Tetrys : Un mécanisme de fiabilisation polyvalent

Pierre Ugo Tournoux, Amine Bouabdallah, Emmanuel Lochin, Jérôme Lacan

Tetrys : Un mécanisme de fiabilisation polyvalent

Pierre Ugo Tournoux[*,**] **— Amine Bouabdallah** [*,**] **— Emmanuel Lochin**[*,**] **— Jerome Lacan**[*,**]

[*] *CNRS ; LAAS ; 7 avenue du colonel Roche, F-31077 Toulouse, France*
[**] *Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ; F-31077 Toulouse, France*

RÉSUMÉ. *Actuellement, une succession d'erreurs qui ne peut être masquée par le mécanisme de fiabilisation (code à effacement) doit attendre au minimum un RTT pour être corrigée, ce qui n'est souvent pas satisfaisant pour les application temps-réel. Les concepts apportés par la théorie du codage réseau (Network Coding) permettent aujourd'hui de combler le fossé entre fiabilisation et fiabilité totale en s'abstrayant du concept d'ARQ. Cet article présente un mécanisme innovant nommé Tetrys dont l'une des caractéristiques est de pouvoir reconstruire les pertes dans un temps paramétrable et indépendant du RTT. A notre meilleure connaissance, c'est la première fois que les propriétés temps réel d'un tel mécanisme sont énoncées et étudiées. Intuitivement, les applications ciblées sont celles nécessitant une fiabilité totale avec contrainte de délai. Il s'avère qu'à taux de redondance égal, des applications telles que la VoIP et la vidéo-conférence sont bien plus performantes lorsque les flux sont protégés par le mécanisme Tetrys que par les mécanismes FEC ou H-ARQ classiques. Après un rapide rappel des points clés relatifs à FEC et H-ARQ, nous décrivons le principe de Tetrys et montrons son possible déploiement. Nous comparons les performances de FEC, H-ARQ et Tetrys du point de vue applicatif à l'aide d'un prototype et suivant des métriques de délai et dans le cadre de la VoIP, de qualité de la transmission (MOS).*

ABSTRACT. *Currently, packets which are not rebuilt by the erasure codes' schemes are delayed of at least one RTT before they reach the receiver side application. Most of the time this additional delay do not satisfy the real-time applications requirements. Recently a new concept emerged from network coding allowing to fill the gap between one-way-delay and full reliability. In other words this concept will enable full reliability whithout ARQ schemes. This paper aims at introducing such an innovative scheme that we call Tetrys and which one of the most interesting characteristic is to rebuild all the packets within a delay independant of the RTT. To the best of our knowledge, this is the first time that the real-time characteristics of those kinds of schemes are stated and studied. Intuitively, the applications that may benefit from Tetrys are the ones that request full reliability under a constraint of delay (less than* $1.5 * RTT$*). Indeed it seems that with the same coding ratio, applications such as VoIP and video-conferencing show performances that are significantly better when the latter are protected by Tetrys than by FEC or H-ARQ. After a short reminder of the key points in relation to FEC and H-ARQ, we will describe how Tetrys works and show that it is possible to deploy it from the complexity side. We will compare FEC, H-ARQ and Tetrys' performances from the application point of view, by using the MOS as metrics in the case of VoIP, and the packet delivery delay in the case of applications requiring full reliability.*

MOTS-CLÉS : *Code à effacement, fiabilité, bout-en-bout, application temps réel*

KEY WORDS: *Network Coding, Erasure Codes, real-time application, reliability, end-to-end delay, best effort*

