) address the problem of scalability, security (confidentiality and integrity), lifetime of networks, instability of wireless transmissions, and their adaptation to applications.

Our protocol, called MultiPath OLSR (MP-OLSR), is a multipath routing protocol based on OLSR [1]. The *Multipath Dijkstra Algorithm* is proposed to obtain multiple paths. The algorithm gains great flexibility and extensibility by employing different link metrics and cost functions. In addition, *route recovery* and *loop detection* are implemented in MP-OLSR in order to improve quality of service regarding OLSR. The backward compatibility with OLSR based on IP source routing is also studied. Simulation based on Qualnet simulator is performed in different scenarios. A testbed is also set up to validate the protocol in real world. The results reveal that MP-OLSR is suitable for mobile, large and dense networks with large traffic, and could satisfy critical multimedia applications with high on time constraints.

© 2010 Elsevier B.V. All rights reserved.

## 1. Introduction

Staying connected anywhere to a network is really the main objective of mobile technologies. Mobile Ad hoc NETwork (MANET) may provide a solution. With MANET, all nodes are routers and forward packets without any infrastructure. This kind of network is spontaneous, self-organized and self-maintained. In this context, routing the data is the big challenging task since many issues are covered: scalability, security, lifetime of network, wireless transmissions, increasing needs of applications.

Many routing protocols have been developed for ad hoc networks [2]. They can be classified according to different criteria. The most important is by the type of route discovery. It enables to separate the routing protocols into two categories: proactive and reactive. In reactive protocols, e.g. Dynamic Source Routing (DSR [3]) and Ad hoc On-demand Distance Vector routing (AODV [4]), the routing request is sent on-demand: if a node wants to communicate with another, then it broadcasts a route request and expects a response from the destination. Conversely, proactive protocols update their routing information continuously in order to have a permanent overview of the network topology (e.g. OLSR [1]).

Another criterion for ad hoc routing protocol classification is the number of routes computed between source and destination: multipath and single path routing protocols. Unlike its wired counterpart, the ad hoc network is more prone to both link and node failures due to expired node power or node mobility. As a result, the route used for routing might break down for different reasons. To increase the routing resilience against link or/and node failures, one solution is to route a message via multiple disjoint paths simultaneously. Thus, the destination node is still able to receive the message even if there is only one surviving routing path. This approach attempts to mainly address the problems of the scalability, mobility and link instability of the network. The multipath approach takes advantage from the large and dense networks.

Several multipath routing protocols were proposed for ad hoc networks [5]. The main objectives of multipath routing protocols are to provide reliable communication and to ensure load balancing as well as to improve quality

* Corresponding author. Tel.: +33 240683044.
*E-mail addresses:* Jiazi.Yi@univ-nantes.fr (J. Yi), Asmaa.Adnane@univ-nantes.fr (A. Adnane), Sylvain.David@univ-nantes.fr (S. David), Benoit.Parrein@univ-nantes.fr (B. Parrein).

1570-8705/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.adhoc.2010.04.007

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

of service (QoS) of ad hoc and mobile networks. Other goals of multipath routing protocols are to improve delay, to reduce overhead and to maximize network life time.

Multiple paths can be used as backup route or be employed simultaneously for parallel data transmission (like round robin). The multiple paths obtained can be grouped into three categories:

1. *Disjoint*: This group can be classified into node-disjoint and link-disjoint. In the node-disjoint multipath type, there are no shared nodes between the calculated paths that links source and destination. The link-disjoint multipath type may share some nodes, but all the links are different.
2. *Inter-twisted*: The inter-twisted multipath type may share one or more route links.
3. *Hybrid paths: The* combination of previous two kinds.

Of all the multipath types, the node-disjoint type is the most disjointed, as all the nodes/links of two routes are different i.e. the network resource is exclusive for the respective routes. Nevertheless, the pure disjoint approach is not always the optimal solution, especially for sparse networks and multi-criteria computing. As we will see, our Multipath Dijkstra Algorithm is more flexible when keeping all the solutions in the shortest paths algorithm.

In this paper, we started from the MultiPath Optimized Link State Routing protocol (MP-OLSR) presented in [6] which was thoroughly revisited and upgraded. Contributions are multiple. First, a major modification of Dijkstra algorithm allows for multiple paths both for sparse and dense topology. Two cost functions are used to generate node-disjoint or link-disjoint paths. Second, the OLSR proactive behavior is changed for an on-demand computation. MP-OLSR becomes a source routing protocol. Third, to support the frequent topology changes of the network, auxiliary functions, i.e. *route recovery* and *loop check*, are implemented. The contribution of these two functions is quantified in terms of quality of service parameters and compared with OLSR. Fourth, the backward and forward compatibility study with its single-path version (OLSR) is proposed. The cooperation between the two protocols is expected here to facilitate the application and deployment of the new protocol. Simulations and real testbed demonstrate all the contributions.

The remainder of the paper is organized as follows. In Section 2, related works on multipath routing protocols are summarized. In Section 3, we introduce our protocol MP-OLSR and its auxiliary functionalities. Simulation and performance evaluation are presented in Section 4. Section 5 presents the testbed and provides related test results. Compatibility between OLSR and MP-OLSR is studied in Section 6. Finally, we conclude this paper.

## 2. Related works

In this section, we will first present the current situation of OLSR standardization, which includes both OLSR version 1 and OLSR version 2. Then some typical multiple path routing protocols for MANET are presented. And a related study based on testbed for MANET is introduced at the end.

### 2.1. OLSR version 1 and OLSR version 2

OLSR, the most popular proactive routing protocol for ad hoc networks and OLSR version 1 (OLSRv1), has been standardized as an experimental RFC [1]. It is a link state protocol in which each node will send out *HELLO* and Topology Control (*TC*) messages periodically. It reduces the overhead of flooding link state information by requiring just Multi Point Relay (MPR) to forward the *TC* messages. A routing table is maintained to keep the next hop information to all the possible destination nodes.

OLSR version 2 (OLSRv2) has the same algorithm and ideas as OLSRv1. Being modular by design, OLSRv2 is made up from a number of generalized building blocks, standardized independently and applicable also for other MANET protocols. Currently, RFC 5148 – Jitter Considerations in mobile ad hoc networks [6], RFC 5444 – Generalized MANET Packet/Message Format [7] and RFC 5497 – Representing Multi-Value Time in mobile ad hoc networks (MANETs) [8] are published as RFCs, with the remaining constituent parts (MANET Neighborhood Discovery Protocol [9] and OLSRv2 [10]) being in the final phases of standardization. It has a more modular and extensible architecture, and is simpler and more efficient than OLSRv1. The multipath and its compatibility that we propose can also exist as additional modules in the OLSRv2 framework.

### 2.2. Multipath routing protocol for ad hoc networks

Most of the proposed multipath protocols are based on the single-path version of an existing routing protocol: AODV and AOMDV [11], DSR and SMR [12].

Most of these protocols are based on a reactive routing protocol (AODV [4] or DSR [3]). In fact, reactive multipath routing protocols improve network performances (load balancing, delay and energy efficiency), but they also have some disadvantages:

- *Route request storm: Multipath* reactive routing protocols can generate a large number of route request messages. When the intermediate nodes have to process duplicate request messages, redundant overhead packets can be introduced in the networks [13].
- *Inefficient route discovery:* To find node-disjoint or link-disjoint paths, some multipath routing protocols prevent an intermediate node from sending a reply from its route cache [14]. Thus, a source node has to wait until a destination replies. Hence, the route discovery process of a multipath routing protocol takes longer compared to that of DSR or AODV protocols.

Compared to reactive routing, the proactive routing protocols need to send periodic control messages. Hence, several researchers consider proactive routing protocols as not suitable for ad hoc networks [5]. For a network with low mobility and network load, the reactive routing protocols generate fewer control messages. However, given a network with high mobility and large traffic, the cost of

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

*route discovery* and *route maintenance* will rise significantly. On the other hand, the proactive protocols try to keep a routing table for all possible destinations and therefore provides a transmission delay shorter than reactive routing protocols [15]. Furthermore, because the proactive protocols try to maintain the information of the whole network by periodical control messages, they can discover multiple routes more efficiently without much extra cost.

Few studies were interested in multipath routing based on the OLSR protocol. Kun et al. [16] propose another version of multipath OLSR using IP-source routing. Based on Dijkstra algorithm, the node calculates multiple paths to the destination. The calculated paths are strictly node-disjoint. The path is inserted in the IP header of the packet before sending. Based on these multiple paths, the paper introduces an algorithm of *load_assigned* to transmit data through the paths based on the congestion information of all the intermediate nodes on each path. The congestion information of one path is measured as the maximal size of the queue of the intermediate nodes (the queue size of a node is encapsulated in *HELLO* packets and advertised in *TC* messages). The algorithm of *load_assigned* selects two paths to transmit data according to their congestion information, and balances the load on the selected paths. In [17], the authors also propose a similar algorithm to calculate node-disjoint multiple paths by removing used nodes from the topology information base. However, the mere source routing is problematic especially with topology changes which will be analyzed in the following section. In both studies, strict node-disjoint routes are not always necessary and the suppression of nodes in multiple calls of Dijkstra algorithm could not work for sparse networks. The node-disjoint multiple paths are not suitable for partition or fusion group of nodes that can temporarily imply a single link for connection. Furthermore, the backward compatibility is not considered, which might be very important for the deployment of the new protocol.

In [18], the authors propose a multipath calculation based on the *shortest-widest path algorithm* for the protocol *QOLSR*, where *QOLSR* is an enhancement of the OLSR routing protocol to support multiple-metric routing criteria (bandwidth and delay). The proposed algorithm computes multiple loop-free and node-disjoint paths with a small correlation factor based on the delay and bandwidth metrics. The correlation factor is defined as the number of links connecting two disjointed paths. It is calculated to minimize interference between the multiple paths in order to achieve better QoS guarantees to applications and improve network resource utilization. However, the authors did not prove that the correlation factor can be correctly calculated. Indeed, OLSR nodes cannot have a global view of the network, but only links advertised in *HELLO* and *TC* messages (by default, the advertised link set in *TC* of the node is limited to the MPR selector set [1]). Moreover, this approach assumes a freshness of the measures (bandwidth, delay) which is difficult to obtain and maintain in practice. Some other metrics might be easier to obtain as mentioned in Section 3.1.

In our work, we propose a new Multipath Dijkstra Algorithm, which provides node-disjoint or link-disjoint paths when necessary by adjusting distinct cost functions. Additional functionalities are used to adapt to the topology changes.

### 2.3. Testbed for ad hoc networks

A high number of network protocols are only assessed through simulators due to implementation difficulties. This might induce two problems: firstly, with current simulation technology, it is not easy to simulate the exact real world scenario, especially the behavior in the physical layer model. In [19], the authors use intelligent ray tracing model to simulate a more realistic physical layer with a very high cost (3 days on a 50-node PC cluster to produce 120 GB of output data for just one scenario). The results show that there are differences between the commonly used physical layer model (free space or two-ray ground) and the more realistic physical layer.

Secondly, some of the techniques and network parameters are easy to achieve in a simulator, but not in practice. For example, some of the protocols use extra information, such as delay and bandwidth as link metrics [18] to improve the performance of the network without mentioning how to obtain and maintain this kind of real-time information. This information might be easy to get in the simulator, but it is not very practical for a general usage.

Consequently, we believe it is important not only to test the protocol with the simulator, but also to validate it in a real testbed to ensure that it is practical and feasable with current technology.

In [20], the authors describe their experiments building a multi-hop wireless ad hoc network of eight nodes based on DSR protocol driving around a 700 m by 300 m site. The jitter introduced by the network is measured and a push-to-talk voice service is tested. For OLSR, one of the most sophisticated implementations is *OLSR daemon* (*olsrd* [21]) which is highly portable and scalable. It now runs on community mesh networks of up to 2000 nodes (*Athens wireless network* [22]) and 400 nodes (*FunkFeuer.at net* [23]). The team from Niigata University, Japan, implemented OLSRv2 in a testbed with 50 nodes on their campus [24] and concluded that address compression and Link Layer Notification can improve the performance of OLSRv2.

The implementation of multipath routing is rare in the literature. In [25,26], the authors propose the multipath testbed based on multipath DSR in the following of [20]. And in [27], a multipath testbed is implemented based on multipath AODV. In [28], the authors set up a testbed in OMF framework based on Greedy Dominating Set algorithm [29] for gateway placements in mesh networks. The results obtained from these testbeds show that the multipath routing protocol can provide shorter average route recovery time and higher throughput. For our study, in addition to the simulation results, we have implemented MP-OLSR in a real testbed to validate our protocol, which will be introduced in the rest of the article.

## 3. Multipath OLSR – functionalities

The MP-OLSR can be regarded as a kind of hybrid multipath routing protocol which combines the proactive and

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

reactive features. It sends out *HELLO* and *TC* messages periodically to detect the network topology, just like OLSR. However, MP-OLSR does not always keep a routing table. It only computes the multiple routes when data packets need to be sent out.

The core functionality of MP-OLSR has two parts: *topology sensing* and *route computation*. The *topology sensing* is to make the nodes aware of the topology information of the network. This part benefits from *MPRs* like OLSR. The *route computation* uses the *Multipath Dijkstra Algorithm* [30,15] to calculate the multipath based on the information obtained from the *topology sensing*. The source route (all the hops from the source to the destination) is saved in the header of the data packets.

The *topology sensing* and *route computation* make it possible to find multiple paths from source to destination. In the specification of the algorithm, the paths will be available and loop-free. However, in practice, the situation will be much more complicated due to the change of the topology and the instability of the wireless medium. So *route recovery* and *loop detection* are also proposed as auxiliary functionalities to improve the performance of the protocol. The *route recovery* can effectively reduce the packet loss, and the *loop detection* can be used to avoid potential loops in the network as depicted in Sections 3.3 and 3.4.

In this section, we discuss both the core functionalities and auxiliary functionalities.

### 3.1. Topology sensing

To get the topology information of the network, the nodes use *topology sensing* which includes link sensing, neighbor detection and topology discovery, just like OLSR [1].

Link sensing populates the local link information base (*link set*). It is exclusively concerned with OLSR interface addresses and the ability to exchange packets between such OLSR interfaces. Neighbor detection populates the neighborhood information base (*neighbor set* and *2-hop neighbor set*) and concerns itself with nodes and node main addresses. Both link sensing and neighbor detection are based on the periodic exchange of *HELLO* messages. Topology Discovery generates the information base which concerns the nodes that are more than two hops away (*topology set*). It is based on the flooding of the *TC* messages (optimized by selecting the MPR set).

Through topology sensing, each node in the network can get sufficient information of the topology to enable routing. The link state protocol tries to keep the link information of the whole network as mentioned above. By default, the path quality is measured by the number of hops according to [1]. It can also be measured by other metrics such as Bit Error Rate (BER) [31] or the queue length. In our previous work [31], the BER metric showed better performance in certain scenarios, but the benefit is not obvious in various situations (such as urban areas). ETX metric [32] is also proposed as a MANET Internet Draft and is bound to become a standard. It has been extensively used in mesh networks around the world. In [33], the authors present a comparison between OLSR–RFC default hysteresis/hop count metric and OLSR–ETX metric in a mesh network testbed. However, their results reveal the ETX metric to be fundamentally flawed when estimating optimal routes in large dense mesh network and worse than the OLSR–RFC standard. From the lower-layer point of view, the metrics still need to be further studied. And we also believe that the way the metrics are used for path selection can be service-dependent to improve the QoS.

What kind of link metric to use and how it can be used properly in MANET are still open topics [34]. In this paper, we follow the RFC of OLSR [1], which uses hop count as link metric. However, different metrics can be easily appended to *HELLO* or *TC* messages by using the extensible architecture of OLSRv2 [7].

### 3.2. Route computation

In OLSR, routes are determined by nodes each time they receive a new Topology Control messages (TC or *HELLO*). The routes to all the possible destinations are saved in the routing table. For MP-OLSR, an on-demand scheme is used to avoid the heavy computation of multiple routes for every possible destination. First, the multipath computation hypotheses will be introduced prior to presenting the resulting algorithm.

#### 3.2.1. Hypotheses

The aim of the multipath algorithm is to build a set $\mathscr{K}$ of $N$ paths, with no loops, joining a source node (noted $s$) and a destination node (noted $d$).

An ad hoc network can be represented by a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E}, c)$, where $\mathscr{V}$ is the set of vertices, $\mathscr{E} \subset \mathscr{V} \times \mathscr{V}$ the set of arcs and $c : \mathscr{V} \to \mathscr{R}^{*+}$ a strictly positive cost function. We assume the graph to be initially undirected i.e. $(v_1, v_2) \in \mathscr{E} \Rightarrow (v_2, v_1) \in \mathscr{E}$ and $c(v_1, v_2) = c(v_2, v_1)$ and loopless, i.e. no arcs join a node to itself. We also assume that no pair of vertices can be connected by more than one arc. Given an ordered pair of distinct vertices $(s, d)$ we can define a path between $s$ and $d$ as a sequence of vertices $(v_1, v_2, \ldots, v_m)$ so that $(v_q, v_{q+1}) \in \mathscr{E}$, $v_1 = s$ and $v_m = d$.

The above representation necessarily implies to define what the cost function $c$ refers to in an ad hoc context. The cost is incremental and the smaller the link, the better. Different metrics can be applied as mentioned in Section 3.1.

#### 3.2.2. Multipath Dijkstra Algorithm

For a source node $s$in the network, MP-OLSR will keep an *updated flag* for every possible node in the network to identify the validity of the routes to the corresponding node. Initially, for every node $i$, the $updatedFlag_i$ is set to *false*, which means the route to the corresponding destination does not exist or needs to be renewed. When there is a route request to a certain node $i$, the source node will first check the $updatedFlag_i$.

- If the $updatedFlag_i$ equals *false*, the node will perform Algorithm 1 to get the multiple paths to node $i$, save it into the multipath routing table, and renew the corresponding $updatedFlag_i$ to *true*.
- If the $udpdatedFlag_i$ equals *true*, the node will find a valid route to node $i$ in the multipath routing table.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

Every time the node receives a new *TC* or *HELLO* message and results in the changes in the topology information base, all the *updatedFlag*s will be set to *false*.

The algorithm to obtain the $N$ paths from $s$ to $d$ is detailed in Algorithm 1.

**Algorithm 1.** Calculate $N$ routes in $\mathscr{G}$ from $s$ to $d$

*MultiPathDijkstra*$(s, d, \mathscr{G}, N)$
$c_1 \leftarrow c$
$\mathscr{G}_1 \leftarrow \mathscr{G}$
**for** $i \leftarrow 1$ to $N$ **do**
  $SourceTree_i \leftarrow Dijkstra(\mathscr{G}_i, s)$
  $P_i \leftarrow GetPath(SourceTree_i, d)$
  **for all** arcs $e$ in $\mathscr{E}$
    **if** $e$ is in $P_i$ OR $Reverse(e)$ is in $P_i$ **then**
      $c_{i+1}(e) \leftarrow f_p(c_i(e))$
    **else if** the vertex $Head(e)$ is in $P_i$ **then**
      $c_{i+1}(e) \leftarrow f_e(c_i(e))$
    **else**
      $c_{i+1}(e) \leftarrow c_i(e)$
    **end if**
  **end for**
  $\mathscr{G}_{i+1} \leftarrow (\mathscr{V}, \mathscr{E}, c_{i+1})$
**end for**
**return** $(P_1, P_2, \ldots, P_N)$

The proposed algorithm is applied to a graph $\mathscr{G} = (\mathscr{V}, \mathscr{E}, c)$, two vertices $(s, d) \in \mathscr{E}^2$ and a strictly positive integer $N$. It provides an $N$-tuple $(P_1, P_2, \ldots, P_N)$ of $(s,d)$-paths extracted from $\mathscr{G}$. $Dijkstra(\mathscr{G}, n)$ is the standard Dijkstra algorithm which provides the source tree of the shortest paths from vertex $n$ in graph $\mathscr{G}$; $GetPath(SourceTree, n)$ is the function that extracts the shortest path to $n$ from the source tree *SourceTree*; $Reverse(e)$ gives the opposite edge of $e$;$Head(e)$ provides the vertex edge to which $e$ points.

The incremental functions $f_p : \mathscr{R}^{*+} \rightarrow \mathscr{R}^{*+}$ and $f_e : \mathscr{R}^{*+} \rightarrow \mathscr{R}^{*+}$ are used at each step to get a disjoint path between $s$ and $d$. $f_p$ is used to increase the costs of the arcs that belong to the previous path $P_i$ (or the opposite arcs belonging to it). This will make future paths tend to use different arcs. $f_e$ is used to increase the costs of the arcs that lead to vertices of the previous path $P_i$. Therefore, there are three possible settings:

- if $id = f_e < f_p$, paths tend to be arc-disjoint;
- if $id < f_e = f_p$, paths tend to be vertex-disjoint;
- if $id < f_e < f_p$, paths also tend to be vertex-disjoint, but when not possible they tend to be arc-disjoint.

where $id$ is the identity function.

By using the cost functions, we can expect to find diversity in the $N$ paths regarding the network topology. But contrary to providing strictly node-disjoint paths, the multiple paths generated by our algorithm do not need to be completely disjoint. The reason for this choice is that the number of disjoint paths is limited to the $(s,d)$ minimal cut (defined as the size of the smallest subset of edges one cannot avoid in order to connect $s$ to $d$). This minimal cut is often determined by the source and destination neighborhoods. For example, if $s$ only has three distinct neighbors, one cannot generate more than three disjoint paths from $s$ to $d$. As a consequence, this limitation of diversity may be local, the rest of the network being wide enough to provide far more than three disjoint paths. Another drawback of completely disjoint paths algorithms is that it may generate very long paths since every local "cut-off" can only be used once.

For example, in Fig. 1, node $S$ is trying to get multiple paths to node $D$. For *MultiPath Dijkstra Algorithm*, we use the number of hops as link cost metric and set $f_p(c) = 3c$ and $f_e(c) = 2c$ (more penalty to the used links). Initially, the cost for all the links is set to 1. For the first step, the shortest path S → A → B → G → D will be found. Then the cost functions will be used to increase the cost of the related arcs:

- S → A, A → B, B → G and G → D will be changed from 1 to 3 by using $f_p$.
- S → C and F → G will be changed from 1 to 2 by using $f_e$.

Then for the next step, the second shortest path S → C → E → F → G → D will be found. If we use the algorithm proposed in [17], and we delete the intermediate nodes A, B and G after the first step, it is impossible to obtain the second path.

As illustrated above, another benefit of using cost functions is that we can get a different multiple path set (node-disjoint or link-disjoint) by choosing different cost functions according to our preference and the network requirements. The network topology in Fig. 2 is presented as an example.

If we choose $f_p(c) = 3c$ and $f_e(c) = c$ (penalty is only applied to the used links), the paths we obtain are two link-disjoint paths: S → A → C → B → D and S → E → C → H → D.

If we choose $f_p(c) = 3c$ and $f_e(c) = 2c$ (penalty is also applied to used nodes), then the algorithm tends to search for node-disjoint paths. Then S → A → C → B → D and S → E → F → G → H → D will be found.

### 3.3. Route recovery

By using the scheme of the *Topology Sensing*, we can obtain the topology information of the network with the exchange of *HELLO* and *TC* messages. All this information is saved in the topology information base of the local node: link set, neighbor set or topology set. Ideally, the topology information base can be consistent with the real topology of the network. However, in reality, it is hard to achieve, mainly because of the mobility of the ad hoc network.

