Université « POLITEHNICA » București
Université « Jean Monnet » St. Etienne

Thèse pour obtenir le grade de Docteur
de l'Université « POLITEHNICA » București et
de l'Université « Jean Monnet » St. Etienne

MESURE DE LA QUALITE

DANS LES RESEAUX INFORMATIQUES

Présentée et soutenue publiquement par

Răzvan Beuran

Le 26 juillet 2004
Directeurs de thèse : Prof. Vasile Buzuloiu et Prof. Jean-Marie Becker

Membres du jury :

M. Ion Bănică (rapporteur)
M. Jean-Marie Becker (directeur de thèse)
M. Philippe Bolon (rapporteur)
M. Vasile Buzuloiu (directeur de thèse)
M. Teodor Petrescu (président)
M. CongDuc Pham (rapporteur)

Motto

Two – were immortal twice –
The privilege of few –
Eternity – obtained – in Time –
Reversed Divinity –

That our ignoble Eyes
The quality conceive
Of Paradise superlative –
Through their Comparative.

Emily Dickinson, « Two – were immortal twice – », 1863

Remerciements

Ce travail n'aurait certainement pas pu être mené à bien sans le soutien, le concours et les conseils de nombreuses personnes. Je tiens à les remercier très sincèrement.

Je remercie tout d'abord M. Teodor Petrescu, professeur à l'Université « POLITEHNICA » București, qui me fait l'honneur de présider le jury de thèse.

Ma gratitude va également aux MM. Ion Bănică, professeur à l'Université « POLITEHNICA » București, Philippe Bolon, professeur à l'Université de Savoie, et CongDuc Pham, maître de conférences à l'Université Claude Bernard de Lyon, pour le temps précieux qu'ils m'ont accordé en acceptant d'être mes rapporteurs.

Je tiens à remercier tout particulièrement M. Vasile Buzuloiu, professeur à l'Université « POLITEHNICA » București, pour l'opportunité qu'il m'a présentée de poursuivre mon doctorat au CERN, la confiance témoignée et l'attention précieuse portée en tant que directeur de thèse roumain. Je apporte également des remerciements spéciaux à M. Jean-Marie Becker, professeur à l'Ecole Supérieure de Chimie Physique Electronique de Lyon, mon directeur de thèse français, pour les remarques très utiles qui m'ont permis d'enrichir ce manuscrit.

Mes remerciements les plus vifs vont au regretté Bob Dobinson, professeur à l'Université de Londres, et à M. Brian Martin, qui m'ont accueilli au sein de leur groupe au CERN, ont veillé de près à ce que mon travail se dirige dans de bonnes directions, et ont mis à ma disposition tous les moyens nécessaires.

Un grand merci à M. Neil Davies, professeur à l'Université de Bristol, pour les discussions extrêmement fécondes et les conseils judicieux qui ont précisé et clarifié les orientations de ma thèse.

Merci à tous les membres du groupe. Certains ont travaillé sur les mêmes projets que moi, comme Mihai Ivanovici (avec qui j'ai partagé aussi le bureau), Matei Ciobotaru, Micheal Levine, Jamie Lokier, Cătălin Meiroșu et Ștefan Stancu. D'autres m'ont simplement rendu la vie plus gaie, comme Alexandra Oltean ou Gilad Kent.

Je remercie aussi vivement tous les autres amis, tels que Michelle Connor, et ma famille, qui de près ou de loin m'ont soutenu tout au long de mon travail.

Table des matières

Résumé

La qualité est une notion essentielle pour les réseaux informatiques d'aujourd'hui par le fait qu'ils sont utilisés sur une échelle de plus en plus large par une gamme toujours plus grande de personnes. Son étude est très importante pour pouvoir juger de l'utilité et de la valeur des réseaux informatiques et des services qu'ils fournissent.

La qualité en général porte sur l'adéquation entre les attentes d'un sujet par rapport aux propriétés d'un objet. La qualité dans les réseaux informatiques comporte deux aspects : la qualité de service (QoS) et la qualité perçue par les utilisateurs (UPQ). La qualité de service traite l'adéquation mentionnée en termes de paramètres de performance du réseau : débit, délai et perte de paquets – un triplet avec deux degrés de liberté. La qualité perçue fournie un point de vue plus élevé que la QoS, celui des utilisateurs, et concerne leurs attentes en terme de performance des applications. Dans ce cas, des métriques spécifiques à chaque application sont définies, qui permettent d'évaluer de la manière la plus objective possible le degré de satisfaction des utilisateurs.

Nous avons développé une méthodologie et des systèmes de mesure qui permet d'évaluer la QoS et l'UPQ pour les réseaux informatiques et les applications qui tournent à travers eux. D'un côté nous avons pu évaluer la dégradation induite par les éléments du réseau et la différenciation de service qui est introduite par les mécanismes QoS courants. Dans ce dernier cas nous avons mis en évidence les inconvénients profondément inhérents à ces mécanismes. De l'autre côté nous avons établi la corrélation qui existe entre la QoS dans un réseau informatique et l'UPQ pour l'application qui tourne sur ce réseau par une évaluation simultanée des métriques associées à ces deux aspects de la qualité. L'étude effectuée sur le transfert de fichiers et la téléphonie IP nous a permis d'établir de manière objective les conditions que le réseau doit remplir afin que ces applications fournissent aux utilisateurs un niveau de satisfaction donné.

Abstract

Quality is an essential concept for computer networks today given that they are used on a larger and larger scale by an ever growing variety of people. Its study is of utmost importance for enabling judgments on the usefulness and value of computer networks and the services they provide.

Quality in general is about the correspondence between the expectations of a subject and the properties of an object. Quality in computer networks has two aspects: quality of service (QoS) and user-perceived quality (UPQ). Quality of service treats the above mentioned correspondence in terms of network performance parameters: throughput, loss and delay – a triplet with two degrees of freedom. Perceived quality asserts a higher point of view, that of users, and concerns their expectations in terms of application performance. In this case, application specific metrics are defined, that permit assessing in the most objective possible way the degree of user satisfaction.

We developed a methodology and test systems that allow assessing the QoS and UPQ for computer networks and applications running over them. On one hand we were able to evaluate the amount of degradation induced by network elements and the service differentiation introduced by current QoS mechanisms. In the latter case we showed the drawbacks that these mechanisms still have. On the other hand we established the correlation that exists between the network QoS and the application UPQ by simultaneously assessing the metrics associated to these two aspects of quality. The studies on file transfer and IP telephony allowed us to establish in an objective fashion the requirements the network must meet for applications to deliver their users a specified satisfaction level.

1 Préface

La qualité est une notion essentielle dans le monde actuel. Les gens s'intéressent à la qualité de toutes les choses essentielles à leur vie : l'air, l'eau, la nourriture. Les réseaux informatiques sont de plus en plus utilisés de nos jours. Et il ne s'agit pas seulement d'une utilisation restreinte, pour des raisons exclusivement scientifiques comme aux origines de son existence. L'Internet a pénétré à tous les niveaux – on l'utilise aujourd'hui pour communiquer, pour acheter, pour s'informer. Certains ne peuvent plus concevoir le monde sans Internet, voire que si on n'est pas connecté, on n'existe plus. Le temps est donc venu de se pencher véritablement sur la question de la qualité dans les réseaux informatiques.

1.1 La qualité

Le terme qualité a plusieurs sens aujourd'hui. Le dictionnaire de l'Académie Française définit la qualité comme « manière d'être d'une chose, bonne ou mauvaise, grande ou petite, chaude ou froide, blanche ou noire, etc. ». Le dictionnaire Hachette, de son côté, définit cette même notion comme étant une « propriété suscitant un jugement favorable ou défavorable de quelque chose, ce qui détermine sa nature, ce qui le rend propre à tel ou tel usage ». Mais elle peut aussi être un « critère permettant de donner une échelle des valeurs dans une même gamme de produits ». La définition la plus proche de la connotation que nous donnons à la qualité est : « chacun des aspects (positifs) de quelque chose qui font qu'elle correspond (au mieux) à ce que l'on attend ».

En effet, la qualité présuppose un sujet et un objet, et représente l'adéquation entre les attentes du sujet par rapport aux propriétés de l'objet. Mais elle ne fait pas proprement parler partie ni du sujet, ni de l'objet ; elle constitue donc une catégorie à part.

La qualité en soi a été étudiée par la philosophie depuis l'Antiquité en tant que propriété sensible et non mesurable qui détermine la nature d'un objet. Plus récemment, Robert Pirsig a conçu une métaphysique de la qualité, la situant comme fondement de tout ce qui existe. Dans [Pir-84] il arrive à la conclusion que la qualité est un événement dans lequel le sujet devient conscient de l'objet. Cet état de conscience le conduit à postuler que l'existence même du sujet et de l'objet est déduite de l'événement qualité, qui est donc leur cause ultime. Le même auteur trace

un parallèle [Pir-94] entre cette conception sur la qualité est la philosophie de la complémentarité de Niels Bohr et l'interprétation de Copenhague de la théorie quantique [Fol-85]. Dans ce contexte on considère que les particules mesurées ne sont pas « réelles » dans le sens où leurs attributs sont créés ou réalisés dans l'acte de mesure. D'une certaine façon on peut dire que c'est l'observation qui crée la réalité. Au niveau macroscopique et pratique cela revient au fait que toute mesure influence le système mesuré et que cette influence doit être minimisée.

Dans le contexte de la qualité, la mesure intervient dans l'estimation des propriétés de l'objet sur lequel portent les attentes du sujet. Ces propriétés doivent être mesurables au sens large, c'est-à-dire évaluables, que ce soit de manière subjective ou objective. Naturellement, dans le domaine scientifique l'évaluation objective est préférable et peut être construite en tant qu'équivalent statistique d'une évaluation subjective par un ensemble représentatif de sujets.

Une autre facette de la qualité est qu'elle met en évidence la valeur, une autre catégorie philosophique importante qui fait l'objet d'étude de l'axiologie. Car la qualité dans notre acception peut être assimilée à certains catégories du Bien, comme définie par Aristote dans [Ari-88]. L'utilitarisme de John Stuart Mill [Mil-94] mesure la qualité d'une action, le fait qu'elle est bonne ou pas, par son utilité et ses conséquences pour ceux qui en sont affectés. Tout ceci est transférable aux réseaux informatiques et aux mécanismes qui leurs sont associés. Il faut examiner quel est leur degré d'utilité, dans quelle mesure ils répondent aux attentes des utilisateurs afin de pouvoir décider sur leur valeur et leur qualité.

1.2 La qualité dans les réseaux informatiques

Les réseaux informatiques et surtout l'Internet ont été utilisés depuis de nombreuses années par les scientifiques, principalement pour faire de la recherche sur les réseaux ou pour échanger des informations entre eux [Pet-99]. L'accès à distance, le transfert de fichiers et le courrier électronique ont été parmi les applications les plus populaires. Le modèle sur lequel se basent les réseaux, les paquets individuels qui sont envoyés de manière indépendante à leurs destinations, marche très bien pour de telles applications. Mais « la Toile » a changé fondamentalement l'Internet, de nos jours le plus grand réseau public du monde. De nouvelles applications, comme la visioconférence, le commerce électronique ou la téléphonie IP se développent avec

une vitesse sans précédent. En ce XXIème siècle, l'Internet est destiné à devenir l'infrastructure de communication globale universelle.

Le terme *convergence* qualifie les réseaux multiservices qui intègrent le transport de la voix, de la vidéo et des données sur la même infrastructure [Cis-01]. La convergence apporte souplesse et évolutivité aux réseaux informatiques. Des applications, comme la téléphonie IP, la visioconférence, la sauvegarde en réseau ou la formation à distance profiteront de la convergence pour accélérer leur pénétration.

1.2.1 La qualité de service

Naturellement, afin de pouvoir déployer tous ces services, les fournisseurs doivent assurer une certaine qualité de service (Quality of Service, QoS) pour les applications. Les réseaux et leurs éléments ont une capacité finie et introduisent une dégradation de la qualité. La « quantité intrinsèque » de qualité doit être partagée entre utilisateurs et la gestion de la QoS est équivalente à la gestion de la dégradation induite par le réseau. Afin de représenter une certaine utilité, l'assurance de la qualité de service doit se faire en termes des garanties statistiques valables pour des intervalles de temps relativement courts, de même ordre de grandeur que la durée d'exécution des applications.

Mais l'intérêt présenté par la QoS dans les réseaux informatiques dépasse la simple utilisation d'Internet. Elle est aussi une question importante pour des applications spécifiques, tournant sur des réseaux dédiés, telle que le système de collection de données développé au CERN dans le cadre de l'expérience ATLAS [Lev-03].

Dans ce contexte il devient d'abord impératif d'avoir un cadre très précis pour la mesure de la QoS dans les réseaux informatiques. Les standards des divers organismes, comme le IETF, sont centrés principalement sur les performances globales des dispositifs réseau. Les propriétés de différenciation de service ne sont pas encore standardisées de manière satisfaisante. C'est la raison pour laquelle nous définissons dans notre travail un cadre pour la mesure des paramètres QoS pour les nœuds réseau. L'approche que nous prenons est celle de la *mesure active*. Ceci implique l'utilisation d'un système de test pour la génération de trafic artificiel avec des propriétés contrôlées par rapport à la taille des paquets, à l'espacement entre paquets et au débit. Ce trafic est introduit dans les éléments réseaux sous test et, après sa réception au niveau du système de test, tous les paramètres QoS essentiels sont

mesurés. Ces paramètres sont le débit, la perte de paquets et le délai, trois paramètres dans une relation à deux degrés de liberté – si un d'eux est fixé, une dépendance est créée entre les deux autres.

Par les mesures actives effectuée dans le cadre de ce travail nous avons pu établir que le comportement réel des éléments du réseau est rarement en parfait accord avec les modèles théoriques qui sont à la base de leur implémentation ou avec les garanties explicites ou implicites (mais souvent floues !) offertes par les fabricants.

1.2.2 La qualité perçue par les utilisateurs

La qualité de service est seulement un aspect de la qualité dans les réseaux informatiques qui se réfère à la relation qui existe entre les propriétés d'un système réseau et les attentes concernant ses propriétés en termes de paramètres de performance du réseau.

L'autre aspect de la qualité concerne l'adéquation entre propriétés et attentes à un niveau plus haut, le niveau de l'utilisateur des applications réseau. Il est extrêmement important, car c'est seulement à ce niveau qu'on peut vraiment faire des jugements sur l'utilité et la valeur.

Ce qui pousse la recherche dans ce domaine est bien sûr la nécessité d'améliorer les performances des applications réseau qui sont utilisées à une échelle de plus en plus large. Pour pouvoir quantifier les effets que les paramètres QoS et leur variation ont sur la qualité perçue par les utilisateurs (User Perceived Quality, UPQ) pour des applications spécifiques, il faut également définir des mesures de cette qualité. Si les paramètres QoS essentiels sont assez bien connus et qu'il existe de nombreux standards régularisant leur mesure, il en va tout autrement en ce qui concerne l'UPQ.

Chaque application a ses propres caractéristiques et demandes au niveau du réseau, qui se reflètent différemment au niveau de l'utilisateur. Par exemple, pour la téléphonie par Internet l'effet des paramètres QoS se traduit par une variation de la qualité sonore de la communication – cette qualité sonore sera donc la mesure la plus adéquate pour l'UPQ dans ce cas. Pour un transfert de fichier, le temps nécessaire pour le transfert ou l'efficacité peuvent être les facteurs d'intérêt.

1.2.3 Corrélation entre QoS et UPQ

Les deux aspects de la qualité, la QoS et l'UPQ, ne sont pas indépendants et pour étudier les applications réseau il faut mettre en relation les paramètres QoS avec la qualité au niveau de l'application. A présent il n'existe pas beaucoup de systèmes capables de corréler la QoS fournie par le réseau avec l'UPQ des applications spécifiques, comme la téléphonie par Internet ou le transfert des fichiers. Connaître ces demandes permet de prédire si une certaine connexion est valide pour une certaine application et quelle est la qualité perçue anticipée pour cette application. Dans le cadre de notre travail nous avons conçu un système qui permet l'accomplissement de cette fonction. L'approche dans ce cas est différente et implique une *mesure passive*. Après l'introduction du système de monitoring du trafic dans le réseau, les paramètres QoS sont calculés et corrélés avec l'UPQ pour les applications tournant par l'intermédiaire du réseau étudié. Dans notre travail nous avons utilisé un émulateur réseau pour obtenir un haut degré de contrôlabilité et de reproductibilité.

Les mesures passives ont permis l'étude d'une série d'applications réseau. Pour le transfert de fichiers nous avons démontré l'utilisation de notre système pour l'évaluation de la performance de ce type d'application, employable par exemple pour la comparaison des diverses améliorations. L'accent a été mis sur la téléphonie IP pour laquelle l'UPQ a été mesurée à l'aide d'une métrique objective, le score PESQ [ITU-862]. Pour cette application nous avons établi les frontières dans lesquelles les paramètres QoS du réseau doivent s'inscrire afin de fournir aux utilisateurs des niveaux de qualité perçue spécifiés.

1.3 Structure du mémoire

Le mémoire a la structure suivante. Le chapitre 2 présente les notions de base liées à la qualité de service dans les réseaux informatiques. Le chapitre 3 aborde la même question de la qualité, mais cette fois dans la perspective des utilisateurs d'applications réseau. Le chapitre 4 traite des systèmes de test utilisés pour la mesure active et les résultats obtenus, principalement dans les tests sur commutateurs Gigabit Ethernet. Les systèmes utilisés pour la mesure passive et les résultats obtenus dans l'étude du transfert des fichiers et la téléphonie IP sont présentés dans le chapitre 5. Ce mémoire se termine par un chapitre dédié aux conclusions (chapitre 6), les listes des figures et des tableaux, une liste d'abréviations et la bibliographie.

2 La qualité de service dans les réseaux informatiques

Quand on mesure une tension et qu'on obtient 12 V, on ne peut rien dire sur la qualité de cette source de tension. Le jugement sur la qualité intervient seulement lorsque on se préoccupe du contexte de l'utilisation de cette tension : alimentation d'une montre, d'une voiture ou d'une ampoule électrique. Il faut donc voir à quoi cette tension sert, et quelles sont les attentes par rapport à sa valeur.

De la même manière, la qualité de service représente les caractéristiques de performance d'un système réseau vues comme la fidélité du comportement observable du système par rapport aux attentes [PNSol], en termes de paramètres de performance mesurables : le débit, la perte de paquets et le délai.

Ce chapitre débute avec une présentation générale de la qualité de service dans les réseaux informatiques, soulignant les principales demandes pour un environnement réseau tenant compte de QoS. Puis nous présentons les paramètres QoS, tels que définis par les organismes de standardisation. La section suivante se concentre sur les solutions QoS au niveau des nœuds réseau, les éléments de construction de n'importe quelle architecture QoS, un des points d'intérêt principaux de notre travail.

Pour donner une vision globale sur la QoS, cette section est suivie par une discussion des solutions qui ont émergé pour permettre le contrôle de la QoS dans l'Internet de bout en bout, c'est-à-dire entre les ordinateurs situés aux extrémités des connexions (mon rapport de doctorat [Beu-02] peut être consulté pour plus de détails sur ce sujet). A la fin de la section on trouvera une discussion sur les questions d'autorisation, d'authentification et de facturation, suivie par quelques conclusions sur la QoS.

2.1 Introduction

Le succès phénoménal de l'Internet a créé de nouveaux défis. Plusieurs nouvelles applications imposent des demandes différentes par rapport à celles pour lesquelles l'Internet a été conçu. Une question importante est d'assurer la performance. Le modèle du datagramme, sur lequel l'Internet est basé, n'offre pas assez de manières de gérer les ressources dans le réseau – il ne peut pas offrir aux applications des garanties au niveau des ressources. Accéder à un site Web ou faire un appel

téléphonique par Internet peut s'avérer très difficile si certaines parties du réseau sont si occupées que les paquets ne peuvent pas passer. Les applications en temps réel, comme la visioconférence, demandent aussi un niveau minimum de ressources afin de pouvoir fonctionner effectivement. Au fur et à mesure que l'Internet devient un outil indispensable dans notre vie et notre travail, son manque de prédictibilité devient un problème nécessitant de manière urgente un traitement.

Une autre question est la différenciation des services. Etant donné que l'Internet traite tous les paquets de la même manière, il ne peut offrir qu'un seul niveau de service. Les applications, par contre, ont des exigences différentes. Les applications interactives, comme la téléphonie par Internet sont sensibles au délai et à la perte de paquets. Lorsque le délai ou le taux de perte dépassent certaines limites, ces applications sont rendues littéralement inutilisables. Par contraste, un transfert de fichier peut tolérer un bon degré de délai, et même une faible perte de paquets, sans que la performance observée soit trop dégradée. Les demandes des clients varient aussi par rapport à l'utilisation qu'ils donnent aux réseaux. Les organisations qui utilisent l'Internet pour des transactions bancaires ou pour le contrôle d'équipements industriels sont probablement disposées à payer plus pour que leur trafic réseau reçoive un traitement préférentiel. Pour beaucoup de fournisseurs de services réseau, pouvoir offrir plusieurs niveaux de service est vital pour leur succès. La capacité à fournir l'assurance des ressources et la différenciation des services constituent le fondement de la QoS.

La convergence sur les réseaux multiservices est essentielle du point de vue économique, pour éviter les dépenses occasionnées par la création de structures parallèles pour toutes les fonctions vitales d'une entreprise : téléphonie, courriel, visioconférence, commercialisation etc. L'Internet ne deviendra véritablement un réseau multiservice que lorsqu'il lui sera possible de supporter la différenciation des services. Implémenter de telles capacités dans l'Internet et les réseaux en général a été un des plus grands défis, touchant presque à tous les aspects des technologies réseau et demandant des modifications de l'architecture de base de l'Internet. Pendant plus d'une décennie, la communauté Internet a fait des efforts continus pour traiter ces questions et a développé un bon nombre de technologies nouvelles afin d'enrichir l'Internet avec des capacités QoS. Ces efforts continuent actuellement.

2.1.1 Exigences techniques

Indépendamment de la taille et de le visées de création d'un réseau IP, la QoS de bout en bout est constituée par la concaténation des QoS entre les limites de chaque domaine par lequel le trafic passe. Finalement, la QoS de bout en bout dépend des caractéristiques QoS à chaque nœud au long du chemin correspondant. Par exemple, la QoS éprouvée par une application téléphonique à l'intérieur d'un LAN dépend seulement du LAN ; par contre, une application téléphonique à longue distance dépend de tous les LAN et fournisseurs d'Internet aux deux extrémités, et du réseau IP au milieu.