1. Introduction

Les codes FEC (Forward Error Correction), ou encore code FEC de niveau application (AL-FEC), ont gagné en popularité ces dernières années car ils permettent de réduire de façon drastique les pertes sur un canal à effacement. Leur principe est le suivant : à un ensemble de k paquets sources, les codes AL-FEC associent $n - k$ paquets de redondance de manière à ce que la récupération d'au moins k paquets parmi les n initiaux permettent la récupération des k paquets originels. En pratique, seul les codes optimaux (aussi appelés MDS [RIZ 97]) possèdent cette propriété tandis que d'autres type de codes (tels que LDPC, Fountain ou Raptor Codes) nécessitent de recevoir plus que k paquets afin de reconstruire la séquence initiale. Parmi les codes optimaux, certains sont dits de type systématiques et envoient les k paquets initiaux en clair (non modifiés) et les $n - k$ paquets de redondance qui contiennent une combinaison des k paquets initiaux. A l'inverse certains codes transforment les k paquets initiaux en n paquets encodés et aucun d'eux ne contient l'information initiale. Dans le second cas, on ne pourra récupérer que k paquets si le décodage réussit ou aucun paquet source si le décodage échoue alors que les codes systématiques permettent l'utilisation des paquets en clairs correctement reçus et ce, même si plus de $n-k$ pertes ont été observées. On note le taux de redondance le ratio $(n-k)/n$ et le taux d'encodage le rapport k/n. Ainsi, un taux d'encodage proche de 1 signifie une faible quantité de redondance et donc une faible tolérance aux pertes tandis qu'un taux proche de 0 signifie une forte tolérance aux pertes mais également synonyme d'une forte surcharge dans le réseau.

Quel qu'en soit l'utilisation, les avantages des AL-FEC reposent sur deux points principaux. Le premier est le fait que les pertes deviennent anonymes, dans le sens où $n-k$ pertes parmi n peuvent être tolérées avec uniquement $n-k$ paquets de redondance. Le second avantage se fait au niveau du délai de récupération d'une perte par rapport aux schémas ARQ. Il est important de noter que ceci n'est valable que pour les codes MDS (comme les codes Reed-Solomon). De manière générale, il est nécessaire de recevoir $(1+\epsilon) * k$ paquets pour récupérer les k paquets d'information. La perte de x paquets (avec $x < k$) peut être reconstruite après la réception d'au moins k paquets parmi les n initiaux. Les débits courants étant suffisamment grand, la durée entre l'instant de la première perte parmi les n paquets et l'instant de réception du $k$ème paquet est souvent inférieure comparé au temps requis pour demander à la source une retransmission. Ceci est particulièrement vrai lorsque le RTT est important. Cette seconde propriété rend l'utilisation des AL-FEC particulièrement avantageuse dans le cas d'applications temps réel où par définition, les données deviennent obsolètes si elles ne sont pas reçues avant l'écoulement d'un certain délai.

1.1. *Positionnement de Tetrys*

On constate donc que les propriétés qui rendent les FEC adaptées aux applications temps-réel sont rompues par H-ARQ puisque des retransmissions sont incontournables et donc un délai d'au moins un RTT est nécessaire pour reconstruire des pertes qui n'ont put être masquées par les FEC. Le but du présent article est de s'affranchir de ses retransmissions par l'introduction d'un concept de codage élastique, permettant le maintien des caractéristiques de FEC pour la fiabilité totale à savoir la reconstruction des pertes en un temps indépendant du RTT. Ce mécanisme autorise un faible délai de reconstruction et peut opérer sur des réseaux bi-directionnels de type *best-effort*, sur liens

asymétriques et caractérisés par un taux de perte important. Le terme de fenêtre élastique provient du fait que contrairement aux codages classiques, le nombre k de segments de données encodés dans les paquets de redondance n'est pas fixe et les informations contenues d'une opération de codage à l'autre ne sont pas forcement disjointes. Ces paquets de redondances contiennent une combinaison de toutes les informations envoyées par l'émetteur et dont la réception n'a pas encore été accusée (les acquittements sont optionnels). Notre système de codage de la fenêtre utilise des coefficients choisis aléatoirement dans le corps fini $\mathbb{F}_q$. La probabilité d'inversion de la matrice nécessaire au décodage demeure suffisamment importante pour ne pas avoir d'impact sur les performances du mécanisme [KAH 01]. Les acquittements permettent à l'émetteur de savoir quels paquets ont été reçus et donc ceux qu'il pourra supprimer de la fenêtre d'encodage. Seul le nombre de paquets de données inclus dans les paquets de redondance joue sur la complexité du mécanisme en termes de temps de calcul. C'est cette propriété qui permet à Tetrys d'être tolérant face aux pertes de ses acquittements ainsi qu'aux réseaux fortement asymétriques. Dans la suite de l'article, le mécanisme est décrit en détail avant de présenter les compromis associés entre terme de taille de buffer, de complexité et de délai de reconstruction de perte. Les performances du mécanisme sont ensuite décrites pour la VoIP qui illustre le cas des applications à fiabilité partielle. Enfin, nous présentons les performances de Tetrys dans le cadre d'applications nécessitant une fiabilité totale.