Firstly, for the *HELLO* and *TC* messages, there are certain intervals during each message generation (2s for *HELLO* and 5s for *TC* by default [1]). During this period, the topology might change because of the movement of the nodes. Secondly, when the control messages (especially the *TC* messages) are being transmitted in the network, delay or collision might happen. This will result in the control message being outdated or even lost.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

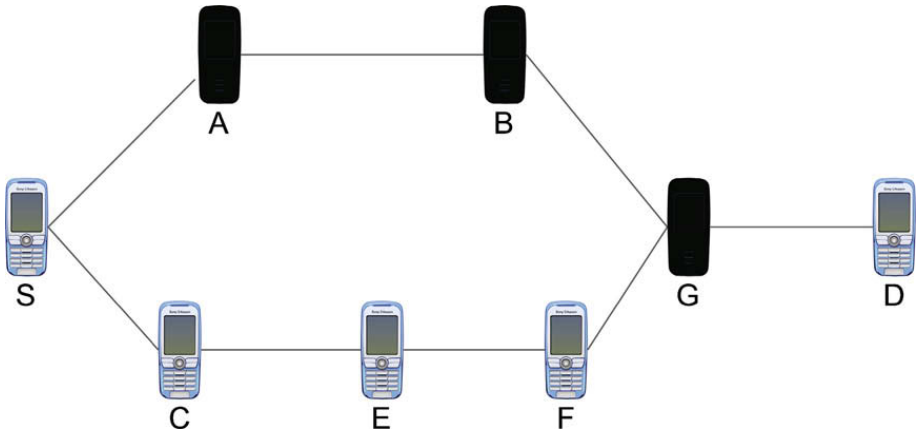



**Fig. 1.** Multiple Dijkstra Algorithm in sparse case. The node-disjoint path is non-desirable after nodes A, B and G are removed.

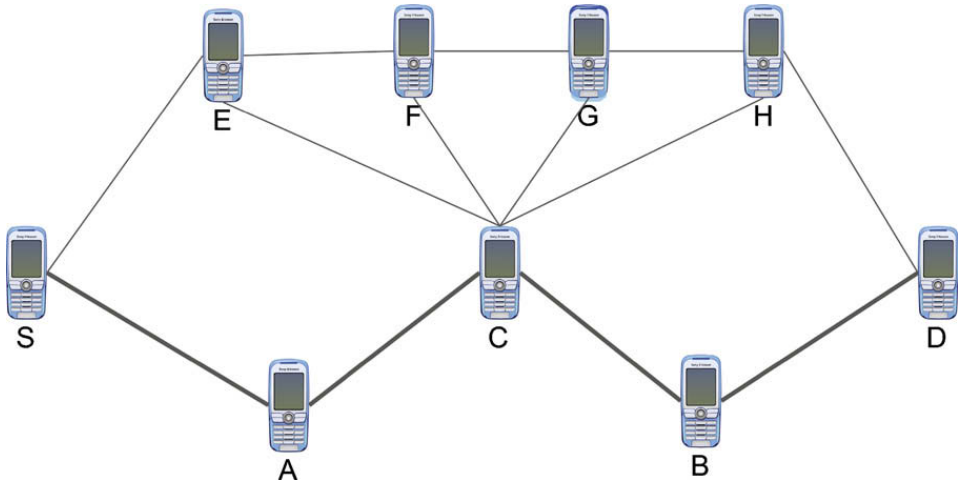



**Fig. 2.** Obtaining different path sets by using different cost functions. Path S → A → C → B → D will first be chosen but the second one might be link-disjoint or node disjoint (the bold lines) depending on the choice of cost functions.

Both of the two reasons mentioned above will result in the inconsistency between the real network topology and the node's topology information base. This means that when a node is computing the multiple paths based on the information base, it might use links that do not exist anymore, and cause the route failure.

Several techniques already exist in the literature to deal with the route failures in source routing. DSR handles route errors using route maintenance, mainly by sending RERR messages, which will increase the end-to-end delay significantly. In [35], the authors propose another method to avoid the effect of short term link deterioration by using opportunistic paths in mesh networks.

For MP-OLSR, we propose *route recovery* to overcome the disadvantage of the source routing. The principle is very simple: before an intermediate node tries to forward a packet to the next hop according to the source route, the node first checks whether the next hop in the source route is one of its neighbors (by checking the neighbor set). If so, the packet is forwarded normally. If not, then it is possible that the "next hop" is not available anymore. Then the node will recompute the route and forward the packet by using the new route.

In Fig. 3 we present an example of route recovery. Node *S* is trying to send packets to *D*. The original multiple paths we have are S → A → B → D and S → C → E → G → D. However, node *G* moves out of the transmission range of node *E* and makes the second path unavailable. The source node *S* is not able to detect the link failure immediately (because of the delay and long interval of *TC* messages) and keeps sending the packets along the path, and all these packets are dropped during this period if only the source routing is used. With *route recovery*, when the packet arrives, node *E* will first check if node *G* is still one of its neighbors, before forwarding the packet according to the source route. If not, node *E* will recompute the route to node *D*, and obtain E → F → D. Then the following packets will be sent through the new path.

Because the *route recovery* just checks the topology information saved in the local node, it will not introduce much extra delay. And most importantly, it will effectively improve the packet delivery ratio of the network. In our simulation, the delivery ratio of the protocol with *route recovery* is 50% higher on average than the one without *route recovery*. In fact, the SR-MP-OLSR [17] also has a very low delivery ratio like MP-OLSR without *route recovery* in

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

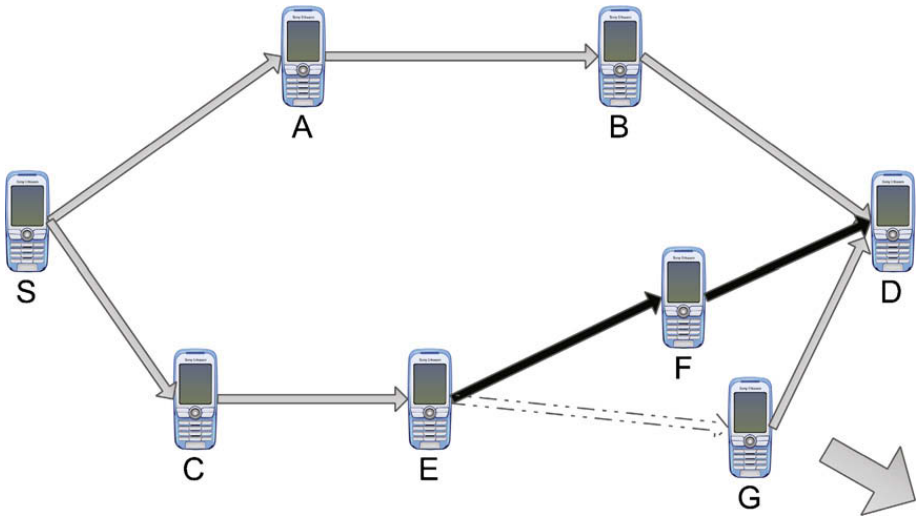



**Fig. 3.** An example of route recovery. The movement of node G makes the link *E–G* unavailable. Then node F is chosen as next hop of node E by using route recovery.

our settings. This means that the mere source routing based on OLSR is not adapted.

### 3.4. Loop detection

Loop in the network is always an important issue in routing. It is important to mention the Link Layer Notification (LLN) before taking the problem of the loops of the protocol. LLN is an extended functionality defined in [1], and implemented in different OLSR or MP-OLSR simulations and implementations [30,36]. If link layer information describing connectivity to neighboring nodes is available (i.e. loss of connectivity through absence of a link layer acknowledgement), this information can be used in addition to the information from the *HELLO* message to maintain the neighbor information base and the MPR selector set. The routing protocol can act on the acknowledgement from LLN (mainly the loss of links), and remove the corresponding links from its information base. The results of the real OLSRv2 testbed [24] and our previous work [30] based on NS2 [37] simulation show that LLN is very important and effectively improves the packet delivery ratio of the OLSR and MP-OLSR protocol.

In theory, the paths generated by the Dijkstra algorithm in MP-OLSR are loop-free. However, in reality, the LLN and *route recovery* which are used to adapt to the topology changes make the loops possible in the network. With LLN, when a node tries to send a packet over a link but fails in the end, the link layer will send feedback to the routing protocol to notify it of the link loss. This kind of abrupt interruption will result in additional operations on the topology information base rather than just regular *HELLO* and *TC* messages. This means that other nodes cannot be aware of these changes immediately. So, LLN might cause some inconsistency of the topology information in different nodes. And with *route recovery*, which might change the path in intermediate nodes, loops can occur temporarily in the network.

In Fig. 4 we give an example of how a loop is generated in the network. Node *A* is an intermediate node of a path.

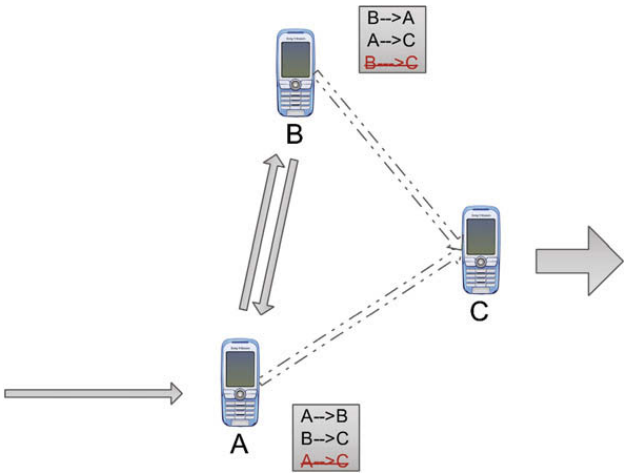



**Fig. 4.** An example of loop in the network. The movement of node C results in inconsistency of the information bases in nodes A and B. One transient loop is formed between A and B.

The packets with source route A → C arrive at node *A* and need to be forwarded to node *C*. Then node *C* moves out of the transmission range of node *A* and node *B*, and makes the links A → C, B → C no longer available.

When the new packets arrive at node *A*, the transmission to node *C* will be failed. Then in node *A*, the routing protocol will be acknowledged by LLN, and it will remove the link A → C from node *A*'s link set. For node *A*, although it can detect the link failure of A → C by LLN, it is hard to know the failure of B → C immediately. This is because link B → C can only be removed when the *NEIGHB_HOLD_TIME* (6 s by default [1]) expires. In the meantime, *route recovery* will be awaken. A new path A → B → C will be established and the following packets will be forwarded along the new path. Then the packets will be redirected to node *B*. The same operation will be performed in node *B*: LLN of the failure of B → C, and *Route Recovery*. Unfortunately, because node *B* cannot detect the link failure of A → C immediately, the new path obtained by *route recovery* is

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

B → A → C. Thus the packet will be returned to node *A*, and from node *A* to *B* again, creating a loop. This is not a permanent loop, but a transient loop which will exist for several seconds and will disappear when the related link expires. However, this kind of temporary loops will block the links in the loop and congest the related transmission area.

In [38], the authors also address the looping issues in OLSRv2, and note that LLN will significantly increase the number of loops. Therefore, the authors introduce two types of loop detection techniques: Mid-Loop Detection (LD-Mid) and Post-Loop Detection (LD-Post). LD-Mid just compares the address of the next hop with the address of the previous hop, so it is only able to detect "two-way" loops between two nodes. LD-Post records all incoming packets that need to be forwarded and compares them with each new incoming packet to see if the same packet has been through this node before. So, it can detect loops that are farther away, by using more memory. When a loop is detected, the Packet Discard strategy is used to drop the packets that are unlikely to reach the destination but only increase the load of the network.

For MP-OLSR, we propose a simple method based on source routing that can effectively detect loops without causing extra cost of memory: after the *route recovery* is performed, a new path will be calculated from the current node to the destination. The algorithm will make use of the new path if there is no loop. Or else it will try to find another path according to the multipath algorithm. If there is no suitable path, the packet will be discarded.

For the example in Fig. 4, node *A* will get a path A → B → C by *route recovery*. Then, when the packet arrives at node *B*, a new path B → A → C will be generated because of link breakage of B → C. Node *B* will compare the new one with the former source route A → B → C in the packet. We will find that the packet has already crossed node *A*, and so there might be a loop. So, the algorithm will try to find if there is any other possible path, or else the packet will be discarded.

Compared with LD-Post, which needs to keep a record of all the incoming packets, our loop detection mechanism could effectively detect the possible loops in the network without consuming extra memory space. By reducing the loops in the network, the network congestion can be reduced. Thus, the performance of the network can be improved, especially the end-to-end delay.

## 4. Simulation and performance evaluation

The simulations are performed to evaluate MP-OLSR which includes both the core functionality and the auxiliary functionality (*route recovery* and *loop detection*). The rest of the section is organized as follows. The simulation environment and assumption are first introduced in Section 4.1. Then we compare the performances between OLSR and MP-OLSR in different scenarios. The difference between the reactive and proactive protocols is also analyzed. The performance evaluation will be done in a real testbed in Section 5.

### 4.1. Environment and assumption

The Qualnet simulator 4.5.1 [39] is used for our simulation. It is the commercial version of GloMoSim and is widely used in academic research and industry. The MP-OLSR protocol is implemented on nOLSRv2 [36]. The scenarios include 81 nodes placed in an area of 1500 m by 1500 m to construct a mobile ad hoc networks. The initial position of the nodes is uniformly distributed like a $9 \times 9$ grid. The nodes will move at certain speed according to the *Random Way Point Model* (*RWP*). The maximum speed is 10 m/s to simulate pedestrian and cycle applications.

For each simulation, the total simulation time is 100 s. A simple Constant Bit Rate (CBR) UDP application runs at several nodes to measure the performance of data transmission. Each CBR application corresponds to a source–destination flow which generates UDP packets of 512 bytes at the source, at different rates. The flow starts 15 s after the simulation begins, to allow enough exchange of the routing messages. The data transmission lasts for 80 s to obtain an average behavior of each flow.

The 802.11b radio is used and the data rate is set to 11 Mbps. We use the two-ray ground pathloss model, the constant shadowing model with a shadowing mean of 4.0 dB, and the transmission power is set to 15 dBm. With these settings, the transmission distance is about 270 m in our simulations. We repeat each simulation 100 times and give the average results. Different random seeds are used to have different scenarios. Different seeds will generate different pseudo-random sequences used in simulation. This will affect the mobility according to the mobility model, back off timers, the interference pattern, etc. The detailed parameters are listed in Table 1 for the purpose of repeatability. Those parameters are widely used in Wi-Fi devices and simulation studies.

The parameters for OLSRv2 and MP-OLSR are presented in Table 2. For the multipath routing, according to previous work [15], 2–4 paths could offer a desired performance with acceptable complexity. Given the node density in our simulation scenario, the algorithm tries to exploit three paths. The Round-Robin packet-distribution scheme is used for packet distribution.

To compare the performances of the protocols, the following metrics are used.

- *Packet delivery ratio: The* ratio of the data packets successfully delivered at destination.
- *Average end-to-end delay: Averaged* over all surviving data packets from the sources to the destinations. This includes queuing delay and propagation delay.
- *Average time in FIFO queue: Average* time spent by packets in the queue.
- *Distribution of delay of received packets: This* measurement can give an idea of the jitter effect.

### 4.2. Comparison between MP-OLSR and OLSR

In this subsection, the performances of MP-OLSR and OLSR in different scenarios with different metrics are compared. The MP-OLSR used in this subsection is always with the *route recovery* and *loop detection* functionalities.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

**Table 1**
Simulator parameter set.

| Parameter | Values |
|---|---|
| Simulator | Qualnet 4.5.1 |
| Routing protocol | OLSRv2 and MP-OLSR |
| Simulation area | 1500 m × 1500 m |
| Mobility | RWP, max speed 0–10 m/s |
| Simulation time | 100 s |
| Applications | CBR |
| Application packet size | 512 bytes |
| Transmission interval | 0.1 s |
| CBR start-end | 15–95 s |
| Transport protocol | UDP |
| Network protocol | IPv4 |
| IP fragmentation unit | 2048 |
| Priority input queue size | 50,000 |
| MAC protocol | IEEE 802.11 |
| MAC propagation delay | uS |
| Short packet transmit limit | 7 |
| Long packet transmit limit | 4 |
| Rtx threshold | 0 |
| Physical layer model | PHY 802.11b |
| Wireless channel frequency | 2.4 GHz |
| Propagation limit | −111.0 dBm |
| Pathloss model | Two ray ground |
| Shadowing model | Constant |
| Shadowing mean | 4.0 dB |
| Transmission range | 270 m |
| Temperature | 290 K |
| Noise factor | 10.0 |
| Receive sensitivity | −83.0 |
| Transmission power | 15.0 dBm |
| Data rate | 11 Mbps |

**Table 2**
OLSR and MP-OLSR parameters.

| Parameter | Values |
|---|---|
| TC Interval | 5 s |
| *HELLO* Interval | 2 s |
| Refresh Timeout Interval | 2 s |
| Neighbor hold time | 6 s |
| Topology hold time | 15 s |
| Duplicate hold time | 30 s |
| Link Layer Notification | Yes |
| No. of path in MP-OLSR | 3 |
| MP-OLSR $f_e$ | $f_e(c) = 2c$ |
| MP-OLSR $f_p$ | $f_p(c) = 3c$ |

#### 4.2.1. Scenario with 81 nodes and 4 sources

In Fig. 5, the data delivery ratio of the two protocols is given. OLSR has a slightly better delivery ratio (about 3%) than MP-OLSR only at the speed of 1 m/s (3.6 km/h). This is because with more paths transmitting packets at the same time, there is a higher possibility of collision at the MAC layer. This inter-path interference can be eliminated by using multichannel techniques, which guarantee a different frequency band for each path [40]. In our case, there is only one channel used, so MP-OLSR has more packets dropped due to the collision at the MAC layer. However, as the speed of the nodes increases, the links become more unstable, and there are also more loops in the network. The delivery ratio of OLSR then decreases quickly and MP-OLSR outperforms OLSR.

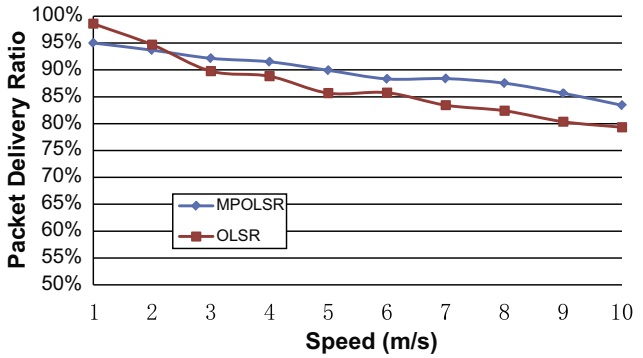



**Fig. 5.** Delivery ratio of MP-OLSR and OLSR in a scenario of 81 nodes and 4 sources.

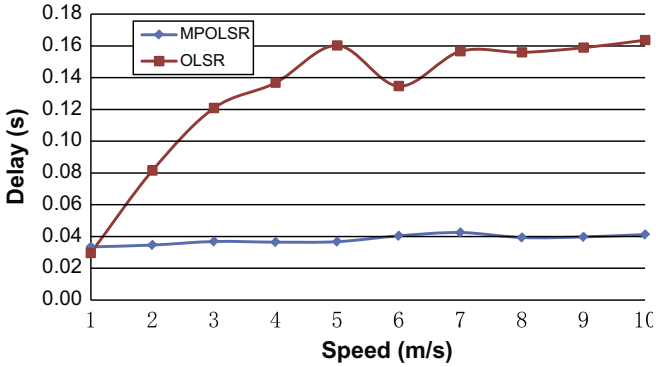



**Fig. 6.** Average end-to-end delay of MP-OLSR and OLSR in a scenario fo 81 nodes and 4 sources.

Compared to the slight gain in the delivery ratio (about 5% at high speed), the multipath protocol performs much better on average end-to-end delay than the single path, as shown in Fig. 6. The delay of OLSR is about four times more than MP-OLSR starting from 4 m/s (14.4 km/h). The end-to-end delay includes the *propagation delay* from the source to the destination and the *queue delay* in every relay nodes. The multipath protocol might have a longer propagation delay because some of the packets are forwarded through longer paths. However, what matters most is that it can effectively reduce the queue delay by distributing the packets to different paths rather than to a single one. In addition, the proposed loop detection mechanism can also reduce the unnecessary transmissions by avoiding the loops. As shown in Fig. 7, the MP-OLSR has much shorter average time in the queue compared to OLSR.

In our simulations, the MP-OLSR also offers more stable delay in different simulations. Figs. 8 and 9 show the distribution of end-to-end delay of all received packets in a scenario with medium mobility (0–5 m/s). The distribution of the delay of OLSR spreads more widely compared to MP-OLSR. In this case, in the 2731 packets received by using OLSR, 1967 packets (82.96%) are received with delay less than 0.1 s. For MP-OLSR, in the 2776 packets received, 2712 packets (97.69%) reach the destination in 0.1 s. In fact, the standard deviation of the delay of OLSR is at least 10 times more than that of MP-OLSR, and even sometimes up to 100 times more in the high mobile scenarios.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

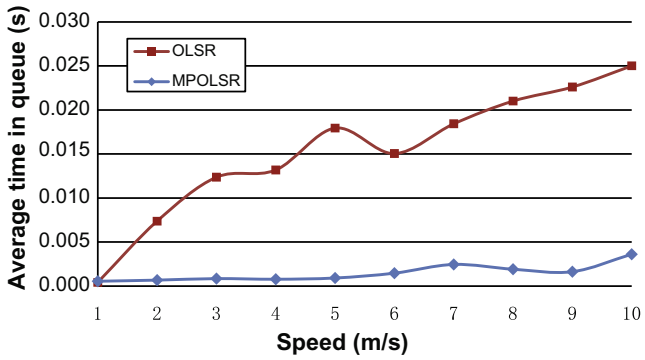



**Fig. 7.** Average time in queue of MP-OLSR and OLSR in a scenario of 81 nodes and 4 sources.

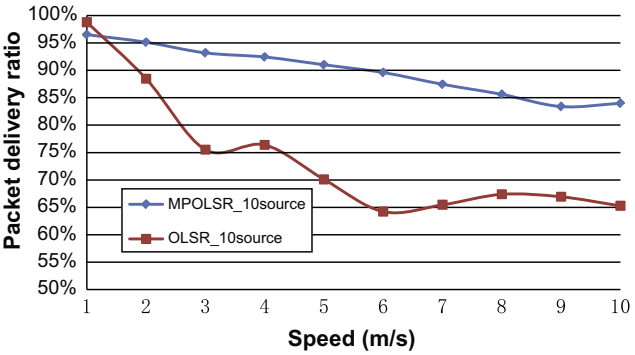



**Fig. 10.** Delivery ratio of MP-OLSR and OLSR in a scenario of 81 nodes and 10 sources.

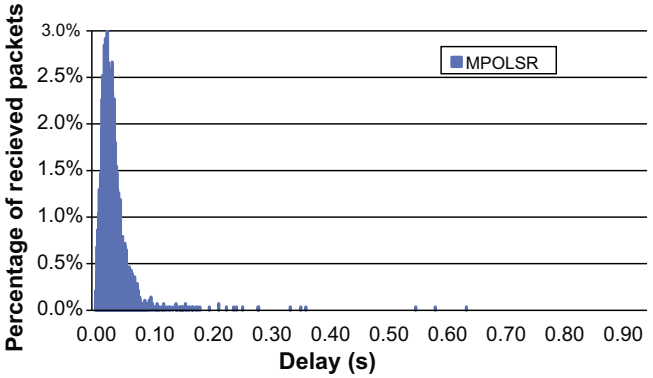



**Fig. 8.** Distribution of delay of received packets of MP-OLSR in a scenario of 81 nodes and 4 sources.

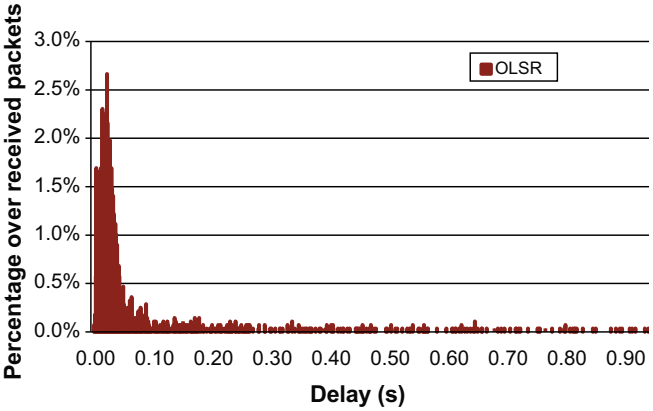



**Fig. 9.** Distribution of delay of received packets of OLSR in a scenario of 81 nodes and 4 sources.

For the routing control message, because the MP-OLSR does not change the topology sensing mechanism of the OLSR protocol, the two protocols tend to have the same number of routing control messages generated. In our simulation scenarios, the number of *HELLO* messages and *TC* messages generated is almost the same.