Dans les réseaux IP d'aujourd'hui, une grande partie des pertes de paquets et de la gigue (« jitter » en anglais), imprédictibles et non différentiées, est due à la manière qu'ont les routeurs traditionnels « meilleur effort » de traiter la congestion interne passagère. Si un certain port de sortie est la destination de deux ou plusieurs flux de trafic, un routeur « meilleur effort » utilise simplement une file d'attente pour les paquets destinés à être servis. Une file d'attente introduit un délai et la possibilité de perte si la file déborde. Quand le trafic est en salves/rafales (« bursty » en anglais), le délai induit par la file d'attente varie considérablement d'un paquet à l'autre, se manifestant ainsi comme gigue au niveau des flux de trafic affectés.

Les réseaux IP à tout niveau doivent transporter du trafic appartenant à une variété croissante de clients avec des demandes diverses : téléphonie IP, réseaux privés, transfert massif de données, commerce électronique. Chaque client a ses exigences spécifiques visant le niveau de prédictibilité de services, même dans l'éventualité d'une congestion passagère causée par d'autres trafics traversant le réseau. La demande de protection absolue ou relative par rapport aux autres flux de trafic dans tous les segments du réseau s'applique aussi bien à un réseau local de haut débit, qu'à une connexion Internet établie par modem et ainsi de suite. Cette demande conduit à trois exigences techniques :

a) La QoS au niveau des nœuds réseau – Le plus petit élément contrôlable du réseau est le nœud (commutateur ou routeur) qui lie deux ou plusieurs connexions. L'architecture de ces nœuds doit être conçue de façon à ce que la gestion des files d'attente permette d'utiliser convenablement les caractéristiques QoS de liens entre nœuds.

b) Signalisation et réservation – La QoS contrôlable au niveau des nœuds réseau et l'expédition par des routes autres que la plus courte sont de moindre intérêt s'ils ne sont pas facilement gérables. Une solution pratique demande un certain degré de distribution automatique de paramètres QoS et/ou des contraintes d'ingénierie du trafic à tous les nœuds du réseau. Ces nouvelles informations sont distribuées chaque fois que le client impose ou modifie ses demandes sur la QoS.

c) Routage et ingénierie du trafic – Lorsqu'il y a plusieurs routes parallèles dans un réseau, distribuer le trafic sur ces routes peut réduire la charge moyenne et la nature en salve du trafic au long de chaque route. Cette pratique améliore la QoS observée du réseau, car pour chaque routeur la probabilité de perdre des paquets ou d'introduire une gigue est plus petite. Il y a nécessité de recourir à des mécanismes de découverte et d'imposition de routes optimales.

Afin de fournir une QoS de bout en bout, deux éléments doivent être simultanément déployés dans le réseau:

- Les solutions QoS au niveau des sauts (voir la section 2.3), qui permettent à chaque nœud de faire face aux demandes QoS de bout en bout et admettent la gestion de la QoS ;
- Les solutions QoS d'une frontière à l'autre (voir la section 2.4), qui assurent un traitement cohérent des flux de données dans une perspective QoS de bout en bout en se basant sur les solutions QoS à niveau de sauts.

Toutes ces questions sont explorées avec davantage de détails dans le reste de ce chapitre. Auparavant, il nous apparaît indispensable de proposer une courte introduction aux réseaux informatiques du point de vue de la QoS. C'est l'objet de la section qui suit.

2.1.2 Les réseaux informatiques

Tout réseau est constitué d'une hiérarchie de composants. Une route d'un point à l'autre est d'habitude formée par la concaténation de routes plus courtes, appelée sauts, au même niveau. Par exemple, le niveau IP est formé par des routeurs fonctionnant comme des points de commutation pour les paquets IP et des connexions qui transportent les paquets IP entre les routeurs. Chaque connexion est un seul saut

IP, mais elle-même peut être composée d'un certain nombre de sauts et connexions. Elle peut être un réseau Ethernet, une connexion virtuelle ATM (CV-ATM) etc. Dans le cas d'un réseau Ethernet, un ou plusieurs commutateurs Ethernet peuvent se situer entre les deux routeurs. Une CV-ATM offre un service de bout en bout entre les limites de la liaison, mais en réalité la connexion peut passer par l'intermédiaire de plusieurs commutateurs ATM au long de la route.

La QoS au niveau IP entre deux points dépend à la fois des routeurs sur le parcours et des caractéristiques QoS de la technologie de chaque connexion. Il est évident que le transport des paquets entre routeurs est basé sur les spécificités de chaque connexion. Si la technologie sous-jacente n'a pas une QoS contrôlable, les routeurs ne peuvent pas beaucoup compenser (c'est le cas, par exemple de l'Ethernet traditionnel). Par contre, en présence de technologies de connexion prenant en compte la QoS (e.g. l'Ethernet avec l'ajout du standard IEEE 802.1p/Q), le comportement des routeurs influence la disponibilité de la QoS au niveau IP. Tous ces éléments sont cruciaux dans les réseaux à haut débit [Cha-01].

La stratification des réseaux est fondamentalement récursive. Par exemple, les caractéristiques QoS d'une CV-ATM dépendent de la prédictibilité des connexions entre commutateurs aussi bien que des commutateurs eux-mêmes. Une CV ATM peut s'étendre sur plusieurs commutateurs en utilisant des circuits SONET/SDH[1] [Bla-02] pour le transport des cellules entre commutateurs. Le circuit SONET/SDH lui-même est formé par un ou plusieurs nœuds à travers différents anneaux et multiplexeurs. Finalement, il se peut que le SONET/SDH aie été multiplexé dans une seule fibre, avec d'autres circuits indépendants utilisant des longueurs d'onde différentes[2]. L'Internet complique davantage le modèle précédent, car beaucoup de routes de bout en bout ne sont pas contenues entièrement dans un seul réseau IP. Il est courant

---

[1] SONET et SDH sont un ensemble de standards pour transmission synchrone de données par réseaux de fibres optiques. SONET signifie « Synchronous Optical NETwork » et SDH est l'acronyme pour « Synchronous Digital Hierarchy ». SONET est la version des Etats Unis du standard publiée par ANSI (« American National Standards Institute ») et SDH et la version internationale publiée par UIT-T.

[2] Par une technologie de multiplexage de fibres optiques (« wavelength division multiplexing » en anglais), qui permet l'utilisation de plusieurs fibres virtuelles par l'intermédiaire d'un seul segment de fibre optique.

qu'une connexion s'étende à plusieurs réseaux IP administrés indépendamment, chacun avec sa propre politique de routage et ses caractéristiques QoS.

2.1.2.1 Dégradation induite de la qualité

Nous discutons et mesurons les propriétés de bout en bout des réseaux en termes de dégradation induite. La dégradation induite entre deux points de mesure quelconques représente le changement dans la qualité de service du réseau entre ces deux points. Nous utilisons la notation ΔQ pour cette dégradation.

La performance de bout en bout d'une application réseau dépend de la dégradation de bout en bout dans le réseau et de la façon dont l'application interagit avec elle. Nous considérons la quantité totale de dégradation comme la « composition » des dégradations locales induite par chaque sous-réseau (voir la figure 1). De point de vue expérimental elle représente l'accumulation des dégradations qui apparaissent dans chaque élément de réseau (commutateur, routeur etc.) sur la route.

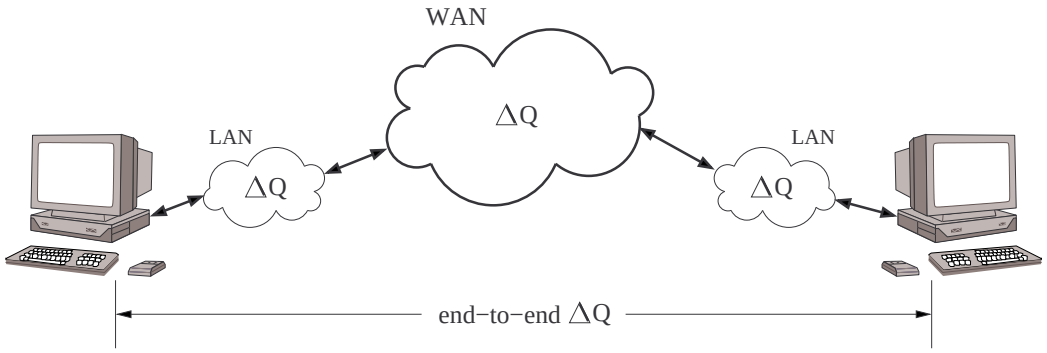


Figure 1 : La dégradation de la QoS de bout en bout est composée par les dégradations sur les sections intermédiaires du réseau.

Déterminer les caractéristiques de performance d'un système réseau est le premier pas dans la compréhension du comportement au niveau de l'application. Chaque application réseau demande un niveau minimum de QoS afin de s'exécuter en conformité avec les attentes des utilisateurs [Beu-03]. Chaque nœud induit une dégradation qui s'accumule. Il y a un maximum acceptable de dégradation de bout en bout induit par le réseau au trafic d'une application, afin que l'utilisateur considère qu'elle tourne de manière satisfaisante. Par exemple, pour la téléphonie IP le délai total à travers le réseau doit être inférieur à 150 ms pour avoir une bonne interactivité [Ser-01], [ITU-1541].

2.1.2.2 Diminution inévitable de la qualité

La qualité est toujours diminuée à travers un réseau et le conflit pour ressources, typiquement due à saturation, en est la cause principale. Afin de pouvoir analyser la qualité de bout en bout cet effet doit être évalué pour les commutateurs, les éléments de construction des réseaux. Une étude exhaustive est nécessaire car, même pour une utilisation moyenne réduite, les conditions de saturation peuvent apparaître à cause de la nature intrinsèque en salve du trafic Internet [Lel-94].

Une approche typique au moment quand les services d'un réseau ne sont plus satisfaisants est de fournir davantage de bande passante aux utilisateurs. Même si le problème semble résolu, il reste toujours là. Car tandis qu'on ne peut pas contrôler les interactions entre les flux de trafic la situation peut se détériorer à n'importe quel moment, quand la bande passante fournie n'est pas suffisante (même s'il s'agit seulement d'un intervalle de temps relativement court). Car un problème supplémentaire est le fait que dans l'Internet le rapport entre utilisation de pointe et utilisation moyenne a été estimé comme étant aux alentour de 1000:1 [Coc-98]. Pour comparaison, dans les réseaux téléphoniques le même rapport est de 5:1. Il résulte que dans les réseaux informatiques il n'est pas possible – de point de vue pratique – de fournir des ressources suffisantes pour assurer un fonctionnement parfait aux moments de pointe. La seule solution est d'utiliser des mécanismes de contrôle de la qualité pour fournir un niveau de QoS adéquat à tout moment.

Il faut noter tout de même que l'utilisation de mécanismes QoS n'est pas une façon d'assurer une qualité globalement améliorée, mais uniquement pour certains flux de trafic, par rapport à celle reçue par eux antérieurement. Les mécanismes QoS sont des techniques de « partage » de dégradation, qui ne délivrent pas de la qualité en soi. Un paquet est soit perdu soit retardé. Aucun de ces effets ne peut être défait : on ne peut pas faire qu'un paquet retardé arrive plus tôt, un paquet perdu ne peut pas être récupéré. Néanmoins, l'utilisation des mécanismes QoS peut empêcher la dégradation de dépasser certaines limites, s'ils sont employés de manière adaptée. Par exemple, la quantité totale de délai et la perte de paquets peuvent être partagés de façon contrôlée entre flux de trafic, introduisant la différenciation. La thèse de doctorat [Ree-03] propose une perspective semblable sur ce problème, mais son auteur prend la voie de la simulation pour l'approfondir.

Lorsque seulement le « meilleur effort » est demandé ou requis, on ne doit pas vraiment se soucier des réseaux intermédiaires au long de la route, pour autant que leurs politiques de routage permettent d'expédier le trafic. Néanmoins, afin de fournir la QoS de bout en bout on doit connaître davantage le comportement dynamique du réseau. Par contre, il n'est pas nécessaire de savoir *comment* chaque réseau atteint ses objectifs QoS. Il suffit de caractériser tout simplement chaque réseau en termes de délai, gigue et probabilité de perte de paquets qui peuvent être imposés au trafic, aussi bien que la bande passante. Etant donné que le réseau d'un fournisseur peut être seulement une connexion pour quelqu'un d'autre, la notion de la QoS de bout en bout est fortement liée à la celle de QoS d'une frontière à l'autre. La QoS atteinte de bout en bout dépend des connexions intermédiaires, avec leurs propres caractéristiques QoS. Les routes internes de chacun de ces réseaux, à leur tour, sont construites par des connexions qui peuvent être des réseaux, de nouveau avec leurs particularités QoS. La capacité de contrôler le comportement QoS d'un réseau dépend de la capacité de maîtriser à la fois les comportements des connexions et nœuds sur le parcours.

2.1.3 Approches d'étude

Notre travail porte, parmi d'autres, sur l'évaluation des performances des réseaux informatiques et des applications réseau. Il y a trois approches classiques pour l'évaluation de performances en général, et chacune d'entre elle est susceptible d'être utilisée suivant la nature du problème et les contraintes qui existent [Bou-04].

Une possibilité consiste à mesurer le système réel, tel qu'il est. Une exigence dans ce cas est que l'influence sur le système soit minime. Les mesures ne sont pas toujours possibles (par exemple si le système n'existe pas encore). Le nombre limité de scénarios sur une échelle de temps courte est un autre inconvénient.

La simulation avec événements discrets est une autre solution. Dans ce cas un modèle simplifié du système et de son environnement est implémenté comme logiciel. Le temps est simulé et « passe » avec quelques ordres de magnitude plus lentement que le temps réel. La performance est mesurée sur le système simulé, sans effets secondaires. Les problèmes proviennent de l'écart qui existe entre réalité et modèle.

L'approche analytique [All-90] implique l'utilisation d'un modèle mathématique sur lequel on fait une analyse numérique. Elle est vue parfois comme une forme de simulation, mais se situe à un niveau plus haut. Le désavantage est que parfois il faut

faire des hypothèses restrictives pour rendre le calcul tout simplement possible. Ceci conduit à un écart potentiellement encore plus grand avec la réalité.

Nous optons pour une approche hybride, celle de l'émulation. Dans ce cas on utilise un émulateur réseau pour reproduire les effets des réseaux, mais d'une façon contrôlable et dans une gamme suffisamment large de scénarios. Dans le même temps on utilise des applications réelles ce qui rend possible l'étude directe de leur performance.

2.1.4 Activités apparentées

Cette section regroupe et discute une série de références sur les activités apparentées à notre travail. Néanmoins, chaque fois que des précisions seront indispensables plus tard sur les spécificités de notre approche, nous fournirons les références nécessaires dans le contexte.

2.1.4.1 Etude de la QoS

Plusieurs projets ont vu le jour ces dernières années autour de l'étude de la QoS, spécialement liés à la différenciation des services. Le projet Quantum a étudié les mécanismes qui peuvent être employé dans des réseaux pour réaliser une différenciation des services [Fer-00]. Dans [ITU-123], UIT-T fait une propose une architecture QoS pour les réseaux IP d'accès à base d'Ethernet. SEQUIN a revu les questions de QoS [Cam-01] et a décrit une topologie pour un environnement de test de la QoS [Cam-02]. L'objective de TEQUILA est l'étude, l'implémentation et la validation des définitions des services réseau et outils d'ingénierie de trafic construites à base de DiffServ pour l'obtention des garanties QoS quantitatives de bout en bout [God-00], [Gri-00], [Man-01]. QBone spécifie et déploie le service « QBone premium », un service IP sur une ligne dédiée virtuelle, construit aussi à base de primitives d'expédition DiffServ [Tei-99]. Internet2 a lancé l'initiative de performance de bout en bout [Int2] ; l'objectif de cette étude est la création d'un environnement réseau prédictible et bien supporté. Tous ces projets sont centrés sur les mécanismes de QoS dans les réseaux, mais nous avons entrepris aussi l'évaluation des exigences des applications afin qu'on puisse répondre aux attentes des utilisateurs. Les mécanismes QoS peuvent être déployés pour essayer de satisfaire à ces exigences.

La relation entre conditions dans le réseau et performance des applications a été étudiée dans une série de projets. Le groupe de travail sur la QoS de Internet2 a publié une analyse sur les besoins des applications par rapport à la QoS, qui est assez vague [Mir-02]. TF-STREAM a tracé les lignes directrices pour le déploiement des applications multimédia en temps réel [Cav-02]. HEAnet a passé en revue certains aspects des l'évaluation quantitative de la qualité perçue pour applications [Reijs]. Généralement toutes ces approches sont qualitatives et imprécises – notre objectif est au contraire de créer une représentation quantitative de l'UPQ qui peut être mise en relation avec les paramètres QoS.

2.1.4.2 Différenciation de service

Actuellement, il manque un cadre standard pour tester les propriétés de différenciation de service dans les réseaux informatiques. La méthodologie de test pour les systèmes de commutation est précisée dans plusieurs RFC (Request For Comments) [RFC-1242], [RFC-1944], [RFC-2285], [RFC-2889]. Ces documents définissent les types de trafic à utiliser pour les tests de commutateurs : unidirectionnel ou bidirectionnel ; avec les ports du commutateur partiellement ou totalement interconnectés. Les paramètres à mesurer sont : capacité, écart entre paquets, délai, taux de perte, débit, taux d'expédition etc. Certains états prédéfinis doivent être testés : « comportement en surcharge », « comportement de redémarrage » « comportement à une seule trame », mais ils ne sont décrits qu'en termes flous (e.g. le « comportement en surcharge » apparaît quand la demande dépasse les ressources disponibles du système). Les tailles des paquets à utiliser sont : 64, 128, 256, 512, 1024, 1280 et 1518 octets. La méthodologie de test proposée par IETF précise aussi le format de présentation des résultats. Par contre, les RFCs ne définissent pas le cadre d'étude de la différenciation de trafic et les conséquences du déploiement des mécanismes QoS. Un fait regrettable, vu que les caractéristiques QoS sont un autre critère important pour comparer les commutateurs, en plus de la commutation *per se*.

Nous allons présenter des résultats et conclusions sur la performance des mécanismes QoS d'ordonnancement, tel qu'ils sont implémentés dans des commutateurs Gigabit Ethernet produits par les leaders du marché. Notre approche de test est complémentaire de celle d'IETF, car nous mettons en évidence les caractéristiques de livraison de QoS des commutateurs. Nous nous concentrons sur les propriétés

émergentes des mécanismes QoS, c.-à-d. leur comportement mesurable. Nous quantifions la dégradation induite sur le trafic différencié, car l'utilisation la plus efficace de ressources du réseau ne peut être obtenue qu'en faisant usage de ces mécanismes. Afin de pouvoir les exploiter pleinement, leur comportement doit être complètement analysé et compris. Quand on emploie des commutateurs pour construire des réseaux locaux spécialisés pour des applications dédiées, il est nécessaire de choisir les commutateurs qui ont la meilleure performance dans des conditions spécifiques.

Il y a plusieurs entreprises engagées dans l'évaluation de commutateurs et l'efficacité des mécanismes QoS dans divers domaines, comme Miercom [Mier] et The Tolly Group [Tolly]. Mais ils n'effectuent que des tests commerciaux, dont le but est de soutenir le marketing. Les tests évaluent seulement les propriétés indiquées par les fabricants et ils ne cherchent pas les éventuels défauts. Ce n'est donc qu'une évaluation incomplète qui est effectuée. En plus, les tests sur commutateurs consistent actuellement en majorité dans des tests d'endurance en conformité avec les RFCs (d'habitude, envoyer du trafic à débit maximum dans une configuration totalement interconnectée). Donc la différenciation de service pour les flux de trafic n'est pas mise en évidence. Nous essayons de comprendre les propriétés spécifiques émergentes des commutateurs lorsqu'ils sont en saturation. Par saturation nous entendons la situation particulière où l'utilisation d'une ressource approche ou dépasse sa capacité et, par conséquent, le service que le système réseau fournit connaît une dégradation sévère (par exemple le délai peut augmenter de quelques ordres de grandeur).

2.1.4.3 Mesure de paramètres QoS

Il y a plusieurs alternatives pour la mesure de paramètres QoS, à partir des applications simples, tel que *ping* et *traceroute*, qui existent dans tous les systèmes d'exploitation, à des applications plus complexes, tel que Iperf [Tirum]. NetIQ Chariot, un produit commercial, émule certains types de trafic et mesure le temps de réponse, le débit etc. Le projet Surveyor [Surv] utilise un système de positionnement global (Global Positioning System, GPS) pour effectuer des mesures de délai unidirectionnel (avec une précision de 20 μs [Kal-99]) et de pertes de paquets en accord avec la méthodologie des métriques de performance IP de IETF. Le service de

mesure avec trafic de test (Test Traffic Measuremetnt, TTM) de RIPE NCC est un système qui monitorise la connectivité d'un site avec d'autre parties de l'Internet [RIPE].

Nous avons construit notre propre système qui peut effectuer des mesures non intrusives des paramètres QoS. Le système est composé de produits commerciaux – diviseurs passifs, cartes réseau programmables – ainsi que des cartes horloge dédiées pour la synchronisation temporelle. Avec ces composants nous parvenons à une précision de 1 μs pour la mesure du délai. Un des avantages importants de notre système est son potentiel d'adaptation. Il peut être utilisé pour tester des éléments réseau et des petits réseaux locaux (LAN) ; il peut aussi tester des larges réseaux locaux, réseaux métropolitains (MAN) ou étendus (WAN), mais dans ce cas des cartes GPS doivent être utilisées pour avoir une référence de temps globale, comme décrit dans [Kor-03]. Nous pouvons mesurer le délai unidirectionnel, qui est plus pertinent que la simple mesure du temps d'aller retour (Round-Trip Time, RTT), vue l'asymétrie habituelle des réseaux. Nos mesures sur applications ne sont pas intrusives : après l'introduction des diviseurs dans le réseau, la circulation du trafic n'est pas perturbée et nous pouvons observer le comportement des applications réseau réelles. En plus notre système peut être reprogrammé pour répondre aux potentiels besoins futurs.

2.1.4.4 Performance des applications

La deuxième partie de notre travail concerne l'étude du comportement des applications et l'influence des conditions dans le réseau sur celui-ci. A noter que certaines applications peuvent être modifiées pour mieux se comporter dans les réseaux actuels. Par exemple, le TCP a été conçu quand les réseaux étaient relativement lents [Jac-88]. Les réseaux actuels ont une bande passante beaucoup plus large, donc de nouveaux mécanismes peuvent être utilisés pour l'augmentation de performance [FAST-02]. Par contre nous nous concentrons sur des applications standard, disponibles à tous les utilisateurs.