2. Présentation du mécanisme

Dans la suite du document, on notera par P_x le $x$ème paquet de données envoyé et P_1 le premier paquet. La taille en nombre d'octets Sz des paquets de données envoyés étant fixe, les données incluses dans P_x sont celles allant de $Sz * (x-1)$ à $Sz * x$ en supposant que la numérotation des octets commence à 0. $R_{(i..j)}$ représente le paquet de redondance qui contient une combinaison linéaire de tous les paquets de données P_k avec k allant de i à j. Chaque paquet de redondance est construit comme suit : $R_{(i..j)} = \sum_{k=i}^{j} \alpha_k^{(i,j)}.P_k$ où les coefficients $\alpha_k^{(i,j)}$ appartiennent au corps fini $\mathbb{F}_q$ et la multiplication avec P_x est réalisée en considérant que chaque paquet est un vecteur d'élément de $\mathbb{F}_q$ [RIZ 97]. Nous choisissons d'envoyer un acquittement à une fréquence donnée : F_{SACK}. Cet acquittement peut contenir un bloc SACK tel que défini dans la RFC 2018 et qui contiendra tous les paquets correctement reçus ou décodés. L'ensemble W_{NACK} contient tous les paquets envoyés mais non accusés par le récepteur.

La notation et l'expression du taux de codage définies pour les AL-FEC reste valable à l'exception de $n-k$ qui vaut toujours 1. Ainsi, après l'émission de k paquets de données, un paquet de redondance $R_{W_{NACK}}$ est construit avec tous les paquets envoyés mais non accusés, c'est à dire l'ensemble des paquets de W_{NACK}. De son coté, le récepteur effectue l'opération de décodage inverse pour reconstruire les paquets perdus. L'exemple suivant illustre le fonctionnement du mécanisme et la récupération sur une perte.

Supposons que la source ait envoyé les six paquets suivants (P_1, P_2, $R_{(1..2)}$, P_3, P_4, $R_{(1..4)}$). Par exemple, si P_2 est perdu, les paquets P_1 et $R_{(1..2)}$ permettent de récupérer P_2. Par contre, si P_1 et P_2 sont les seuls paquets perdus de la séquence, il faut attendre la réception du paquet de redondance supplémentaire (*i.e.* $R(1..4)$) afin de lui soustraire les paquets reçus P_3 et P_4 en effectuant $R'_{(1..4)} = R_{(1..4)} - \alpha_3^{(1..4)}.P_3 - \alpha_4^{(1..4)}.P_4$ pour obtenir $R'_{(1..4)}$ comme deuxième paquet de redondance nécessaire au décodage.

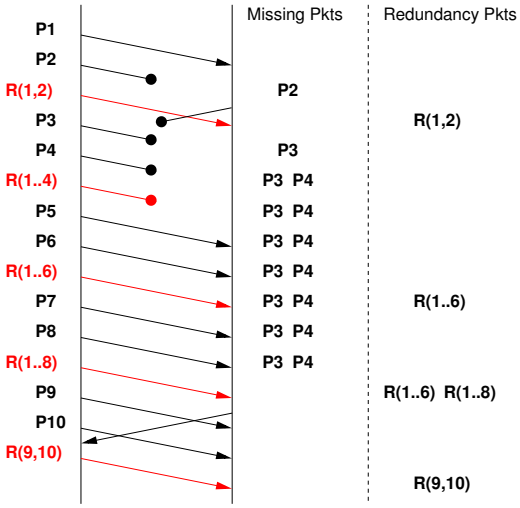


Figure 1. *Exemple simple de transfert unidirectionnel avec acquittements pour un taux d'encodage de* $2/3$

On obtient donc : $(R_{(1..2)}, {R'}_{(1..4)})^T = G.(P_1, P_2)^T$ où G est la matrice 2×2 :

$$G = \begin{pmatrix} \alpha_1^{(1..2)} & \alpha_2^{(1..2)} \\ \alpha_1^{(1..4)} & \alpha_2^{(1..4)} \end{pmatrix} \quad (1)$$