### 4.2.2. Scenario with 81 nodes and 10 sources

In this scenario, a higher load is applied to the network. There are 10 CBR sources transmitting the data packets to the destination, instead of four in the previous scenario in Section 4.2.1.

In Fig. 10, we present the delivery ratio of the two protocols. Compared to Fig. 5 of the 4-source scenario, whose delivery ratio is always superior to 80%, the OLSR protocol is more unstable with the high load. With a speed superior to 6 m/s (21.6 km/h), it drops to about 65%. The MP-OLSR is more robust with high load, and stays at about 85% even with high mobility.

Fig. 11 is the average end-to-end delay. As we can see from the figure, the increase of the delay is much more significant than in Fig. 6, which is more than 1 s at higher speed.

In this subsection, the protocols in different scenarios are simulated. The MP-OLSR and OLSR are compared in different scenarios. In addition to the scenarios presented in this section, we performed other scenarios like *duplex scenario* (two nodes send and receive packets from each other simultaneously, like VoIP application), *short-time scenario* (we have many more source-destination pairs, but very short transmission time, for an application like instant message sending). The results share the same trends and features as Sections 4.2.1 and 4.2.2. The multipath protocol is more adapted to the mobile scenarios.

From the simulation results we obtained, we can conclude that, compared to OLSR, MP-OLSR has almost the same performance with low mobility and low network load. However, when the speed of the nodes or the net-

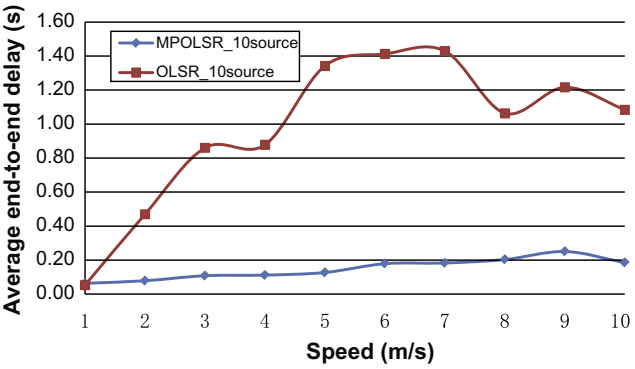



**Fig. 11.** Average end-to-end delay of MP-OLSR and OLSR in a scenario of 81 nodes and 10 sources.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

work load increases, MP-OLSR has a better delivery ratio and a shorter delay than OLSR due to the ability to distribute the packets through different paths. In our settings, the speed is from 1 m/s (3.6 km/h) to 10 m/s (36 km/s). From the trends we observe that MP-OLSR offers more advantages than OLSR at even higher speed.

In real life, the topology changes can be caused by not only node movements, but also by the changes in the environment. Which are not taken into account in the simulation (e.g. the moving objects in the scenario or perturbation). All these will result in link failures. This phenomenon will be presented in the next section with the discussion of the testbed.

### 4.3. Comparison between proactive routing and reactive routing

Because MP-OLSR is a hybrid routing protocol and uses source routing, we are also interested in the difference between proactive routing and reactive routing. The performance of DSR is also measured.

Figs. 12 and 13 present the delivery ratio and delay of DSR respectively (compared to OLSR and MP-OLSR from Section 4.2.1). The DSR has almost the same performance as the others protocols at low node speed (1 m/s and 2 m/s). However, when the mobility increases, the packet loss and delay of DSR increase significantly. In fact, the DSR uses source routing like MP-OLSR and also has a corresponding route recovery mechanism. But the reactive nature of DSR cannot adapt to frequent topology changes. Its route recovery mechanism is based on transmission of explicit RERR (route error) messages, which will increase rapidly as the node speed rises. On the other hand, the *route recovery* of MP-OLSR, which is based on link layer feedback and local network topology information base, does not need extra packet transmission in the network.

There are have been several multipath routing protocol proposed based on DSR, like SMR [12], which relies on the same reactive mechanism. The reactive feature makes those protocols hard to compare with the proactive ones in the scenarios with frequent topology changes. According to the simulation result of SMR [12], even if it can reduce the delay to 1/5 of DSR, it is still much higher than the proactive protocols.

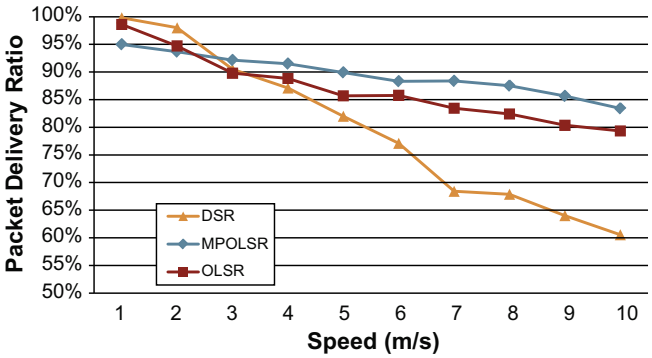



**Fig. 12.** Delivery ratio of DSR in a scenario of 81 nodes and 4 sources.

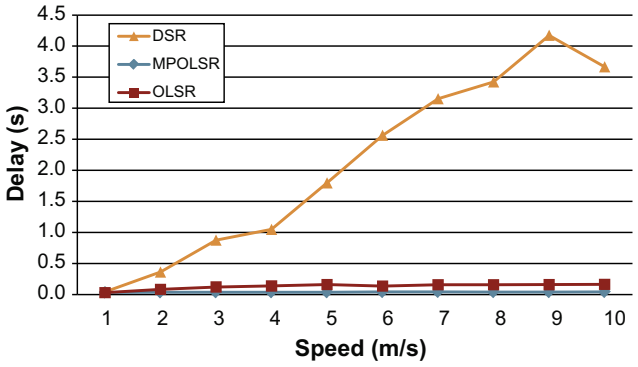



**Fig. 13.** Average end-to-end elay of DSR in a scenario of 81 nodes and 4 sources.

## 5. Testbed

This section presents experimentation results of MP-OLSR based on the real testbed that we implemented. Two different scenarios are proposed in order to verify the MP-OLSR protocol and compare with OLSR. The following test was realized at the *Ecole Polytechnique of University of Nantes, Nantes, France*.

### 5.1. Hardware and configuration

We implemented MP-OLSR based on ASUS 901 EeePcs (Fig. 14) with the parameters shown in Table 3.

At the same time, we used the UFTP (UDP-based file transfer protocol) [41] application to test bi-directional exchanges. We used the following log files to analyze the results of each scenario:

- *Wireshark log: Log* of packets captured by network interface in the network.
- *MP-OLSR log: Log* of routing operations of MP-OLSR (link detection, route calculation with KDijkstra algorithm).
- *UFTP log: Log* of the process of the UFTP application.

**Fig. 14.** Some ASUS eeePCs 901 before deployment at the headquarters of the testbed.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

**Table 3**
eeePC characteristics.

| Parameter | Value |
|---|---|
| CPU | Intel Atom N270 1.6 GHz |
| Memory | DDR2 1024 MB |
| Radio frequency | 2.487 GHz |
| Rate | 54 Mbps with auto bit-rate |
| Physical layer | 802.11 g |
| Operating system | Linux (backtrack three live) |
| Kernel | Linux 2.6.21.5 |
| OLSR version | Olsrd 0.5.6r2 |
| Network protocol analyzer | Wireshark 0.91.6 |

Relying on this information, we measured the following parameters for each test:

- Test duration.
- *Transfer duration: The* duration of UFTP transmission, from the first to the last packet sent.
- *Requested rate: The* rate initialized by the user.
- *Average rate: The* number of data bytes actually delivered during the transfer.

### 5.2. Implementation of multipath routing

The implementation of MP-OLSR is based on *Olsrd* [21]. The structure of the implementation is shown in Fig. 15.

Like OLSR, after the reception of *HELLO* and *TC* messages, MP-OLSR updates the network topology information base. But for MP-OLSR, no further operation is performed because the routing table is not calculated at that moment. To send out user data, the TCP/IP stack sends a request to MP-OLSR to calculate a set of paths to the destination (for the first request or when the topology changes) or to return calculated routes (for the following requests).

Note that for the convenience of development and debugging, the MP-OLSR module currently exists as an application layer program in our setting. In the future, for efficiency sake and practical use, rather than in a testbed, it is better to put the MP-OLSR module in the Linux kernel as a kernel module to avoid frequent context switches after the test for the protocol is completed.

### 5.3. Results

In this subsection, two different scenarios are presented to compare the performance between multipath and single path protocols.

#### 5.3.1. Scenario 1, OLSR and MP-OLSR on four paths

The first scenario presented includes six nodes. The location of the nodes is shown in Fig. 16. The object is to test the multipath algorithm, so we try to find as many paths as possible in this simple scenario. The number of paths for MP-OLSR is set to 4. A node with an IP address 10.0.0.100 is chosen as source, and another node 10.0.0.98 as destination. The distance from the source node to the destination node is about 60 m. There are different kinds of obstacles in this scenario: trees, buildings, and cars moving between the nodes, which will block certain links for a random period of time. *Iptable* rules are employed to block the direct transmission between source and destination to construct a multi-hop scenario.

In the following tests, the data rate is set to 62 KB/s to transfer a file with 17.8 MB. Results are presented in Table 4. For MP-OLSR, the transmission was finished in 6 min 12 s. Nodes with IP addresses 10.0.0.90, 10.0.0.95, 10.0.0.99 and 10.0.0.105 are chosen as intermediate nodes to relay the packets. During the transmission, 9.9% of the packets were lost. For OLSR, only 10.0.0.90 and 10.0.0.95 are used to forward the data packets. The connection is lost after 5 min 17 s of transmission. For the packets sent out, 37.53% were lost.

To compare the network performance of these two protocols, we also analyzed the log file from *Wireshark*. Figs. 17 and 18 show the number of packets sent out to different nodes from the source (10.0.0.100) to the next hops in each tick (1 s per tick) for MP-OLSR and OLSR respectively.

As we can see from Fig. 17, for a fixed source rate (10.0.0.100), the traffic load is distributed over four paths. The transmission is almost continuous even if some of the

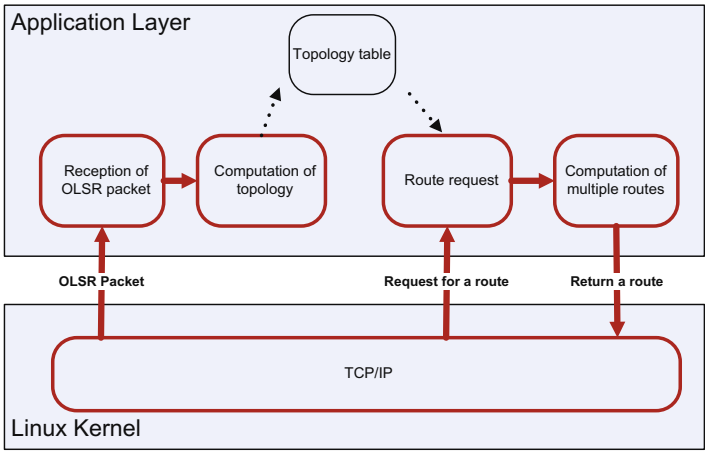



**Fig. 15.** Implementation of MP-OLSR on Linux.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

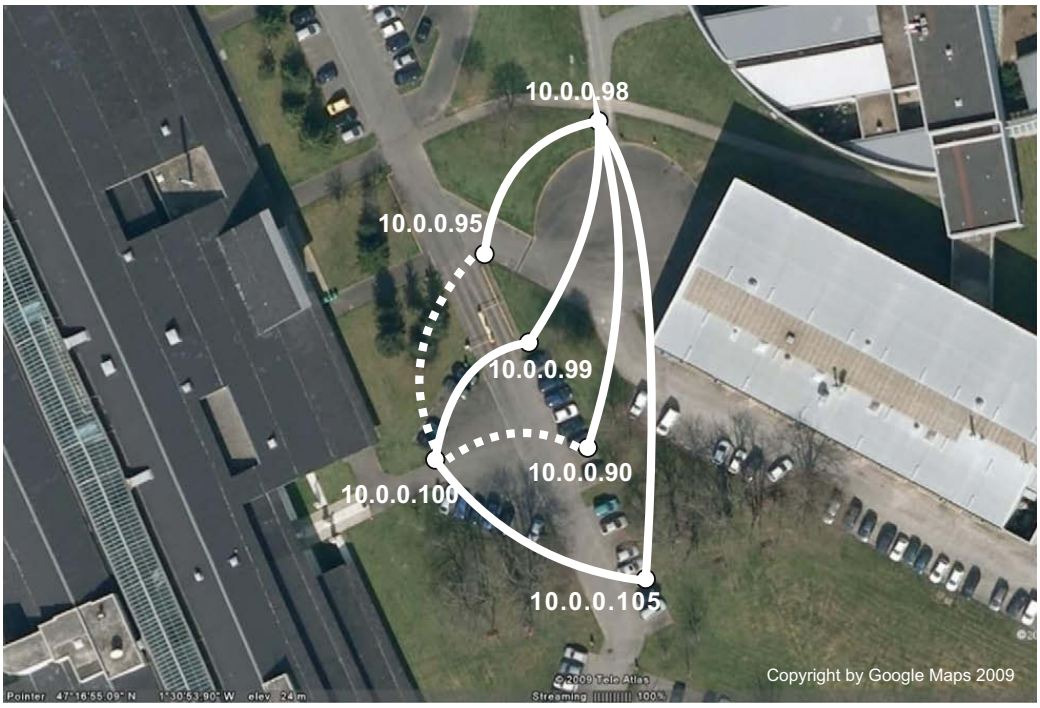



**Fig. 16.** Network topology of the scenario 1: OLSR and MP-OLSR with four paths. 10.0.0.100 is the source and 10.0.0.98 is the destination.

**Table 4**
Results of scenario 1, OLSR and MP-OLSR with four paths, data rate = 62 KB/s.

| Protocol | MP-OLSR | OLSR |
|---|---|---|
| Duration of the transmission | 6 m 12 s | 5 m 17 s (connection lost) |
| Test duration | 7 m 50 s | 6 m 26 s |
| Size of the sent file | 17.8 MB | 17.8 MB |
| Requested rate | 62 KB/s | 62 KB/s |
| Average rate | 48.99 KB/s | 38.73 KB/s |
| Packets sent by 10.0.0.100 | 15,002 | 9784 |
| Packets sent by 10.0.0.90 | 4503 | 5303 |
| Packets sent by 10.0.0.99 | 3715 | 0 (route not used) |
| Packets sent by 10.0.0.95 | 2084 | 2909 |
| Packets sent by 10.0.0.105 | 3726 | 0 (route not used) |
| Packets received by 10.0.0.98 | 13,516 | 6112 |
| Rate of lost packets | 9.90% | 37.53% (connection lost) |

links are unavailable. If certain links break, the traffic will be assigned to other nodes (for example, from 290 s to 300 s and 380 s to 420 s).

Compared to MP-OLSR, the transmission with OLSR (Fig. 18) is just through one path (so, the source rate is the same with the unique path and is not plotted here). The transmission is interrupted for a short period because the only path is unavailable. In this case, the node will try to find another route to the destination. But the data transmission will be stopped during this period (for example, 80–90 s, 115–130 s in Fig. 18). And if this kind of route switch takes too much time, the connection will be lost and result in the failure of the file transmission in the end.

*5.3.2. Scenario 2: OLSR and MP-OLSR routing on three paths*

In the second scenario, we compared OLSR and MP-OLSR with three paths which are three or four hops away. The goal is to test the protocol with longer paths in a complex scenario. The allocation of the nodes is shown in Fig. 19. The distance from the source to the destination is about 200 m. There is a large building between them. To reach the destination, the packets have to travel around the building or go through the hall inside the building.

Several links are unstable, mainly those in the parking-lot and inside the building (Fig. 20). MP-OLSR only uses one or two paths most of the time. However, despite these frequent changes in the network topology, the transmission is successful with MP-OLSR.

With OLSR, the UFTP connection stopped after sending several packets. This is because compared to multiple paths, the unstable paths have more negative effects on the single path routing protocol. So the file transmission failed in the end because of time out. Test results are presented in Table 5.

In this subsection, experimentations are performed to show the efficiency and validity of the MP-OLSR routing protocol in real scenarios. UFTP is taken as application example. Mobility is not considered in the presented scenario, but the network topology still changes due to the failure of links.

### 5.4. Discussion of simulation and testbed results

If we compare the results from the simulator and the real testbed, we will find that the network performance in real testbed is not as good as that in the simulator, but with the same trend. This is reasonable because in the network simulation, we simulate the physical layer in a free space by using the ideal mathematical model (two-ray ground and shadowing). And in the real scenario, there are many more factors that will affect the final results:

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

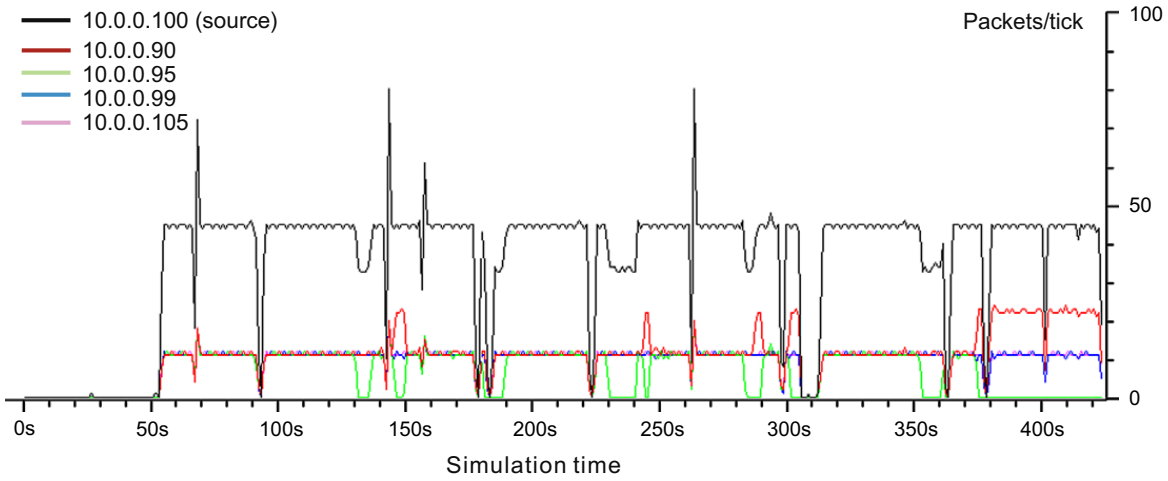



**Fig. 17.** Wireshark trace of scenario 1 with MP-OLSR and rate = 62 KB/s.

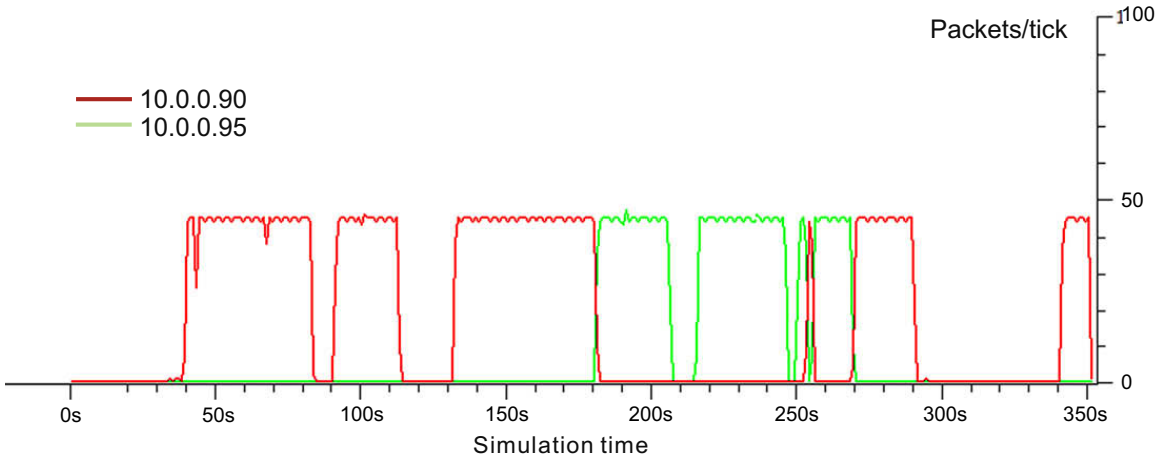



**Fig. 18.** Wireshark trace of scenario 1 with OLSR and rate = 62 KB/s (connection lost after 350 s).

obstacles, radio reflection (much more complex than two-ray ground model), texture in the environment, radio interference (from other Wi-Fi devices, etc.), or even humidity and temperature. However, although it is hard to simulate the exact real scenario with current simulation technology, we can still have a relative evaluation between the protocols through the simulation results.

Because of the limitation of resources, it is very hard for us to perform the tests in the real scenario with a large number of nodes and mobility like in the simulation (for example, tens of nodes moving in an area of 1 $km^2$). As a compromise, to test the performance of the protocols with the real UFTP application in a more complex scenario, we set a semi-realistic testbed based on IPNE (IP Network Emulation) [42] interface, as shown in Fig. 21. The IPNE implements a packet sniffer/injector. It sniffs the packets from the physical layer network, sends them through the Qualnet simulation, and injects them back to the physical network. The virtual Qualnet Network is transparent to the UFTP application.

We launch the UFTP transmission at 100 KB/s, with the same under layer settings as in Section 4. There are also 81 nodes in the network with medium mobility (maximum 5 m/s). The Wireshark traces are shown in Figs. 22 and 23 for OLSR and MP-OLSR respectively. The same thing happens with the tests in the real scenario, the file transmission using OLSR failed after a short period of time (170 s). And for MP-OLSR, although it also suffers from the unstable paths during the same period, the file successfully reached the destination in the end. This result is coherent with the real testbed in Section 5.3 .

Based on the results obtained from the simulator and testbed, we can conclude that MP-OLSR could offer better performances than OLSR, especially in the harsh scenarios (high mobility in simulation and frequent link breakage in the testbed). This is mainly because the proactive-based Multipath Dijkstra Algorithm could find appropriate multiple paths and distribute the traffic into different paths by source routing. And the *route recovery* and *loop detection* could effectively avoid the disadvantage of source routing and the possible loops in the network.