Dans [Cho-03] on trouve une étude sur l'utilisation en convergence des réseaux, dans une approche analytique. Mais cette démarche reste dans un plan trop abstrait. Nous considérons qu'il est nécessaire d'entrer dans plus de détails pour chaque application et évaluer objectivement ses performances.

Le transfert de fichiers par FTP ou HTTP est une application à base de TCP. Le comportement du TCP a été analysé par une multitude de chercheurs. Certains ont pris une approche analytique [Mat-97], [Pad-98], [Pad-00] créant des modèles qui expliquent les propriétés des transferts par TCP par méthodes numériques. Une autre voie est celle de la simulation, où des modèles sont étudiés à l'aide des outils informatiques [Fal-96], [Bre-00]. Il est aussi possible de faire des travaux expérimentaux sur des réseaux réels pour évaluer leur performance [Kor-03] ou pour collecter des traces de trafic [ITA]. Chacune de ces méthodes a certains avantages et désavantages liés à sa précision et la gamme de conditions qui sont analysées. L'émulation est une méthodologie hybride d'évaluation de la performance qui permet des expériences contrôlées avec des applications réelles. Nous considérons cette approche comme étant la plus adaptée et elle fait partie intégrante de notre approche d'étude de la QoS.

Il y a un certain nombre d'études sur les réseaux informatiques qui sont faites dans une perspective de performance de la VoIP. Dans [Jia-03] les auteurs évaluent la disponibilité des services VoIP qui est obtenue typiquement dans l'Internet ; la qualité même de la communication n'est pas prise en compte. Des mesures variées sur le trafic sont utilisées dans [Bou-02] pour quantifier l'impact des défaillances des connexions sur la performance de la VoIP. Ils utilisent une métrique objective de la qualité vocale perçue, le modèle E, dont les résultats n'ont pas une expression analytique standard complète. Les auteurs de [Mar-02] font aussi usage du modèle E, dans une version ajustée plus finement, pour évaluer la qualité de la VoIP à travers des réseaux fédérateurs. Les mesures sont effectuées dans des réseaux réels, elles ne fournissent donc pas une image complète sur le comportement de VoIP dans la gamme entière des conditions réseau possibles. [Con-02] utilise une combinaison de simulation et paquets réels pour arriver plus près de ce but, mais certains détails fins sont perdus, car ils apparaissent seulement avec l'utilisation des implémentations réelles de VoIP pour l'envoi et la réception des paquets. Pour évaluer la qualité de VoIP les auteurs utilisent une variante de PSQM.

2.1.4.5 Notre approche

Notre travail est une étude sur la qualité dans les réseaux informatiques en tant que fidélité des performances mesurables d'un système par rapport aux attentes. La

démarche est faite dans une nouvelle perspective sur la dégradation induite par le réseau et son influence sur les performances des applications.

Notre approche consiste dans la combinaison de quatre points principaux qui permettent de mettre en pratique cette étude de façon optimale:

- L'utilisation d'un système de conception propre pour effectuer la mesure des paramètres QoS dans le réseau avec une très grande précision ;
- L'emploi d'un émulateur réseau pour caractériser une large gamme de conditions réseau ;
- L'utilisation des applications courantes FTP et VoIP pour un maximum de réalisme ;
- L'emploi de métriques objectives pour l'évaluation de l'UPQ (telle que la plus récente métrique VoIP, le score PESQ de UIT-T [ITU-862]), afin d'obtenir des estimations précises de cet aspect de la qualité.

2.2 Paramètres QoS

Le concept de QoS est apparu quand on a fait l'observation que l'Internet traditionnel « meilleur effort » ne peut pas répondre aux exigences de certaines applications. Des solutions ont été proposées et elles sont lentement déployées. Pour déterminer leur influence sur la qualité d'un réseau il faut d'abord pouvoir mesurer ses caractéristiques QoS.

Le premier pas vers ce but est la définition des paramètres QoS qui doivent être mesurés. Etant donné que la plus grande partie des applications opère au niveau IP ou un niveau plus élevé, nous allons nous concentrer sur les paramètres à ce niveau en suivant les deux organismes qui régularisent ce domaine, l'Internet Engineering Task Force (IETF) et l'Union Internationale de Télécommunications (UIT) avec son secteur de standardisation dans le domaine de la télécommunication, UIT-T.

Toutes les demandes, actions et réponses QoS doivent être faites en terme de paramètres QoS. D'un côté, UIT-T adopte des définitions statistiques pour les paramètres QoS. Par contre l'IETF a une philosophie déterministe, avec l'argument que les définitions en terme de probabilité peuvent cacher des suppositions quant aux modèles stochastiques des comportements mesurés. Par la suite, il est indiqué de

fournir pour chaque expérience effectuée non seulement les paramètres mesurés, mais aussi la méthodologie exacte utilisée.

Il va de soi que la mesure des paramètres QoS n'est qu'une étape, et le plus important est de quantifier l'effet de la QoS sur les applications mêmes, sujet traité dans le chapitre 3. La corrélation entre les paramètres QoS et la qualité perçue pour une application particulière (voir le chapitre 5) est un outil pour pouvoir déterminer les configurations optimales des réseaux afin d'arriver à des fins précises par rapport à la qualité perçue.

2.2.1 Eléments de base

Chaque routeur est le plus petit point de convergence et divergence pour des dizaines, centaines et milliers de flux de paquets. Dans la majorité des réseaux de données, l'arrivée simultanée de salves de paquets en provenance de connexions différentes, qui sont tous destinés au même port de sortie (avec une capacité finie), met le routeur dans la situation d'avoir plus de paquets qu'il ne peut traiter immédiatement. Par exemple, le trafic convergeant des multiples connexions Ethernet à 100 Mbps dans un LAN peut facilement dépasser la capacité d'un circuit OC-3/STM-1 à 155 Mbps qui connecte le LAN avec un WAN. Pour faire face à une telle situation, tous les routeurs intègrent des mémoires tampon internes (sous forme de files d'attente) pour stocker l'excès de paquets jusqu'à l'instant où ils peuvent être expédiés. Dans ces circonstances, les paquets passant par le routeur vont subir des délais supplémentaires. On dit qu'un tel routeur subit une congestion temporaire. Le délai de bout en bout d'un paquet est la somme des délais de transmission à travers toutes les liaisons qui composent la route et les délais de traitement dans chaque routeur. Les délais intrinsèques des technologies de transmission, tel que les circuits SONET/SDH, les liaisons spécialisées (« leased line » en anglais) ou CBR (« Constant Bit Rate ») ATM sont assez prédictibles par conception. Par contre, les délais introduits par la mémoire tampon (induits par la congestion temporaire) dans chaque routeur ne sont pas tellement prédictibles. Ils varient avec la nature de la congestion et d'un moment à l'autre, même pour les paquets se dirigeant envers la même destination. Cette composante fluctuante du délai de bout en bout est appelée d'habitude gigue (« jitter » en anglais). Elle peut être assimilée à la variation du délai, ce qui fait qu'on va parfois

parler seulement de celui-ci, en sous-entendant la prise en compte de son évolution temporelle.

Une autre question est la perte de paquets. Etant donné que les routeurs ont seulement une mémoire tampon finie, une période de congestion prolongée peut conduire les files à atteindre leur capacité maximale. Quand des paquets arrivent et ne trouvent plus d'espace de stockage, ils doivent être rejetés jusqu'à la libération de l'espace de stockage. Le routeur traditionnel a effectivement une seule file d'attente pour chaque point interne de congestion et aucun mécanisme pour isoler les différentes classes ou types de trafic des effets de l'autre trafic traité. L'absence de prédictibilité du trafic sans rapport avec le trafic d'intérêt passant par les files d'attente de chaque point interne de congestion va probablement avoir une influence lourde sur le délai, la gigue et la perte de paquets de chaque flux de trafic. Certains types de trafic (e.g. connexions TCP transportant des courriers électroniques) tolèrent le délai mieux que la perte de paquets, suggérant que les files longues sont idéales. Inversement, d'autres types de trafic (comme le UDP transportant des flux vidéo ou audio) préfèrent que les paquets soient rejetés au lieu d'être gardés trop longtemps dans une file, indiquant que les files courtes sont à préférer.

Tout paquet arrivant pour trouver une file partiellement pleine subit un délai supplémentaire car il doit attendre que les paquets qui l'ont précédé soient envoyés. Si un paquet arrive quand la file est pleine il ne peut pas être gardé et doit être rejeté. La gigue provient du fait que le trafic réseau est en salves et non corrélé. Dans un environnement typique IP, les paquets n'ont pas une taille fixe, ajoutant ainsi une variabilité encore plus grande à la relation entre la bande passante de la connexion de sortie, le nombre des paquets dans la file d'attente et les délais de ces paquets. Le délai peut aussi être une conséquence de la technologie du réseau, comme c'est le cas du schéma de recul (« back-off » en anglais) utilisé pour le Ethernet half-duplex. Ce schéma se traduit tout simplement comme un manque de prédictibilité temporelle de la liaison.

Un autre paramètre QoS est le débit, qui est une mesure de la quantité de données transférée par un réseau. Il est d'habitude exprimé en mégabits par seconde (Mbps) et dépend d'une part de la capacité de la liaison et d'autre part, plus importante encore, de l'interaction du flux de trafic d'intérêt avec les autre flux de trafic avec lesquels il partage la connexion.

2.2.1.1 Deux degrés de liberté

Les trois paramètres mentionnés ne sont néanmoins pas indépendants et dans tout réseau il existe une relation entre la perte de paquets, le débit et le délai [Dav-99]. Une capacité illimitée des files d'attente crée la possibilité d'avoir une perte de paquets nulle. Les files infinies sont malgré tout une notion purement théorique (et peuvent conduire à un délai infini). En conséquence, dans tout système de files d'attente finies, la perte de paquets va arriver occasionnellement. En général, permettre plus de pertes et du délai dans un système, de manière contrôlée, peut conduire à une augmentation de la quantité de trafic transportée.

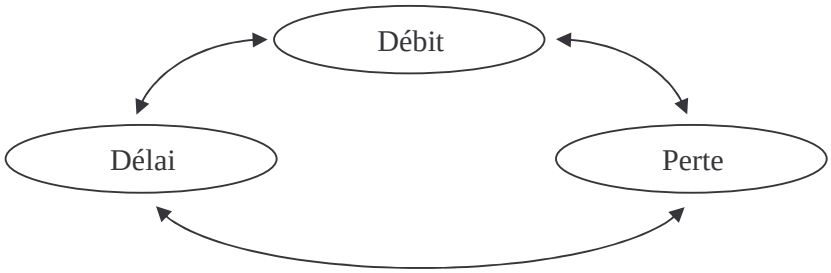


Figure 2 : Les deux degrés de liberté de la relation entre délai, perte et débit.

Le contrôle peut se réaliser par l'intermédiaire des deux degrés de liberté de la relation entre débit, délai et perte (voir la figure 2). Fixer un de ces paramètres crée une dépendance entre les deux autres. Ainsi, pour un taux de perte fixe, réduire le délai implique que le débit doit diminuer. Pour un débit fixe, réduire le délai conduit à une croissance du taux de perte. Pour un délai fixe réduire le taux de perte diminue le débit. La relation mentionnée donne aussi des informations pour la gestion à long terme du réseau, car elle montre quand les exigences par rapport à la perte et au délai ne peuvent pas être satisfaites. A présent, la gestion des réseaux se fait très souvent en fonction seulement des exigences de débit, et parfois sa relation avec le délai. Mais considérer le taux de perte et son contrôle comme des parties importantes du processus de décision permet une gestion plus flexible et plus efficace.

Un autre aspect très important lié à la QoS et ses paramètres mesurables est le fait qu'il existe une loi de conservation en ce qui les concerne. Prenons le cas d'un système avec une seule file d'attente. Le délai que ce système introduit dépend seulement de caractéristiques globales du trafic à l'entrée et à la sortie. Le fait qu'il s'agit d'un seul flux de données ou de plusieurs importe peu. La même chose est valable pour la perte de paquets et pour le débit. Ce que les mécanismes QoS peuvent

faire est de contrôler le partage de la dégradation totale, de manière différente, entre les flux de trafic. Ceci conduit à des délais et taux de perte différents au niveau des flux, mais toujours en observant la loi de conservation de la dégradation totale.

Le délai, la perte et le débit caractérisent de manière intuitive un réseau, mais il est impératif de définir en termes plus précis ce que sont les paramètres QoS. [ITU-350] identifie trois étapes distinctes de l'acte de communication : l'accès aux ressources, le transfert d'information utile et la libération des ressources. Trois critères de performance sont aussi définis : la vitesse, l'exactitude et la fiabilité. Une matrice 3 par 3 est utilisée pour organiser les paramètres de performance. Ils peuvent être primaires (tel que le délai) ou dérivés (tel que la disponibilité du système). Les paramètres dérivés sont obtenus par l'application des seuils sur les paramètres primaires. Neuf paramètres de performance peuvent être définis par la matrice mentionnée, mais nous allons nous concentrer sur ceux liés au transfert d'information, qui sont présentés dans le tableau 1 avec les paramètres QoS primaires correspondants.

Critères de performance	***Paramètres QoS***
Vitesse	Délai, débit
Exactitude	Probabilité d'erreur
Fiabilité	Probabilité de perte

Tableau 1: Les critères de performance et les paramètres QoS relatifs au transfert d'information conformément à l'UIT-T I.350.

2.2.2 Paramètres QoS statistiques : I.380

Dans [ITU-380] la même matrice que celle définie dans [ITU-350] est utilisée. Les paramètres sont particularisés pour les réseaux IP, tenant compte de leur service en mode datagramme ainsi que la possibilité de fragmentation des paquets. Les résultats possibles pour le transfert d'un paquet IP sont :

a) *Transféré avec succès* – le paquet IP arrive à destination dans un intervalle $T_{\max}$ avec un en-tête valide et sans erreurs dans le contenu. Ceci doit être le cas pour tous les fragments du paquet IP, s'il en a.

b) *En erreur* – le paquet arrive à la destination dans un intervalle $T_{\max}$ avec un ou plusieurs champs d'en-tête corrompus ou avec un contenu incorrect. Notons que les paquets avec en-têtes qui entraînent le rejet ou l'arrivage à une autre destination sont considérés comme perdus.

c) *Perdu* – le paquet IP n'est jamais délivré à la destination ou délivré après un intervalle $T_{\max}$. Si un fragment d'un paquet IP est perdu, le paquet IP entier est considéré comme perdu.

d) *Faux* (« spurious » en anglais) – un paquet IP qui crée un événement de réception auquel aucun événement de transmission ne correspond.

Le service en mode datagramme du réseau IP implique les faits suivants :

- Un paquet IP peut quitter une section de réseau vers n'importe quel point de sortie ;
- Un paquet IP peut être fragmenté et les fragments peuvent suivre des routes différentes ;
- Un paquet IP ou un fragment peut être routé vers une section de réseau qu'il a quitté à cause d'un changement dans les tableaux de routage.

2.2.2.1 Paramètres de performance UIT-T I.380

Les paramètres de performance associés au transfert des paquets IP, en accord avec UIT-T sont présentés par la suite.

2.2.2.1.1 Délai de transfert (IPTD)

Le délai de transfert du paquet IP (IP Packet Transfer Delay, IPTD) est défini pour tous les résultats avec succès ou en erreur comme la différence entre le moment de réception et celui de transmission du paquet et doit être inférieur à $T_{\max}$. Dans le cas de fragmentation, $t_{reception}$ correspond au dernier fragment réceptionné.

$$IPTD = t_{reception} - t_{transmission}, \qquad (2.1)$$

où $t_{reception} > t_{transmission}, t_{reception} - t_{transmission} < T_{\max}$. Le IPTD moyen est la moyenne arithmétique des IPTD pour une population d'intérêt. D'autres métriques statistiques peuvent être utilisées aussi, telles que les centiles (99,5% par exemple).

2.2.2.1.2 Variation du délai (IPDV)

La variation du délai (IP Packet Delay Variation, IPDV) pour un paquet IP est la différence entre son IPTD et un IPTD de référence. L'IPTD de référence peut être, par exemple, le IPTD moyen ou le IPTD du premier paquet.

$$IPTV = IPTD - IPTD_{reference} \qquad (2.2)$$

2.2.2.1.3 Taux d'erreur (IPER)

Le taux d'erreur des paquets IP (IP Packet Error Ratio, IPER) est le taux de nombre total des paquets en erreur (N_{en_erreur}) par rapport au nombre total de paquets transférés avec succès (N_{avec_succes}) plus le nombre des paquets en erreur.

$$IPER = \frac{N_{en\_erreur}}{N_{avec\_succes} + N_{en\_erreur}} \qquad (2.3)$$

2.2.2.1.4 Taux de perte (IPLR)

Le taux de perte des paquets IP (IP Packet Loss Ratio, IPLR) est le taux de nombre total des paquets perdus (N_{perdus}) par rapport au nombre total des paquets transmis ($N_{transmis}$).

$$IPLR = \frac{N_{perdus}}{N_{transmis}} \qquad (2.4)$$

Le service IP est défini comme disponible si IPLR ne dépasse pas un certain seuil c_1 pendant un intervalle T_{av} (les valeurs suggérées sont 0,75 pour c_1 et 5 minutes pour T_{av}). Les objectifs de performance pour IPLR ne doivent pas être évalués pour les périodes d'indisponibilité.

2.2.2.1.5 Taux des paquets faux (SIPR)

Le taux des paquets IP faux (Spurious IP Packet Rate, SIPR) est le nombre total de paquets IP faux observés dans un intervalle de temps ΔT divisé par ΔT .

$$SIPR = \frac{N_{faux}}{\Delta T} \qquad (2.5)$$

2.2.2.1.6 Le débit (IPPT)

Le débit des paquets IP (IP Packet Throughput, IPPT) est le nombre total de paquets IP transférés avec succès dans un intervalle de temps ΔT divisé par ΔT .

$$IPLR = \frac{N_{avec\_succes}}{\Delta T} \qquad (2.6)$$

Le débit des paquets IP en octets (Octet based IP Packet Throughput, IPOT) est le nombre total d'octets transmis dans des paquets IP transférés avec succès dans un intervalle de temps ΔT divisé par ΔT .

2.2.3 Paramètres QoS déterministes : IPPM

Le but de IETF avec les métriques IPPM (IP Performance Metrics) [RFC-2330] est le développement d'un ensemble des métriques standard qui fournissent des mesures quantitatives de la qualité, performance et fiabilité des services de livraison de données de l'Internet. Ils vont fournir aux utilisateurs finaux, aux opérateurs réseau et aux fournisseurs des services de l'Internet une compréhension commune et une mesure précise de la performance et fiabilité des routes unidirectionnelles de bout en bout et des sous-réseaux IP. Le projet Surveyor [Surv] est une infrastructure de mesure qui peut déjà être utilisée pour la mesure du délai et perte de paquets en conformité avec IPPM.

2.2.3.1 Le cadre de IPPM

Dès le début IPPM définit les points qui doivent être analysés pour chaque paramètre QoS à étudier [RFC-2330].

2.2.3.1.1 Définir une métrique

La métrique est une quantité spécifiée, liée à la performance et fiabilité de l'Internet et précisée dans des unités de mesure standard (e.g. le bit pour l'information). Les métriques sont à définir dans des termes déterministes plutôt que probabilistes. Les définitions dans des termes probabilistes peuvent cacher des suppositions quant au modèle stochastique du comportement mesuré (e.g. perte de paquets corrélée).

La définition d'une métrique s'applique aux paquets standard, à moins que le contraire soit explicitement spécifié. Les paquets doivent avoir la taille précisée dans leur en-tête, qui inclue l'en-tête et la charge utile, un en-tête valide avec une somme

de contrôle correcte. Le paquet ne doit pas être un fragment IP et ne doit pas contenir des options IP. Son en-tête de transport, s'il en a un, doit être valide.

La valeur de la métrique dépend du type de paquet IP utilisé pour effectuer la mesure. On se réfère à lui comme « paquet de type P », type qui doit être précisé explicitement[1].

2.2.3.1.2 Type de métriques

Les métriques se regroupent dans deux catégories :

- Métriques analytiques – vu le riche cadre analytique existant, il est important de générer des caractérisations du réseau qui sont en accord avec ce cadre et les configurations pratiques à la fois. Ceci permet aux études non empiriques de corréler et comprendre le comportement des réseaux réels en sélectionnant les propriétés des composants du réseau pertinentes pour une métrique analytique donnée (e.g. taux de transmission dans un routeur modélisé avec une file d'attente).
- Métriques empiriques – quand une métrique pertinente n'entre pas dans le cadre analytique à cause du fait qu'elle ne modélise pas convenablement le système réel, une métrique empirique peut être définie à sa place, accompagnée par une méthodologie de mesure de référence (e.g. le meilleur débit réalisable au long d'une route en utilisant un protocole de type TCP conforme au RFC 2001).

2.2.3.1.3 Composition de métriques

Il y a deux formes de composition des métriques, ce qui permet de réaliser des extrapolations :

- Composition spatiale – une métrique peut être appliquée à une route IP complète et/ou aux routes qui la composent (e.g. le délai total comparé aux délais sur certaines connexions).

---

[1] Les paquets de test utilisés par Surveyor [Surv] pour mesurer le délai et la perte sont des paquets UDP de 12 octets avec un numéro de séquence et un timbre-à-date.

- Composition temporelle – une métrique peut être appliquée au moment t_i et/ou aux moments t_j (e.g. le débit durant cinq minutes comparé au débit pendant cinq autres minutes).

2.2.3.2 Instance des métriques

Les métriques peuvent être classifiées en fonction du nombre d'instances utilisées, c.-à-d. le nombre de fois qu'une mesure est effectuée. A noter que la quantité étant mesurée peut impliquer elle-même un certain nombre d'événements.

2.2.3.2.1 Métrique singleton

Une métrique singleton est une métrique pour laquelle une seule instance de la mesure a été utilisée.

2.2.3.2.2 Métrique échantillon

Une métrique échantillon implique un nombre d'instances distinctes de métriques singleton. Les échantillons donnent une indication sur les variations et la cohérence d'une métrique. Les échantillons peuvent être collectés au moins de trois façons :

- Echantillonnage périodique, c.-à-d. à des intervalles de temps fixes. Ce type d'échantillonnage est simple, mais présente trois inconvénients : (i) la métrique elle-même peut présenter une périodicité ; (ii) il est facile à anticiper ; (iii) il peut amener le réseau dans un état de synchronisation.
- Echantillonnage aléatoire additif, c.-à-d. à des intervalles de temps aléatoires en accord avec un distribution $G(t)$. Il évite la synchronisation, mais peut compliquer l'analyse en fréquence (la Transformée de Fourier Discrète suppose en général un échantillonnage périodique) et peut être prédictible, sauf si $G(t)$ est une distribution exponentielle négative.
- Echantillonnage poissonnien, c.-à-d. un échantillonnage à des intervalles de temps aléatoires conformément à une distribution $G(t)$ exponentielle négative. Dans ce cas l'échantillonnage est non biaisé, et aussi asymptotiquement non biaisé, même si l'état du réseau ne l'est pas. Par contre cette distribution est non bornée et peut produire des intervalles très longs – dans ce cas elle peut être remplacée par un échantillonnage uniforme. La valeur du taux moyen d'échantillonnage n'est pas contrainte, sauf aux extrémités : un taux trop grand

va perturber le réseau, un taux trop petit peut faire qu'on rate un comportement intéressant[1].