Ainsi, la reconstruction de (P_1, P_2) à partir de $(R_{(1..2)}, {R'}_{(1..4)})$ n'est possible que si G est inversible car si G^{-1} existe, il en résulte $(P_1, P_2) = G^{-1}.(R_{(1..2)}, {R'}_{(1..4)})$. La probabilité d'inversion de la matrice dépend du choix initial des coefficients de codage α. Ces coefficients sont choisis aléatoirement dans le corps fini $\mathbb{F}_{256}$ ce qui assure une probabilité d'inversion de cette matrice (et donc de décodage) très proche de 1 [KAH 01]. La figure 1 illustre le fonctionnement de Tetrys avec des acquittements. Le taux de redondance est de $2/3$ avec $n = 3$ et $k = 2$ et c'est donc un paquet de redondance qui est construit et envoyé tous les 2 paquets de données. Dans ce nouvel exemple, le paquet P_2 est perdu tandis que le paquet de redondance $R_{(1,2)}$ qui est correctement reçu permet de reconstruire P_2. Le paquet accusant la réception du paquet P_1 est perdu ce qui implique que P_1 continuera a être inclus dans les prochains paquets de redondance sans conséquence sur le devenir du transfert. Les paquets suivant, P_3, P_4 et $R_{(1..4)}$ sont perdus, contrairement au mécanisme H-ARQ, aucun d'entre eux ne nécessitent d'être retransmis puisqu'ils seront reconstruits grâce à la réception des paquets allant de P_5 à $R_{(1..8)}$. En supposant que l'application source soit à débit constant et émet un paquet toutes les $10ms$ et que le délai de transit soit symétrique et égal à $100ms$ (soit un RTT de $200ms$)[1], le délai observé entre la génération de P_3 et sa réception effective serait de $160ms$ contre un minimum $320ms$ pour H-ARQ. Comme pour la matrice (1) le récepteur peut reconstruire P_3, P_4 en soustrayant dans un premier temps les paquets source de manière à obtenir $({R'}_{(1..6)}, {R'}_{(1..8)})$.

1. A noter que la figure 1 présente un diagramme de séquence symbolique sans axe des temps à l'échelle.

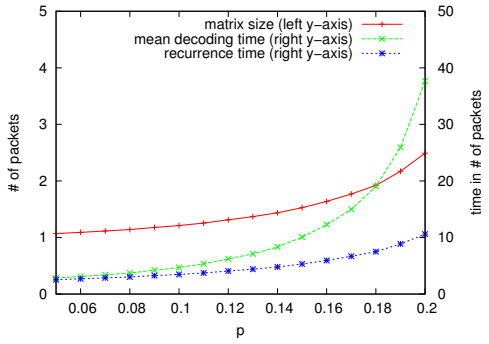


Figure 2. *Evolution du temps de récurrence, du délai et de la taille de la matrice à inverser en fonction du taux de perte pour un taux d'encodage de 3/4*

L'étape suivante consiste alors en l'inversion puis la multiplication par $(R'_{(1..6)}, R'_{(1..8)})$ de la matrice 2×2. Enfin, cet exemple illustre bien que la voie retour n'est utilisée que pour diminuer la complexité d'encodage au niveau de la source, en effet, à la réception du second acquittement (le premier ayant été perdu), la source recommence son processus de codage à partir du paquet #9 ce qui permet à la fois au récepteur de diminuer la complexité du codage et de s'assurer que les précédents paquets source ont bien été reçus.