## 6. Compatibility between MP-OLSR and OLSR

As presented in the related work, most of the multipath routing protocols proposed are based on single-path ver-

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

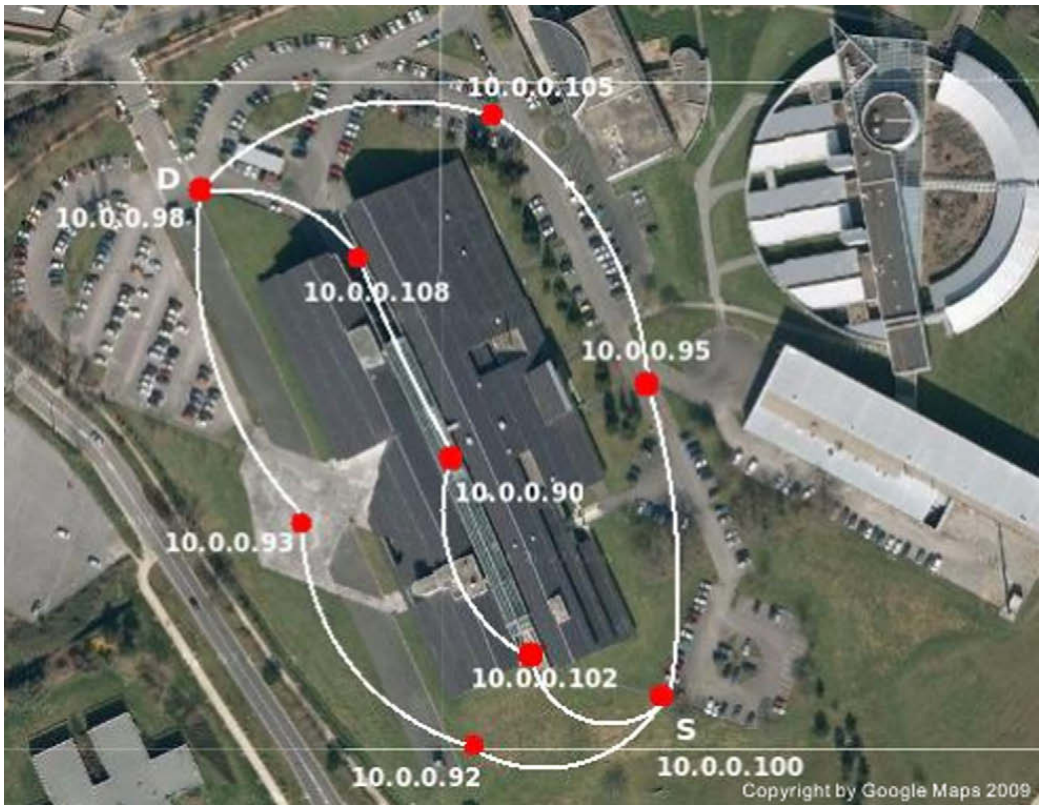



**Fig. 19.** Network topology of scenario 2: OLSR and MP-OLSR with three paths, 10.0.0.100 is the source and 10.0.0.98 is the destination.

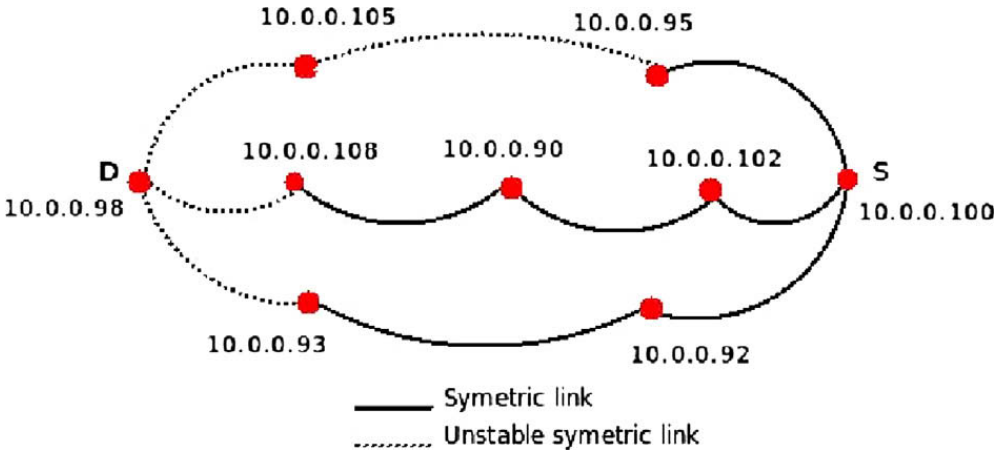



**Fig. 20.** Quality of links in scenario 2: OLSR and MP-OLSR with three paths.

**Table 5**
Results of scenario 2, OLSR and MP-OLSR routing with three paths.

| Protocol | MP-OLSR | OLSR |
|---|---|---|
| Duration of the transmission | 9 m 40 s | 8 m 43 s (connection lost) |
| Test duration | 14 m 6 s | 9 m 6 s |
| Size of the sent file | 17.8 MB | 17.8 MB |
| Requested rate | 62 KB/s | 62 KB/s |
| Average rate | 31.42 KB/s | 34.85 KB/s |
| Packets sent by 10.0.0.100 | 14,528 | 12,548 |
| Packets received by 10.0.0.98 | 9145 | 8544 |
| Rate of lost packets | 37.05% | 31.90% (connection lost) |

sion of an existing routing protocol: AODV and AOMDV [11], DSR and SMR [12]. However, the backward compatibility of those protocols with its single-path version is not considered. In fact, because the reactive-based multipath routing requires extra operations during the route discovery to gather enough information for the multipath construction, the compatibility is not easy to achieve.

Given OLSR, the standardized protocol and the distributed/heterogeneity property of ad hoc networks, it will be interesting to study the compatibility between MP-OLSR and OLSR which will facilitate the application and deployment of the multipath routing.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

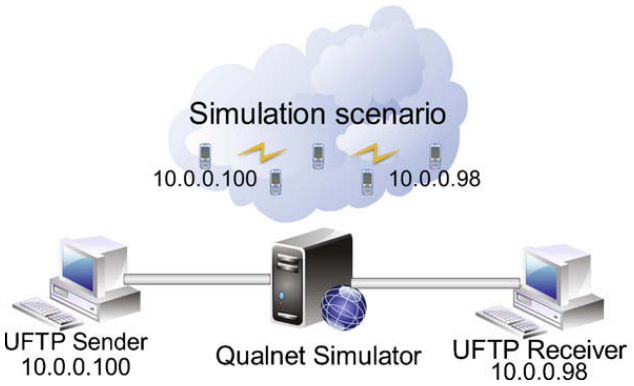



**Fig. 21.** Semi-realistic IPNE testbed, with UFTP application.

### 6.1. The problem of compatibility

As presented in Section 1, MP-OLSR is based on the OLSR protocol. In fact, the two protocols follow the same steps for the detection of neighborhood and network topology, but they are different in routing the data packets. In OLSR, the source node calculates the shortest path to the destination and sends the packets to the next hop. The intermediate OLSR nodes forward the packets according to their routing table. In MP-OLSR, the source node calculates different paths, and specifies one path in each packet (source routing) before sending it to the next hop. And the intermediate MP-OLSR nodes will forward the packet according to the source routing initialized by the source node.

The study of backward compatibility makes the deployment of the new protocol much easier because it can make use of the network that already exists. Moreover, it allows to return to the single-path version if necessary (basically with no mobility and low traffic). It is important to point out that we are studying the mutual compatibility between OLSR and MP-OLSR. In the following, we show that each protocol can use the nodes of either protocol to perform routing, with respect to QoS parameters.

To ensure the compatibility between the two protocols OLSR and MP-OLSR, we propose an implementation of MP-OLSR protocol based on *IP-source routing* [43].

The IP-source routing allows to partially or completely specify the path in the data packets. This option is mostly used by network administrators to test the routes. The IP protocol supports two forms of source routing:

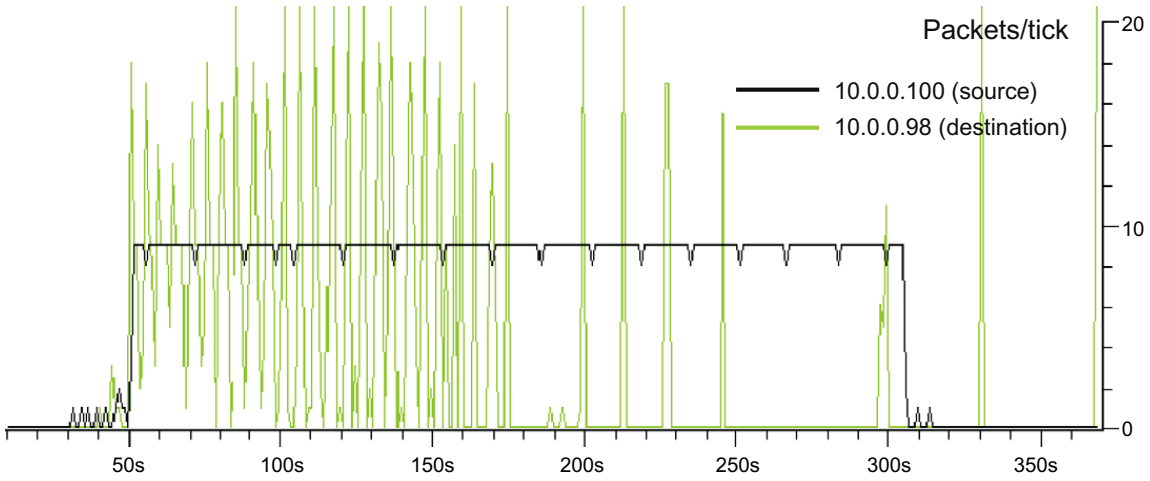



**Fig. 22.** Wireshark trace of UFTP source and destination nodes, 81 nodes OLSR ad hoc network.

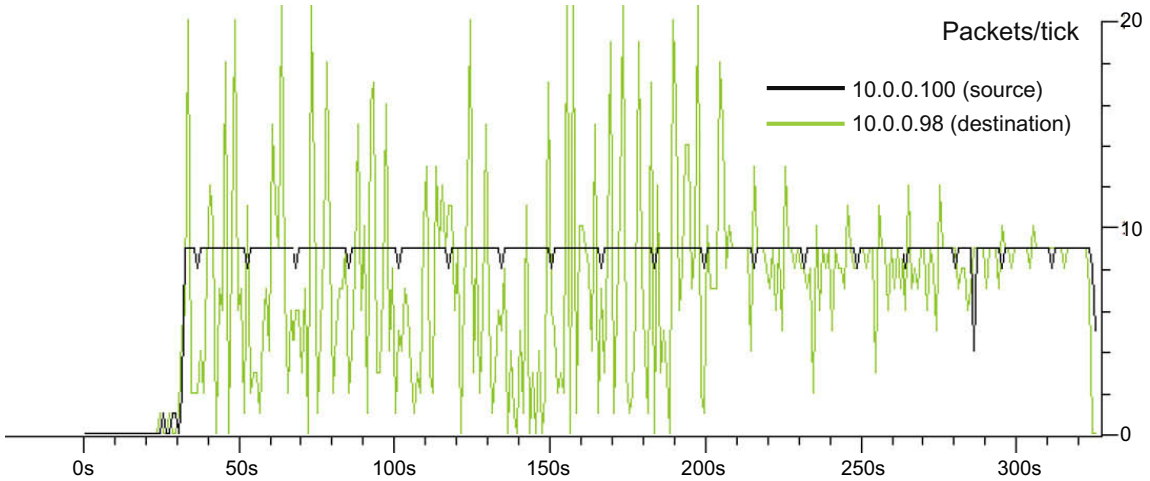



**Fig. 23.** Wireshark trace of UFTP source and destination nodes, 81 nodes MP-OLSR ad hoc network.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

1. *Strict source routing: The* exact route of the packet is specified by the sender.
2. *Loose source routing: The* sender gives one or more hops that the packet must go through. This means that before reaching its final destination, the packet should go through the following IP addresses as intermediate destinations. These intermediate destinations are responsible for forwarding it to the next destination.

The IP-source routing option implicitly ensures the compatibility between OLSR and MP-OLSR. Indeed, when OLSR nodes receive an MP-OLSR packet (with source route), they will forward it directly according to the IP-source routing. On the other hand, when an MP-OLSR node receives a packet generated by an OLSR node (without source route), it will recompute the path as the packet comes from its application layer and attach the source route to the packet. So this packet can also take advantage of the *loop detection* in all the following MP-OLSR intermediate nodes.

However, the IP-source routing option accepts only a maximum of nine addresses (in total, 11 nodes including source and destination nodes = 10 hops) because of the limitation of the length of the IP head. Therefore, when the route contains more than 10 hops (very large network), other solutions must be proposed. To solve this problem, we propose two possible solutions:

- The first is by using the *loose source routing*: the source node just specifies 10 "key" hops that the packet needs to travel through to reach the destination and allows each intermediate node to choose a route to the next hop. This solution can guarantee the source routing as defined by the source node. But it requires the *loose source routing* support from the IP layer and the MP-OLSR protocol to maintain a routing table just like OLSR.

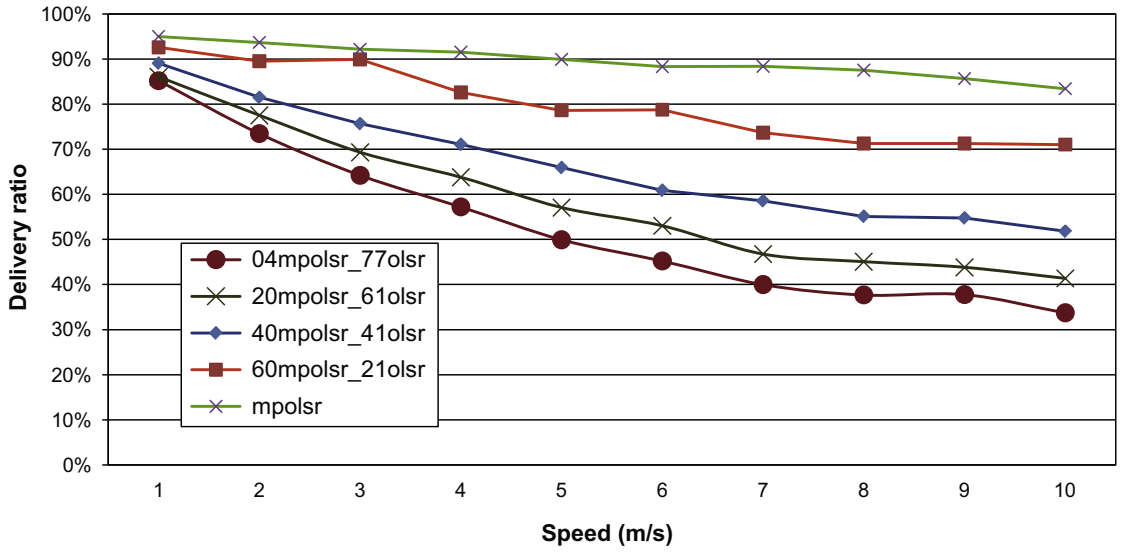



**Fig. 24.** Ratio of delivered packets for a network of 81 OLSR and MP-OLSR mobile nodes, with MP-OLSR source nodes.

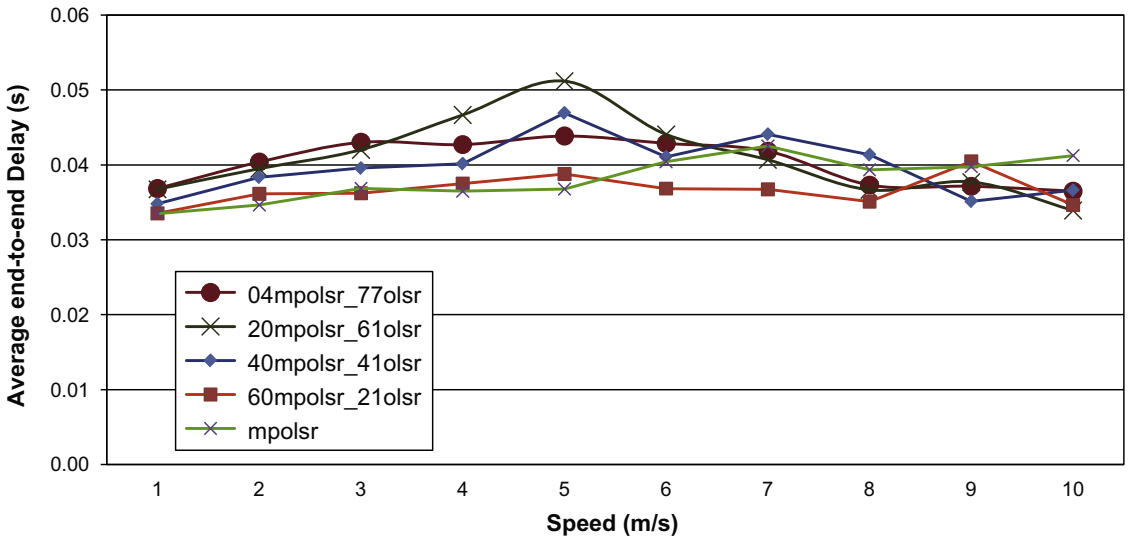



**Fig. 25.** Average end-to-end delay for a network of 81 OLSR and MP-OLSR mobile nodes, with MP-OLSR source nodes.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

- The second solution is that if the path found by MP-OLSR is more than 10 hops, it will just forward packets to the next hop, instead of using the source routing. The next hop will decide the rest of the route, no matter whether it is an MP-OLSR node or an OLSR node. This solution is easy to implement but does not guarantee the multiple paths as defined by the source node. In the following simulation, results are based on this solution.

### 6.2. Simulation results

In this section, we present routing performances when OLSR and MP-OLSR protocols cooperate in the same network in order to check the backward compatibility.

The following results are obtained with Qualnet simulator, and scenarios of 81 mobile nodes using a strict IP-source routing. Nodes were uniformly placed initially on a square grid of 1500 m × 1500 m. Table 1 summarizes the simulation settings.

#### 6.2.1. Scenario 1: network with MP-OLSR source nodes

In the first scenario, the simulation results are obtained for a network of four MP-OLSR source nodes. We change the density of the OLSR nodes. We start by studying a network of four MP-OLSR sources and all the rest 77 hosts are OLSR nodes (denoted *4mpolsr_77olsr*), then we replace OLSR nodes by MP-OLSR nodes, and for each scenario we note the number of MP-OLSR and OLSR nodes in the network (*20mpolsr_61olsr*, *40mpolsr_41olsr*, *60mpolsr_21olsr*). Finally, we give the results for a network of only MP-OLSR nodes.

Fig. 24 shows the delivery ratio when the OLSR nodes are involved in the routing by carrying the packet gener-

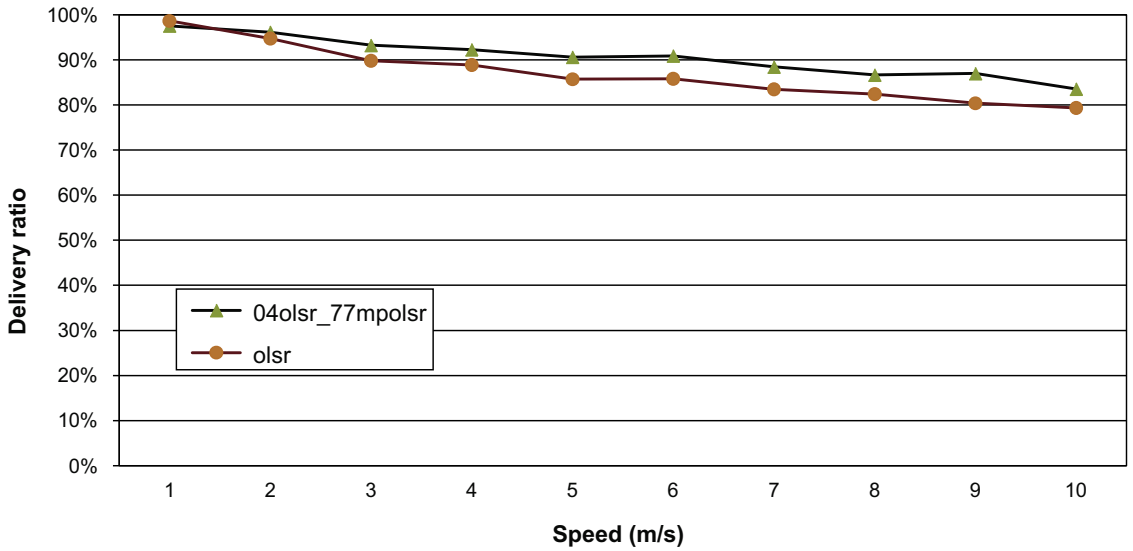



**Fig. 26.** Ratio of delivered packets for a network of 81 OLSR and MP-OLSR mobile nodes, with OLSR source nodes.

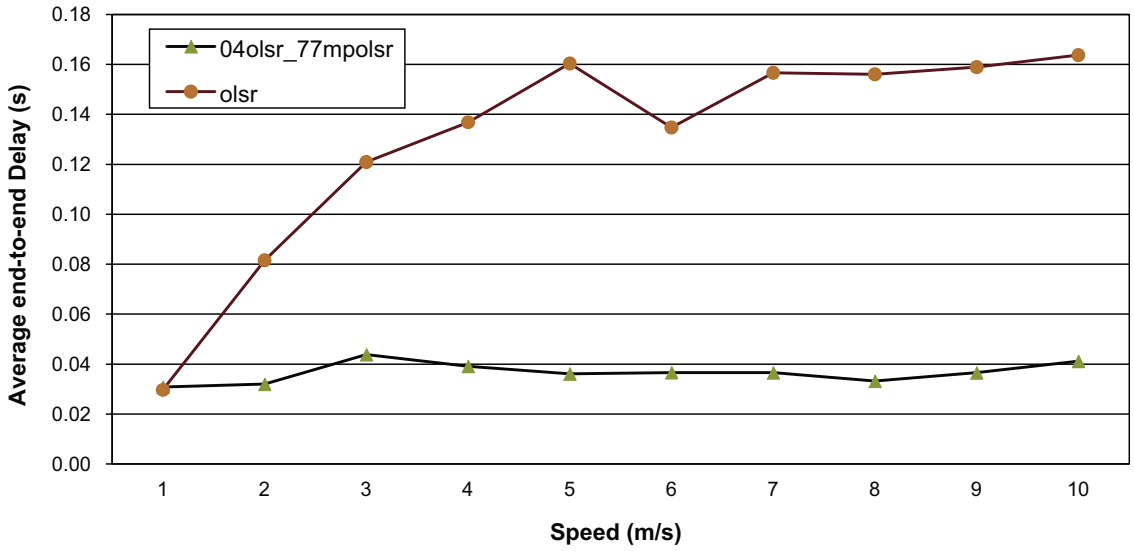



**Fig. 27.** Average end-to-end delay for a network of 81 OLSR and MP-OLSR mobile nodes, with OLSR source nodes.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

ated by the MP-OLSR source nodes. In general, the OLSR nodes have no problem in forwarding the source routing packets generated by MP-OLSR nodes. However, the OLSR intermediate nodes cannot have the same performance with MP-OLSR nodes. This is mainly because OLSR cannot perform *route recovery* and *loop detection* for the packets. So, in the scenarios where the density of OLSR nodes is too high, it is better for MP-OLSR nodes to just forward the packet to the next hop without appending the source route. On the other hand, OLSR nodes do not significantly affect the average end-to-end delay of the MP-OLSR protocol (Fig. 25).

*6.2.2. Scenario 2: network with OLSR source nodes*

In the second scenario, we simulate the case in which the sources are OLSR nodes. Here, we have four OLSR source nodes with 77 MP-OLSR nodes, and we compare this scenario with that of all OLSR nodes. In Figs. 26 and 27, we present the delivery ratio and average end-to-end delay respectively. As we can see in these figures, OLSR nodes have no problems in sending packets to MP-OLSR nodes. Furthermore, with the help of the *loop detection* and the multipath feature of MP-OLSR, we can increase the packet delivery ratio and reduce the end-to-end delay of the network.