2.2.3.2.3 Métrique statistique

Une métrique statistique est dérivée d'une métrique d'échantillonnage par le calcul de statistiques sur les valeurs définies par la métrique singleton échantillonné (e.g. la moyenne).

2.2.3.2.4 Distribution statistique

Les distributions statistiques, ainsi que les centiles, sont une autre méthode de description des échantillons. Elles sont réalisées par le calcul de la fonction de distribution (cumulative) empirique (FDE)[2]. Elle est préférée aux histogrammes, qui ne donnent aucune indication relative aux moments de temps auxquels les événements se sont produits.

2.2.3.3 Les métriques IPPM

Dans ce qui suit, les principales métriques IPPM sont présentées, en conformité avec les RFCs correspondants.

2.2.3.3.1 Délai unidirectionnel

Le délai unidirectionnel [RFC-2679] est préféré au temps d'aller-retour à cause de l'asymétrie qui peut exister dans les caractéristiques des routes (même pour celles symétriques) aux niveaux de l'ordonnancement, de la fourniture de ressources et de la réservation. Il est défini comme la différence entre le temps de réception du dernier bit d'un paquet et le temps de transmission de son premier bit.

$$delai = t_{reception} - t_{transmission} \qquad (2.7)$$

---

[1] Un flux de trafic de type Poisson avec un λ de 4 s^{-1} est utilisé dans Surveyor pour mesurer les pertes et le délai [Surv].

[2] Considérons un nombre N d'observations, $X_1 \ldots X_n$. $FDE(x)$ est $n(x)/N$, ou $n(x)$ est le nombre de points X_i inférieurs à x. Un centile est la plus petite valeur x pour laquelle $FDE(x)$ est inférieure ou égale à un pourcentage donné. Pour un N impair, le centile 50% équivaut à la médiane.

La méthodologie doit préciser depuis quelle limite supérieure de temps le paquet est considéré comme perdu[1]. A noter que, si le paquet est fragmenté et l'assemblage n'a pas lieu, le paquet entier est considéré comme perdu. Le délai unidirectionnel est mesuré par l'intermédiaire des timbres-à-date. Les statistiques suggérées sont les centiles, la médiane, le minimum et le centile inverse. Le délai moyen donne aussi une indication sur le comportement du système en état d'équilibre.

2.2.3.3.2 Délai bidirectionnel

Le délai bidirectionnel [RFC-2681] est l'écart entre le temps d'apparition du premier bit d'un paquet « requête » au transmetteur et le temps de réception du dernier bit du paquet « réponse » au transmetteur, considérant que la destination, à la réception du paquet « requête », envoie le paquet « réponse » sans aucun délai. Il est équivalent au RTT.

2.2.3.3.3 Variation du délai (ipdv)

La variation du délai des paquets IP (IP Packet Delay Variation, *ipdv*) [RFC-3393] pour deux paquets est défini comme la différence entre les délais unidirectionnels de deux paquets, choisis parmi les paquets du flux de trafic par une fonction de sélection (e.g. paquets consécutifs, avec des indices prédéterminés, avec les valeurs maximale et minimale pour le délai etc.).

$$ipdv = D_m - D_n \qquad (2.8)$$

Le terme « gigue » est utilisé dans ce contexte pour la valeur absolue d'un *ipdv* calculé pour des paquets consécutifs. C'est la définition que nous préférons aussi, et dans ce cas la gigue moyenne, *G*, devient :

$$G = \frac{1}{N-1}\sum_{i=2}^{N}\left|D_i - D_{i-1}\right|, \qquad (2.9)$$

où N est le nombre total des paquets VoIP et D_i et le délai unidirectionnel du paquet d'indice i, $i = \overline{1,N}$.

[1] Pour Surveyor une valeur de 10 secondes est utilisée [Surv].

2.2.3.3.4 Perte unidirectionnelle

La perte des paquets unidirectionnelle [RFC-2680] est définie comme zéro quand un paquet est reçu dans un délai unidirectionnel fini. La méthodologie doit distinguer entre un paquet perdu et un délai large (mais fini) en sélectionnant un seuil « raisonnable[1] » pour le délai unidirectionnel. Les paquets corrompus, même s'ils sont reçus, ou les paquets fragmentés pour lesquels l'assemblage n'a pas eu lieu sont considérés comme perdus.

Le taux de perte ne donne aucune indication sur la répartition en temps des événements. Par exemple la transmission TCP a lieu en salve et induit la perte des paquets, s'adaptant ultérieurement en fonction de cette perte.

Même si la perte moyenne est mentionnée dans [RFC-2680], IETF préfère une approche plus déterministe et [RFC-3357] définit des motifs (« patterns » en anglais) de perte. Une perte en salve signifie la perte de paquets consécutifs dans un flux. La distance de perte est la différence entre les numéros de séquence des deux paquets perdus, séparés ou non par des paquets réceptionnés. Une période de perte est la période qui commence avec la perte d'un paquet (le paquet précédent étant reçu) et finit avec le dernier paquet perdu (le paquet suivant étant réceptionné). Des mesures dérivées possibles sont : la perte observable (quand la distance de perte est inférieure à un certain seuil), le total des périodes de perte, la longueur de la période de perte, la longueur des périodes entre pertes.

2.2.3.3.5 Capacité de transfert massif

La capacité de transfert massif (« bulk » en anglais) [RFC-3148] est une mesure de la capacité du réseau à transférer des quantités significatives de données par une seule connexion de type TCP. Elle est définie comme le rapport entre le nombre de bits de données transférés effectivement (c.-à-d. sans les bits des en-têtes ou ceux retransmis) et le temps écoulé pour le transfert respectif.

$$BTC = \frac{quantite\_de\_donnees}{temps\_de\_transfert} \qquad (2.10)$$

---

[1] Dans ce contexte raisonnable signifie convenable pour le système étudié. Par exemple pour un commutateur le délai maximum possible est d'ordre de millisecondes, tandis que pour un réseau WAN il peut être de l'ordre de secondes.

D’autres métriques incluent la connectivité [RFC-2678] et la mesure de la performance du réseau pour des flux périodiques, tel que ceux produits par des applications de diffusion en continu [Rai-00].

2.2.3.4 Discussion

On peut remarquer que IPPM de IETF et I.380 de UIT-T sont très similaires, mis à part le fait que les paquets en erreur sont distingués de ceux qui sont perdus par I.380, ce qui peut être utile pour les applications qui font la différence entre les deux cas, comme la téléphonie IP. Mais UIT-T n’offre pas des informations si précises relativement aux méthodes d’échantillonnage.

Le travail continu de UIT-T sur la définition de la QoS pour IP est une combinaison de son point de vue sur la QoS dans ATM et IntServ. La convergence des deux méthodes est achevée par la quantification des limites sur les paramètres QoS qui doivent être obtenus par différents services d’expédition des paquets. Chaque classe de QoS crée une combinaison spécifique de limites sur les valeurs de performance (voir la section 3.2). A noter que les objectifs de performance sont donnés en termes statistiques. Par exemple, l’objectif du délai de transfert d’un paquet est une limite supérieure sur le IPTD moyen – la conséquence est que certains paquets auront en effet des temps de transfert supérieurs à cette valeur.

2.2.4 Mesure des paramètres QoS

L’activité de IETF sur la mesure de la QoS est très complète et concrète et les résultats sont déjà appliqués et utilisés dans une série de projets, parmi lesquels Surveyor [Surv]. Par comparaison, le travail de UIT-T semble plus théorique et moins précis.

2.2.4.1 Le cadre IETF

IPPM définit très clairement toutes les notions liées à la mesure de la QoS.

2.2.4.1.1 Méthodologies de mesure

Au moins quatre méthodologies de mesure sont identifiées (à noter que toutes les méthodes peuvent introduire un parti pris en fonction des propriétés spécifiques de chacune d’entre elles):

- La mesure directe d'une métrique en utilisant du trafic de test injecté dans le réseau;
- La projection d'une métrique à base de mesures de niveaux plus bas ;
- L'estimation d'une métrique constituée par plusieurs mesures agrégées ;
- L'estimation d'une métrique au temps *t'* à base de métriques observées au temps *t.*

Les propriétés de toute méthodologie sont : (i) la répétabilité ; (ii) la continuité ; (iii) la cohérence avec elle-même ; (iv) la qualité de l'approximation (« goodness of fit » en anglais). La continuité est définie comme suit : une petite variation dans les conditions du test résulte dans une petite variation des résultats de la mesure. La métrique est appelée continue dans ce cas. La qualité de l'approximation d'une distribution peut se mesurer, par exemple, par le test Anderson-Darling [RFC-2330], qui indique si un échantillon de données provient d'une population ayant une distribution précisée.

Pour chaque méthodologie, les incertitudes et erreurs doivent être minimisées, documentées et quantifiées.

2.2.4.1.2 Problèmes liés au temps

Pour une horloge on peut déterminer son écart (« offset » en anglais), asymétrie (« skew » en anglais), dérive (« drift » en anglais) et précision. Deux horloges peuvent être comparées du point de vue de leur écart ou asymétrie relatifs.

Les mesures dans l'Internet sont souvent effectuées par des ordinateurs hôtes qui introduisent leurs propres effets matériels dans la mesure et produisent ce qu'on appelle le temps hôte. Pour un paquet P, à l'ordinateur hôte H et une connexion C on définit le temps d'arrivée sur la liaison de P à H sur C comme le moment auquel le premier bit de P est apparu au point de mesure de H sur C. Le temps de sortie de P à H sur C est le moment auquel tous les bits de P sont passés au point de mesure du H sur C.

2.2.4.2 Le cadre UIT-T

La méthodologie de mesure de la performance est assez imprécise dans [I.380] et toujours en cours d'étude. En accord avec ce document, les conditions suivantes doivent être documentées pour chaque test :

- Les sections du réseau testées ;
- Le temps de mesure et le nombre d'échantillons ;
- Les caractéristiques exactes du trafic de test, incluant la taille des paquets IP et le motif de trafic ;
- Le type de mesure : pendant ou en dehors d'un régime opératoire normal, active ou passive etc. ;
- Le sommaire des mesures effectuées, incluant les moyennes, les pires cas et les centiles empiriques.

2.2.4.3 Autres projets

Parmi les autres projets de mesure, le TTM (Test Traffic Measurements) de RIPE (Réseau IP Européen) [RIPE] est l'équivalent européen de Surveyor. D'autres projets, tel que QBone [QBone] aux Etats-Unis et TF-TANT [TFT] en Europe se concentrent sur la mesure de DiffServ par l'intermédiaire des paramètres de performance sous-jacents.

2.3 Comportement prédictible au niveau des sauts

Un environnement qui prend en compte la QoS a pour objectif de rendre possible la livraison de services prédictibles à certaines classes ou types de trafic, sans tenir compte du reste du trafic traversant le réseau considéré. Ceci signifie la création d'une solution IP multiservice dans laquelle le trafic traditionnel en salve peut partager la même infrastructure (routeurs, commutateurs, connexions) que le trafic pour lequel des demandes plus rigoureuses sont imposées, en termes de délai, gigue, bande passante et perte de paquets. Indépendamment du fait qu'on se concentre sur le réseau d'une entreprise, d'accès ou fédérateur (ou une quelconque combinaison d'entre eux), la route de bout en bout suivie par les paquets appartenant à un seul utilisateur est tout simplement une séquence des connexions et routeurs. Ainsi l'attention doit être portée au début vers la dynamique du comportement d'expédition de chaque routeur. Malgré

le fait que le modèle de routeur traditionnel est centré sur la destination des paquets[1], les routeurs des réseaux IP tenant compte de la QoS doivent permettre également le contrôle de l'instant auquel les paquets sont expédiés.

2.3.1 Classification, gestion et ordonnancement

Les caractéristiques de délai, débit et perte de paquets de chaque réseau IP sont finalement déterminées par les propriétés QoS des connexions et la dynamique de l'utilisation et gestion des files d'attente à l'intérieur de chaque routeur. Si la charge du réseau dépasse le taux de service, une seule file en chaque point interne de congestion ne suffit plus. Dans ce cas on a besoin d'une file d'attente pour chaque classe de trafic pour laquelle des caractéristiques indépendantes de délai, débit et taux de perte sont requis. Chacune de ces files doit avoir sa propre politique de rejet de paquets (par exemple des seuils en dessus desquels les paquets sont rejetés de façon aléatoire). Les files multiples pour chaque interface de sortie sont cependant inutiles sans un mécanisme pour diriger les paquets vers les files appropriées. Une méthode de classification est nécessaire en plus de la simple tâche consistant à trouver le saut prochain des routeurs traditionnels. Finalement les files doivent toutes partager la capacité finie de la connexion de sortie qu'elles servent. Ce besoin implique l'ajout d'un mécanisme d'ordonnancement pour le mixage des paquets de chaque file, et ainsi accorder l'accès à la connexion d'une manière contrôlable et prédictible. Les exigences précédentes peuvent être résumées dans l'affirmation que les réseaux tenant compte de QoS demandent des routeurs qui peuvent classifier, gérer et ordonnancer (CGO) de manière différente tous les types de trafic selon leurs besoins [Arm-00] (voir la figure 3). On dit que de tels routeurs ont une architecture CGO.

---

[1] Dans les routeurs, les décisions d'expédition de paquets sont basées sur l'adresse destination de chaque paquet et des tableaux d'expédition locaux.

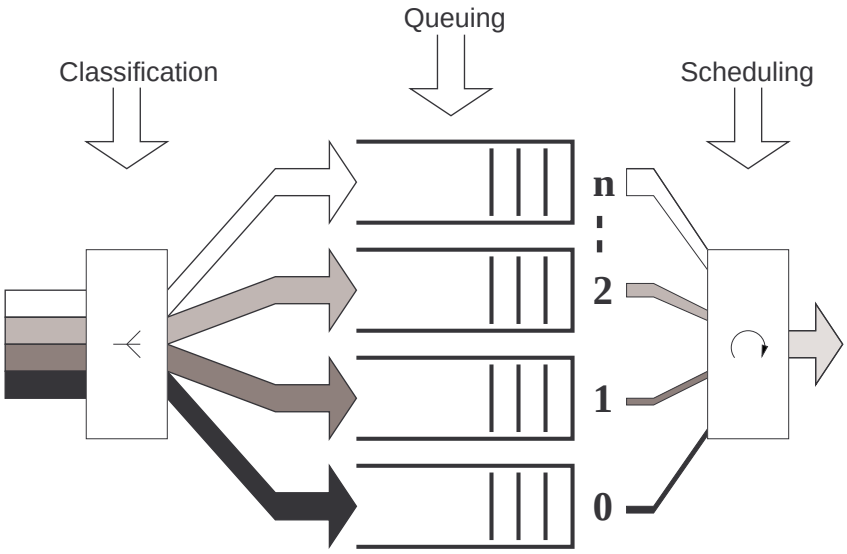


Figure 3 : L'architecture COS d'un routeur.

Pour chacune de ces fonctions il existe une pléthore d'algorithmes, chacun avec ses avantages et inconvénients. De plus, les implémentations dans les routeurs peuvent s'éloigner de la version théorique, par exemple pour des raisons de complexité de calcul, en conduisant à une déviation des caractéristiques et garanties par rapport aux résultats théoriques. Ce fait est la principale raison des tests actifs sur éléments de réseau que nous avons effectué et qui sont présentés dans le chapitre 4.

2.3.1.1 Classification

Les routeurs classifient les paquets pour déterminer le flux auquel ils appartiennent et pour décider quel est le niveau de service qu'ils vont recevoir. La classification peut se faire sur un nombre arbitraire de champs dans l'en-tête du paquet. Effectuer une classification rapide dans de telles conditions est une tâche reconnue comme difficile, et la performance dans le pire cas est faible. En dépit du fait que la classification est une fonction normale de tout routeur, sa performance peut avoir un impact plus important sur les routeurs QoS, à cause de la complexité potentielle plus grande des critères de classification dans ce cas.

La forme de classification la plus connue est celle qui est utilisée pour router les paquets IP. Il y a plusieurs autres services réseau qui nécessitent la classification des paquets, tel que le contrôle de l'accès dans des pare-feu, le routage à base de priorité, la fourniture de services différenciés ou la facturation du trafic.

Dans chaque cas il faut déterminer la classe à laquelle appartient un paquet qui arrive afin de décider le type de service qu'il va recevoir. La fonction de catégorisation est effectuée par un *classificateur de flux* (appelé aussi classificateur de paquets) qui maintient un ensemble de règles de classification. Le processus est basé sur le contenu des en-têtes des paquets.

Une taxonomie des algorithmes de classification est fournie ci-après ; pour plus de détails voir [Gup-01] :

1) Algorithmes avec structures de données élémentaires
    - Recherche linéaire ;
    - « Tries[1] » hiérarchiques ;
    - Elaguer ensembles de « tries » ;
2) Algorithmes géométriques
    - Maille de « tries »
    - Produit vectoriel;
    - Schémas de classification 2D
    - Arbre quaternaire basé sur des régions ;
    - Arbre inversé des segments ;
3) Algorithmes heuristiques
    - Classification récursive des flux ;
    - Segmentation hiérarchique intelligente ;
    - Recherche dans l'espace de « tuples[2] » ;
4) Algorithmes au niveau matériel

---

[1] Un « tries » est une structure de données de type arbre pour le stockage des chaînes, avec un sommet (nœud) pour chaque préfixe commun. Les chaînes elles-mêmes sont stockées dans les feuilles (nœuds terminaux).

[2] Un « tuple » est une structure de données qui contient plusieurs champs vus de manière unitaire, comme un seul enregistrement.

- Mémoires ternaires adressables par contenu ;
- Intersection des représentations binaires.

Suite à la classification, les paquets sont placés dans des files d'attente. Le routeur tenant compte de la QoS doit avoir une file d'attente pour chaque classe de priorité et gérer les paquets de manière adéquate.

2.3.1.2 Gestion

Des mécanismes de gestion liés à la mémoire tampon doivent être mis au point pour décider de la gestion des paquets placés dans des files d'attente. C'est la façon de contrôler la perte de paquets, avec une influence considérable sur les caractéristiques QoS globales. Les mécanismes de gestion active sont concentrés aussi sur le contrôle de la congestion, une cause importante de la dégradation de la QoS.

La gestion active contrôle le nombre de paquets dans une file d'attente en décidant quand et quels paquets sont rejetés lorsqu'une file éprouve la congestion passagère [RFC-2309]. Par conséquent elle permet le contrôle de l'accès des flux d'une certaine classe de service à une ressource limitée du routeur : la mémoire tampon. Les techniques de gestion de files d'attente les plus utilisées sont présentées par la suite.

2.3.1.2.1 Rejet du dernier paquet

Le rejet du dernier paquet (« tail drop ») est en effet l'équivalent d'une gestion passive de la mémoire tampon. Un paquet arrivant dans une file dont les ressources sont complètement épuisées est rejeté. Cette méthode est très facile à implémenter, mais elle a des caractéristiques faibles par rapport à l'absorption des salves de trafic.

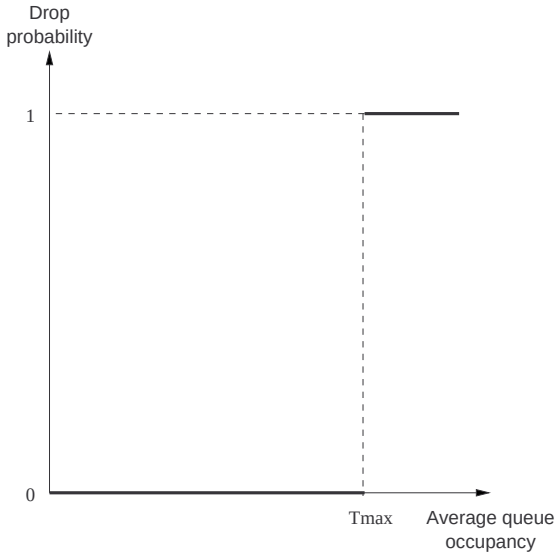


Figure 4 : Profil de perte pour le rejet du dernier paquet.

2.3.1.2.2 Détection aléatoire avancée (RED)

La détection aléatoire avancée (Random Early Detection, RED) est une technique de gestion active déployée actuellement dans les grands réseaux IP. RED utilise un profil de perte de paquets pour contrôler l'agressivité du procès de rejet des paquets. Ce profil définit des intervalles de probabilité de perte à travers l'espace d'occupation de la file ; la probabilité de perte augmente généralement avec l'occupation de la file une fois qu'un certain seuil a été dépassé. La configuration de cette technique pour atteindre une performance prédictible peut s'avérer difficile, mais une fois qu'elle est bien configurée, cette méthode fournit de bonnes caractéristiques de délai et performance du TCP.

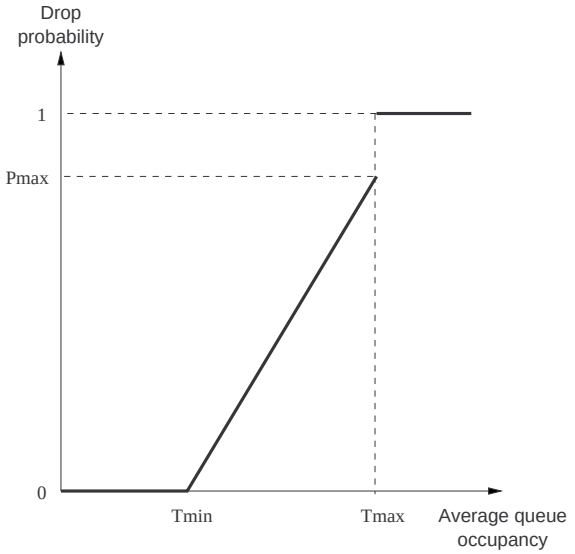


Figure 5 : Profil de perte pour RED.