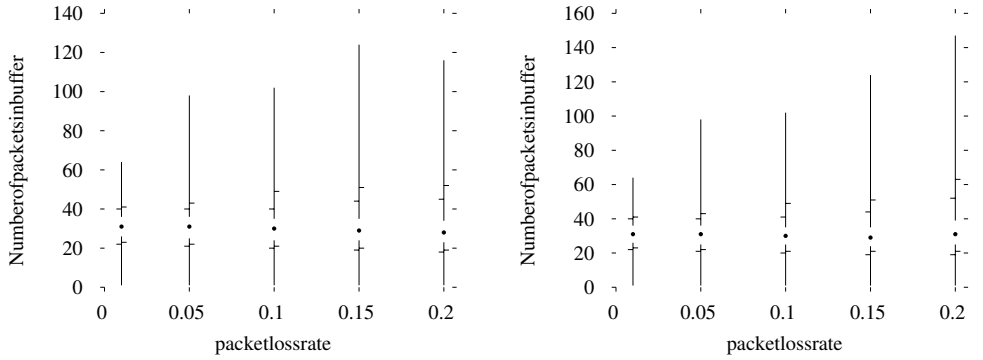
3. Performance et complexité

Il est clair que le point clé de la performance du mécanisme est lié au délai entre l'instant ou un paquet de données est perdu et l'instant ou le récepteur aura collecté suffisamment de paquet de redondance pour le reconstruire. Plus formellement, on nomme "temps de récurrence" le délai entre le moment ou une perte est détectée et le moment ou le nombre de perte devient inférieur ou égal au nombre de paquets de redondance reçus. Ce délai va être impacté par trois éléments : le taux de codage, le taux de perte et la distribution des pertes (uniformes ou par rafales). Dans le cas de pertes uniformes, le délai moyen peut être obtenu de manière analytique par l'étude de la chaîne de Markov de la marche aléatoire de la distance entre le nombre de perte et le nombre de paquet de redondance (voir [LAC 09]). La figure 2 montre l'importante du temps récurrence ainsi que sa tendance lorsque le taux de perte approche le taux de redondance (sur un canal à perte uniforme).

3.1. *Complexité en temps*

En termes de complexité en temps, on utilisera comme unité de base, le temps requis pour l'addition d'un paquet de données à un paquet de redondance. Du coté émetteur, la complexité pour l'encodage d'un paquet de redondance sera donc en $O(BS_t)$ avec BS_t le nombre de paquets de données présents dans la fenêtre d'encodage. Cette quantité qui est au minimum égale au produit du RTT et du débit d'émission est directement liée à la fréquence d'envoi d'acquittement ainsi qu'au taux de perte sur la voie retour. Du coté récepteur, la complexité va dépendre du nombre de pertes L

à reconstruire. Ce nombre de paquets étant égal au nombre de paquets de redondance impliqués dans l'opération de décodage, il faut compter de l'ordre de $L * BS_t$ opérations pour l'étape de soustraction. Quant à l'inversion et le produit matriciel, ils sont de l'ordre de L^2 opérations. Un décodage complet (pour les L paquets) est donc en $O(L * BS_t + L^2)$.



(a) Nombre de paquets de données en mémoire

(b) Nombre totale de paquets en mémoire (données et redondance)

Figure 3. *min max et (5, 10, 25, 50, 75, 90, 95) percentiles des tailles de buffer necessaire à Tetrys en fonction du taux de perte avec un taux d'encodage de* $3/4$

3.2. *Complexité en espace*

Concernant le taux d'occupation des buffers, du coté émetteur, le mécanisme nécessite de garder en mémoire $|BS_t|$ paquets. Cette valeur qui est fonction du RTT, du débit d'émission et de la fréquence des acquittements demeure stable dans une gamme de valeurs similaire à celle du buffer d'émission de TCP. Lorsque que le taux de perte est proche du taux de redondance, le nombre de paquets en attente de reconstruction peut augmenter à l'infini, augmentant ad-infinitum la taille de BS_t. Ce cas de figure est de toute manière à proscrire pour des raisons de performances. Coté récepteur, deux buffers sont nécessaires, un pour les données correctement reçues mais toujours incluses dans les paquets de redondance ; et un autre qui contient tous les paquets de redondance reçus entre le moment où une perte a été détectée et le moment où cette même perte est reconstruite. Tant que le taux de perte ne tend pas vers le taux de redondance, le buffer de redondance reste dans des valeurs acceptables. Ainsi, le buffer des données croit en fonction de $(RTT + F_{SACK}) *$ débit d'émission.

Les figures 3(a) et 3(b) montrent respectivement l'évolution de la taille du buffer de données et la taille totale des buffers $(donnees + redondance)$ en fonction du taux de perte, ces dernières étant uniformément distribuées. Ces évaluations montrent qu'à taux d'encodage donné $(3/4)$ et sur un taux de perte allant de 1 à 20%, la taille des buffers (*i.e.* corresponsant au total des paquets sources et redondances) reste stable (autour de 30 paquets). En effet, plus le taux de perte est élevé, plus il y a de paquets de redondance à mémoriser du côté récepteur et moins il y a de paquets sources à cause

des pertes accusées par la source. Ceci résulte en un certain équilibre du buffer de réception quelque soit le taux de perte.