In conclusion, the MP-OLSR and OLSR can cooperate within in the same network. This feature makes the deployment of the MP-OLSR protocol much easier because it can make use of the existing OLSR network. Because MP-OLSR nodes can perform multipath routing and *loop detection*, it can improve the performance of the network. For OLSR nodes, the route failure might increase when using source routing, because they do not have *route recovery*. Therefore, when the density of OLSR nodes is very high, it is better for MP-OLSR to forward the packets without source route.

## 7. Conclusion and future works

In this paper, the multipath optimized link state routing (MP-OLSR) protocol is proposed. The extension of the single-path version includes a major modification of the Dijkstra algorithm (two cost functions are now used to produce multiple disjoint or non-disjoint paths), auxiliary functions, i.e. *route recovery* and *loop detection* to guarantee quality of service and a possible backward compatibility based on IP-source routing. The MP-OLSR can effectively improve the performance of the network (especially in the scenarios with high mobility and heavy network load) and also be compatible with OLSR. Simulations and real testbed demonstrate our contributions.

The advantages of a link state multipath approach are clearly exemplified. Classical issues in MANET are covered: scalability, lifetime of the network (by reducing the number of forwarded packets per node) and non reliable wireless transmissions. From the security point of view, we can argue that the multipath approach can increase confidentiality. By a spatial diversity, the classical Man-In-the-Middle (MiM) attack is quite ineffective assuming a large cooperation between applicants. For integrity purpose, a redundant coding can be integrated to the routing protocol in order to ensure an higher delivery rate [44].

The last point is the increasing needs of applications. The best benefit of MP-OLSR for QoS occurs in end-to-end delay and jitter that is precisely required for critical multimedia services. Routing decision based on different types of scalable streams (especially video streams) can be further exploited, combined with the study on the metrics of link quality to fulfil the QoS. This constitutes the subject of future work.

## Acknowledgements

This work is supported by the French Program RNRT (Réseau National de Recherche en Télécommunications) under the Project SEREADMO (Sécurisation des Réseaux ad hoc par Tansformée Mojette, ANR-05-RNRT-028-01). We also wish to thank the reviewers for their valuable comments and suggestions that helped improve the quality and presentation of this paper.

## References

[1] T. Clausen, P. Jacquet, IETF Request for Comments: 3626, Optimized Link State Routing Protocol OLSR, October 2003.
[2] M. Abolhasan, T. Wysocki, E. Dutkiewicz, A review of routing protocols for mobile ad hoc networks, Ad Hoc Networks 2 (1) (2004) 1–22.
[3] D.B. Johnson, Y. Hu, D.A. Maltz, IETF Request for Comments: 4728, The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4, February 2007.
[4] C. Perkins, E. Royer, Ad-hoc on-demand distance vector routing, in: Second IEEE Workshop on Mobile Computing Systems and Applications, 1999, pp. 90–100.
[5] M. Tarique, K.E. Tepe, S. Adibi, S. Erfani, Survey of multipath routing protocols for mobile ad hoc networks, Journal of Network and Computer Applications 32 (2009) 1125–1143.
[6] T. Clausen, C. Dearlove, B. Adamson, IETF Request for Comments: 5148, Jitter Considerations in Mobile Ad Hoc Networks, February 2008.
[7] T. Clausen, C. Dearlove, J. Dean, C. Adjih, IETF Request for Comments: 5444, Generalized Mobile Ad Hoc Network (MANET) Packet/Message Format, February 2009.
[8] T. Clausen, C. Dearlove, IETF Request for Comments: 5497, Representing Multi-Value Time in Mobile Ad Hoc Networks (MANETs), March 2009.
[9] T. Clausen, C. Dearlove, J. Dean, IETF Internet Draft, MANET Neighborhood Discovery Protocol (NHDP), draft-ietf-manet-nhdp-10, July 2009.
[10] T. Clausen, C. Dearlove, P. Jacquet, IETF Internet Draft, The Optimized Link State Routing Protocol Version 2, draft-ietf-manet-olsrv2-10, September 2009.
[11] M.K. Marina, S.R. Das, On-demand multi path distance vector routing in ad hoc networks, in: Proceedings of the Ninth International Conference on Network Protocols, IEEE Computer Society, Washington, DC, USA, 2001, pp. 14–23.
[12] S. Lee, M. Gerla, Split multipath routing with maximally disjoint paths in ad hoc networks, Helsinki, Finland, 2001, pp. 3201–3205.
[13] Z. Yao, J.J. Jiang, P. Fan, Z. Cao, V. Li, A neighbor table based multipath routing in ad hoc networks, in: 57th IEEE Semi Annual Vehicular Technology Conference, vol. 3, 2003, pp. 1739–1743.
[14] Z. Ye, S.V. Krishnamurthy, S.K. Tripathi, A framework for reliable routing in mobile ad hoc networks, in: IEEE INFOCOM, San Francisco, CA, USA, 2003, pp. 270–280.
[15] E. Cizeron, S. Hamma, A multiple description coding strategy for multi-path in mobile ad hoc networks, in: International Conference on the Latest Advances in Networks (ICLAN), Paris, France, 2007.
[16] M. Kun, Y. Jingdong, R. Zhi, The research and simulation of multipath olsr for mobile ad hoc network, in: International Symposium on Communications and Information Technologies (ISCIT), 2005, pp. 540–543.

[17] X. Zhou, Y. Lu, B. Xi, A novel routing protocol for ad hoc sensor networks using multiple disjoint paths, in: 2nd International Conference on Broadband Networks, Boston, MA, USA, 2005.
[18] H. Badis, K.A. Agha, Qolsr multi-path routing for mobile ad hoc networks based on multiple metrics: bandwidth and delay, in: IEEE Vehicular Technology Conference, Los Angeles, CA, USA, 2004, pp. 2181–2184.
[19] I. Stepanov, K. Rothermel, On the impact of a more realistic physical layer on manet simulations results, Ad Hoc Networks 6 (1) (2008) 61–78.
[20] D. Maltz, J. Broch, D. Johnson, Lessons from a full-scale multihop wireless ad hoc network testbed, IEEE Personal Communications 8 (2001) 8–15.
[21] Olsrd, an adhoc wireless mesh routing daemon. <http://www.olsr.org/>.
[22] Athens wireless network. <http://wind.awmn.net/?page=nodes>.
[23] Funkfeuer.at, <http://map.funkfeuer.at/>.
[24] Y. Owada, T. Maeno, H. Imai, K. Mase, Olsrv2 implementation and performance evaluation with link layer feedback, in: Proceedings of the 2007 International Conference on Wireless Communications and Mobile Computing, Honolulu, Hawaii, USA, 2007.
[25] Y. Zhai, O. Yang, W. Wang, Y. Shu, Implementing multipath source routing in a wireless ad hoc network testbed, in: IEEE Pacific Rim Conference on Communications, Computers and signal Processing, 2005, pp. 292–295.
[26] S. Mao, S. Lin, S. Panwar, Y. Wang, A multipath video streaming testbed for ad hoc networks, in: IEEE Vehicular Technology Conference, 2003, pp. 2961–2965.
[27] K. Taniyama, T. Morii, S. Koizumi, K. Noguchi, Experimental evaluation of an on-demand multipath routing protocol for video transmission in mobile ad hoc networks, Journal of Zhejiang University SCIENCE A (2006) 145–150.
[28] Omf, the testbed control and management framework. <http://omf.mytestbed.net>.
[29] J.W. Tsai, T. Moors, Minimum interference multipath routing using multiple gateways in mesh networks, in: IEEE Conference on Mobile ad-hoc and Sensor Systems, Atlanta, Georgia, 2008.
[30] J. Yi, E. Cizeron, S. Hamma, B. Parrein, Simulation and performance analysis of MP-OLSR for mobile ad hoc networks, in: IEEE WCNC: Wireless Communications and Networking Conference, Las Vegas, USA, 2008.
[31] A.-M. Poussard, W. Hamidouche, R. Vauzelle, Y. Pousset, B. Parrein, Realistic SISO and MIMO physical layer implemented in two routing protocols for vehicular ad hoc network, in: IEEE ITST, Lille, France, 2009.
[32] H. Rogge, E. Baccelli, A. Kaplan, MANET Internet-draft: packet sequence number based ETX metric for mobile ad hoc networks, December 2009. <http://www.ietf.org/id/draft-funkfeuer-manet-olsrv2-etx-00.txt>.
[33] D. Johnson, G. Hancke, Comparison of two routing metrics in olsr on a grid based mesh network, Ad Hoc Networks 7 (2009) 374–387.
[34] C. Dearlove, T. Clausen, P. Jacquet, Manet internet-draft: link metrics for olsrv2, July 2009. <http://tools.ietf.org/html/draft-dearlove-olsrv2-metrics-04>.
[35] J.W. Tsai, T. Moors, Opportunistic multipath routing in wireless mesh networks, in: Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 22, 2009, pp. 71–85.
[36] nOLSRv2, Niigata olsrv2 Implementation, Niigata University, Japan.
[37] Network simulator 2. <http://www.isi.edu/nsnam/ns/>.
[38] L. Speakman, Y. Owada, K. Mase, An analysis of loop formation in OLSRv2 in ad-hoc networks and limiting its negative impact, in: IEEE International CQR Workshop, Naples, Florida, USA, 2008.
[39] Qualnet simulator. <http://www.scalable-networks.com/products>.
[40] H. Gharavi, Multichannel mobile ad hoc links for multimedia communications, Proceedings of the IEEE 96 (1) (2008) 77–96.
[41] Udp based ftp with multicast. <http://www.tcnj.edu/bush/uftp.html>.
[42] Scalable network technologies, IP Network Emulation Interface.
[43] IETF Request for Comments: 791, IP: Internet Protocol, September 1981.
[44] N. Normand, B. Parrein, I. Svalbe, A. Kingston, Erasure coding with the fnite radon transform, in: Proceeding of IEEE WCNC, Sydney, Australia, 2010.

**Jiazi Yi** was born in 1984 and received his B.E. degree in Computer Science from Central South University, China in 2005. Then he joined the double-master program between South China University of Technology, China and Ecole Polytechnique of University of Nantes, France, and received the master's degree in Computer Science and Electronic System respectively in 2008. He is currently a PhD candidate in Image–Video-Communication (IVC) team of the Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN UMR 6597), France. His current research interests include routing protocol and QoS of multimedia applications for MANET.

**Asmaa Hassiba Adnane** received the master's degree in Computer Science from Abdelhamid Ibn Badis University (Mostaganem, Algeria, 2004) and the PhD degree in Computer Science from University of Rennes 1 (Rennes, France). She developed her doctoral thesis from January 2006 to December 2008, working in the Security Group of the Information Systems and Networks – SSIR (EA 4039) at the Ecole Superieure d Electricite – Supelec, Rennes, France. She is now lecturer-researcher at the Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN) and Ecole Polytechnique de Nantes, France.

**Sylvain David** was born in Cholet, France in 1986. He received engineer's degree in Computer Science from the graduate school of engineers of the University of Nantes in 2009 and the University of Technology Diploma in Networks and Telecommunications from the University of La Rochelle in 2006. He is currently a research engineer at the Research Institute for Communications and Cybernetics of Nantes (IRCCyN), France.

**Benoît Parrein** was born in Normandy in 1975. After studies in Physics at University of Paris 7, he received the master's degree in Medical Imaging from the Medical School of Angers, France and the PhD degree in Computer Science from the University of Nantes, France in 1998 and 2001 respectively. Since 2004, he is associate professor at the Computer Science department of Polytech'Nantes, engineer school and affiliated to the Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN – UMR CNRS 6597). His main interests include image communication, multiple description coding, error control theory and mobile ad hoc networks.

Please cite this article in press as: J. Yi et al., Multipath optimized link state routing for mobile ad hoc networks, Ad Hoc Netw. (2010), doi:10.1016/j.adhoc.2010.04.007

# The Mojette Transform: Erasure Coding Technique for Distributed File System

Dimitri Pertin[1,2], Sylvain David[2], Pierre Évenou[2], Benoît Parrein[1] and Nicolas Normand[1]
[1]*LUNAM Université, Université de Nantes, IRCCyN UMR CNRS 6597, Nantes, France*
[2]*Fizians SAS, Nantes, France*
{*dimitri.pertin,benoit.parrein,nicolas.normand*}*@univ-nantes.fr,* {*sylvain.david, pierre.evenou*}*@rozofs.com*

Keywords: Distributed File System, RozoFS, Erasure Coding, Mojette Transform, IOzone, Video Editing.

Abstract: Distributed storage systems take advantage of the network, storage and computational resources to provide a scalable infrastructure. But in such large system, failures are frequent and expected. Data replication is the common technique to provide fault-tolerance but suffers from its important storage consumption. Erasure coding is an alternative that offers the same data protection but reduces significantly the storage consumption. As it entails additional workload, current storage providers limit its use for longterm storage. We present the Mojette Transform (MT), an erasure code whose computations rely on fast XOR operations. The MT is part of RozoFS, a distributed file system that provides a global namespace relying on a cluster of storage nodes. This work is part of our ongoing effort to prove that erasure coding is not necessarily a bottleneck for intense I/O applications. In order to validate our approach, we consider a case study involving a storage cluster of RozoFS that supports video editing as an I/O intensive application.

## 1 INTRODUCTION

Distributed storage systems have been used to provide data reliability. Failures in such large systems are considered as the norm, and can come either from hardware or software considerations. They can result in dramatic data loss and/or the crash of the service. The traditional way to deal with data protection is to replicate the data. Once $n$ copies of data are distributed across multiple network nodes, the system is able to face $n-1$ failures. If a node has a breakdown, its data is not accessible anymore, but other copies are still available on other nodes. On the other hand, this technique is expensive, particularly for huge amounts of data. Replication is the default data protection feature included in distributed storage systems.

Erasure coding is an alternative that provides the same data protection but reduces significantly the storage consumption. Optimal codes, called Maximal Distance Separable (MDS), aim at encoding $k$ data blocks into $n$ parity blocks, with $n \geq k$. The encoded blocks hold enough redundancy to recover the former data from any subset of $k$ parity blocks during the decoding process. Compared to replication whose storage overhead is $n$, erasure coding's overhead is defined by $n/k$. However, encoding and decoding require further computations compared to the simple replication technique. Efforts are done today to design efficient codes. The most famous ones are the Reed-Solomon (RS) codes. They rely on Galois field operations. In this arithmetic, addition is fast as it corresponds to exclusive-OR (XOR), but multiplication has many implementations which are much more computational expensive. The storage systems that provide erasure coding by RS suffer from the slowing down of its computations. Many implementations of RS codes exist. Examples of implementation libraries are OpenFEC[1] and Jerasure (Plank, 2007).

We propose the use of the Mojette Transform (MT) (Normand, Kingston, and Évenou, 2006) as an erasure code for distributed storage. The MT relies only on additions and its implementation uses the fast XOR operations. The MT is part of RozoFS[2], an open-source software providing a distributed file system. In RozoFS, when a client wants to store a file, the data is cut into small chunks of 8 KB. Chunks are composed of $k$ blocks of fixed size which depends on the desired protection. These blocks are encoded into $n$ parity blocks, which are discrete projections in the Mojette formalism. These projections are then distributed to storage nodes and the system is able to face

[1]http://openfec.org/
[2]code available at http://www.rozofs.org

$n-k$ failures. Decoding is done client-side after retrieving $k$ blocks out of the $n$ from storage nodes.

The current storage systems provide erasure coding to benefit from the storage capacity saving for longterm storage. When intensive data I/Os are needed, replication is put forward because of the additional workload of erasure codes. Our work differs from this position by exploring the use of an erasure code for intensive applications. We experiment the concept by applying RozoFS to video editing.

The rest of the paper is structured as follows: Section 2 briefly describes some distributed storage systems and their data protection policy. In Section 3, we present the MT and its erasure code characteristics, while Section 4 describes how RozoFS use it to provide a distributed file system. The video editing experiment is presented in section 5 and demonstrates that erasure coding is not a bottleneck for I/O intensive applications. Section 6 concludes the paper with the possible future work.

# 2 RELATED WORK

Erasure Coding receives significant attention from both the business and the research communities. Scality, Caringo, Isilon or Cleversafe are such examples of companies providing private storage solutions based on erasure codes. Data protection for these systems are partially described in technical white papers.

Scality and Caringo provide replication and erasure coding as a way to protect data. Erasure coding is only used for the longterm storage of massive amounts of data. They both recommend replication for intensive applications. Isilon puts forward the use of erasure coding through a Reed Solomon implementation in OneFS. Replication is only used when erasure coding is not applicable (e.g. too few many nodes). Cleversafe provides exclusively data protection through erasure coding. It relies on the Luby's implementation (Blömer et al., 1995) of the Reed Solomon algorithm.

Besides these solutions, famous and stable free alternatives exist. One of the most famous free solution is Apache HDFS. This Distributed File System (DFS) aims at archiving and data-intensive computing (i.e. not really an I/O centric DFS). It relies on the MapReduce framework that divides and distributes tasks to storage nodes. Data protection is done by replication, and triplication is the default protection policy. Carnegie Mellon University has developed a version based on erasure codes, called HDFS-RAID (Fan et al., 2009). It is based on Reed-Solomon codes for generic parity computations. HDFS-RAID has been applied in practice (e.g. Facebook Warehouse). However, only a small part of the warehouse's data has been translated by erasure coding. Warm data (i.e. frequently accessed data) is still replicated. Once the data have not been used for a while, the raid-node daemon encodes it and erases the corresponding replicated blocks to make room. GlusterFS[3] and Ceph (Weil et al., 2006) are examples of popular I/O centric DFS. Replication is currently their standard for data protection, but erasure coding is a hot topic on the roadmap.

Erasure coding is already included in private solutions but it is not to be efficient enough for intensive applications. It is exclusively used today for longterm storage and systems benefit from its low data consumption compared to replication. Open-source storage solutions are still actively looking for high speed erasure code implementations as an alternative to replication.

Our contribution, RozoFS, is an open-source software jointly developed by Fizians SAS and the University of Nantes. It provides a POSIX DFS whose file operations are exclusively done by erasure coding. This code is based on the MT whose computations rely entirely on the very fast XOR operations. Thus, we expect RozoFS to be efficient enough for intensive applications.

# 3 MOJETTE TRANSFORM

The Mojette Transform (MT) is a mathematical tool based on discrete geometry. Long developed at the University of Nantes, its first definition as an erasure code remains in (Normand, J. Guédon, et al., 1996). A first design for distributed storage relying on the MT was proposed in (J. P. Guédon, Parrein, and Normand, 2001).

## 3.1 Algorithm

Fundamentally, an erasure code should compute efficiently a desired amount of redundant information from a set of data. This operation is done by a linear operation that should be efficiently inverted. The inverse operation reconstructs the former data from a subset of the computed information.

The Mojette encoding is done as follows. A set of data fills a discrete grid whose elements are identified by $(i, j)$, where $i$ and $j$ are integers. The linear transform consists in summing these elements following different discrete directions. Each direction is defined

[3] http://www.gluster.org/

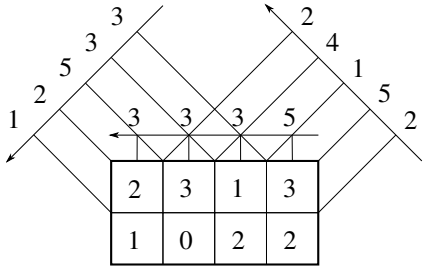



Figure 1: The Mojette Transform, applied to two lines of 4 integers, computes here the following set of 3 projections: $\{(p_i,q_i)=(-1,1),(0,1),(1,1)\}$.

by a couple of integers $(p,q)$. The elements aligned in a direction are summed to form a bin. The set of bins defined by the same direction is called a projection. Hence, it is possible to use the Mojette Transform to compute a desired number of projections from a set of lines of data. If $k$ and $n$ are respectively the number of lines of the grid, and the number of projections, and if $n > k$, the set of projections holds redundant information. Figure 1 shows the computation of 3 projections from a set of 2 lines of data.

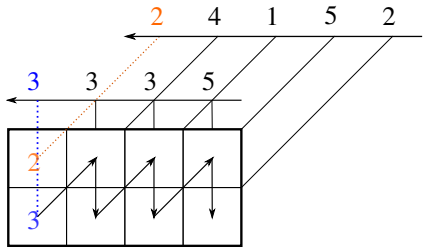



Figure 2: Inverse Mojette Transform as described in (Normand, Kingston, and Évenou, 2006).

The decoding is the inverse operation. It consists in filling the empty grid with the only projection data. Data reconstruction is possible if the Katz's criterion (Normand, Kingston, and Évenou, 2006) holds for the rectangular-shaped grid. This criterion was extended for any convex shape in (Normand, J. Guédon, et al., 1996). Reconstruction is straightforward for the elements in the corners of the grid. Indeed these elements match entirely a projection bin. (Normand, Kingston, and Évenou, 2006) showed that if a reconstruction path can be found to fill a side of the grid, then it can be applied several time to easily reconstruct the grid. Figure 2 represents such reconstruction considering the example of Figure 1. According to the Katz criterion if the following condition is sufficient to validate the reconstruction:

$$\sum_{i=0}^{r} q_i \geq k \qquad (1)$$

where $r$ is the number of projections that remains, $q_i$ depends on this projection set, and $k$ is the number of lines of the grid. For instance, in Figure 2, $\sum_{i=1}^{2} q_i = 2$ which equals $k$, so reconstruction is possible. By modifying the number of lines $k$ and the number of projections $n$, it is possible to set the desired fault-tolerance threshold.

### 3.2 Implementation

In storage application we cut the input stream into $k$ fixed size blocks. Each line of the grid in the Mojette representation is such a data block. The shape of the grid must be rectangular then. The set of projections is set so that $q_i = 1$. If we suppose that $l$, the size of blocks (i.e. grid columns) is higher than $k$, the number of blocks (i.e. grid lines), then the Katz's criterion guarantees that reconstruction is possible from any subset of $k$ projections.

The strength of the Mojette Transform is that encoding and decoding rely entirely on additions. Compared to classical codes like Reed Solomon, there is no need to compute expensive multiplications or divisions in Galois fields. The elements of the grid should fit computer words to improve computation performance. Then, addition is done by efficient exclusive-OR (XOR). Most recent Intel-based processors can perform very fast computations with elements of 128 bits as it fits the dedicated SSE (Streaming SIMD Extensions) registers.

In traditional solutions like Reed Solomon codes, the size of the parity blocks is fixed. The Mojette Transform relaxes this constraint. The size $B$ of each projection varies slightly with the angle of projection, and is given by the following formula:

$$B(k,l,p,q) = (k-1) \mid q \mid +(l-1) \mid p \mid +1 \qquad (2)$$

where $k$ is the number of the grid lines and $l$ the size of blocks. The MT is then considered as $(1+\varepsilon)$ MDS (Parrein, Normand, and J. P. Guédon, 2001), where $\varepsilon$ is the quotient of the number of bins required for decoding by the number of element in the grid. The set of projections is taken such as $q = 1$ and $p$ varies around 0 in order to minimize the value of $\varepsilon$.

## 4 ROZOFS

RozoFS is an open source software solution providing a scalable distributed file system. Once mounted, the file system yields a unique name space (i.e. an hierarchy of directories and files), that relies on clusters of commodity storage nodes. RozoFS manages scalability by adding storage nodes to expand the global resources of the system. For data reliability, it relies on the Mojette Transform as an erasure code.

## 4.1 Information Dispersal

We consider a network of commodity computers (or nodes) communicating by one or several links. The more the links, the more the paths for packets, and the more reliable and high-performance the system is. Reliability comes from the capacity to communicate even if a link is down. Again, failure probabilities are non negligible and should be considered as the norm. Multiple links induce high-performance because packets can be sent in parallel.