2.3.1.2.3 Détection aléatoire avancée (WRED)

La détection aléatoire avancée avec poids (Weighted Random Early Detection, WRED) est une extension du RED qui permet d'assigner des profils de perte RED différents pour différents types de trafic. Elle offre ainsi une granularité accrue du contrôle du fait que certaines classes de trafic sont traitées de manière moins agressive que d'autres.

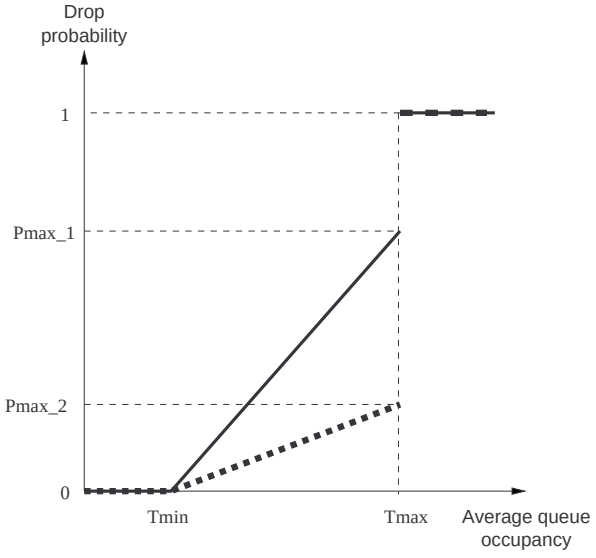


Figure 6 : Profil de perte pour WRED.

2.3.1.2.4 RED dans *et* en dehors *des limites (RIO)*

La méthode RED *dans* et *en dehors* des limites (RED with In and Out, RIO) est une technique qui emploie deux algorithmes RED avec des profils de perte différents. Les mots *dans* et *en dehors* se réfèrent au contrat de prestation (Service Level Agreement, SLA) et ses spécifications de prestation [God-02]. Un des algorithmes est utilisé pour le paquet *dans* les limites (pour lesquels la bande passante ne dépasse pas le taux de SLA négocié) et l'autre pour les paquets *en dehors* des limites, c'est-à-dire les excédant.,

Dans ce cas pour chaque file il y a deux seuils. Tant que l'occupation de la file d'attente est en dessous du premier seuil aucun paquet n'est rejeté. Quand l'occupation est entre les deux seuils, seuls les paquets *en dehor*s des limites sont rejetés. Si l'occupation dépasse le second seuil, indiquant une possible congestion du réseau, les paquets *dans* et *en dehors* des limites à la fois sont rejetés de façon aléatoire, mais les paquets *en dehors* sont rejetés plus agressivement. Dans la technique de signalisation DiffServ (voir 2.4.3) il y a un champ qui contient un bit indiquant si un paquet et *dans* ou *en dehors* des limites.

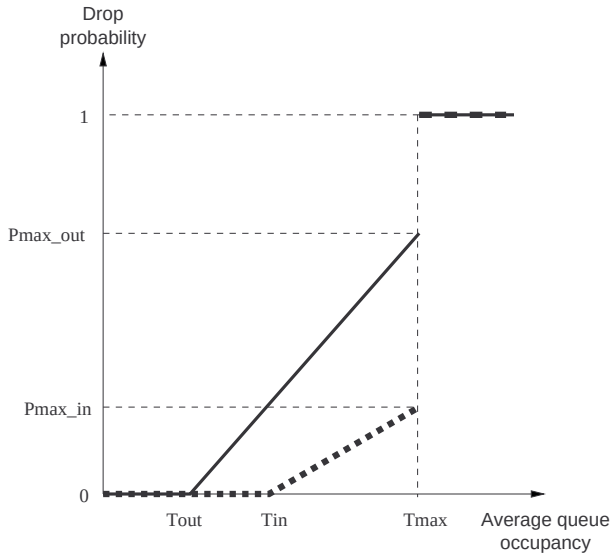


Figure 7 : Profil de perte pour RIO.

2.3.1.2.5 Notification explicite de congestion (ECN)

La notification explicite de congestion (Explicite Congestion Notification, ECN) est un ajout expérimental à l'architecture IP, conçu pour fournir une approche différente de la gestion active de files. ECN répond à la congestion en marquant les paquets et

en demandant au nœud destination l'envoi d'une notification explicite de congestion au nœud source. A la réception d'une telle notification, la source est censée réduire son taux de transmission.

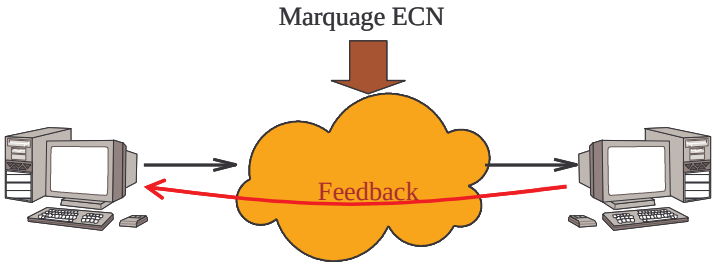


Figure 8 : Le mecanisme de notification explicite de la congestion.

2.3.1.3 Ordonnancement

L'ordonnancement est le mécanisme par lequel la décision d'expédition d'un paquet est prise. Elle est une composante très importante de l'architecture d'un routeur QoS, puisque c'est le stade où le délai et le débit peuvent être contrôlés. Les paquets sont sélectionnés afin d'atteindre un traitement différencié du trafic.

L'ordonnancement gère la largeur de bande allouée à chaque classe de service sur un port de sortie, contrôlant ainsi l'accès des classes de service à une ressource du réseau limitée : la bande passante. Voici les disciplines de file d'attente rencontrées les plus souvent dans des implémentations réelles.

2.3.1.3.1 Premier entré, premier sorti (FIFO)

Premier entré, premier sorti (First-In First-Out, FIFO) est la discipline d'ordonnancement la plus élémentaire dans laquelle tous les paquets sont traités de façon égale en les plaçant dans une seule file d'attente et en les expédiant dans l'ordre dans lequel ils ont été placés dans la file. Elle a une complexité de calcul très réduite, mais ne permet pas une différenciation de services entre flux de données.

Figure 9 : Exemple de fonctionnement de l'ordonnancement FIFO.

2.3.1.3.2 Priorité stricte (SP)

La priorité stricte (Strict Priority, SP, ou Priority Queuing, PQ) est la base d'une classe d'algorithmes d'ordonnancement qui sont conçus pour fournir une façon relativement simple de supporter des classes de services différenciés. Les paquets sont d'abord classifiés par le système, puis placés dans des files d'attente différentes par priorité. Ils sont sélectionnés depuis la tête d'une file (de manière FIFO) uniquement si les files de priorité plus haute sont vides. La complexité de calcul est de nouveau assez faible, mais un volume excessif de trafic de priorité haute peut conduire à un complet refus d'allocation des ressources pour le trafic de priorité plus basse.

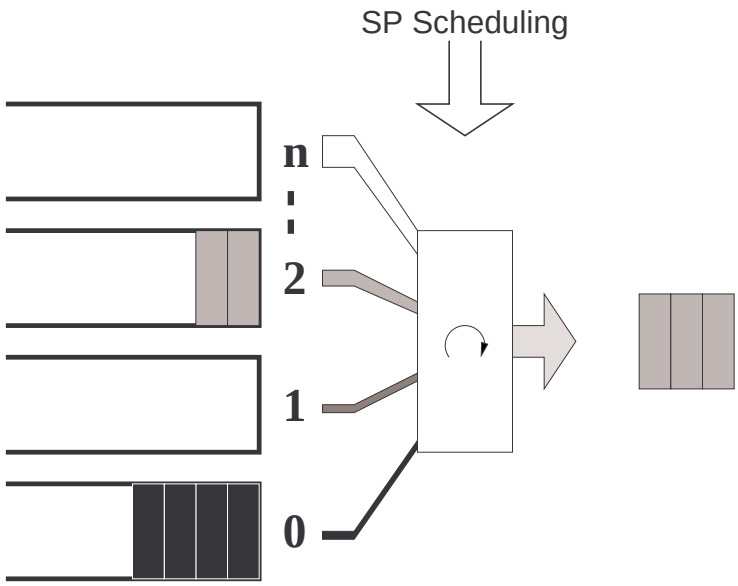


Figure 10 : Exemple de fonctionnement de l'ordonnancement SP.

2.3.1.3.3 Le tourniquet (RR)

L'ordonnancement par l'algorithme du tourniquet (Round Robin, RR, ou Fair Queueing, FQ) est à la base de toute une classe de disciplines d'ordonnancement conçues pour assurer à chaque flux un accès équitable aux ressources du réseau et empêcher les flux en salve de consommer plus que leur quote-part de bande passante du port de sortie. Les paquets sont d'abord classifiés dans des flux ; ils sont affectés ensuite à des files d'attente dédiées à ces flux. Ultérieurement les paquets sont sélectionnés par l'algorithme du tourniquet, en omettant les files vides. Dans RR les flux en salve ou avec un comportement incorrect ne dégradent pas la QoS délivrée aux autres flux, mais RR ne supporte pas des flux ayant des demandes différentes par rapport à la bande passante.

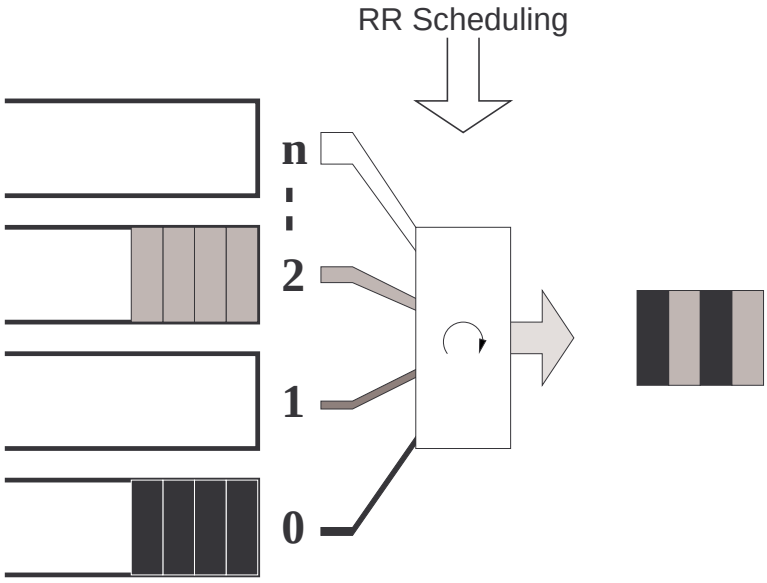


Figure 11 : Exemple de fonctionnement de l'ordonnancement RR.

2.3.1.3.4 Le tourniquet avec poids (WRR)

Le tourniquet avec poids (Weighted Round Robin, WRR, ou Class-Based Queueing, CBQ) est une technique conçue pour dépasser les limitations des modèles SP et RR. Les paquets sont classifiés dans des classes de service avec files d'attente dédiées et sont sélectionnés par l'algorithme du tourniquet. L'allocation de bande passante différente est faite soit en permettant aux files avec une largeur de bande supérieure d'expédier plus d'un paquet à chaque reprise de service soit en visitant ces files plusieurs fois dans le cadre d'une reprise.

WRR peut être implémentée au niveau matériel, offrant un contrôle imprécis mais efficace du pourcentage de bande passante de sortie alloué à chaque classe de service. Le contrôle est imprécis parce que le modèle est conçu au niveau des paquets et donne des résultats incorrects pour des flux ayant des paquets de taille variable.

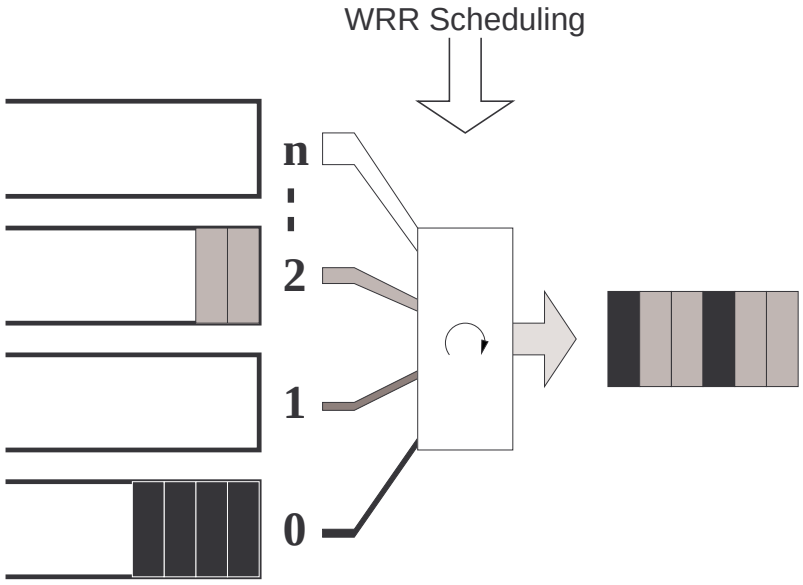


Figure 12 : Exemple de fonctionnement de l'ordonnancement WRR (poids 2 pour la file d'attente 2, et 1 pour la file d'attente 0).

Une variante de WRR utilise des quanta en lesquels les paquets sont divisés, et l'ordonnancement se fait par quanta et non par paquets, ce qui permet une plus grande précision dans le contrôle de la bande passante.

2.3.1.3.5 Ordonnancement équitable avec poids (WFQ)

L'ordonnancement équitable avec poids (Weighted Fair Queueing, WFQ) est une discipline d'ordonnancement basée sur FQ qui supporte des flux ayant des besoins différents en bande passante en associant à chaque file un poids qui lui assigne un pourcentage différent de la bande passante du port de sortie. Elle supporte des paquets de taille variable par une approche de tourniquet à niveau de bits (une version pratique de la technique Generalized Processor Sharing [Par-93], [Par-94]).

WFQ peut fournir de fortes garanties pour la limite supérieure du délai induit, au détriment d'une complexité de calcul relativement grande. Ceci induit aussi une limite sur le nombre total de classes qui peuvent être gérées. Des perfectionnements variés ont été proposés afin de réduire la complexité de l'algorithme, tout en maintenant une précision aussi bonne que possible.

2.3.1.3.6 Le tourniquet avec poids et déficit (DWRR)

Le tourniquet avec poids et déficit (Deficit Weighted Round Robin, DWRR) est une classe de disciplines qui essaye de dépasser les limitations des modèles WRR et

WFQ. Elle supporte avec exactitude une distribution juste de la bande passante même pour des files qui contiennent des paquets de taille variable (à l'opposé de WRR). DWRR définit une discipline avec une complexité de calcul inférieure à celle de WFQ et peut être implémentée au niveau matériel. Elle utilise un poids pour contrôler la bande passante et un compteur de déficit pour spécifier le nombre maximum d'octets qu'une file peut transmettre à chaque reprise de sélection. Un quantum de service proportionnel au poids de la file d'attente est défini et utilisé pour incrémenter le compteur de déficit.

2.3.1.4 Guarantee of Service (GoS)

La dégradation de qualité est dûe à la compétition pour des ressources finies. Il s'agit de la compétition pour partager la connexion, qui détermine le débit, de la compétition pour entrer dans les files d'attente, qui entraîne la perte de paquets, et de la compétition pour sortir des files d'attente, qui influence le délai. Mais, comme mentionné auparavant, ces compétitions ne sont pas indépendantes [Dav-99]. La société U4EA Technologies [U4EA] a conçu sur ce nouveau modèle la technologie GoS (Guarantee of Service).

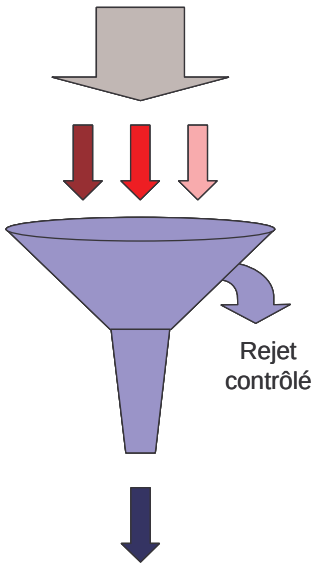


Figure 13 : Principe de fonctionnement de GoS.

Le fonctionnement du GoS a comme base le contrôle simultané des paramètres de performance comme suit :

- Il identifie et sépare les flux d'entrée en fonction des exigences QoS ;

- Il limite les taux à des niveaux configurés ;
- Il assigne des niveaux permis de perte et délais aux paquets ;
- Il contrôle le rejet par flux pour éviter le débordement incontrôlé ;
- Il mélange les flux en respectant les contraintes de délai et en maximisant l'efficacité d'utilisation de la connexion.

2.3.1.5 Discussion

Nous présentons dans le tableau qui suit une comparaison synthétique des méthodes de gestion des flux de trafic les plus répandues, du point de vue du contrôle qu'elles exercent sur les paramètres de performance.

Approche	*Contrôle du débit*	*Contrôle de la perte*	*Contrôle du délai*
FIFO	non	non	non
SP	non	non	**OUI**
RED	non	**OUI**	non
WRR	**OUI**	non	non
GoS	**OUI**	**OUI**	**OUI**

Tableau 2 : Comparaison des méthodes de gestion de flux de trafic les plus répandues.

On voit que pour un contrôle total, certaines méthode doivent être combinées, mais le problème est qu'elles sont parfois mutuellement exclusives (par exemple, SP et WRR, toutes les deux techniques d'ordonnancement).

2.3.2 QoS au niveau des connexions

Parfois l'ordonnanceur d'un routeur doit faire plus que le simple mixage du trafic au niveau des paquets IP [Arm-00]. La capacité de l'ordonnanceur à mélanger le trafic appartenant à différentes files dépend de la vitesse avec laquelle la connexion de sortie peut transmettre chaque paquet. Pour des liaisons à haut débit (comme les circuits OC-3/STM-1 à 155 Mbps), moins de 80 μs sont nécessaires pour transmettre un paquet IP de 1500 octets. Cela permet à l'ordonnanceur de diviser la bande passante de la connexion en créneaux de largeur temporelle supérieure ou égale à 80 μs – un nombre très raisonnable, qui descend à 20 μs pour une connexion à 622 Mbps

(circuits OC-12/STM-4). Néanmoins, aux frontières de l'Internet il y a des connexions opérant à 1 Mbps ou moins, dans la gamme 56-128 kbps pour ISDN (« Integrated Services Digital Network ») et aussi bas que 28,8 kbps pour les connexions par l'intermédiaire du modem.

Pour un paquet IP de 1500 octets, la transmission prend environ 94 ms sur une connexion à 128 kbps, bloquant complètement la connexion pendant tout ce temps-là. Sans égard au fait que le trafic sensible à la gigue a été classifié dans une autre file, ces paquets vont éprouver une gigue de 94 ms lorsque l'ordonnanceur transmet le paquet de 1500 octets d'une autre file. Manifestement cela pose certains problèmes si des applications sensibles à la QoS doivent être supportées à l'extrémité d'une connexion typique à bas débit. La solution de base est d'effectuer une segmentation supplémentaire des paquets IP au niveau de la connexion, d'une manière transparente au niveau IP même. L'architecture QoS est par la suite appliquée au niveau connexion en gérant des segments au lieu de paquets, permettant ainsi à l'ordonnanceur de mixer au niveau des segments. En choisissant la taille de segments d'une manière adaptée, une telle approche permet au trafic IP sensible à la gigue d'éviter d'attendre derrière de longs paquets IP. Pourtant la segmentation fait baisser l'efficacité globale du transfert, car chaque segment doit porter son propre en-tête pour permettre la recombinaison ultérieure de segments.

En dépit du fait que l'ATM a été conçu à l'origine pour des connexions à haut débit, son design reflète un souci similaire de minimiser l'intervalle pendant lequel le trafic d'une certaine classe peut monopoliser la connexion. La cellule ATM est petite par conception, et chaque commutateur ATM est un exemple d'architecture COS. Les cellules arrivantes sont mises dans des files d'attente pour transmission en fonction du contenu des champs de leur en-tête. Les bons commutateurs ATM consacrent des files à chaque classe de trafic au niveau des ports et des ordonnanceurs transmettant les paquets depuis chaque file en conformité avec les garanties de bande passante données à chaque classe.

Les architectures COS peuvent être implémentées de manière différente, chaque solution ayant des conséquences spécifiques pour les caractéristiques QoS du réseau IP entier. La fonctionnalité fondamentale de chaque routeur est donc de :

- Savoir où transmettre le paquet (expédition conventionnelle) ;

- Savoir quand transmettre le paquet (une exigence supplémentaire de la QoS) ;
- Finaliser les tâches précédentes indépendamment de l'autre trafic partageant le même routeur (prédictibilité).

2.4 Comportement prédictible d'une frontière à l'autre

Comme mentionné auparavant, tout service réseau de bout en bout est construit à la fois par la concaténation et la superposition des comportements d'une frontière à l'autre et au niveau des sauts. Les opérateurs réseau se trouvent face à un défi : offrir une QoS améliorée à leurs utilisateurs alors qu'ils ont une large gamme de comportements possibles à combiner. Une série d'organisations ont proposé des modèles de service et mécanismes pour répondre à la demande pour QoS [Hun-02]. Chaque type de solution reflète un ensemble différent de hypothèses et compromis par rapport au COS et aux capacités des routeurs dans le réseau.

2.4.1 Modèles « frontière et cœur »

La conception d'une bonne architecture COS, aussi bien dans le hardware que dans le software, est généralement non triviale. Dans beaucoup d'implémentations au niveau logiciel, l'introduction de la classification, de la gestion et de l'ordonnancement influence souvent la performance globale du système. Les implémentations au niveau matériel viennent seulement de devenir usuelles et jusqu'à un passé récent une implémentation COS fixée dans le hardware était trop risquée du point de vue commercial. Le modèle « frontière et cœur » permet aux routeurs situés au centre du réseau d'utiliser des implémentations matérielles afin d'avoir une vitesse accrue, laissant les traitements plus complexes (et plus lents) à la charge des routeurs de frontière de nature logicielle. Les routeurs de frontière peuvent classifier et gérer indépendamment des centaines et milliers de classes de trafic, tandis que les routeurs au cœur du réseau seront limités à une dizaine de classes.

Un nombre limité de files d'attente dans les routeurs de cœur produit une nouvelle exigence, que les routeurs de frontières soient capables de réduire la nature en salve du trafic entrant dans le réseau. Dans la discussion précédente sur le contrôle de la QoS au niveau de sauts, on considérait que les classes individuelles de trafic pouvaient être complètement imprédictibles, dans l'hypothèse où on peut les isoler et les ordonnancer à chaque point de congestion. Pourtant, même si un modèle

« frontière intelligente/cœur simple » peut avoir la granularité requise aux frontières, il ne l'a pas au cœur. Des classes de trafic multiples se verront agrégées dans des files partagées par les routeurs de cœur. Le potentiel pour une interférence mutuelle imprévisible est grand, à moins que le réseau impose un certain niveau de prédictibilité avant que le trafic arrive aux routeurs de cœur. La solution consiste à permettre aux routeurs de frontière de manipuler les caractéristiques temporelles des classes individuelles de trafic (et, par conséquent, de l'agrégation de ces classes) avant qu'elles n'entrent le cœur du réseau.