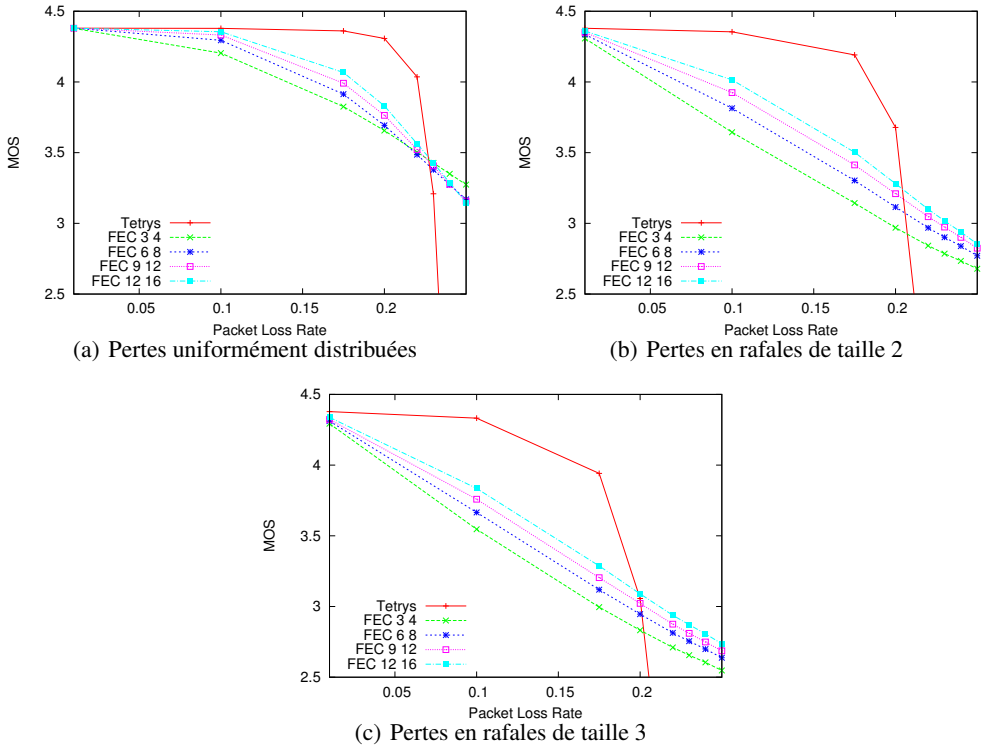
4. Application à fiabilité partielle

En guise d'exemple d'application temps-réel ne requérant qu'une fiabilité partielle, nous avons choisi la téléphonie sur IP (VoIP) et avons évalué ses performances avec pour métrique le MOS (Mean Opinion Score) développé par l'ITU pour l'encodeur G.711. Le MOS est une métrique subjective permettant d'obtenir une estimation moyenne de la qualité que percevrait un utilisateur. Ce score possède une échelle allant de 1 à 5 qui se traduit en termes de qualité audio phonique par les qualificatifs de mauvais, pauvre, passable, bon et excellent. L'ITU a développé le E-Model qui permet de calculer le MOS en fonction de deux paramètres que sont le délai accusé par chaque paquet et le taux de perte. Des paramètre spécifiques à chaque encodeur permettent d'adapter le modèle à ces derniers. Pour plus de détails sur l'utilisation du E-Model, le lecteur peut se référer à [ITU].

Nous avons effectué plusieurs simulations de 1000 secondes chacune en faisant varier le taux de perte de 1% à 25% pour un taux d'encodage fixe de $3/4$ avec des pertes uniformément distribuées et des pertes en rafales de taille moyenne respectives de 2 et 3. Le mécanisme FEC étant sensible à ce paramètre, les résultats sont donnés pour des blocs de taille 4, 8, 12 et 16. Pour chacun de ces cas, nous calculons le MOS obtenu par FEC et Tetrys. Le RTT est fixé à $200ms$ (équivalent au RTT entre La Réunion et la métropole). L'utilisation de H-ARQ n'est pas envisageable car une retransmission, pouvant provoquer un phénomène d'écho, résulterait forcément en une diminution du score final.