RozoFS considers flows of information of fixed size. A small data chunk of 8 KB fills a Mojette grid as seen in Figure 1. The protection settings, called layouts, define the value of $n$ and $k$ (respectively the number of projections and the number of lines in the Mojette grid).

Table 1: The protection settings provided by RozoFS.

| Layout | $k$ | $n$ | storage nodes | fault-tolerance |
|---|---|---|---|---|
| 0 | 2 | 3 | 4 | 1 |
| 1 | 4 | 6 | 8 | 2 |
| 2 | 8 | 12 | 16 | 4 |

Currently, three layouts are designed in RozoFS. The table 1 displays the relative information for these configurations. Each layout corresponds to a storage consumption of $n/k = 1.5$ times the size of the input. For instance, we consider the write operation of 1 GB of data in an exported volume of RozoFS, set with the layout 0. It results that 3 projection files of around 500 MB (i.e. plus $\varepsilon$) are distributed across physical disks.

## 4.2 Stackable Architecture

RozoFS relies on three components:

- *exportd*: the daemon that manages metadata;
- *storaged*: the daemon in charge of the projections;
- *rozofsmount*: used by the clients to mount RozoFS locally.

For the sake of modularity, these components are stackable. For instance, it is either possible to collocate them on a single node, storing projections on different disks, to obtain protection over disk failure (similar to some RAID configurations). In a large network topology, dedicated nodes are much more appreciated for the sake of scalability.

**RozoFS Client.** Each client can use RozoFS as a standard POSIX file system thanks to the *rozofsmount* process. It relies on FUSE[4] to operates in the user-mode. Clients handle two kinds of operations: (i) the metadata operations (*lookup*, *getattr*, etc), interfacing the export server; (ii) the file I/O operations (read/write the projections), interfacing the storage nodes through the *storcli* subprocesses. The encoding and decoding workload are managed by the clients. The network design should take care of reliability. Services and links must be replicated to provide availability facing failures.

[4]http://www.fuse.sourceforge.net

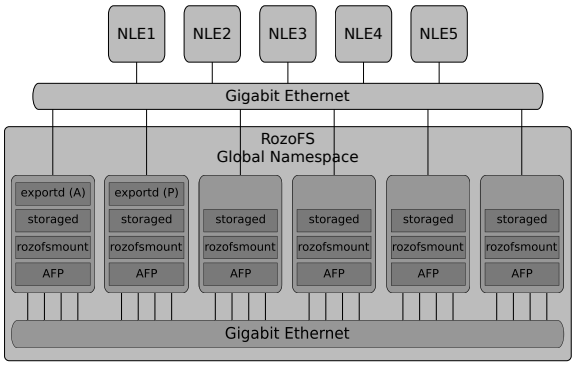



Figure 3: The RozoFS cluster is composed of 6 nodes that hold different services. They provide the DFS to 5 clients running a Non-Linear Editing (NLE) application.

**Metadata Server.** This node manages the *exportd* process that services the metadata operations. Its configuration file defines the exported file systems. It keeps statistics of storage nodes to provide a data distribution policy based on the storage capacity. The metada server supplies clients with two lists: (i) the list of storage nodes servicing the mounted volume; (ii) the list of storage nodes associate with a regular file (for read/write operations). This service should be replicated to guarantee high availability.

**Storage Nodes.** These entities hold the *storaged* daemon. Two services are provided by storage nodes: (i) the management services to exportd; (ii) the projection I/O services to the clients, called the *storio* processes. Each storio process can listen to a specific network interface. A storcli groups all storio of a storage node in a load-balancing group to improve availability.

# 5 EVALUATION

In this section, we explore the capacity of RozoFS to scale as the global workload increases.

**Experiment Setup.** For our evaluation, we employ a RozoFS cluster of 6 similar servers. Each machine contains an Intel Xeon CPU @ 2.40 GHz, 12 GB of RAM, and 10 disks (7200 rpm SAS) with 2 TB each. These drives are managed by a RAID-5

controller, providing better I/O performances and local reliability. Each node holds $8 \times 1$ Gbit Ethernet ports. The exported volume is set to layout 0 providing fault-tolerance against a single failure. The services are distributed as follows:

- the 6 nodes serve as storage servers;
- one among them serves as active metadata server;
- another is a passive/replicated metadata server;
- the 6 nodes mount RozoFS and re-export the file system through the AFP protocol for client access;

Figure 3 displays the platform used here. For high speed communications and reliability, 4 Ethernet ports are reserved for storio processes. RozoFS manages itself the packet scheduling for load-balancing and availability. Because the metadata server is a potential point of failure, it is necessary to set a high-availability strategy. Here, we use DRBD[5] to synchronise the active metadata server with the passive one. Pacemaker[6] is used as a cluster management software that manages the failover mechanisms in case of failure.

**The Study Case.** We use the previous RozoFS cluster as a storage solution for video editing. Non-Linear Editing (NLE) applications entail intensive workloads for computers. Multiple source editing is particularly file I/O intensive since multiple multimedia contents are accessed simultaneously. For this experimentation, we use the famous NLE software Apple Final Cut Pro[7]. Remote clients, running on Apple Mac OS, attach the RozoFS file system to their own machine using the AFP protocol. Once mounted, the file system provides an easy access to the source media stored on RozoFS. In our case, the system stores video encoded with the lossy compressed format Apple ProRes 422. It requires at least a rate of 200 Mbit/s (i.e. 25 MB/s) to avoid frame drops. We consider that editing involves 5 input video files simultaneously and outputs a single one. The software is both designed for sequential operations (e.g. read the video stream) and direct access to a part of the video.

## 5.1 IOzone Benchmark

IOzone[8] is a filesystem benchmark that can output file I/O performance according to a set of stressing tests. In particular, the software can perform sequential and random read/write tests, which are in accordance with our study case. We explore the capacity of RozoFS to adapt, facing a growing number of clients accessing the file system simultaneously.

[5]Distributed Replicated Block Device (www.drbd.org)
[6]http://clusterlabs.org/
[7]http://www.apple.com/final-cut-pro/
[8]http://www.iozone.org

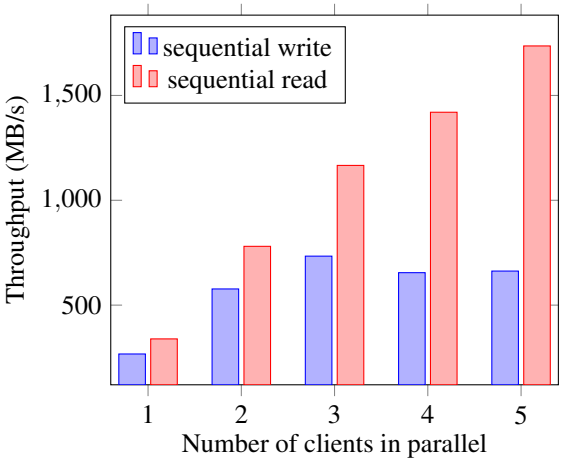



Figure 4: Accumulated throughput (in MB/s) recorded by the threads, depending on the number of clients working in parallel, for sequential access operations.

IOzone takes the following parameters. The file size is set to 1 GB which is larger than the CPU cache, and smaller than the RAM capacity. To fit the multi-source editing, each client involved in the test manages 5 threads. A thread reads or writes the 1 GB file according to the desired access strategy. For instance, in a writing test, each node induces the writing of 5 GB in RozoFS (i.e. which represents 7.5 GB in the physical disks).

We measure the accumulated throughput recorded by the threads as the number of clients grows. Each thread should access data with a rate of at least 25 MB/s to validate the study case. Figure 4 and 5 respectively display the average results from 5 test iterations for sequential and random access.

**Benchmarking Sequential I/Os.** The write operations are asynchronous and multiple write requests are done simultaneously. Figure 4 shows that as the workload grows, the performance for sequential write scales up to 3 nodes. When more clients are added, the nodes are overwhelmed by the requests and cache misses slow down the performances as data need to be fetch on the disks. The sequential read operations significantly benefit from fast access as data is pre-fetched in the cache. The performance scales up as the number of clients increases. For instance, when 5 clients are involved, the 25 threads record an accumulated throughput of 1800 MB/s. Each client receives $1800/5 = 360$ MB/s and each thread receives $360/5 = 72$ MB/s. In any case in this benchmark, we validate the study case as each thread receive more than 25 MB/s. Mechanical disks are particularly sen-

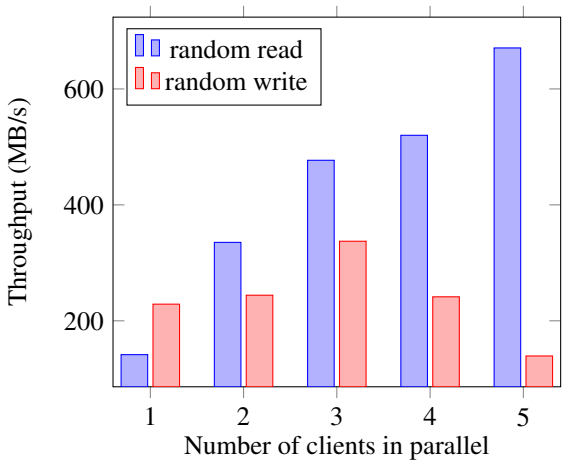



Figure 5: Accumulated throughput (in MB/s) recorded by the threads, depending on the number of clients working in parallel, for random access operations.

sitive to the type of access. For sequential access, the disk's arm move the read-and-write head to reach the correct track on the physical disk. Once there, the following accesses are straightforward.

**Benchmarking Random I/Os.** Random operations are slowed down by intermediate seeks between operations, increasing the latency of disk's head. Figure 5 shows that the throughput suffers from these random accesses. During random read operations, the performance scales as it benefits from small cache effects and each thread receive at least 25 MB/s. However, the random write test is clearly the worst case and shows the limit of hardware as the performances collapse. For 3 clients, each thread receives $350/3/5 = 23$ MB/s. For more clients, performances get worse. We should note that 5 writing threads per client does not fit our study case. For video editing, each editors outputs a single file. It would correspond to a single write thread in our case.

# 6 CONCLUSIONS

In this work, we have presented the Mojette Transform (MT) as an efficient alternative to classical erasure codes since it relies on fast computations, and a great candidate to handle intensive applications. We have set the MT as the I/O centric process in RozoFS with good expectations for practice efficiency. Finally, we have designed an evaluation based on a RozoFS cluster. The platform is able to handle multiple parallel access from intensive I/O applications (i.e. multiple source video editing). The evaluation has revealed that erasure coding is not a bottleneck for intensive applications and should not be limited to longterm storage.

More specific measures should be done to reveal the real cost of the MT in the computational time. Our evaluation could be extended to more intensive applications like virtualization, which access data over block devices. There is clearly a need for comparisons with other existing solutions. The MT must be compared to erasure code libraries such as OpenFEC and Jerasure. Further erasure code aspects beyond computational considerations should be explored, such as the node repair problem.

# ACKNOWLEDGEMENTS

This work has been supported by the French Agence Nationale de la Recherche (ANR) through the project FEC4Cloud (ANR-12-EMMA-0031-01).

# References

Blömer, J. et al. (1995). *An XOR-based erasure-resilient coding scheme*. Tech. rep. TR-95-048. International Computer Science Institute.

Fan, B. et al. (2009). "DiskReduce: RAID for data-intensive scalable computing". In: *Proc. PDSW 2009*. Portland, Oregon: ACM. DOI: 10.1145/1713072.1713075.

Guédon, J. P., B. Parrein, and N. Normand (2001). "Internet Distributed Image Information System". In: *Integr. Comput.-Aided Eng.* 8.3, pp. 205–214. ISSN: 1069-2509.

Normand, N., J. Guédon, et al. (1996). "Controlled redundancy for image coding and high-speed transmission". In: *Proc. VCIP 1996*. Vol. 2727. Orlando, FL. DOI: 10.1117/12.233180.

Normand, N., A. Kingston, and P. Évenou (2006). "A geometry driven reconstruction algorithm for the Mojette transform". In: *DGCI*. Vol. 4245. LNCS. Springer Berlin Heidelberg. DOI: 10.1007/11907350_11.

Parrein, B., N. Normand, and J. P. Guédon (2001). "Multiple description coding using exact discrete Radon transform". In: *Proceedings of the Data Compression Conference*. DCC '01. Washington, DC, USA: IEEE Computer Society. DOI: 10.1.1.19.5708.

Plank, J. S. (2007). *Jerasure: A library in C/C++ facilitating erasure coding for storage applications*. Tech. rep. CS-07-603. University of Tennessee.

# Performance evaluation of the Mojette erasure code for fault-tolerant distributed hot data storage

Dimitri Pertin
*Université de Nantes*
*IRCCyN UMR 6597*
*Rozo Systems*

Didier Féron
*Rozo Systems*

Alexandre Van Kempen
*Université de Nantes*
*IRCCyN UMR 6597*

Benoît Parrein
*Université de Nantes*
*IRCCyN UMR 6597*

## Abstract

Packet erasure codes are today a real alternative to replication in fault tolerant distributed storage systems. In this paper, we propose the Mojette erasure code based on the Mojette transform, a formerly tomographic tool. The performance of coding and decoding are compared to the Reed-Solomon code implementations of the two open-source reference libraries namely ISA-L and Jerasure 2.0. Results clearly show better performances for our discrete geometric code compared to the classical algebraic approaches. A gain factor up to 2 is measured in comparison with the ISA-L Intel . Those very good performances allow to deploy Mojette erasure code for hot data distributed storage and I/O intensive applications.

## 1 Introduction

Storage systems rely on redundancy to face ineluctable data unavailability and component failures. For its simplicity, data replication is the de facto standard to provide redundancy. For instance, three-way replication is the storage policy adopted by major file systems such as Hadoop Distributed File System [11] and Google File System [1]. While being straightforward to implement, plain replication typically incurs high storage overheads. It has now been acknowledged that erasure codes can significantly reduce the amount of redundancy compared to replication while offering the same data protection [13].

However, these storage savings come at a price in terms of additional complexity, as data must be encoded during write operations, and decoded during read operations. Very efficient coding operations are thus keys to maintain transparent operations for I/O intensive applications. Since data replication has higher storage costs but performs faster than erasure codes, storage systems tend to differentiate between *cold* data (i.e. not frequently accessed, such as in long term storage) and *hot* data, typically data that is frequently accessed. In practice, plain replication is used for I/O intensive applications due to fast data accesses while erasure codes are limited to long-term storage because of their extra complexity.

*Reed-Solomon* (RS) are the most popular codes as they provide deterministic general-purpose codes without limit on the parity level. They are mostly implemented in their systematic form, meaning that the information data is a part of the encoded data. In addition, they are known to be Maximum Distance Separable (MDS) thus providing the optimal reliability for a given storage overhead. The former definition of RS codes are based on Vandermonde matrices and expensive Galois field operations. Implementing such codes in an efficient manner is therefore challenging. One of the best known implementation is provided by Jerasure [9], an open-source library that relies on Cauchy generating bit-matrices to only perform XOR operations, thus avoiding the costly multiplications. Recently, Intel released ISA-L, a performance-oriented open-source library [4] that implements RS codes leveraging SIMD instructions. To the best of our knowledge, these two are the most efficient implementations publicly available.

In this paper, we propose to use the *Mojette* transform [2], a formerly tomographic tool, to implement a high-performance erasure code. The Mojette transform is a discrete and exact version of the Radon transform and relies on discrete geometry, contrary to the classic algebraic code definition. By nature, the Mojette transform provides a non-systematic erasure code. The geometric approach coupled with an optimized implementation help to perform very fast encoding and decoding operations, handling I/O intensive applications such as virtualization or databases, that access small blocks of data (4 KB or 8 KB) in a random pattern [8]. Those block sizes fit the general-purpose file systems requirements such as *ext4* or *Btrfs*.

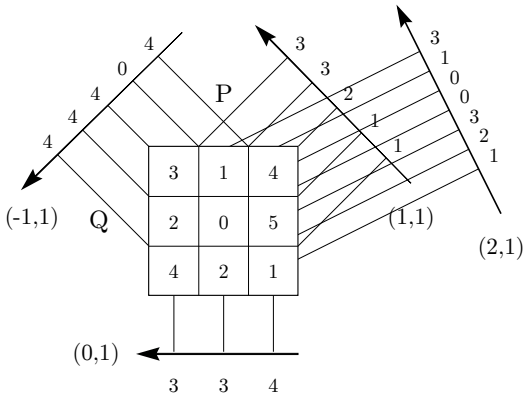



Figure 1: Mojette transform of a $3 \times 3$ image for directions $(p,q)$ in the set $\{(-1,1),(0,1),(1,1),(2,1)\}$. Addition is done here modulo 6.

## 2 The Mojette Erasure Code

This section presents how the Mojette transform is used to encode data, the uniqueness conditions of the reconstruction solution and its inverse algorithm enabling the decoding. Finally, the end of this section details how the Mojette transform is used as an erasure code in practical systems.

### 2.1 Forward Mojette Transform

The forward Mojette transform is a linear operation that computes a set of 1D projections at different angles, from a discrete image $f : (k,l) \mapsto \mathbb{N}$, composed of $P \times Q$ *pixels*. A projection direction is defined by a couple of co-prime integers $(p,q)$. Projections are vectors of variable sizes whose elements are called *bins*. A bin in the Mojette transform of $f$ is characterised by its position $b$ in the projection which corresponds to a discrete line of equation $b = -kq + lp$. Its value is the sum of the centered pixels along the line:

$$(M_{(p,q)}f)(b) = \sum_{k=0}^{P-1} \sum_{l=0}^{Q-1} f(k,l)[b = -kq + lp], \qquad (1)$$

where, $[\cdot]$ is the Iverson bracket ($[P] = 1$ whenever $P$ is true, 0 otherwise). The number of bins $B$ of a projection depends on the projection direction $(p,q)$ and the lattice size $P \times Q$:

$$B(p,q,P,Q) = |p|\,(Q-1) + |q|\,(P-1) + 1. \qquad (2)$$

Figure 1 gives an example of the forward Mojette transform for a $3 \times 3$ integer image. The process transforms the 2D image into a set of $I = 4$ projections along the directions of the following set: $\{(-1,1),(0,1),(1,1),(2,1)\}$. For the sake of this example, addition is arbitrarily done modulo-6 (but any addition works). The complexity $\mathcal{O}(PQI)$ is linear with the number of projections and the number of grid elements. Note that some border bins are the exact copy of some pixels. This remark will help to understand how starts the inverse transform algorithm.

### 2.2 Inverse Mojette Transform

In this section, we first expose the reconstruction criterion on the projection set which yields to a unique reconstructed image. Then, we detail how is implemented the reconstruction algorithm.

**Reconstruction Criterion** Katz has shown that for a $P \times Q$ lattice, the reconstruction is possible given a projection set $S_I$ if one of the following criterion is verified [5]:

$$P \leq \sum_{i=0}^{I-1} |p_i| \text{ or } Q \leq \sum_{i=0}^{I-1} |q_i|, \qquad (3)$$

where $I$ is the number of projections involved in the reconstruction process.

In the example of the Figure 1, we see that each subset of 3 projections $\{(p_0,q_0),\ldots,(p_2,q_2)\}$ is such that $\sum_{i=0}^{2} |q_i| = 3$. Thus, the 4 projections in Figure 1 depicts a redundant representation of the image, where any 3 projections among these 4 can be used for reconstruction.

**Inverse Mojette Algorithm** The reconstruction algorithm aims at finding a reconstructible bin and to write its value in the image by back-projection. Bins are reconstructible when they result from a unique pixel of the image. Once a bin is reconstructed, its contribution is subtracted from all the projections involved in the reconstruction, thus paving the way to reconstruct further bins. As the forward algorithm, the Mojette inverse is linear with the number of projections $I$ and the number of elements $P \times Q$ in the grid.

Observing that the reconstruction propagates from the image corners to its center, Normand et al. [6] showed that given an image domain and a projection set, a dependency graph between the image pixels can be found. Within this graph, considering that a single projection is dedicated to the reconstruction of a single line of the image, a reconstruction path can be pre-determined. We refer the interested reader to [6] due to lack of space.

### 2.3 Properties of the Mojette Erasure Code

The Mojette erasure code extends the application of the Mojette transform, originally designed for images, to any

type of data. As the Mojette transform creates a redundant representation, it appears to be an appealing candidate to provide failure tolerance in storage systems. In classic coding theory words, we consider the $k$ lines of the Mojette array as the input data packets, and we compute $n$ projections as the set of encoded packets. The Mojette erasure code is therefore non-systematic here. Note that a systematic version of the Mojette is currently under development. Since the size of projection varies with the parameters $(p,q)$ we consider for each projection that $q_i = 1$ to limit the bin overhead (as proposed in [7]). Then the Katz criterion proves that if we get any $k$ out of the $n$ projections, it is possible to exactly reconstruct the array. This way, the storage system is able to face the unavailability of up to any $n-k$ storage nodes.

In practice, we can observe that some bins are never used during reconstruction whatever the projection set used for the process [12]. Removing these bins from the encoding process, particularly when $p$ increases, significantly limits the projection size variation and therefore yields to a negligible storage overhead relative to the MDS case.

# 3 Erasure Code Micro-benchmark

In this section, we evaluate the performance of our new erasure code compared to the Jerasure and ISA-L libraries. Firstly, we describe our Mojette implementation and then present our two competitors. Secondly, we detail the experiment setup to finally depict the results and analysis in the last section.

**Mojette** We implemented a non-systematic version of the Mojette erasure code in C. In practice, pixel size should fit a computer word to improve performance based on XOR operations. Since x86 architectures provide Streaming SIMD Extensions (SSE) instruction set, pixel and bin sizes are set to 128 bits to benefit from high-performance XOR computations. The Mojette encoding requires at most $k-1$ XORs per computed bin (and zero XOR for bins at projection edges). Similarly for decoding, at most $k-1$ XORs are required per reconstructible pixel. The progressive reconstruction from left to right of connected pixels (as proposed in [6]), coupled with a drastic reduction of updates, guarantees spatial memory locality, thus high-performance computation.

**Jerasure** The first competitor is the systematic Vandermonde implementation of RS codes from the open-source Jerasure 2.0 library [9]. We choose their Vandermonde implementation since it performs better than their Cauchy-based implementation for such small packets size. The Galois field size is set to $w = 8$ to fit our erasure code configuration $(n,k)$.

**Intel ISA-L** The second competitor is the RS implementation provided in Intel ISA-L open-source library [4]. It is one of the fastest systematic erasure code implementation since it makes intensive usage of the x86 architecture features such as *xmm* registers and SSE instructions.

## 3.1 Experiment Setup

We conducted all experiments on small data blocks of *BlockSize* equals to 4 KB and 8 KB (that fit block-based file-system like *ext4*). For encoding, we consider a single data block filled with random data. For decoding, we record the performance as we increase the number of erasures up to the failure tolerance. With systematic codes (implementations of ISA-L and Jerasure), erasures only concern data packets (no decoding otherwise).

Two erasure code configurations are considered for the benchmark: $(n,k)$ equals $(6,4)$ and $(12,8)$, preventing from 2 and 4 failures respectively. All the computations are performed in memory, with no disk I/O operation. Furthermore, we do not take into account pre-computations such as the matrix inversion or deterministic reconstruction path respectively for the RS and Mojette implementations. Since we measure optimized encoding and decoding functions that are mostly computation-bounded, with the data entirely located in L1 and L2 cache, we use the *RDTSC* instruction that returns the *time stamp counter* (TSC) which is incremented on every CPU clock cycle [3]. For all tested implementations, the standard deviation is too negligible to be represented (less than 1%).

All the experiments are done on a single processor running Linux 3.2 and Debian Wheezy over an x86-64 architecture. It embeds a 1.80 GHz Intel Xeon processor, with 16 GB of RAM and cache sizes of 32, 256 and 10240 KB for respectively L1, L2 and L3 cache levels.