2.4.1.1 Conditionnement et régulation

Le principal but d'une architecture COS est la protection du trafic dans chaque file d'attente, du trafic potentiellement en salve dans d'autres files. Au niveau des sauts il est évident que, étant donnée une séparation du trafic dans des files d'attente différentes, un ordonnanceur doit garantir qu'un certain intervalle maximal entre services (ou, de manière équivalente, une largeur de bande minimale) est respecté dans le pire cas. Si une capacité en surplus est disponible, on peut s'attendre à ce qu'un « bon » routeur accorde cette capacité à toutes les files ayant des paquets en attente. Néanmoins cette pratique n'est pas toujours désirable dans une perspective globale au niveau de tout le réseau.

Le simple fait de vider une file aussi vite que possible peut augmenter la nature en salve perçue par les routeurs en aval de la route. Par suite, un problème sérieux peut apparaître si les routeurs suivants ne différentient pas le trafic de granularité aussi élevée que le routeur local. En plus, les fournisseurs de services réseau peuvent souhaiter limiter le débit maximal qu'un utilisateur peut envoyer à travers le réseau. Si le client reçoit souvent plus de largeur de bande que le minimum garanti (par exemple si le réseau est nouveau et/ou sous chargé), une question de perception émerge : le client commence à associer la performance typique avec celle qu'il a payée. Si la capacité disponible diminue, l'utilisateur va simplement obtenir une performance d'une frontière à l'autre plus proche du minimum garanti. Toutefois le client va percevoir une dégradation du service et être insatisfait.

Gérer les attentes des clients est une partie importante d'une affaire, et dans ce cas le contrôle du débit est l'outil technologique qui peut être utilisé. Le fait de placer une limite supérieure sur le débit (ou un écart minimum entre paquets) pour une classe de

trafic donnée est connu sous le nom de conditionnement. Un conditionneur est configuré afin de fournir à la fois une valeur minimale et une valeur maximale pour l'intervalle de service[1]. Les paquets séparés avec un intervalle plus petit que celui qui est autorisé seront mis dans des files d'attente, leur nature en salve étant ainsi réduite. L'intervalle maximal garantit une largeur de bande minimale (et un délai limité). Une forme simple de conditionneur est le seau percé (« leaky bucket » en anglais), car indépendamment de la vitesse d'arrivage des paquets, ils peuvent « fuir » seulement avec un taux fixe. Malgré le fait que cette solution de transmission des paquets également espacés est la plus évidente, des recherches comme celle de [Dav-99] montrent que l'utilisation d'un espacement entre paquets de type poissonnien est plus appropriée, car il facilite l'ordonnancement dans les routeurs suivants.

Le conditionnement n'est pas simple à introduire dans un routeur « meilleur effort », puisque cette fonction suppose l'existence d'une architecture COS adaptée. Une alternative, peut être moins élégante, a été d'introduire un comportement lié au rejet des paquets, qui est sensible à l'excès de salves dans le trafic d'une classe. Lorsque trop de paquets arrivent dans un court intervalle, ils sont tout simplement rejetés. Ce processus est connu sous le nom de régulation. La régulation peut être implémentée sans files ou ordonnanceur, quoiqu'une certaine forme de classification soit nécessaire afin de différencier les règles correspondant aux différentes classes de trafic. Dans sa forme la plus simple on limite le nombre des paquets qui peuvent être transmis dans un certain intervalle au moyen d'un compteur ayant une valeur maximale. La régulation permet à l'opérateur du réseau de simuler une congestion pour une certaine classe de trafic, avant que la véritable congestion apparaisse plus loin sur le trajet. Même si le trafic n'utilise pas un protocole adaptatif, la régulation protège le reste du réseau en continuant de rejeter les paquets qui dépassent les paramètres permis.

Le conditionnement et la régulation sont tous deux des instruments très utiles pour les concepteurs dans leur tâche de contrôle du réseau. On ne peut pas permettre aux classes individuelles de trafic d'être imprédictibles, sauf si on peut les isoler avec précision en tous les points potentiels de congestion. Si cette fonctionnalité manque, on doit essayer d'imposer un certain degré de prédictibilité avant chaque point de congestion. Dans le modèle « frontière intelligente/cœur simple », la solution est de

[1] L'intervalle de service est le temps entre la transmission des paquets de la même file.

conditionner et/ou réguler chaque classe de trafic avant qu'elle ne pénètre le cœur du réseau pour imposer un niveau général d'ordre, lissage et prédictibilité dans le cadre de chaque classe (et par la suite au niveau de leur agrégats). Le conditionnement peut aussi s'avérer utile à la sortie d'un réseau, dans des situations où la régulation agressive du réseau suivant pourrait être dommageable.

2.4.1.2 Marquage et réarrangement

Bien que le conditionnement puisse être une solution sophistiquée pour lisser le trafic en salve, la simple régulation est un instrument assez brutal. De nombreuses variations ont été introduites pour adoucir l'effet de la régulation des routeurs de frontière. Un nœud régulateur peut choisir de marquer uniquement les paquets dont la nature en salve dépasse un certain seuil, au lieu de les rejeter immédiatement. Les routeurs au long de la route reconnaissent les paquets ainsi marqués et leur attribuent une priorité plus petite que celle des paquets non marqués. Si les files d'attente commencent à se saturer à cause d'une congestion temporaire, l'algorithme de gestion peut rejeter les paquets marqués avant ceux qui ne le sont pas.

L'impact de cette stratégie sur le cœur du réseau est plus faible que celui obtenu par la simple régulation, car un bon nombre de paquets dans une salve seront marqués au lieu d'être rejetés. L'avantage de cette approche est que, en absence de congestion au cœur du réseau, cette classe de trafic peut utiliser la bande passante disponible.

2.4.2 Routage d'une frontière à l'autre

Il n'y a pas de restrictions particulières qui régissent la façon dont les routeurs et liens sont connectés entre eux pour former un réseau IP. Les mécanismes de routage de l'Internet, qui choisissent la route la plus courte, sont basés sur la supposition que la topologie du réseau est dynamique. Chaque routeur peut avoir plus d'une interface de sortie par laquelle il peut envoyer des paquets ; le rôle des protocoles de routage est d'établir une seule interface par laquelle le paquet est effectivement transmis. Pour réduire la complexité des calculs, ce choix a été généralement fait par des algorithmes utilisant une seule métrique pour définir la route la plus courte. Deux problèmes connexes apparaissent dans le contexte de la QoS. D'abord le fait qu'une seule métrique peut ne pas être adaptée pour tout trafic traversant une section particulière du réseau. Deuxièmement, forcer des sous-ensembles de trafic à suivre des routes

alternatives dans une topologie de réseau donnée est rendu difficile par le paradigme d'expédition basé uniquement sur la destination.

2.4.2.1 Routage à base de QoS

Les protocoles de routages à base de QoS tentent de prendre en compte des métriques multiples en créant les tableaux de routage du réseau. Car le routage avec une seule métrique a des limitations lorsqu'on essaye de répondre aux demandes variées de qualité d'un environnement multiservice.

Une métrique peut être considérée comme un type de coût, avec chaque connexion (saut) associée à un coût. Les protocoles de routage essayent de trouver des routes ayant un coût total minimal résultant de l'addition des coûts des toutes les connexions jusqu'à la destination. Pourtant ce coût peut ne pas représenter les intérêts et besoins de tous les types de trafic. Est-ce qu'il devrait représenter le délai de la connexion, la bande passante disponible, la probabilité de perte de paquets ou peut-être le coût réel d'envoi du paquet à travers la connexion ? Un choix peut être approprié pour un certain type de trafic, et en même temps représenter un gaspillage de ressources pour d'autres trafics.

Le routage basé sur QoS crée des arbres multiples de la route la plus courte, couvrant la même topologie de routeurs et connexions avec chaque arbre, mais en utilisant différentes combinaisons des paramètres comme métriques de la connexion. Ainsi l'adjectif « court » aura des significations différentes. Le but est de minimiser la coexistence non nécessaire dans le même routeur de trafic avec des exigences QoS considérablement différentes. Les paquets ayant des exigences strictes par rapport au délai seront transmis en utilisant l'arbre construit avec le délai en tant que métrique. Les paquets sans demandes de temps réel seront transmis au moyen d'un arbre construit, par exemple, afin de minimiser le coût financier de la route.

2.4.2.2 Le contrôle explicite de la route

Les topologies internes de beaucoup de réseaux sont organisées de telle sorte que plusieurs routes puissent être trouvées entre la majorité des points. Une limitation importante de l'expédition IP conventionnelle est que les arbres de la route la plus courte basés sur une seule métrique n'utilisent qu'une des routes possibles vers chaque destination. Parce que les routes alternatives, avec un chargement léger, ne

sont pas utilisées, les routeurs qui existent dans les arbres de la route la plus courte peuvent être surchargés – ils deviennent des goulots d'étranglement, limitant potentiellement la capacité du réseau à fournir une différenciation des services adaptée, même si les routeurs eux-mêmes ont une architecture COS. Quand la charge moyenne sur un tel routeur s'accroît, la probabilité de perte de paquets et de gigue augmente aussi. Pour pallier ce problème l'opérateur du réseau a deux alternatives :

- Mettre à jour les routeurs et les connexions afin qu'ils opèrent plus vite, ce qui peut demander des investissements relativement grands, ce qui est généralement une solution à court terme ;
- Utiliser des mécanismes d'expédition des paquets supplémentaires qui permettent de scinder le trafic en lui faisant emprunter des routes différentes (une partie des quelles peuvent être aussi courtes que la route la plus courte « officielle », d'autres pouvant être plus longues par rapport à la métrique utilisée).

Quand le réseau même est construit avec une technologie bon marché, avec une bande passante réduite ou moyenne, la première solution peut être entièrement satisfaisante. Construire la capacité équivalente peut aussi se faire par parallélisme – une topologie IP riche en routeurs et connexions de bas débit à travers lesquelles la charge agrégée peut être distribuée.

La deuxième approche donne naissance à la question suivante : s'ils renoncent à l'information fournie par les protocoles de routage existants, les opérateurs du réseau doivent fournir une source externe d'information afin de contrôler le routage. Ceci peut se faire, par exemple, en ajoutant des étiquettes aux paquets et en les expédiant à travers des routes construites à l'avance par des routeurs modifiés.

2.4.2.3 Signalisation

Supposant qu'on peut fournir une gestion et un ordonnancement différenciés au niveau des sauts, et que les couches de connexion sous-jacentes sont contrôlables de façon adéquate, la question est maintenant de savoir comment établir et modifier le comportement actuel du réseau. Ce problème demande la coordination des comportements effectifs de tous les éléments le long de chaque route. Un terme générique pour ce processus est *signalisation* – l'acte d'informer chaque saut au long

d’une ou plusieurs routes de la manière de reconnaître le trafic qui nécessite un traitement spécial et le type de ce traitement. La signalisation peut être réalisée d’un grand nombre de façons avec des degrés variés de temps de réponse, flexibilité et intervention humaine (qui ne sont pas toutes considérées comme signalisation *per se*).

A un extrême se trouve la signalisation dynamique d’une frontière à l’autre, quand le réseau est informé chaque fois qu’une nouvelle classe de trafic nécessite un support spécial. Le réseau lui-même répond en établissant une information supplémentaire (ou en modifiant l’information existante) à chaque saut, pour assurer le comportement demandé d’une frontière à l’autre. Des exemples de signalisation à la demande sont les protocoles User Network Interface (UNI) et Network Node Interface (NNI) de ATM, et celui de IETF, Resource reSerVation Protocol (RSVP).

Certaines technologies n’ont pas de protocoles de signalisation complètement dynamiques ou ne sont pas assez mûres pour que des implémentations fiables de leurs protocoles de signalisation existent. Dans ces circonstances, les réseaux doivent être configurés chaque fois pour les nouveaux services, impliquant souvent l’intervention humaine au niveau des nœuds au long des routes affectées. Ce processus peut être vu comme une forme de signalisation, même si la durée de réponse est de quelques ordres de magnitude plus grande que celle de la signalisation dynamique.

Un compromis est l’utilisation des contrôleurs ou serveurs centralisés pour lesquels la configuration est effectuée. Ces contrôleurs distribuent ensuite automatiquement les règles adaptées aux connexions et nœuds dans le réseau sans l’intervention d’un opérateur humain. Les designs encore plus avancés permettent à ces contrôleurs de réagir automatiquement aux conditions variables dans le réseau en accord avec des politiques générales imposées. Bien que ces méthodes centralisées constituent des mécanismes dynamiques, elles diffèrent par rapport à la signalisation d’une frontière à l’autre par le fait qu’elles ne sont pas contrôlées par les applications elles-mêmes.

2.4.2.3.1 Classes de service

Les classes de service sont une façon de gérer le trafic dans un réseau par le groupement des types de trafic similaires (par exemple courrier électronique, vidéo en continu, voix, transfert de fichiers). Chaque type est traité comme une classe avec son propre niveau de priorité de service ; sa fonction peut donc être assimilée à la signalisation. Il y a deux technologies principales de classe de service :

- IEEE 802.1p/Q ;
- Précédence IP (Type of Service, ToS).

La spécification IEEE 802.1p définit les trois bits dans le champ IEEE 802.1Q (étiquette VLAN) de l'en-tête de niveau Ethernet des paquets pour la désignation de la priorité. Cette approche manque d'une vue globale sur les caractéristiques de la connexion, car l'Ethernet n'est généralement utilisé que dans les LAN.

Les mécanismes QoS au niveau IP fournissent un contrôle sur les configurations QoS de bout en bout. La première tentative de IETF d'utiliser le ToS de l'en-tête IPv4 a été [RFC-1349]. La plus récente version s'appelle « précédence IP » et est définie dans [RFC-1812]. Trois bits du ToS sont utilisés pour créer 8 niveaux de priorité et quatre sont utilisés pour signaler la sensibilité au délai, bande passante et perte de paquets. Les mêmes bits peuvent aussi être employés comme une indication de la façon dont un paquet doit être expédié, connue sous le nom de « comportement par saut », comme c'est le cas dans DiffServ.

2.4.3 Les solutions de base

Le support pour QoS dans les réseaux IP a son origine dans l'article sur le modèle de services intégrés dans les réseaux à base de paquets (Integrated Service Packet Network) par Clark, Schenker et Zhang [Cla-92]. Ce modèle est construit sur quatre « piliers » :

a) Une spécification des demandes QoS, qui peut être vue comme une description d'un contrat de prestation qui doit être honoré par une certaine architecture QoS ;

b) Des mécanismes pour le contrôle de l'admission et le conditionnement du trafic quand les ressources sont finies et la congestion peut apparaître ;

c) Mise en place de l'ordonnancement et d'autres mécanismes dans les nœuds du réseau pour appliquer une expédition et un traitement préférentiel des paquets. Des ressources suffisantes doivent être présentes pour assurer la QoS spécifiée.

d) Des interfaces de signalisation et service pour communiquer l'information sur les préférences et attentes par rapport au traitement et l'expédition des paquets

de la part des applications aux éléments qui contrôlent les ressources du réseau et en arrière aux applications.

Une architecture QoS qui intègre les quatre piliers dans une solution de bout en bout est nécessaire. Mais cette condition ne signifie pas nécessairement que, par exemple, un seul protocole de signalisation pour la réservation des ressources doit être utilisé de bout en bout. En effet il est probable qu'une architecture de gestion de la QoS consistera en plusieurs architectures compatibles concaténées, plutôt qu'une seule infrastructure QoS globale.

Chaque pilier peut cependant être mis en pratique différemment. Par exemple les ressources peuvent être demandées en fonction de la granularité des flux de trafic. La signalisation reflétera de manière similaire le niveau de granularité et peut être implicite ou explicite, par l'intermédiaire d'un protocole tel que RSVP. Le contrôle de l'admission peut aussi être explicite ou implicite, par fourniture excédentaire ou conditionnement etc.

2.4.3.1 Allocation de ressources

Les solutions QoS peuvent être classifiées comme étant dirigées vers le contrôle de l'allocation de ressources ou vers l'optimisation des performances [Wan-01].

Il manque actuellement à l'Internet une allocation active, dynamique des ressources. Ce concept est pourtant très important, car les ressources sont ce qui détermine les caractéristiques d'une connexion. Les principales solutions d'une frontière à l'autre qui sont basées sur l'allocation des ressources sont :

a) Integrated Services (IntServ) [RFC-2211], [RFC-2212] – utilise un protocole de réservation de bout en bout, RSVP, à niveau de flux pour la route de bout en bout [RFC-2205], [RFC-2210];

b) Differentiated Services (DiffServ) [RFC-2475] – utilise une combinaison de conditionnement aux frontières, fourniture et utilisation de priorités du trafic pour rendre possible la différenciation de services [RFC-2474], [RFC-2638].

c) Integrated Services sur Differentiated Services [RFC-2998] – un cadre qui assure la QoS d'un bout du réseau à l'autre dans une approche hybride [Wro-01] (voir la figure 14).

Dans [RFC-2990] les auteurs admettent que « both the Integrated Services architecture and the Differentiated Services architecture have some critical elements in terms of their current definition which appear to be acting as deterrents to widespread deployment [...] There appears to be no single comprehensive service environment that possesses both service accuracy and scaling properties ». Ce propos souligne les raisons d'opter pour une architecture hybride, composée des régions IntServ et DiffServ (avec les problèmes associés liés aux procédures de mise en relation et de fonctionnement combiné) et la nécessité d'améliorer les paradigmes IntServ et DiffServ.

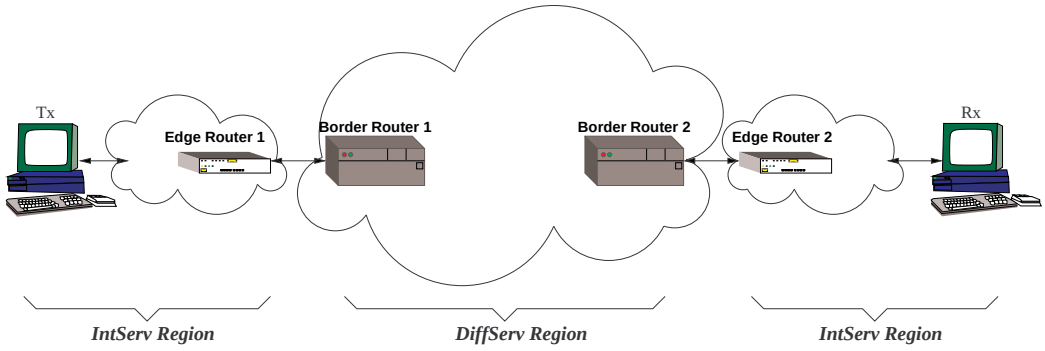


Figure 14 : Une configuration possible de réseau pour le déploiement de IntServ sur DiffServ.

2.4.3.2 Optimisation de performance

Les solutions privilégiant l'optimisation de la performance sont centrées sur la modification des architectures de routage IP conventionnelles par l'introduction des fonctionnalités supplémentaires pour le routage IP et de nouveaux outils de gestion de la performance. Les directions principales sont la commutation multi protocole et l'ingénierie du trafic.

Les solutions de commutation multi protocole (par exemple MPLS) utilisent pour l'expédition de paquets un algorithme à base d'étiquettes. Une étiquette est un champ de petite taille dans l'en-tête du paquet utilisée pour l'identification d'une classe équivalente d'expédition (Forwarding Equivalent Class). Une étiquette est une sorte d'identificateur de la connexion et représente une classe de paquets qui sont expédiés par la même route, même si leur destination finale est différente.

Les algorithmes d'expédition à base d'étiquettes nécessitent de classer les paquets au point d'entrée du réseau et qu'une étiquette initiale soit assignée au paquet. Dans le réseau, les commutateurs ignorent l'en-tête de réseau du paquet et utilisent

exclusivement l'étiquette. Le commutateur à la sortie du réseau élimine l'étiquette et expédie le paquet de manière traditionnelle. La route créée est équivalente à un circuit virtuel entre les points d'entrée et de sortie du réseau, circuit suivi par tous les paquets appartenant à la même classe.

Multi-Protocol Label Switching (MPLS) [RFC-3031] a été conçu en 1997 comme une approche alternative pour le support de IP sur ATM. Les techniques précédentes qui s'attaquaient à cette tâche étaient toutes trop complexes et avaient des problèmes d'extensibilité. MPLS permet au protocole de routage IP de prendre le contrôle sur les commutateurs ATM, permettant ainsi une intégration aisée dans l'infrastructure existante.

MPLS définit de nouveaux protocoles de signalisation et de distribution d'étiquettes [RFC-3036] ainsi que d'extensions des protocoles existants. Les protocoles de contrôle sont basés sur les adresses IP, ils peuvent donc être facilement intégrés aux protocoles IP correspondants. Ceci crée une architecture unifiée qui utilise MPLS pour l'ingénierie de trafic au cœur du réseau et le routage IP pour l'extensibilité.

MPLS peut aussi être utilisé conjointement avec DiffServ pour fournir la QoS demandée dans les réseaux IP [Fau-01]. Dans cette architecture une route séparée est créée pour chaque classe de trafic et le mode opératoire des routeurs est similaire à celui qui a été décrit pour l'architecture DiffServ. Il est aussi possible de réduire le nombre de routes en agrégeant celles qui ont les mêmes extrémités.

2.4.3.3 Discussion

La remarque essentielle est que les concepteurs de réseau doivent trouver un compromis entre le nombre de classes de trafic dans leur réseau et le nombre de classes de trafic que les architectures COS de leurs routeurs peuvent traiter. Certaines solutions sont basées sur des architectures distribuées entre le cœur et les frontières du réseau, dans lesquelles au cœur du réseau, sont placés des routeurs rapides ayant des fonctions COS limitées, tandis qu'aux frontières sont placés des routeurs plus lents mais avec des fonctions COS avancées.

Une deuxième observation est que les algorithmes de recherche de la route la plus courte existant dans l'Internet ne sont pas nécessairement optimaux pour toutes les classes de trafic et à travers un groupe de routeurs et connexions. Une métrique

unique peut ne pas être adaptée à tout le trafic traversant une section particulière du réseau. En plus, le paradigme d'expédition basé sur la destination rend difficile le fait de forcer les sous-trafics à suivre des routes alternatives dans une topologie de réseau donnée.