Les figures 4(a), 4(b) et 4(c) présentent l'évolution du MOS en fonction du taux de perte pour le codec G.711 avec une fréquence d'échantillonnage de $10ms$. On peut observer que le MOS de Tetrys demeure stable et plus élevé que celui de FEC et ce tant que le taux de perte reste inférieur à 20%. Ceci est du au fait que tant que le taux de perte reste sensiblement inférieur au taux de redondance, le temps de récurrence n'est pas significatif et le délai qu'il implique ne dégrade pas la qualité de la voix. A l'inverse, certaines pertes ne peuvent être reconstruites par FEC lesquelles ont un impact négatif sur le MOS. Ces observations, vraies lorsque les pertes sont uniformément distribuées, le sont d'autant plus lorsque la taille des rafales de pertes augmente. A l'inverse, lorsque le taux de perte approche le taux de redondance (25%), le délai de reconstruction des paquets perdus par Tetrys peut croître au point qu'aucune perte ne puisse être reconstruite dans les temps et être jouable par l'application. Cependant, il est à noter qu'au moment où les performances de Tetrys décroissent en dessous de celles de FEC, le MOS de FEC est déjà inférieur au seuil de qualité "pauvre" de l'E-model.

A taux de redondance égal et pour des plages d'utilisation satisfaisantes pour les utilisateurs, Tetrys est donc toujours meilleur que les différentes variantes de FEC. A performance égale, Tetrys peut donc être utilisé avec un taux de redondance moins important impliquant une surcharge moindre pour le réseau.



(a) Pertes uniformément distribuées

(b) Pertes en rafales de taille 2

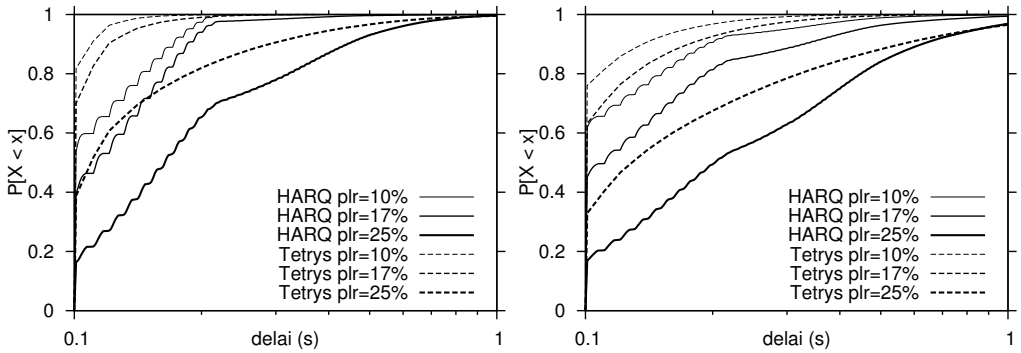
(c) Pertes en rafales de taille 3

Figure 4. *Variation du MOS en fonction du taux de perte pour le codec G711. Le taux d'encodage est de* $3/4$ *et les performances de FEC sont montrées pour diverses tailles de bloc*

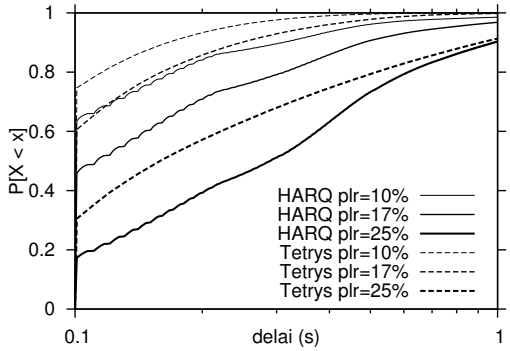
5. Application à fiabilité totale

Pour les applications requérant une fiabilité totale, Tetrys ne peut pas être comparé aux FEC car ces derniers n'ont pas cette propriété. C'est donc uniquement les performances d'H-ARQ qui seront étudiées. Il existe plusieurs variantes d'algorithmes H-ARQ adaptatifs. Nous avons utilisé un algorithme H-ARQ de type II où le nombre de paquets de redondance supplémentaires transmis à chaque requête du récepteur correspond au nombre de paquets manquant du bloc [LIN 83].