## 3.2 Results

We now present the results of encoding and decoding throughput for various *BlockSize* and code parameters. For the sake of comparison, we plot the optimal performance recorded by the **memcpy()**. More precisely, the optimal encoding is given by the **memcpy()** of $n$ packets of $\frac{BlockSize}{k}$ bytes while the optimal decoding is the **memcpy()** of only $k$ packets among the $n$ encoded. Once again, note that the Mojette is implemented as a non-systematic code, thus increasing the overall computation compared to the two other systematic codes.

**Encoding** Figure 2 shows the encoding performance recorded for 4 KB (top) and 8 KB (bottom) data blocks for the $(6,4)$ and $(12,8)$ codes. The first observation is that the Mojette erasure code outperforms the two other

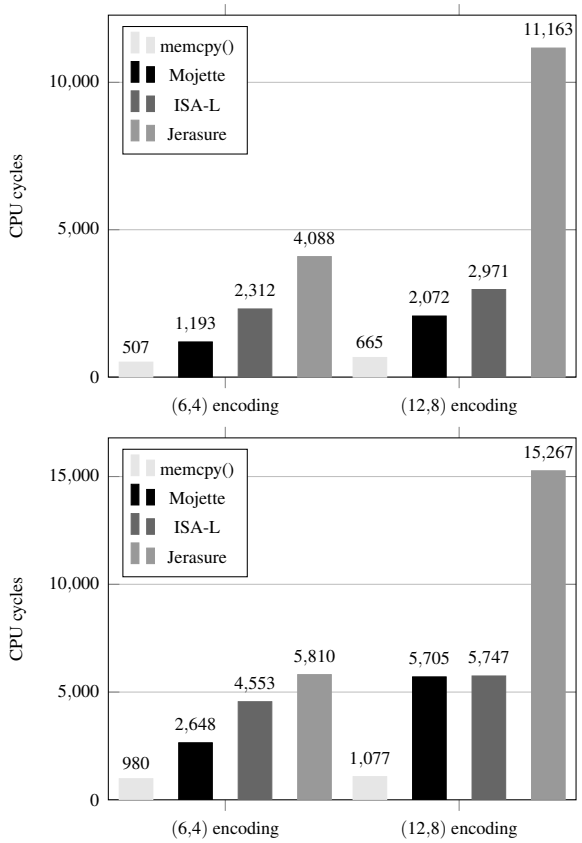



Figure 2: Encoding performance for an input data block of 4 KB (**top**) and 8 KB (**bottom**) depending on the code parameters $(n = 6, k = 4)$ or $(n = 12, k = 8)$.

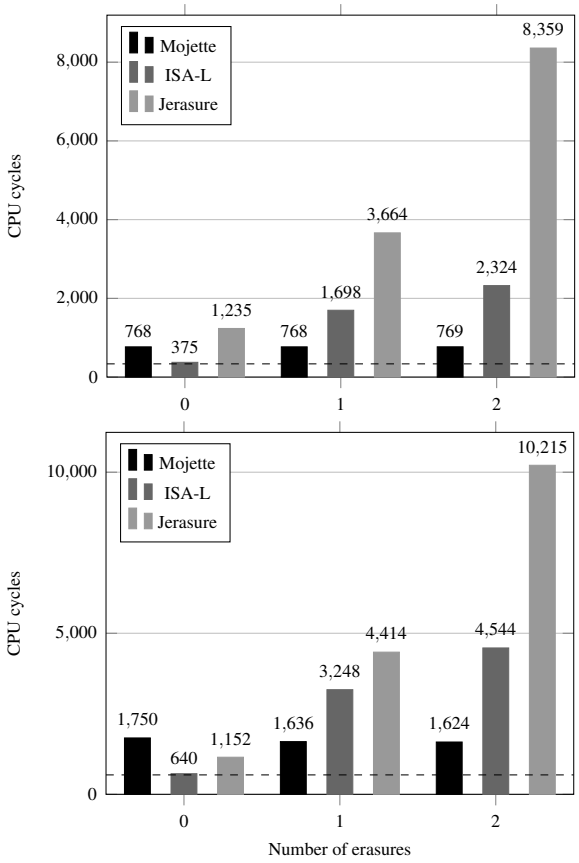



Figure 3: Decoding performance of a $(n = 6, k = 4)$ code for an input data block of 4 KB (**top**) and 8 KB (**bottom**) depending on the number of failures. The dashed line depicts the optimal value of the **memcpy()** respectively measured at 336 (4 KB) and 603 (8 KB).

implementations in every tested settings. For example, to encode a 4 KB block with a $(6,4)$ code, the Mojette implementation divides the number of CPU cycles by a factor of 1.94 and 3.42 when respectively compared to ISA-L and Jerasure. While the Mojette implementation still provides the closest performance from the optimal value of the **memcpy()**, the improvements are mitigated for the code $(12,8)$, and especially versus ISA-L for a block of 8 KB. This is mainly due to our non-systematic design. Indeed, to encode a 8 KB block with a $(12,8)$ code, the encoder splits 8 KB into 8 packets of one kilobyte and produces 4 encoded packets for systematic codes, while non-systematic codes have to compute 12 encoded packets thus performing 3 times more computations. Finally, we notice that, as expected, the CPU cycles number linearly increases with the *BlockSize*, as well as with the number of blocks to be encoded thus experimentally confirming the linear complexity of the Mojette transform.

**Decoding** We respectively plot in Figures 3 and 4 the number of CPU cycles required to decode the data for the same codes as before, depending on the number of failures. We still emphasize here the differences between systematic and non-systematic implementations. Since RS codes are systematic, **when no failure occurs**, they should achieve optimal performance (equivalent to a **memcpy()**) as the decoding process boils down to the copy of $k$ data blocks in memory. For example, we see that ISA-L delivers the optimal performance for every 0-erasure settings. Note that our ongoing implementation of the Mojette in systematic-form would also provide the same results.

We now focus on the results in the presence of failures, when decoding operations are therefore involved (i.e. we do not just retrieve the data packets in memory). Results for the code $(n = 6, k = 4)$ on a 4 KB block, depicted on top of the Figure 3 show that the number of CPU cycles is divided by a factor of 2.2 and 4.8 when respectively compared to ISA-L and Jerasure for a single failure. These factors are even higher when two erasures occurred. In fact, due to its non-systematic form, the

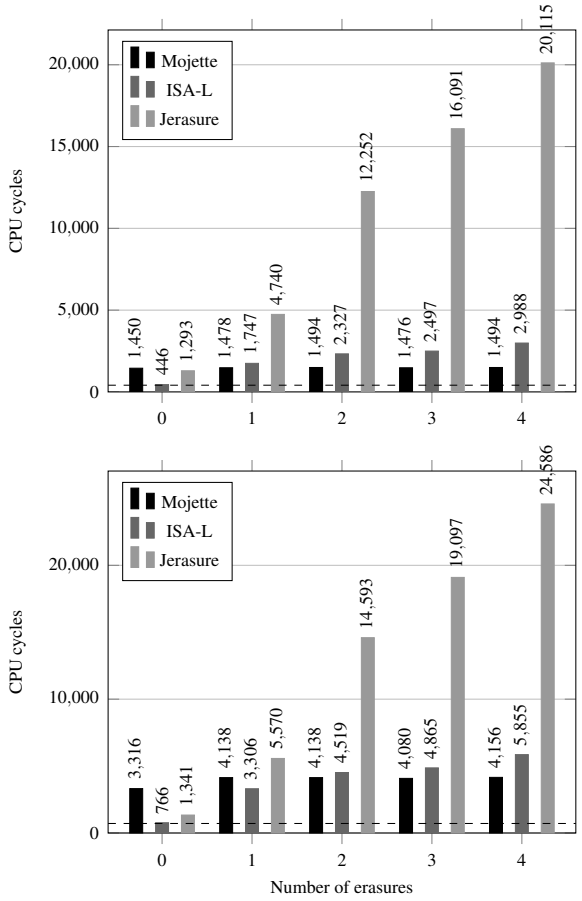



Figure 4: Decoding performance of a $(n = 12, k = 8)$ code for an input data block of 4 KB (**top**) and 8 KB (**bottom**) depending on the number of failures. The dashed line depicts the optimal value of the **memcpy()** respectively measured at 411 (4 KB) and 711 (8 KB).

set of projections used has no influence on the decoding performances of the Mojette. On the contrary, the performances of Jerasure and ISA-L progressively decrease with the number of erasures. Although this performance gap is reduced for the code $(n = 12, k = 8)$, especially versus ISA-L, results presented in Figure 4 still confirm the above observations.

## 4 Conclusion

Erasure codes are well known to incur a high computational penalty due to their inherent coding operations, thus preventing them from being deployed in I/O intensive applications. In this paper, we advocated that the *Mojette transform* is a particularly suitable tool to design high-performance erasure code. We implemented and evaluated our new erasure code compared to the best-known implementations, namely ISA-L and Jerasure. Results show that this paradigm shift towards a geometric approach enables the *Mojette*-based implementation to significantly improve the throughput of coding and decoding operations. As non-systematic, the proposed code can still bring better throughputs in a foreseeable future. A Mojette erasure code implementation is currently deployed in an open-source project RozoFS [10]. We believe that this new code paves the way to the use of erasure codes in I/O intensive applications.

## 5 Acknowledgements

This material is based upon work supported by the Agence Nationale de la Recherche (ANR) through the project FEC4Cloud (ANR-12-EMMA-0031-01).

## References

[1] GHEMAWAT, S., GOBIOFF, H., AND LEUNG, S.-T. The Google file system. In *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles* (New York, NY, USA, 2003), SOSP '03, ACM, pp. 29–43.

[2] GUÉDON, J. P., AND NORMAND, N. The Mojette transform: The first ten years. In *Discrete Geometry for Computer Imagery*, E. Andres, G. Damiand, and P. Lienhardt, Eds., vol. 3429 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2005, pp. 79–91.

[3] INTEL. Using the RDTSC instruction for performance monitoring. Tech. rep., Intel Corporation, 1997.

[4] ISA-L. https://01.org/intel%C2%AE-storage-acceleration-library-open-source-version.

[5] KATZ, M. *Questions of uniqueness and resolution in reconstruction from projections*. Springer-Verlag Berlin ; New York, 1978.

[6] NORMAND, N., KINGSTON, A., AND ÉVENOU, P. A geometry driven reconstruction algorithm for the Mojette transform. In *DGCI*, vol. 4245 of *LNCS*. Springer Berlin Heidelberg, 2006, pp. 122–133.

[7] PARREIN, B., NORMAND, N., AND GUÉDON, J. P. Multiple description coding using exact discrete Radon transform. In *Proceedings of the Data Compression Conference* (Washington, DC, USA, 2001), DCC '01, IEEE Computer Society, pp. 508–.

[8] PERTIN, D., DAVID, S., ÉVENOU, P., PARREIN, B., AND NICOLAS, N. Distributed file system based on erasure coding for I/O-intensive application. *The 4th International Conference on Cloud Computing and Services Science, CLOSER 2014*, 13-16 (Apr. 2014).

[9] PLANK, J. S., AND GREENAN, K. M. Jerasure: A library in C facilitating erasure coding for storage applications–version 2.0. Tech. rep., Technical Report UT-EECS-14-721, University of Tennessee, 2014.

[10] ROZOFS. https://github.com/rozofs/rozofs.

[11] SHVACHKO, K., KUANG, H., RADIA, S., AND CHANSLER, R. The Hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on* (May 2010), pp. 1–10.

[12] VERBERT, P., RICORDEL, V., AND GUÉDON, J. P. Analysis of Mojette transform projections for an efficient coding. In *Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)* (Lisboa, Portugal, Apr 2004), pp. –.

[13] WEATHERSPOON, H., AND KUBIATOWICZ, J. Erasure coding vs. replication: A quantitative comparison. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems* (London, UK, 2002), IPTPS '01, Springer-Verlag, pp. 328–338.

# Table des figures


3.1 Transformation Mojette d'une image $f$ avec une série de projections discrètes $S_4 = \{(0,1),(1,1),(2,1),(-1,1)\}$ (extrait du Chapitre 3 du livre Mojette [32]). 27

3.2 Représentations graphiques de la série de Farey $F_4$ (extraites du Chapitre 1 du livre Mojette [32]). . . . . 31

3.3 Transformation Mojette de (a) l'image unitaire $f_1$ et de (b) l'image des index uniques $f_i$ (extrait de [32]). . . . . 33

3.4 Mise à jour des trois transformations $\mathcal{M}_{-1,1}f$, $\mathcal{M}_{-1,1}f_1$ et $\mathcal{M}_{-1,1}f_i$ après reconstruction du pixel $(0,0)$ situé en bas à droite de l'image $4 \times 4$ (extrait de [32]). . . 33

3.5 (a) Graphe de dépendance inter-pixelique pour une image de hauteur 4 ($Q = 4$). (b-e) Les 4 graphes possibles de reconstruction (les parcours sont représentés en gras) (extrait de [52]). . . . . 34

3.6 Construction du code binaire Mojette(3,5) non systématique : le buffer géométrique d'entrée possède 3 lignes et 5 projections sont calculées en sortie de codage. 37

3.7 Construction du code binaire Mojette (3,5) systématique : le buffer géométrique d'entrée possède 3 lignes et 2 projections sont calculées. Le codage est composé des 3 lignes du buffer initial et des 2 projections calculées. . . . . 39

3.8 Phase d'initialisation des projections pour un décodage binaire Mojette (3,5) systématique en fonction des pertes de lignes du buffer géométrique : les 2 lignes perdues sont exactement compensées par 2 projections. Ces 2 projections sont initialisées par un XOR bin à bin avec les 2 projections recalculées en fonction du motif de lignes reçues (ici la deuxième ligne). . . . . 40

3.9 Support hexagonal et projections correspondantes pour un code binaire Mojette non-systématique de paramètre $(3,5)$. Les 5 projections obtenues sont de taille constante (9 bins). . . . . 41

3.10 Schéma illustrant la protection inégale Mojette. Chaque région du buffer géométrique est reconstructible par un nombre graduel de projections. Dans ce schéma, la région 1 (resp. 2 et 3) est reconstructible par 3 projections (resp. 4 et 5). La région 1, plus facilement reconstructible, est mieux protégée que les régions 2 et 3 vis-à-vis des pertes ou effacement de projections (extrait de [57] Chapitre 5). . . 44

4.1 Illustration de la complexité des opérations de lecture et écriture pour l'approche par codes à effacement (a) comparée à l'approche par réplication (b) pour différents niveaux de pannes (extrait de [28]). . . . . 48

4.2 Débit d'encodage (en Mo/s) pour le code $(4, 6)$ et le code $(8, 12)$ (première campagne de tests). . . . . 50
4.3 Débit de décodage (en Mo/s) sur des blocs de 4Ko pour les codes $(4, 6)$ et $(8, 12)$ respectivement à gauche et à droite de la figure (première campagne de tests). . . 51
4.4 Nombre de cycles CPU pour le codage (figure de gauche) et décodage (figure de droite) pour la Mojette et pour la librairie ISA-L. L'instruction `memcopy` est donnée à titre de référence pour le codage et le décodage (deuxième campagne de tests). . . . . 52
4.5 Stockage distribué Mojette original de 2001 [34]. . . . . 54
4.6 Stockage distribué RozoFS (schéma simplifié). . . . . 54
4.7 Schéma général d'une plateforme stockage distribué RozoFS incluant les *daemons* `exportd` et `storaged` ainsi que le point de montage client `rozofsmount`. . . . . 55
4.8 Cluster de machines dédiées au montage vidéo en ligne. . . . . 56
4.9 Débit cumulé (en Mo/s) enregistré en fonction du nombre de clients pour la lecture (séquentielle et aléatoire). . . . . 58
4.10 Déploiement des réplicats (ici au nombre de 2) au sein du système de fichier CephFS (voir `http://ceph.com/docs/master/architecture/`). . . . . 59
4.11 Débit cumulé moyen (en Mo/s) pour CephFS et RozoFS pour la lecture et l'écriture séquentielle. . . . . 60
4.12 Nombre moyen d'entrées-sorties cumulées par seconde (IOPS) pour CephFS et RozoFS pour la lecture et l'écriture aléatoire. . . . . 60
4.13 Répartition de la charge et capacité totale du cluster pour le stockage de 10 Go de fichiers utiles pour CephFS et RozoFS (estimation pour RozoFS v2). . . . . 61
4.14 Exemple d'un traitement analytique de type *wordcount* (ou comptage de nombre d'occurrences d'un mot) au travers de l'algorithme MapReduce - Extrait de `http://blog.trifork.com/2009/08/04/introduction-to-hadoop/`. . . . . 61
4.15 Temps exécution (sec.) des traitements *sort*, *wordcount* et *CloudBurst* en mode MapReduce (extrait de [103]). . . . . 63
4.16 Interactions entre l'algorithme MapReduce et le système de fichier distribué HDFS (source : `http://www.datafactz.com/blog/2014/11/12/`). . . . . 64
4.17 Temps d'exécution moyen pour les deux traitements analytiques *wordcount* et *grep* effectués sur RozoFS et sur HDFS. . . . . 65

5.1 Schéma général du protocole P2PWeb incluant de manière hybride un serveur de contenu (à gauche) et une population hétérogène de clients et de pairs (à droite). 67
5.2 Plateforme de tests du projet P2PWeb composé de deux clusters de 23 et 50 nœuds. 69
5.3 Comparaison des temps de téléchargement entre une architecture client/serveur simple et une architecture hybride P2PWeb. . . . . 70
5.4 Répartition des connexions du pair de mesure pour 10 clients et 63 pairs. . . . . 71
5.5 Débits descendants et montants enregistrés pour le scénario avec 10 clients et 63 pairs. . . . . 71
5.6 Répartition des connexions du pair de mesure pour 25 clients et 48 pairs. . . . . 72

5.7 Débits descendants et montants enregistrés pour le scénario avec 25 clients et 48 pairs. . . . 72
5.8 Débits descendant et montant enregistrés dans le cas d'un *tracker* passif (figure de gauche) et d'un *tracker* actif (figure de droite). . . . 73
5.9 Débits descendant et montant enregistrés dans le cas de la fourniture d'un chunk aléatoire (figure de gauche) et d'un chunk rare (figure de droite). . . . 74
5.10 Schéma général du protocole P2PWeb incluant un code à effacement systématique. 75

6.1 Format d'un paquet de données OLSR (extrait de [99]). . . . 79
6.2 Format d'un paquet MP-OLSR (extrait de [99]). . . . 79
6.3 Description modulaire du simulateur Qualnet avec des exemples de protocoles mis en œuvre. . . . 81
6.4 Taux de paquets délivrés en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR, enregistrés pour les protocoles MP-OLSR et OLSR (100 simulations de 100 s). . . . 83
6.5 Taux de paquets délivrés en fonction de la mobilité pour le scénario à 81 nœuds et 10 sources de type CBR enregistrés pour les protocoles MP-OLSR et OLSR (100 simulations de 100 s). . . . 84
6.6 Délais de bout-en-bout en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR mesurés (en secondes) pour les protocoles MP-OLSR et OLSR (100 simulations de 100 s). . . . 84
6.7 Comparaison des taux de paquets délivrés pour les protocoles DSR, AODV, OLSR et MP-OLSR en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR. . . . 85
6.8 Comparaison des délais de bout-en-bout mesurés (en secondes) pour les protocoles DSR, AODV, OLSR et MP-OLSR en fonction de la mobilité pour le scénario à 81 nœuds et 4 sources de type CBR. . . . 85
6.9 Photos de la journée d'expérimentation SEREADMO (2 juin 2009). Du haut à droite en bas à gauche : Sylvain David (Université de Nantes), Matthieu Fournier (Université de Poitiers), Jiazi Yi (Université de Nantes), moi-même (Université de Nantes) expliquant en salle le déroulant de la journée, Xavier Lecourtier (société Keosys) et Sylvain David. . . . 86
6.10 Vue aérienne de l'expérimentation du protocole MP-OLSR autour du bâtiment IRESTE sur le campus de la Chantrerie à Polytech Nantes. Exemple de topologie à 9 nœuds situés à l'intérieur et à l'extérieur du bâtiment. . . . 87
6.11 Qualités des liens radios (entre stables et instables) lors de l'expérimentation du scénarios de la figure 6.10. . . . 88
6.12 Séquences vidéos utilisées pour l'expérimentation. Du coin haut-gauche au coin bas-droit, une image des séquences *Skatefar*, *Burzoom*, *Aspen* et *Shadowboxing*. . 89
6.13 Information spatiale et temporelle (SI/TI) des quatre séquences sélectionnées pour l'expérimentation. . . . 90
6.14 Débit applicatif (vidéo et protection) mesuré pour les 4 séquences vidéos sélectionnées et les 3 protocoles de routages OLSR, MP-OLSR et MP-OLSR UEP (Unequal Error Potection). . . . 91

6.15 Résultats en matière de qualité d'usage VQM (Video Quality Metric) pour les protocoles OLSR, MP-OLSR et MP-OLSR utilisant de la protection inégale (MP-OLSR UEP) sur les séquences de référence : Skatefar, Burnzoom, Aspen, Shadowboxing. Plus le critère VQM est faible, plus la vidéo est de meilleure qualité. . . . . 92
6.16 Taux de décodage de chacune des séquences vidéos en fonction du protocole de routage et de la protection utilisés. . . . . 93

# Index


Élément Structurant à 2 Pixels (ES2P), 29

algorithme de Dijkstra, 77
algorithme de Shamir, 96
Asynchronous Transfert Mode (ATM), 25

buffer géométrique, 22, 36
    hexagonal, 40

Ceph, 22, 46, 57
chiffrement à clé secrète, 23
Chord, 66
Cloud
    Computing, 60, 62
    Mobile Cloud Computing, 97
    Storage, 23, 46, 66
codage canal, 30
    déterministe, 30, 35
    probabiliste, 30, 35
    systématique, 63, 91
codage conjoint source-canal, 25
code
    $(1+\varepsilon)MDS$, 35
    Fountain, 35
    Goppa, 96
    LDPC, 22
    MDS, 22, 25, 35, 47, 89
    Raptor, 22
    Reed-Solomon, 22, 25, 95
    Tornado, 35
code Mojette
    $(1+\varepsilon)MDS$, 38, 49
    $(1+\varepsilon)MDS$, 95
    non systématique, 36, 49
    systématique, 38, 75
Colloque Francophone pour l'Ingénierie des Protocoles (CFIP), 72, 75
compression d'image
    Local Adaptive Resolution (LAR), 23, 44
    sans perte basé Mojette, 23
compression vidéo
    H.264/AVC, 23, 44
    H.264/SVC, 89, 90, 94
Content Delivery Networks (CDN), 68, 72
corps de Galois, 29, 49
critère de reconstruction
    de Katz, 29
    de Normand, 29
crypto-système de McEliece, 96

description multiple, 20
DFS, 46
Dietmar, 33

Farey
    séries, 31
FEC4Cloud, 22, 50, 64, 95

Grid5000, 22, 56, 60, 96
Groupe de Recherche (GDR)
    Arch. Sys. et Réseaux (ASR), 21
    Information Signal Image viSion (ISIS), 21
    Informatique Mathématique (IM), 21

haute disponibilité, 47
HDFS, 22, 62, 96

IDA, 66
image des index, 33
image unitaire, 33
Internet Engineering Task Force (IETF), 21, 76, 77, 97
Internet Research Task Force (IRTF), 80
ISA-L, 22, 49, 52
ixels, 36

Jerasure, 22, 62

Louis Pouzin, 21

MANET *Working Group*, 76, 77, 94, 97
MapReduce, 61, 96
matrice
    de Cauchy, 26, 49
    de Vandermonde, 26
montage vidéo en ligne, 54

NetFilter, 80, 86
NS2, 80

P2PWeb, 67–69, 95
protection inégale Mojette, 44
protocole
    AODV, 76, 81, 83
    BitTorrent, 67, 70
    DASH, 97
    DSR, 76, 81, 83
    MP-OLSR, 76–79, 82
    NTP, 86
    OLSR, 76, 77, 79, 82

Qualité
    d'usage (QoE), 66, 80, 92, 97
    de Service, 97
    de Service (QoS), 66, 80, 82, 85, 91
Qualnet, 80, 82, 89

réseaux
    auto-organisés, 21
    de capteurs, 21
    mobile ad hoc, 21, 76, 96
    P2P, 21, 66
RAID-6, 46
recommandation ITU P.910, 89
reconstructibilité
    théoreme, 30
redondance Mojette, 28
Reed-Solomon, 63
Rozo Systems, 22
RozoFS, 22, 49, 52, 57, 95

Sereadmo, 22, 80, 85, 86, 93
Software Defined
    Network (SDN), 21
    Storage (SDS), 23, 46, 53
stockage Mojette, 54
système complexe, 53

taille des projections, 28, 37, 42
tomographie, 30
transformation de Radon finie (FRT), 45, 89, 91
transformation Mojette, 25
    $nD$, 29
    Dirac, 27
    Spline, 27

Unequal Error Protection, 44, 91–93

Video Quality Expert Group (VQEG), 89
Video Quality Metric, 92, 94

# Bibliographie


[1] A.A. Alatan, M. Zhao, and A. N. Akansu. "unequal error protection of SPIHT encoded image bit streams". *IEEE Journ. on Sel. Areas in Commun.*, 18(6) :814–818, Juin 2000.