Toute solution réaliste par rapport à la QoS pour réseaux IP doit tenir compte de demandes souvent conflictuelles : une implémentation facile, une quantité d'informations d'états réduite, l'optimisation de l'utilisation des ressources du réseau, l'adaptation dynamique aux changements de routes et le fonctionnement dans un environnement où le routage et la signalisation sont découplés.

2.5 Autorisation, authentification et facturation

Si l'on offre à quelqu'un une place dans un train de première classe pour le prix d'une place de seconde, il la prendra vraisemblablement. On peut se demander où est le piège, mais, à la fin, personne ne refuse un meilleur service s'il a le même prix avec un service plus mauvais, peu importe s'il s'agit d'une place dans un train, le temps d'expédition d'un paquet ou la bande passante d'accès au réseau local.

Toute technologie de réseau qui offre une différenciation des niveaux de service doit aussi prendre en charge le droit de chaque personne d'utiliser un niveau de service particulier. Si tout le monde avait le droit d'utiliser le meilleur niveau de service en même temps, ceci conduirait à un épuisement des ressources et/ou une obligation de modifier le réseau. En général les ressources du réseau sont limitées à des niveaux de service variés. Par la suite, une tâche essentielle consiste à consentir ou empêcher l'accès d'utilisateurs spécifiques à des niveaux de service en fonction de leurs droits. Le droit d'utilisation peut être établi d'un certain nombre de façons – par exemple l'imputation d'un coût ou l'attribution administrative (en fonction de l'importance de l'utilisateur). Un fournisseur de services commercial serait enclin à utiliser des frais : on reçoit le service qu'on paye. Dans le réseau d'une entreprise on peut déterminer les allocations des services, au moins partiellement, en fonction du statut de l'utilisateur (ou son département) dans la société.

Etablir et surveiller le droit d'une personne à faire convenablement usage de certains niveaux de service : voilà une problématique que l'industrie de l'Internet commence seulement d'aborder. On commence par les questions d'autorisation : identifier les classes de service que les utilisateurs ont le droit de négocier. Puis il y a

l’authentification : démontrer que l’entité utilisant le réseau est l’utilisateur prétendu, à la fois pendant les négociations concernant les droits d’utilisation et pendant la transmission de trafic ultérieure. Il y a aussi la question de facturation : obtenir le règlement des frais dûs par l’utilisateur correct. La facturation présente un intérêt même dans le cadre d’un réseau d’entreprise, assurant un niveau plus fin de contrôle de l’utilisation que seul permet la connaissance du statut de l’utilisateur.

Ces trois questions sont étroitement liées à la signalisation du réseau car ce système doit établir les niveaux de service requis d’une frontière à l’autre et les associer au trafic en provenance de l’utilisateur. Quand l’utilisateur fait usage d’une signalisation dynamique d’une frontière à l’autre, le protocole doit être couplé avec les mécanismes d’autorisation, authentification et facturation. Le réseau doit être en mesure d’authentifier toutes les demandes d’utilisateur pour des niveaux spécifiques de service. Les utilisateurs ne doivent pas être facturés pour des services qu’ils ne demandent pas et bien sûr être facturés correctement pour les niveaux de service qu’ils demandent. Si le mode de facturation de l’opérateur s’appuie, au moins partiellement, sur la quantité d’utilisation, la consommation doit être également suivie et authentifiée.

Un opérateur peut essayer de déduire l’identité d’un utilisateur de ses points de connexions physiques au réseau, mais en cette époque de connexion par modem et de nœuds mobiles (« wireless » en anglais) cette approche est bien peu viable. En absence de ces fonctions, l’utilisateur et le fournisseur de services sont obligés de se contenter de canaux plus traditionnels pour la négociation des niveaux de service : le fax, le téléphone, le courrier. Comme alternative, le fournisseur peut simplement espérer que les utilisateurs ne vont pas essayer de prétendre être quelqu'un d’autre au moment de la commande des niveaux de service.

Deux problèmes sont soulevés par la décision d’intégrer une composante liée à la quantité d’utilisation dans les frais liés au droit d’utilisation. Premièrement, il n’y a pas de consensus émergeant dans l’industrie sur la métrique la plus réaliste : le seul nombre de paquets, la nature en salve, la bande passante maximale ou moyenne, ou alors une mesure complexe du délai et de la gigue ? Deuxièmement, après avoir décidé d’une métrique compréhensible par les utilisateurs, il faut pouvoir la mesurer de façon précise et associer de manière fiable les mesures à des utilisateurs particuliers.

Appréhender les rôles joués par les modèles de gestion de l'autorisation, l'authentification des utilisateurs et la facturation comme des composantes essentielles d'une solution QoS globale pour les réseaux IP et avoir la capacité de comprendre ce que l'industrie met à disposition sont des questions clé pour tous ceux qui désirent implémenter et fournir la QoS.

2.6 Conclusions

Le besoin pour des capacités QoS dans l'Internet dérive du fait que le service « meilleur effort » et le routage des paquets ne répondent pas aux exigences de beaucoup d'applications nouvelles, qui nécessitent un certain degré d'assurance de ressources afin d'opérer de manière satisfaisante. Les demandes diverses des clients créent aussi la nécessité que les fournisseurs offrent des niveaux différents de service dans l'Internet.

La communauté Internet a développé un bon nombre de technologies pour traiter ces questions. IntServ et DiffServ offrent de nouvelles architectures pour l'allocation des ressources. IntServ utilise des réservations pour fournir des garanties pour ressources à chaque flux. L'architecture DiffServ combine le conditionnement aux frontières, la fourniture de ressources et l'utilisation de priorités pour fournir des niveaux de service différenciés aux utilisateurs.

MPLS et l'ingénierie du trafic traitent la question de la fourniture de ressources et de l'optimisation des performances dans les réseaux fédérateurs. Le mécanisme explicite de routage de MPLS ajoute une fonction importante aux réseaux IP. Combiné avec le routage à base de contraintes pour l'ingénierie de trafic, cette méthode peut aider les fournisseurs à faire le meilleur usage des ressources disponibles et réduire les coûts.

On peut conclure que, même si les exigences strictes des applications fixes et mobiles, présentes ou futures, ne sont pas satisfaites entièrement par les solutions existantes et qu'il demeure des questions encore en suspens, les mécanismes de base sont définis et vont être déployés de plus en plus intensivement avec la croissance de la demande des utilisateurs en QoS. Au niveau de nœuds, l'essentiel est représenté par les mécanismes de type COS, les éléments de base de la QoS, sans lesquels aucune solution ne produira de résultats. Dans notre travail nous nous sommes penchés surtout sur cet aspect du contrôle de la QoS et ses divers aspects sont détaillés par l'intermédiaire des résultats présentés dans le chapitre 4.

3 La qualité perçue par l'utilisateur

Les applications constituent le moteur du développement des techniques QoS. C'est pourquoi il est vital de regarder les applications du point de vue de leur interaction avec le réseau et de corréler les paramètres QoS du réseau avec la qualité perçue par les utilisateurs, l'UPQ.

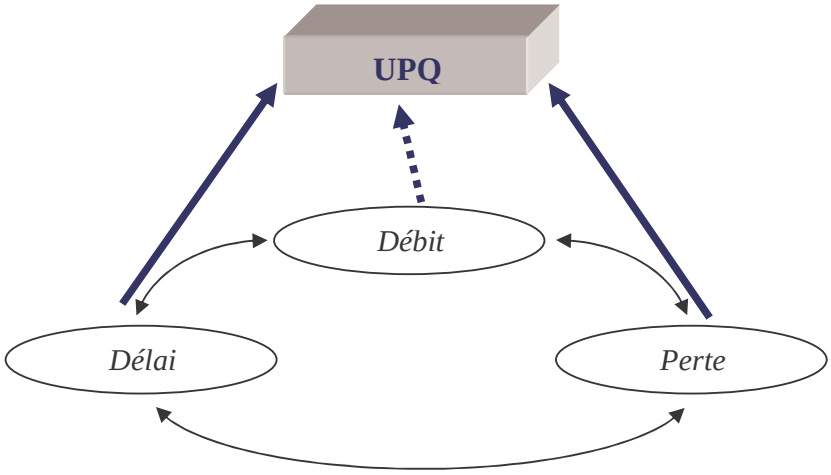


Figure 15 : L'impact de la QoS au niveau de l'utilisateur, en tant que UPQ.

La mesure de cette qualité est fonction de l'application, et les métriques appropriées doivent être définies pour tous les types d'applications. Avoir une information concrète sur la façon dont l'UPQ change en fonction de la variation des paramètres QoS est donc une question très importante. Des études sur les exigences que les applications imposent sur le réseau ont déjà été publiées, mais la dépendance exacte qui en résulte n'a pas été étudiée à notre connaissance de manière exhaustive.

3.1 Classes d'applications

Si on divise les applications réseau en classes, il devient possible de travailler avec elles, au lieu de considérer chaque application séparément. Cette division est utile avant tout dans l'étape de définition des métriques. Puisque pour déterminer l'influence exacte que les conditions dans le réseau ont sur une application, on doit effectuer des expériences pour cette application particulière.

On peut distinguer les classes d'application suivantes :

- Applications unidirectionnelles ;

- Applications unidirectionnelles avec contraintes temporelles ;
- Applications bidirectionnelles ;
- Applications bidirectionnelles avec contraintes temporelles.

Pour chaque classe d'application nous allons d'abord examiner l'influence de paramètres QoS et ensuite suggérer des métriques pour l'UPQ.

On peut aussi diviser les applications en deux types, en fonction de la manière dont elles utilisent les ressources du réseau et concernant leur adaptabilité:

- Applications élastiques, qui s'adaptent aux conditions dans le réseau ;
- Applications inélastiques, qui ne s'adaptent pas aux conditions dans le réseau.

3.1.1 Applications unidirectionnelles

Dans cette classe nous incluons les applications qui sont de manière générale unidirectionnelles. Des exemples typiques sont les applications de type transfert de données par TCP, telles que celles qui s'appuient sur les protocoles FTP (pour le transfert de fichiers) ou HTTP (pour l'accès Web).

Pour le FTP, les paramètres liés au délai ne sont pas particulièrement importants, encore que le temps d'achèvement du transfert puisse être l'un des paramètres utilisés pour juger la qualité perçue. Par contre l'influence de la perte de paquets est assez significative, premièrement à cause de son effet sur le temps d'achèvement, supérieur à l'effet du délai même, et deuxièmement dû à l'influence sur l'efficience d'utilisation de la bande passante du réseau. Pour une telle application le débit utile moyen (le nombre total de bits d'information reçus à la destination divisé par le temps de transfert) est une métrique UPQ importante.

Même si le HTTP est à plusieurs égards similaire au FTP – étant donné que les requêtes ont d'habitude une taille plus petite que les réponses (qui peuvent représenter des images, du son et d'autre formes de multimédia) –, le délai semble être plus important du point de vue de l'utilisateur étant donnée l'interactivité des applications Web. Même en l'absence de contraintes de temps réel, le temps de réponse d'un serveur Web peut être considéré comme un paramètre UPQ important dans ce cas.

3.1.2 Applications unidirectionnelles avec contraintes temporelles

Dans cette catégorie, l'exemple typique est la diffusion en continu. Une telle application, purement audio ou plus généralement multimédia (e.g. MPEG-2, MPEG-4), dépend essentiellement de la perte de paquets. Le délai et sa variation sont importants dans la mesure où ils doivent être compensés par l'utilisation d'une mémoire tampon. La diffusion de vidéo en continu a des exigences particulièrement élevées sur la bande passante, comme précisé dans le tableau 3.

Application	***Nécessaire de bande passante [Mbps]***	***Version MPEG***
Vidéo dans une fenêtre	< 1,5	MPEG-1
Qualité VHS (plein écran)	1 – 2	MPEG-2
Diffusion NTSC	2 – 3	MPEG-2
Diffusion PAL	4 – 6	MPEG-2
PAL professionnel	8 – 10	MPEG-2
Diffusion HDTV	12 – 20	MPEG-2
HDTV professionnel	32 – 40	MPEG-2
NTSC brut	168	*Non comprimé*
PAL brut	216	*Non comprimé*
HDTV brut	1000-1500	*Non comprimé*

Tableau 3 : La bande passante requise pour la diffusion vidéo en continu.

3.1.3 Applications bidirectionnelles

Les applications bidirectionnelles sont celles pour lesquelles les deux directions de communication sont également significatives. Des exemples de ce type sont : NFS (Network File System), DNS (Domain Name System), transferts HTTP courts, l'accès à distance (telnet). Généralement elles ne sont pas des applications en temps réel, mais imposent des demandes d'interactivité chaque fois qu'un utilisateur est directement impliqué.

A noter que pour l'accès à distance il existe des versions en mode graphiques, tel que le TightVNC, cas où il y a disparité entre les quantités de données circulant dans

chacune des deux directions, et où l'application devient essentiellement unidirectionnelle.

3.1.4 Applications bidirectionnelles avec contraintes temporelles

Ce type d'application est probablement le plus spectaculaire et le plus exigeant en termes de contraintes imposées aux réseaux. Ces applications font usage des flux RTP/UDP (par exemple H.323). En fonction de la quantité de données transférée (c.-à-d. la bande passante nécessaire) on peut déceler deux sous-catégories :

- Application avec un usage réduit de bande passante (e.g. la téléphonie IP) ;
- Applications avec un usage plus élevé de bande passante (e.g. la visioconférence).

La téléphonie IP (nommée aussi Voice over IP, VoIP) est l'une des applications qui commence être de plus en plus utilisée, stimulant les efforts de fourniture de QoS. Son interactivité est telle que les utilisateurs percevront d'infimes variations des paramètres QoS. On dit que la perte est plus importante que la gigue qui est à son tour plus importante que le délai, mais une étude plus approfondie était nécessaire et nous l'avons faite.

Déterminer l'UPQ pour VoIP n'est pas une tâche facile, mais UIT-T a développé un cadre pour l'évaluation subjective et objective de la qualité vocale, basée sur la qualité perceptuelle auditive. Ces standards, qui peuvent être utilisés aussi pour le VoIP, sont [ITU-800], [ITU-861], [ITU-107] et [ITU-P.862] (voir aussi 3.4.2).

La visioconférence (Video Teleconferencing, VTC) est une application de nature similaire qui envoie aussi des données vidéo. Plusieurs standards existent dans ce cas aussi : H.323, H.322, H.261, H.263 etc. Les demandes de bande passante sont plus grandes que pour VoIP, mais pas aussi significatives que pour la diffusion vidéo en continu (inférieures à 384 kbps). Des nouveaux problèmes apparaissent par rapport à la diffusion audio seule, comme la synchronisation entre vidéo et audio (par exemple pour les lèvres, où la désynchronisation ne doit pas dépasser 1-2 trames vidéo, c.-à-d. environ 50 ms). Une recommandation qui peut être utilisée pour une évaluation subjective de l'UPQ pour la vidéo est [ITU-J.143].

3.1.5 Applications élastiques

Les applications élastiques sont celles dont la base est constituée de protocoles adaptatifs du type TCP. Ce protocole essaye d'occuper une partie aussi large qu'il peut gérer de la bande passante disponible. Le taux de transmission est adapté aux conditions réseau courantes en utilisant un mécanisme de contrôle et évitement de la congestion [Jac-88] (voir 3.3.1 pour plus de détails). Il s'agit donc des applications dont le trafic est régulé.

3.1.6 Applications inélastiques

Les applications inélastiques sont, par exemple, celles qui sont fondées sur des protocoles de diffusion en continu de type UDP. Ce type de protocole utilise une bande passante fixe et n'a pas des mécanismes de correction des erreurs intrinsèques, donc le trafic est non-régulé. Il ne peut pas s'adapter aux conditions courantes dans le réseau, donc ses exigences QoS sont strictes.

3.2 Valeurs de performance UIT-T

UIT-T a essayé d'établir des limites sur les paramètres QoS qui sont censés assurer une bonne UPQ pour quelques classes d'applications [ITU-1541]. Ces valeurs de performance IP doivent être réalisées pour les paramètres de performance IP par l'utilisation de techniques d'ingénierie du réseau appropriées (routage à base de contraintes etc.) et gestion des files d'attente (séparation par priorité, mécanismes d'ordonnancement etc.).

Les six classes de service sont :

- 0 – applications de temps réel, sensibles à la gigue, avec une haute interactivité (e.g. VoIP, VTC) ;
- 1 – applications de temps réel, sensibles à la gigue, interactives (e.g. VoIP, VTC de qualité basse) ;
- 2 – applications très interactives, données de transaction (e.g. signalisation) ;
- 3 – applications interactives, données de transaction ;
- 4 – applications nécessitant un niveau réduit de perte de paquets (e.g. transactions courtes, transferts massifs, diffusion vidéo en continu) ;

- 5 – applications traditionnelles « meilleur effort ».

Le tableau qui suit présente, pour plusieurs applications, les valeurs de performance IP pour les paramètres de performance UIT-T en accord avec les classes de service correspondant à chaque application. Les valeurs indiquées sont des limites supérieures sur les valeurs moyennes, sauf pour le IPDV où la limite est sur la différence entre le centile $1 - 10^{-3}$ de l'IPTD et l'IPTD minimum.

	Applications en temps réel	***VoIP***	***WWW, service « meilleur effort »***	***Vidéo en continu (qualité VHS)***
IPTD	150 ms	400 ms	*Non défini*	400 ms
IPDV	50 ms	50 ms	*Non défini*	17 ms
IPLR	10^{-3}	10^{-3}	*Non défini*	10^{-5}
IPER	10^{-4}	10^{-4}	*Non défini*	10^{-4}

Tableau 4: Valeurs de performance IP pour diverses applications.

3.3 Le transfert de fichiers

Le transfert de fichier est une des applications importantes dans un réseau, car il permet de communiquer des volumes importants de données à des emplacements lointains. A la base, le transfert des fichiers (par FTP par exemple), utilise le protocole TCP pour le transfert proprement dit ; il s'agit donc d'une application de type élastique.

En plus, ceci conduit au fait qu'il y a une relation étroite entre l'étude des transferts de fichiers et celle des applications Web, car le protocole utilisé sur le Web, le HTTP, se base aussi sur TCP. La différence principale concerne la quantité de données expédiée : alors que les fichiers transférés peuvent avoir des tailles très importantes, les pages Web,elles, ont d'habitude des dimensions plus petites.

3.3.1 Le protocole TCP

A la base du TCP se trouve le mécanisme de contrôle et évitement de la congestion défini dans [Jac-88]. Le transmetteur envoie des paquets et attend les confirmations de réception. Chaque fois qu'une série de paquets a été réceptionnée intégralement avec succès, la quantité de données transmise à la fois dans une salve (appelée fenêtre du protocole) est incrémentée. La fenêtre du TCP a une taille maximale qui peut limiter

artificiellement le débit, pour des raisons de gestion de la connexion, par exemple, dans le cadre du partage entre plusieurs utilisateurs. Par contre, si des pertes ont lieu (signalées par un manque de confirmation dans l'implémentation classique) la taille de la fenêtre décroît de moitie.

Ainsi le taux de transmission croît d'une façon linéaire dans les intervalles sans pertes et décroît d'une manière exponentielle s'il y a des pertes dans le réseau. Les principaux facteurs externes qui influencent la performance de ce mécanisme sont le temps d'aller-retour (RTT) et la capacité de la ligne. L'utilisation maximale de celle-ci est contrôlée, comme mentionné auparavant, par l'intermédiaire de la fenêtre TCP maximale.

Depuis 1988, une série d'améliorations ont été proposées, essayant d'adapter ce mécanisme aux réseaux actuels qui peuvent avoir un RTT grand (de l'ordre de centaines de ms) mais aussi un haut débit (1 Gbps et même plus). Notre travail, par contre, se centre sur l'étude du mécanisme classique, le plus répandu sur les ordinateurs en usage courant.

L'adaptabilité du TCP signifie qu'il peut opérer dans une large gamme de conditions réseaux ; cependant, pour arriver à une utilisation efficace du réseau et pour avoir des temps raisonnables de finalisation des opérations, certaines conditions doivent être satisfaites par le réseau.

Une des relations de base pour le TCP est celle qui donne la dimension optimale de la fenêtre de transmission, $W_{optimale}$, en fonction de la bande passante du goulot d'étranglement de la connexion[1], *BP*, et du temps d'aller-retour, RTT :

$$W_{optimale} = BP \cdot RTT. \qquad (3.1)$$

Cette relation résulte de la façon dont le TCP transmet les paquets après la réception de confirmations. Ainsi, pour saturer la bande passante *BP*, il faut continuellement envoyer une quantité de données équivalente à $BP \cdot RTT$, car *RTT* est le temps nécessaire pour recevoir les confirmations.

[1] Il s'agit de la section du réseau, tout au long de la connexion, avec le débit maximal potentiel qui à la valeur minime pour cette connexion.

Une autre caractéristique importante du TCP, qui en explique l'utilisation massive, est qu'il fournit une façon de transférer de données avec confiance car il inclut le mécanisme de détection de pertes à base de confirmations de réception. Chaque fois qu'un paquet est considéré comme perdu, il est retransmis.

3.3.2 UPQ pour le transfert des fichiers

Pour les protocoles de transfert de fichiers nous proposons l'usage de deux métriques QoS : le débit utile et la performance de transfert.

3.3.2.1 Débit utile

Le débit utile (« goodput » en anglais), D_{utile}, quantifie l'efficacité du transfert au niveau du réseau. Il est calculé comme suit :

$$D_{utile} = \frac{B_{\min}[octets]}{B[octets]}, \qquad (3.2)$$

où $B_{\min}$ est le nombre minimum d'octets nécessaires pour le transfert (incluant les en-têtes Ethernet, IP, TCP et FTP) et B est le nombre d'octets transmis effectivement. B peut être supérieur à $B_{\min}$ lors de l'apparition des pertes de paquets dans le réseau, ce qui entraîne leur retransmission.

Les valeurs du débit utile sont sur une échelle de 0 à 1, 1 signifiant une efficacité maximale du transfert. Le débit utile décroît à cause des retransmissions des paquets quand il y a des pertes de paquets. Evidement D_{utile} ne dépend pas des paramètres temporels du transfert (e.g. la durée du transfert, le RTT), mais seulement du nombre d'octets transmis ; une autre mesure est donc nécessaire pour prendre cet aspect en compte.