Le délai est la métrique utilisée pour comparer les performances de ces différents mécanismes. Il correspond au temps entre l'émission du paquet par la source et celui où il est transmis par le mécanisme de fiabilité (dans l'ordre) à la couche supérieure. Les figures 5(a),5(b) et 5(c) présentent la fonction de répartition (CDF) des délais de chaque paquet avec une échelle logarithmique pour l'axe des abscisses. Ce choix de représentation se justifie par le fait que les applications sont différemment impactées par les variations de délais et une simple moyenne n'aurait pas été suffisamment descriptive. Les paramètres de simulation utilisés sont les suivants : un taux de codage de $2/3$; chaque simulation dure 1000 secondes correspondant à une réception de 10^5 paquets ; les pertes sont res-



(a) Pertes uniformément distribuées (b) Pertes en rafales de taille 2



(c) Pertes en rafales de taille 3

Figure 5. *CDF du délai de livraison des paquets en ordre pour Tetrys et H-ARQ avec un taux de redondance de* $1/3$*, un RTT de* $200ms$ *et un débit de* 10 *paquets par seconde*

pectivement distribuées de manière uniforme et par rafales de moyenne 2 et 3 pour les figures 5(a), 5(b) et 5(c). Pour chacune de ces figures, la distribution des délais est présentée pour H-ARQ et Tetrys et pour des taux de perte de 10%, 17% et 25%. On constate qu'à délai égal, la probabilité d'être inférieur à ce dernier est toujours significativement supérieure pour Tetrys. Cela reste vrai lorsque l'on compare Tetrys avec 17% de pertes et H-ARQ avec 10% de pertes. A titre d'exemple, avec un taux de perte de 17% et des pertes uniformément distribuées, la probabilité d'obtenir un délai inférieur à $150ms$ est de 67% pour H-ARQ contre 94% pour Tetrys. Il est à noter que lorsque le taux de perte approche du taux de redondance, le temps de récurrence est tel que les délais rencontrés par Tetrys seront supérieurs à ceux d'H-ARQ. Cependant, comme pour la VoIP, il est probable que les performances d'H-ARQ ne soient également pas acceptable avec un tel ratio entre le taux de redondance et le taux de perte.

6. Conclusion et perspectives

Cet article décrit un mécanisme de fiabilité nommé Tetrys consistant en un encodage élastique permettant une fiabilité totale dans sa forme générique. Tetrys est résistant aux pertes d'acquittements et permet d'obtenir des délais de récupération des paquets perdus paramétrables et indépendant du RTT. Bien que ce concept de codage élastique ait récemment émergé dans des équipes différentes [LAC 09] [SUN 09], les caractéristiques de ce dernier demeurent méconnues. Il est à noter que cette étude présente la première utilisation d'un tel mécanisme afin de répondre à des contraintes de délai. Nous avons montré qu'à taux de redondance égal, ses performances sont meilleures que celle obtenues par FEC pour la fiabilité partielle. Pour la fiabilité totale, nous avons montré qu'au titre d'une charge plus légère pour le réseau (Tetrys ne fait aucune retransmission), Tetrys permet un délai de récupération également inférieur a celui des H-ARQ de type II. Ses caractéristiques en termes de complexité en temps et en espace autorisent son implémentation sans restriction particulière sur son environnement d'exécution. Le point faible du mécanisme, qui est l'augmentation du temps de récurrence lorsque le taux de perte approche le taux de redondance, peut être contourné par de simples solutions d'ingénierie telles que l'adaptation du taux de redondance en fonction du taux de perte observé de manière à ce que le délai reste dans les bornes requises par l'application. Bien que cet article se soit attaché à montrer que ce mécanisme est idéal pour la fiabilisation partielle ou totale de flux unicast temps réels, l'utilisation de ce mécanisme pourrait être un atout essentiel dans d'autres domaines tels que le multicast/anycast fiable.

6.1. *Bibliographie*

7. Bibliographie

[ITU] ITU, T-REC-G.107-200503-I, The E-Model, a computational model for use in transmission planning.

[KAH 01] KAHN J., KOMLÓS J., « Singularity Probabilities for Random Matrices over Finite Fields », *Combinatorics, Probability and Computing*, vol. 10, 2001, p. 137 - 157.

[LAC 09] LACAN J., LOCHIN E., TOURNOUX P., BOUABDALLAH A., ROCA V., « On-the-fly coding for time-constrained applications », avril 2009, ISAE Tech. Report http ://arxiv.org/abs/0904.4202.

[LIN 83] LIN S., COSTELLO D., *Error Control Coding : Fundamentals and Applications*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[RIZ 97] RIZZO L., « Effective erasure codes for reliable computer communication protocols », *ACM Computer Communication Review*, , 1997.

[SUN 09] SUNDARARAJAN J. K., SHAH D., MEDARD M., MITZENMACHER M., BARROS J., « Network coding meets TCP », *INFOCOM*, , 2009.