[2] A. Albanese, J. Blömer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. *IEEE Transactions on Information Theory*, 42(6) :1737–1744, November 1996.

[3] Florent Autrusseau, Benoît Parrein, and Myriam Servières. Lossless compression based on a discrete and exact Radon transform : A preliminary study. In *International Conference on Acoustics, Speech and Signal Processing ICASSP 06*, volume 2, pages 425–428, May 2006.

[4] Marie Babel, Benoît Parrein, Olivier Déforges, Nicolas Normand, Jean-Pierre Guédon, and Véronique Coat. Joint source–channel coding : Secured and progressive transmission of compressed medical images on the internet. *Computerized Medical Imaging and Graphics*, 32(4) :258–269, 2008.

[5] Thomas Beth. Finite versions of the radon-transform based on finite geometric structures. In *Mathematical Aspects of Computerized Tomography*, pages 7–12. Springer, 1981.

[6] J. Blömer, M. Kalfane, R. Karp, M. Karpinski, M. Luby, and D. Zuckerman. An xor-based erasure-resilient coding scheme. Technical Report TR-95-048, International Computer Science Institute, Berkeley, 1995.

[7] Fadi Boulos, Wei Chen, Benoît Parrein, and Patrick Le Callet. A new H. 264/AVC error resilience model based on regions of interest. In *Packet Video Workshop, 2009. PV 2009. 17th International*, pages 1–9. IEEE, 2009.

[8] Fadi Boulos, Wei Chen, Benoît Parrein, and Patrick Le Callet. Region-of-interest intra prediction for H. 264/AVC error resilience. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 3109–3112. IEEE, 2009.

[9] Fadi Boulos, Benoît Parrein, Patrick Le Callet, and David Hands. Perceptual effects of packet loss on H. 264/AVC encoded videos. In *Fourth International Workshop on Video Processing and Quality Metrics for Consumer Electronics VPQM-09*, 2009.

[10] Nassima Bouzakaria, Majd Ghareed, Benoît Parrein, and Soufiane Rouibia. Une architecture hybride client/serveur et pair-à-pair pour le streaming vidéo sur l'internet. In *CFIP/NOTERE 2012*, pages pp–1, 2012.

[11] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A digital fountain approach to reliable distribution of bulk data. Technical Report TR-98-013, International Computer Science Institut, Berkeley, California, 1998.

[12] Brad Calder, Ju Wang, Aaron Ogus, Niranjan Nilakantan, Arild Skjolsvold, Sam McKelvie, Yikang Xu, Shashwat Srivastav, Jiesheng Wu, Huseyin Simitci, et al. Windows azure storage : a highly available cloud storage service with strong consistency. In *Proceedings*

*of the Twenty-Third ACM Symposium on Operating Systems Principles*, pages 143–157. ACM, 2011.

[13] Bin Cheng, Hai Jin, and Xiaofei Liao. Rindy : a ring based overlay network for peer-to-peer on-demand streaming. In *Ubiquitous Intelligence and Computing*, pages 1048–1058. Springer, 2006.

[14] Seongho Cho, Joonho Cho, and Sung-Jae Shin. Playback latency reduction for internet live video services in cdn-p2p hybrid architecture. In *Communications (ICC), 2010 IEEE International Conference on*, pages 1–5, May 2010.

[15] Eddy Cizeron. *Routage multichemins et codage à description multiple dans les réseaux ad hoc*. PhD thesis, Université de Nantes, 2009.

[16] Thomas Clausen, Christopher Dearlove, Philippe Jacquet, and Ulrich Herberg. The optimized link state routing protocol version 2 (OLSRv2). IETF RFC 7181 (Proposed Standard), April 2014.

[17] Thomas Clausen and Philippe Jacquet. Optimized Link State Routing Protocol (OLSR). IETF RFC 3626, October 2003.

[18] Bastien Confais. Expérimentation de Rozofs sur Grid5000. Projet Recherche et Développement, 5ème année, Polytech Nantes, Département Informatique, Février 2015.

[19] Peter Corbett, Bob English, Atul Goel, Tomislav Grcanac, Steven Kleiman, James Leong, and Sunitha Sankar. Row-diagonal parity for double disk failure correction. In *Proceedings of the 3rd USENIX Conference on File and Storage Technologies*, pages 1–14, 2004.

[20] Laurent Dairaine, Jérôme Lacan, Laurent Lancérica, and Jérôme Fimes. Content-access qos in peer-to-peer networks using a fast MDS erasure code. *Computer Communications*, 28(15) :1778 – 1790, 2005. ICON 2004 : Advances and Directions in Quality of Service Management.

[21] Sylvain David, Pierre Evenou, Jeanpierre Guédon, Nicolas Normand, and Benoît Parrein. Erasure coding and decoding using mojette projection. *Brevet EP 13306435.2*, Oct. 2013.

[22] Sylvain David, Pierre Evenou, Jean-Pierre Monchanin, and Didier Féron. Système et procédé de traitement de données. *Brevet FR-14 52585*, Mars 2014.

[23] Olivier Déforges, Marie Babel, Nicolas Normand, Benoît Parrein, Joseph Ronsin, Jean-Pierre Guédon, and Laurent Bédat. Le LAR aux Mojettes. In *CORESA 04-COmpression et REprésentation des Signaux Audiovisuels*, pages 165–168. CORESA, 2004.

[24] Edsger W. Dijkstra. A note on two problem in connection with graphs. *Numerische Mathematik*, 1, 1959.

[25] L D'Acunto, JA Pouwelse, and HJ Sips. A measurement of nat and firewall characteristics in peer-to-peer systems. In *Proc. 15-th ASCI Conference*, volume 5031, pages 1–5. Advanced School for Computing and Imaging (ASCI), 2009.

[26] Dietmar Eggeman, Jeanpierre Guédon, and Nicolas Normand. Transferts de documents multimédias sur réseaux à qualité de service. In *CORESA*, Lannion, January 1998.

[27] Edwin O Elliott. Estimates of error rates for codes on burst-noise channels. *Bell system technical journal*, 42(5) :1977–1997, 1963.

[28] Bin Fan, Wittawat Tantisiriroj, Lin Xiao, and Garth Gibson. Diskreduce : Replication as a prelude to erasure coding in data-intensive scalable computing. Technical report, Parallel Data Laboratory, Carnegie Mellon University, 2011.

[29] Majd Ghareeb, Soufiane Rouibia, Benoît Parrein, Mohamad Raad, and Cédric Thareau. P2PWeb : A client/server and P2P hybrid architecture for content delivery over internet. In *Communications and Information Technology (ICCIT), 2013 Third International Conference on*, pages 162–166. IEEE, 2013.

[30] Vivek Goyal. Muliple description coding : Compression meets the network. *IEEE Signal processing magazine*, 18(5) :74–93, September 2001.

[31] Jeanpierre Guédon, editor. *The Mojette Transform : Theory and Applications.* ISBN : 9781848210806. Wiley-ISTE, Janvier 2009.

[32] Jeanpierre Guédon. *The Mojette transform : theory and applications.* John Wiley & Sons, 2013.

[33] Jeanpierre Guédon, Dominique Barba, and Nicole Burger. Psychovisual image coding via an exact discrete Radon transform. In Lance T. Wu, editor, *Proc. Visual Communications & Image Processing (VCIP)*, volume 2501, pages 562–572, Taipei, Taiwan, May 1995.

[34] Jeanpierre Guédon, Benoît Parrein, and Nicolas Normand. Internet distributed image information system. *Integrated Computer-Aided Engineering*, 8(3) :205–214, 2001.

[35] JP. Guédon and D. Barba. Décomposition psychovisuelle et transmission rapide d'images. *CORESA*, page 6 pages, Janvier 1995.

[36] Cheng Huang, Huseyin Simitci, Yikang Xu, Aaron Ogus, Brad Calder, Parikshit Gopalan, Jin Li, Sergey Yekhanin, et al. Erasure coding in Windows Azure storage. In *USENIX Annual Technical Conference*, pages 15–26. Boston, MA, June 2012.

[37] Mohamed Jawad, Patricia Serrano-Alvarado, and Patrick Valduriez. Supporting Data Privacy in P2P Systems. In Richard Chbeir and Bechara Al Bouna, editors, *Security and Privacy Preserving in Social Networks*, page 50 pages. Springer, August 2013. Chapter of 50 pages.

[38] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In *Mobile computing*, pages 153–181. Springer, 1996.

[39] Myron Bernard Katz. *Questions of uniqueness and resolution in reconstruction from projections*, volume 26 of *Lecture Notes in Biomath.* Springer-Verlag, Berlin, RFA, 1978.

[40] Jacques Klossa, Marc Tarin, P. Horain, and Benoît Parrein. Procédé de dématriçage d'image, European Patent, EP04291269, May 2004.

[41] Mao Kun, Yu Jingdong, and Ren Zhi. The research and simulation of multipath olsr for mobile ad hoc network. In *International Symposium on Communications and Information Technologies (ISCIT)*, pages 540–543, 2005.

[42] J Lacan and J Fimes. Systematic MDS erasure codes based on Vandermonde matrices. *IEEE Communications Letters*, 8(9) :570–572, September 2004.

[43] M. Luby, M. Mitzenmacher, A. Shokrollahi, D. Spielman, and V Stemann. Pratical loss-resilient codes. *Proceedings of the 29th ACM Symposium on Theory of computing*, 1997.

[44] Michael Luby. LT codes. In *Proc. 43rd Ann. IEEE Symp. Found. Comp. Sci.*, pages 271–280, November 2002.

[45] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes.* North Holland, 1983.

[46] František Matúš and Jan Flusser. Image representation via a finite Radon transform. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 15(10) :996–1006, October 1993.

[47] Robert J McEliece. A public-key cryptosystem based on algebraic coding theory. *DSN progress report*, 42(44) :114–116, 1978.

[48] Robert J. McEliece and Dilip V. Sarwate. On sharing secrets and Reed-Solomon codes. *Communications of the ACM*, 24(9) :583–584, 1981.

[49] A.E. Mohr, E.A. Riskin, and R.E. Ladner. Unequal loss protection : graceful degradation of image quality overpacket erasure channels through forward error correction. *Selected Areas in Communications*, 18, Issue 6 :819 – 828, June 2000.

[50] N. Normand, JP. Guédon, O. Philippé, and D. Barba. Controlled redundancy for image coding and high-speed transmission. In R. Ansari and MJ. Smith, editors, *Proc. SPIE Visual Communications and Image Processing 1996*, volume 2727, pages 1070–81. SPIE, February 1996.

[51] Nicolas Normand and Jeanpierre Guédon. La transformée Mojette : une représentation redondante pour l'image. *Comptes-Rendus de l'Académie des Sciences*, 326 :123–126, January 1998.

[52] Nicolas Normand, Andrew Kingston, and Pierre Évenou. A geometry driven reconstruction algorithm for the Mojette transform. In A. Kuba, LG. Nyúl, and K. Palágyi, editors, *Discrete Geometry for Computer Imagery*, volume 4245 of *Lecture Notes in Computer Science*, pages 122–133, Szeged, Hungary, October 2006. Springer Berlin / Heidelberg.

[53] Nicolas Normand and Benoît Parrein. Description multiple et codage source-canal sur un canal à effacement avec la transformation mojette. *17° Colloque sur le traitement du signal et des images (GRESTI)*, January 1999.

[54] Nicolas Normand, Imants Svalbe, Benoit Parrein, and Andrew Kingston. Erasure coding with the finite radon transform. *Wireless Communications and Networking Conference (WCNC), IEEE, Sydney, Australia*, April 2010.

[55] Frédérique Oggier and Anwitaman Datta. Coding techniques for repairability in networked distributed storage systems. *Foundations and Trends® in Communications and Information Theory*, 9(4) :383–466, 2012.

[56] ITU-T Recommandation P.910. Subjective video quality assessment methods for multimedia applications, April 2008.

[57] Benoît Parrein. *Description multiple de l'information par transformation Mojette.* thèse de doctorat, Université de Nantes, November 2001.

[58] Benoît Parrein, Fadi Boulos, Patrick Le Callet, and Jeanpierre Guédon. Priority image and video encoding transmission based on a discrete radon transform. In *Packet Video 2007*, pages 105–112. IEEE, 2007.

[59] Benoit Parrein, Jeanpierre Guédon, and Nicolas Normand. Multimedia forward error correcting codes for wireless LAN. *Annals of Telecommunications*, 3(4) :448–463, 2003.

[60] Benoıt Parrein, Nicolas Normand, and Patrick Le Callet. Transmission prioritaire jpeg2000 sur lien sans fil. In *20° Colloque sur le traitement du signal et des images, FRA, 2005.* GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2005.

[61] Benoît Parrein, Marc Tarin, and Patrick Horain. Demosaicking and JPEG2000 compression of microscopy images. In *International Conference on Image Processing*, Oct. 2004.

[62] Benoît Parrein and N. Normand. Multiplexage de flux multimédias sur IP par description multiple mojette. *CORESA, Sophia Antipolis*, pages 211–218, juin 1999.

[63] Benoît Parrein, N. Normand, and J. P. Guédon. Multiple description using exact discrete Radon transform. *IEEE Data Compression Conference (DCC)*, page 508, mars 2001.

[64] Benoît Parrein, Nicolas Normand, and Dominique Barba. Description multiple par transformation de Radon discrète exacte. In *18° Colloque sur le traitement du signal et des*

*images, FRA, 2001*. GRETSI, Groupe d'Etudes du Traitement du Signal et des Images, 2001.

[65] Charles Perkins, E Belding-Royer, and Samir Das. Ad hoc on demand distance vector (AODV) routing (rfc 3561). *IETF MANET Working Group*, August 2003.

[66] Dimitri Pertin, Sylvain David, Pierre Evenou, Benoît Parrein, and Nicolas Normand. Distributed file system based on erasure coding for i/o-intensive application. In *Proc. of the 4th International Conference on Cloud Computing and Services Science (CLOSER), Barcelona, Spain*, April 2014.

[67] Dimitri Pertin, Didier Féron, Alexandre Van Kempen, and Benoît Parrein. Performance evaluation of the Mojette erasure code for fault-tolerant distributed hot data storage. *ArXiv e-prints (1504.07038)*, April 2015.

[68] Dimitri Pertin and Nicolas Normand. Re-projection without reconstruction. In *GeoDIS, Reims*, Novembre 2014.

[69] Dimitri Pertin, Benoît Parrein, and Nicolas Normand. The Mojette erasure code for distributed file systems. In *ACM Eurosys (poster session)*, Amsterdam, April 13-16 2014.

[70] Olivier Philippé. *Représentation d'images pour le codage conjoint source-canal sur réseaux à qualité de service*. PhD thesis, Université de Nantes, IRESTE, 1998.

[71] J. S. Plank, K. M. Greenan, and E. L. Miller. Screaming fast Galois Field arithmetic using Intel SIMD instructions. In *FAST-2013 : 11th Usenix Conference on File and Storage Technologies*, San Jose, February 2013.

[72] J. S. Plank, J. Luo, C. D. Schuman, L. Xu, and Z. Wilcox-O'Hearn. A performance evaluation and examination of open-source erasure coding libraries for storage. In *FAST-2009 : 7th Usenix Conference on File and Storage Technologies*, February 2009.

[73] J. S. Plank, S. Simmerman, and C. D. Schuman. Jerasure : A library in C/C++ facilitating erasure coding for storage applications - Version 1.2. Technical Report CS-08-627, University of Tennessee, August 2008.

[74] J. S. Plank and L. Xu. Optimizing Cauchy Reed-solomon Codes for fault-tolerant network storage applications. In *NCA-06 : 5th IEEE International Symposium on Network Computing Applications*, Cambridge, MA, July 2006.

[75] Michael O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *Journal of Association for Computing Machinery (ACM)*, 36(2) :335–348, April 1989.

[76] Johann Radon. Über die bestimmung von functionen durch ihre integralwerte langs gewisser mannigfaltigkeiten. *Berichte sachsische academie der wissenchaften, Leipzig, Math.-Phys. K1*, 69 :262–267, 1917.

[77] Johann Radon. Über die bestimmung von funktionen durch ihre integralwerte längs gewisser mannigfaltigkeiten. *Classic papers in modern diagnostic radiology*, 5, 2005.

[78] D. Radu, C. Avram, A. Astilean, B. Parrein, and Jiazi Yi. Acoustic noise pollution monitoring in an urban environment using a vanet network. In *Automation Quality and Testing Robotics (AQTR), 2012 IEEE International Conference on*, pages 244–248, May 2012.

[79] Dan Radu. *Efficient communication strategies for mobile ad hoc networks*. PhD thesis, Technical University of Cluj-Napoca, Romania, Octobre 2012.

[80] Dan Radu, Jiazi Yi, and Benoît Parrein. QoE enhancement for H.264/SVC video transmission in MANET using MP-OLSR protocol. In *Proceedings of ISIVC 2012, the 6th International Symposium on signal, Image, Video and Communications*, pages 1–4, Valenciennes, France, 2012.

[81] I. Reed and G. Solomon. Polynomial codes over certain finite fields. *Journal of the Society for Industrial and Applied Mathematics*, 8(2) :300–304, 1960.

[82] S. Rhea, C. Wells, P. Eaton, D. Geels, B. Zhao, H. Weatherspoon, and J. Kubiatowicz. Maintenance-free global data storage. *Internet Computing, IEEE*, 5(5) :40–49, Sep 2001.

[83] Sean C Rhea, Patrick R Eaton, Dennis Geels, Hakim Weatherspoon, Ben Y Zhao, and John Kubiatowicz. Pond : The oceanstore prototype. In *FAST*, volume 3, pages 1–14, 2003.

[84] Luigi Rizzo. Effective erasure codes for reliable computer communication protocols. *SIGCOMM Comput. Commun. Rev.*, 27(2) :24–36, April 1997.

[85] Soufiane Rouibia, Majd Ghareed, Benoît Parrein, Marco Biazzini, Raziel Carvajal-Gomez, Adriana Perez-Espinosa, and Patricia Serrano-Alvarado. Towards a Hybrid Client/Server and P2P Architecture for Content Delivery over the Internet. In *CFIP/NOTERE*, page pp.1, Bayonne, France, October 2012.

[86] Myriam Servières. *Reconstruction tomographique Mojette*. PhD thesis, Université de Nantes, 2005.

[87] Adi Shamir. How to share a secret. *Communications of the ACM*, 22(11) :612–613, 1979.

[88] A. Shokrollahi. Raptor codes. *IEEE Transactions on Information Theory*, 52(6) :2551–2567, June 2006.

[89] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord : A scalable peer-to-peer lookup service for Internet applications. In *Proceedings of the ACM SIGCOMM '01 Conference*, August 2001.

[90] Mark W Storer, Kevin M Greenan, Ethan L Miller, and Kaladhar Voruganti. Pergamum : Replacing tape with energy efficient, reliable, disk-based archival storage. In *Proceedings of the 6th USENIX Conference on File and Storage Technologies*, page 1. USENIX Association, 2008.

[91] Mark W Storer, Kevin M Greenan, Ethan L Miller, and Kaladhar Voruganti. Potshards–a secure, long-term storage system. *ACM Transactions on Storage*, 5(2) :5, 2009.

[92] Andrzej Szwabe, Adam Nowak, Emmanuel Baccelli, Jiazi Yi, and Benoît Parrein. Multipath for optimized link state routing protocol version 2. IETF Internet Draft, MANET WG, Novembre 2010. draft-szwabe-manet-multipath-olsrv2-01.txt.

[93] Andrzej Szwabe, Adam Nowak, Emmanuel Baccelli, Jiazi Yi, and Benoît Parrein. Multipath for optimized link state routing protocol version 2. IETF Internet Draft, MANET WG, Mai 2011. draft-szwabe-manet-multipath-olsrv2-02.txt.

[94] Pierre Verbert, Vincent Ricordel, and J. P. Guédon. Analysis of Mojette transform projections for an efficient coding. In *Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, Lisboa, Portugal, Apr 2004.

[95] Hakim Weatherspoon and John D. Kubiatowicz. Erasure coding vs. replication : A quantitative comparison. *In Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS 2002*, 2002.

[96] Sage A. Weil, Scott A. Brandt, Ethan L. Miller, Darrell D. E. Long, and Carlos Maltzahn. Ceph : A scalable, high-performance distributed file system. In *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, OSDI '06, pages 307–320, Berkeley, CA, USA, 2006. USENIX Association.

[97] Stephen Wolf and Margaret Pinson. *Video Quality Measurement Techniques*. NTIA Report 02-392, 2002.

[98] Taka Maeno Yasunori Owada, Kenta Tsuchida and Hiroei Imai. nOLSRv2, Niigata olsrv2 implementation, Niigata university, Japan. http ://www.net.ie.niigata-u.ac.jp/ yowada/v2/, 2006.

[99] Jiazi Yi. *Protocole de routage à chemins multiples pour des réseaux ad hoc.* PhD thesis, Université de Nantes, 2010.

[100] Jiazi Yi, Asmaa Adnane, Sylvain David, and Benoît Parrein. Multipath optimized link state routing for mobile ad hoc networks. *Ad Hoc Networks*, 9(1) :28–47, 2011.

[101] Jiazi Yi and Benoît Parrein. Multi-path extension for the optimized link state routing protocol version 2 (olsrv2) draft-ietf-manet-olsrv2-multipath-02. IETF Internet Draft, MANET WG, Octobre 2014.

[102] Jiazi Yi, Benoît Parrein, and Dan Radu. H.264/SVC video transmission over multipath routing protocol in ad hoc networks. In *14th IEEE Wireless Personal Multimedia Communications Symposium (WPMC'11)*, 2011.

[103] Zhe Zhang, Amey Deshpande, Xiaosong Ma, Eno Thereska, and Dushyanth Narayanan. Does erasure coding have a role to play in my data center ? *Microsoft Research Tech. Report, May*, May 2010.

[104] X. Zhou, Y. Lu, and B. Xi. A novel routing protocol for ad hoc sensor networks using multiple disjoint paths. In *2nd International Conference on Broadband Networks*, Boston, MA, USA, Oct. 2005.