3.3.2.2 Performance de transfert

La performance de transfert (Transfer-Time Performance, TTP) permet l'évaluation de l'efficacité temporelle d'un transfert :

$$TTP = \frac{T_{th}[s]}{T[s]} = \frac{B_{\min}[octets]}{L[bps] \cdot T[s]}, \qquad (3.3)$$

où T_{th} et la durée théorique du transfert et T est la durée mesurée. La durée théorique du transfert est le rapport entre le nombre minimum d'octets nécessaire pour un

transfert, B_{min}, et la bande passante de la ligne, L (dans notre cas 100 Mbps). T est calculé comme la différence entre l'instant où le dernier paquet d'un transfert a été reçu et celui auquel le premier paquet a été émis.

TTP a aussi des valeurs sur une échelle de 0 à 1, 1 désignant la performance optimale, idéale. Les délais de retransmission des paquets font diminuer le *TTP*. *TTP* dépend aussi indirectement de tous les paramètres qui influencent la durée du transfert, tel que le RTT, les mécanismes internes du TCP etc.

3.4 La téléphonie IP

Cette section présente certains principes de base liés au VoIP et à son utilisation. Puis nous présentons brièvement les métriques qui peuvent être utilisées pour calculer l'UPQ prédite pour les applications VoIP. En plus nous introduisons les méthodes d'évaluation de l'UPQ pour VoIP. Pour une discussion plus ample liée des questions spécifiques au VoIP voir la section 5.2.2.

3.4.1 Eléments de base

La communication vocale par Internet, i.e. la téléphonie IP, est actuellement une des applications réseau interactives les plus utilisées parmi celles qui impliquent directement la perception humaine. Pour désigner ce service, un terme anglais consacré est « Voice over IP » ou VoIP. Même si les exigences de VoIP par rapport à la bande passante sont relativement réduites, d'habitude en dessous de 64 kbps de données utiles, son interactivité implique une sensibilité accrue au délai et à la gigue. La perte de paquets a aussi une influence importante sur la performance de VoIP.

La raison pour laquelle VoIP est une application intéressante de notre point de vue est sa facilité de déploiement, ce qui fait qu'elle peut se répandre très rapidement. Donc, avec un minimum d'effort et de dépenses, on peut utiliser l'infrastructure réseau existante pour faire des appels téléphoniques essentiellement gratuits. Par contre, afin que cette alternative au système téléphonique traditionnel soit largement acceptée de la part des utilisateurs, il est indispensable que la qualité de service offerte soit équivalente.

Une des différences principales entre les réseaux informatiques et les réseaux téléphoniques est que les premiers sont commutés au niveau des paquets, tandis que les autres sont commutés au niveau des circuits. D'un côté, dans un réseau

informatique les applications « tournant » aux extrémités des connexions produisent des paquets qui sont expédiés à travers le réseau. Conceptuellement, ces paquets font partie des flux de données logiques de chaque application ; cependant une fois qu'ils ont pénétré le niveau de commutation, ils sont traités de manière indépendante. Pour chaque paquet, des décisions de routage sont prises et le paquet est expédié vers sa destination. Par la suite, les paquets transportant la voix vont partager le réseau (et les ressources associées) avec les paquets d'autres applications et vont tous être traités de manière indépendante.

D'un autre côté, les réseaux téléphoniques étant commutés au niveau des circuits, une fois que la connexion est faite, le circuit créé est réservé pour cette communication. Cela signifie qu'une certaine quantité de ressources est allouée au circuit correspondant, de sorte qu'il n'y a pas d'interférences avec les autres communications. En outre, une fois que le circuit est établi, on a une très haute probabilité[1] de n'avoir aucune interruption ou variation significative de la qualité pendant toute la durée de la communication.

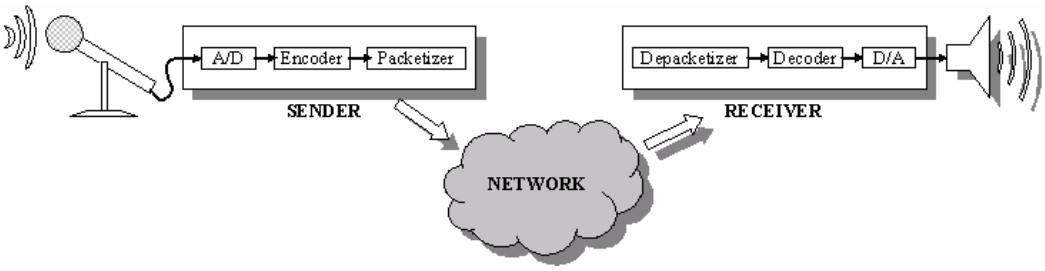


Figure 16 : La route de bout en bout pour une communication VoIP.

La figure 16 montre une description de la route de bout en bout pour une communication VoIP. Un dispositif d'entrée audio, tel qu'un microphone, est requis au transmetteur. A cause de la commutation par paquets des réseaux informatiques, les données audio doivent être codées et divisées en paquets avant la transmission. Le codage, ainsi que le décodage, est fait par des codeurs qui transforment les échantillons de voix en une représentation spécifique au niveau du réseau ou vice

[1] Le standard industriel, connu aussi sous le nom de « cinq neufs », demande une disponibilité de 99,999%, ce qui est équivalent à environ 5 minutes de temps en panne par système et par année [Mit-03].

versa. La plupart des codeurs est définie par des standards de UIT-T. Chacun a des propriétés différentes concernant la bande passante nécessaire, mais aussi par rapport à la qualité du signal vocal obtenu. Les codeurs que nous avons testés sont G.711, G.726, GSM et G.729. Le tableau 5 indique les paramètres principaux des codeurs les plus utilisés conformément à [Cis-03] (le score moyen d'opinion, MOS, représente la qualité perceptuelle moyenne obtenue pour le codeur correspondant, voir 3.4.2.1).

Standard de compression	***Nécessaire de bande passante [kbps]***	***Délai de compression [ms]***	***Score moyen d'opinion [0-5]***
G.711	64	0,75	4,1
G.723	5,3 – 6,3	30	3,65 – 3,9
G.726	32	1	3,85
G.728	16	3-5	3,61
G.729a	8	10	3,7

Tableau 5 : Vue d'ensemble de la compression pour VoIP.

Après le codage et la division en paquets de l'information binaire au niveau du transmetteur, les paquets contenant la voix sont expédiés à travers le réseau. Comme mentionné auparavant, les paquets de VoIP interagissent dans le réseau avec les paquets d'autres applications qui sont routés par des connexions partagées vers leur destination.

Au point de réception, les paquets sont déballés et décodés. Le décodage peut être suivi par d'autres étapes aussi, la plus typique étant la compensation de gigue. D'autres exemples sont la correction d'erreurs et la dissimulation de perte de paquets. Le flux de données numériques est ensuite converti dans une forme analogique et joué sur un dispositif de sortie, typiquement un haut-parleur.

A noter que pour la communication VoIP, qui est bidirectionnelle, la même route existe en direction opposée. Une interaction entre ces routes apparaît uniquement si l'annulation d'écho est employée.

A présent, on trouve des téléphones IP de forme très similaire à celle des téléphones habituels ; mais, au lieu d'être connectés à une prise téléphonique, ils sont connectés à une prise réseau. Par conséquent, l'acte de faire un appel téléphonique en utilisant

VoIP peut être identique à celui d'utiliser un téléphone normal[1]. Par contre, la qualité de la communication même peut être assez différente et elle est l'aspect le plus important de la transition de réseaux téléphoniques standard à la téléphonie par Internet.

Une raison autre que financière pour une telle transition est le fait que la communication par VoIP est plus flexible que la téléphonie standard. En faisant le choix approprié du codeur on peut contrôler la bande passante requise et déterminer la qualité intrinsèque associée de la communication[2]. En outre, en gérant de manière adéquate le réseau on peut utiliser le codeur qui fournit le niveau désiré de qualité, car les ressources du réseau peuvent être allouées de manière différente pour des utilisations différentes.

Néanmoins, étant donné que le canal de communication n'est pas réservé mais partagé avec d'autres applications, les paquets contenant la voix peuvent arriver au récepteur avec un espacement entre paquets différent de celui qu'ils avaient au départ, dans le mauvais ordre, certains pouvant même être perdus. Evaluer la relation précise entre ces facteurs, quantifiés par le moyen des paramètres QoS, et l'UPQ pour les communications VoIP a été un des buts de notre recherche (voir 5.4).

3.4.2 Evaluation de l'UPQ pour VoIP

Les réseaux de télécommunication modernes fournissent un large ensemble de services vocaux utilisant beaucoup de systèmes de transmission. Le déploiement rapide des technologies numériques en particulier a conduit à un besoin accru d'évaluation des caractéristiques de transmission des nouveaux équipements de communication.

UIT-T a défini plusieurs standards qui permettent une évaluation de la qualité de la communication vocale qui seront décrits ici brièvement, en ordre chronologique. La première a été une métrique subjective, ultérieurement des tentatives successives ont été faites pour définir des métriques objectives aussi.

---

[1] A noter tout de même que, d'habitude, les utilisateurs sont devant leurs ordinateurs et emploient de casques avec microphone pour réaliser les appels VoIP.

[2] En principe on peut effectuer une communication vocale à la qualité perceptuelle d'un CD audio. Mais il y a des contraintes à cause de la disponibilité usuelle en matière de bande passante.

3.4.2.1 Score moyen d'opinion (MOS)

En 1996, UIT-T a défini la méthodologie de déterminer le degré de satisfaction concernant une certaine ligne téléphonique, sous le nom de score moyen d'opinion (Mean Opinion Score, MOS) [ITU-800]. Cette méthode peut s'appliquer à toute forme possible de dégradation : perte de paquets, distorsions de circuit, erreurs de transmission, distorsions environnementales, écho, distorsions de codage etc.

La procédure d'évaluation est basée sur des tests subjectifs dans lesquels la qualité est notée par un « panel » d'expérimentateurs. Les valeurs suivantes sont assignées en fonction de la qualité perçue de la connexion :

$$Excellent = 5;\ \ Bien = 4;\ \ Acceptable = 3;\ \ Pauvre = 2;\ \ Mauvais = 1. \qquad (3.4)$$

On fait la distinction entre le score lié à la conversation, MOS_C, et le score lié à la simple écoute (« listening » en anglais), MOS_L. Dans le deuxième cas, seule la qualité intrinsèque est prise en compte, tandis que dans le premier cas le score inclut l'opinion de l'expérimentateur par rapport au niveau d'interactivité.

3.4.2.2 Mesure de la qualité perceptuelle vocale (PSQM)

L'étape suivante a été la standardisation en 1998, toujours par UIT-T, d'une méthode objective pour la mesure de la qualité des codeurs vocaux dans la bande téléphonique, nommée mesure de la qualité perceptuelle vocale (Perceptual Speech Quality Measure, PSQM) [ITU-861]. Cette méthode a été initialement développée par la société KPN aux Pays-Bas. Après l'avoir comparée à d'autres métriques, UIT-T a tiré la conclusion que les résultats de PSQM sont les plus corrélés avec la qualité subjective de la voix codée.

PSQM est conçu pour des tests d'écoute seulement. Il peut être utilisé pour des codeurs ayant des débits supérieurs à 4 kbps, mais il n'y a pas assez d'information concernant sa performance par rapport à certains facteurs, tel que les erreurs de canal de transmission, et il ne doit pas être utilisé en conjonction avec le délai, par exemple.

PSQM utilise des représentations psychophysiques[1] qui sont aussi près que possible des représentations internes humaines pour les signaux vocaux. Une valeur zéro de PSQM signifie qu'aucune dégradation n'est présente, et une valeur de 6,5 signifie un canal totalement inutilisable. Les valeurs PSQM peuvent être transposées sur une échelle de type MOS afin de pouvoir obtenir une estimation de la qualité subjective.

3.4.2.3 Le modèle E

Le modèle E (E-model), apparu en 2000, est un modèle de calcul à utiliser dans la planification de transmission [ITU-107]. C'est un modèle d'appréciation de la qualité de transmission qui peut être utilisé pour garantir que les utilisateurs seront satisfaits par la performance de transmission de bout en bout. Le modèle intègre les facteurs de dégradation qui affectent l'équipement de transmission, incluant le délai et les codeurs bas débit. Ces dégradations sont calculées en fonction d'une série de paramètres d'entrée pour lesquels des valeurs par défaut et des gammes permises sont précisées. Ils doivent être utilisés si la situation de dégradation correspondante apparaît.

Le score MOS équivalent (de type conversation), sur une échelle de 1 à 4,5, peut être obtenu du facteur R, qui est le résultat du modèle E, par les formules suivantes :

$$\begin{cases} MOS = 1, & R < 0 \\ MOS = 1 + 0{,}035R + R(R-60)(100-R)\cdot 7\cdot 10^{-7}, & 0 \le R \le 100. \\ MOS = 4.5, & R > 100 \end{cases} \qquad (3.5)$$

Le graphique de la dépendance entre MOS et facteur R est donné ci-dessous. A noter que le score maximum qu'on peut obtenir dans ce cas est 4,5, le score moyen qui résulte typiquement de la suite de tests subjectifs pour une qualité excellente, car il est connu que les notes d'expérimentateurs varient entre 4 et 5 dans ces conditions.

[1] La psychophysique est une branche de la psychologie qui étudie l'effet des processus physiques (comme l'intensité de stimulation) sur les processus mentaux d'un organisme humain.

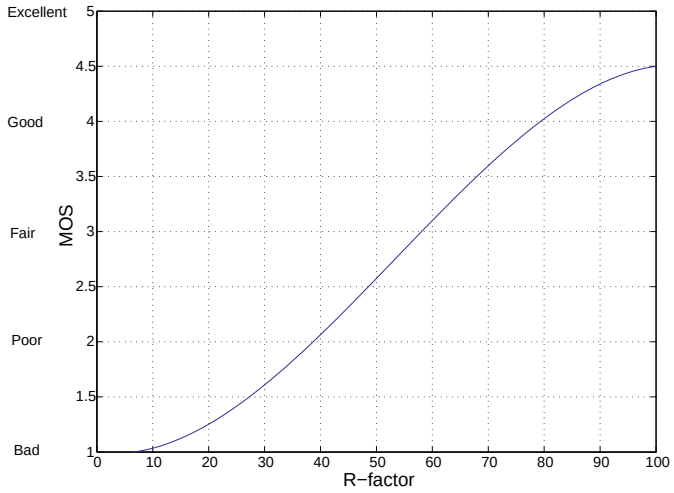


Figure 17 : MOS comme fonction du facteur R.

La correspondance entre satisfaction des utilisateurs et facteur R est donnée dans le tableau 6.

Facteur R (valeur inférieure)	***MOS (valeur inférieure)***	***Satisfaction des utilisateurs***
90	4,34	Très satisfaits
80	4,03	Satisfaits
70	3,60	Certains satisfaits
60	3,10	Beaucoup non satisfaits
50	2,58	Presque tous non satisfaits

Tableau 6 : Correspondance entre facteur R, score MOS et satisfaction des utilisateurs.

3.4.2.4 Système d'analyse/mesure perceptuelle (PAMS)

Le système d'analyse/mesure perceptuelle (Perceptual Analysis/Measurement System, PAMS) est une méthode développée par British Telecom pour déterminer la qualité vocale anticipée pour un système de transmission à travers n'importe quel réseau, y compris ceux où apparaissent des pertes de paquets ou une variation de délai [PAMS].

Le processus de calcul de PAMS utilise un modèle qui combine une description mathématique des propriétés psychophysiques de l'ouïe humaine avec une technique d'analyse qui prend en compte la subjectivité des erreurs dans la perspective de la perception humaine. Le processus PAMS compare deux versions d'un même signal,

original et dégradé, et détermine des prédictions MOS, sur une échelle de 1 à 5, pour la qualité de l'écoute et l'effort d'écoute.

Des tests subjectifs extensifs ont été menés avec des sujets humains sur une large variété de technologies de transport de la voix pour valider le modèle PAMS. L'algorithme a été vérifié dans une large gamme de conditions, incluant les codeurs à bas débit. Il en est résulté que le score calculé par PAMS est proche à 0,5 près du MOS typique déterminé par des tests subjectifs contrôlés de laboratoire.

3.4.2.5 Evaluation perceptuelle de la qualité vocale (PESQ)

En 2001, UIT-T a rendu public son système d'évaluation perceptuelle de la qualité vocale (Perceptual Evaluation of Speech Quality, PESQ) [ITU-862], une méthode objective de prédiction de la qualité subjective pour la téléphonie avec bande passante réduite et codeurs vocaux. PESQ combine les meilleurs aspects de PSQM et PAMS, étant développé en commun par KPN et British Telecom. Comparé au PSQM, PESQ prend en compte, en plus, le filtrage, le délai variable, les distorsions de codage et les erreurs de canal.

Suite à des tests menés de façon extensive, PESQ s'avère fournir des précisions acceptables dans les types d'applications suivants : l'évaluation et la sélection des codeurs, tests dans des réseaux actifs (avec connexions numériques ou analogiques pour la communication VoIP), tests des réseaux émulés et prototype.

Le processus clé de PESQ, comme pour les méthodes apparentées, est la transformation des signaux original et dégradé en représentations psychophysiques proches de celles des signaux auditifs du système auditif humain.

Le score PESQ est produit sur une échelle similaire au MOS, avec des valeurs situées entre -0,5 et 4,5. Les valeurs habituelles sont entre 1,0 et 4,5, les scores MOS obtenus dans des expériences subjectives sur la qualité de l'écoute.

En accord avec [Ser-01] et l'interprétation du MOS [ITU-800], la relation entre les scores PESQ et la qualité audio est la suivante :

- Des scores PESQ entre 3 et 4,5 désignent une qualité perçue acceptable (avec 3,8 comme seuil de la qualité dans les systèmes téléphoniques traditionnels) – on va se référer à ce niveau comme qualité « bonne » ;

- Des valeurs entre 2 et 3 indiquent qu'un effort est nécessaire pour la compréhension du parler – on va se référer à ceci comme qualité « basse »;
- Scores inférieurs à 2 signifient que la dégradation a rendu la communication très difficile ou même impossible, par conséquent la qualité est « inacceptable ».

Etant la métrique la plus récente, PESQ est supérieure aux précédentes, tel PSQM et PAMS. A la différence du modèle E, elle n'exige pas de connaissance préalable du réseau et n'utilise que les signaux original et dégradé pour le calcul du score PESQ. Par la suite nous le considérons comme la solution optimale pour nos expériences, qui font usage d'un émulateur réseau. Pour calculer le score PESQ nous avons utilisé dans notre travail une implémentation fournie par Malden Electronics, Ltd. [Malden].

4 Mesure active de la qualité

La mesure active de la QoS consiste à générer un certain trafic qui est injecté dans des éléments de réseau afin de permettre l'analyse de leur réponse. Le but principal de cette méthode est l'évaluation des éléments réseau pour déterminer l'adéquation de leur déploiement dans des conditions spécifiques.

Une série de tests a été effectuée sur divers commutateurs pour déterminer leurs propriétés et l'adéquation de leur utilisation pour le réseau employé au niveau 2 de l'expérience ATLAS au CERN [ATLAS].

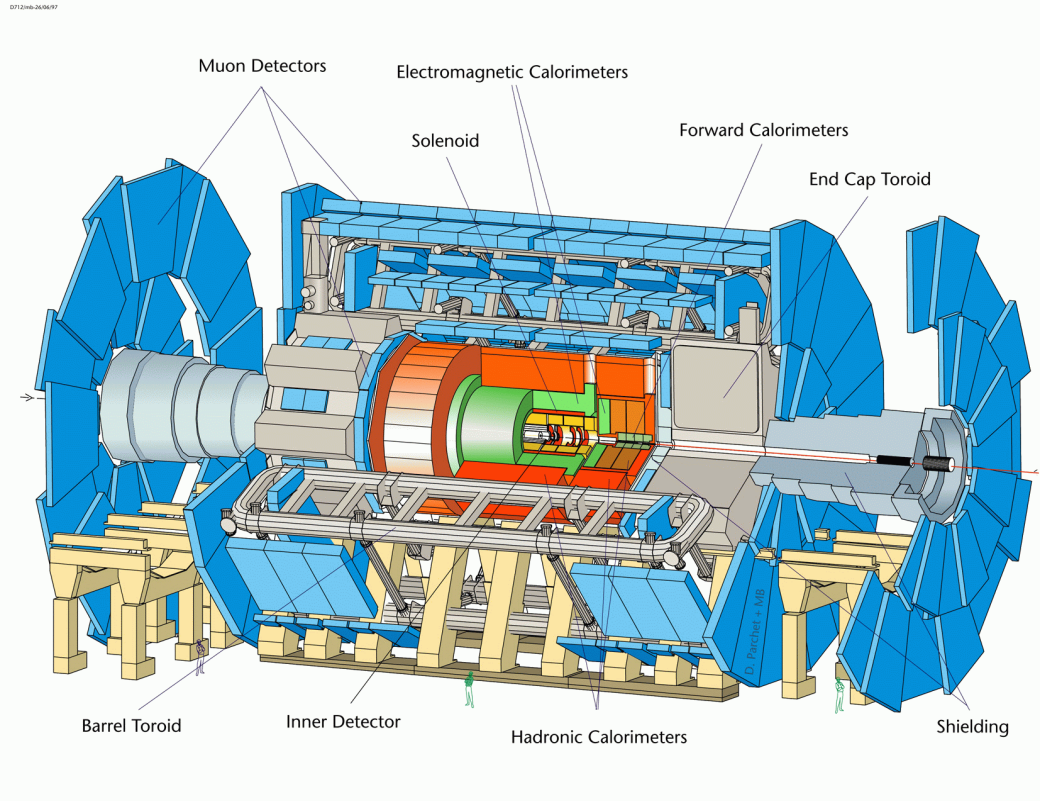


Figure 18 : Image en section de l'expérience ATLAS sur l'accélérateur LHC de CERN.

4.1 Système de test à base de FPGA

Le système de test à base de FPGA[1], nommé Enet32, a été conçu et réalisé au CERN pour tester des réseaux FastEthernet [Bar-01].

[1] Field Programmable Gate Array, technologie de circuits numériques programmables à très haute intégration.

4.1.1 Architecture du système

Ce système implémente 32 ports FastEthernet (100 Mbps) full-duplex en utilisant des FPGAs Altera Flex 10K pour l'implémentation des contrôleurs d'accès au medium (MAC) ainsi que les autres fonctions décrites ici. Tous les FPGA sont programmés presque exclusivement en Handel-C [Celox], [Beu-01], un langage de programmation de haut niveau disponible commercialement. La décision de construire ces systèmes au lieu d'utiliser des testeurs réseau commerciaux a eu certaines raisons financières ; mais la motivation la plus importante a été celle de pouvoir dans une phase ultérieure générer du trafic identique à celui qui sera présent dans le système d'acquisition de données d'ATLAS, ce qui n'est pas possible en utilisant des testeurs du commerce. L'architecture du système est présentée sur la figure 19.

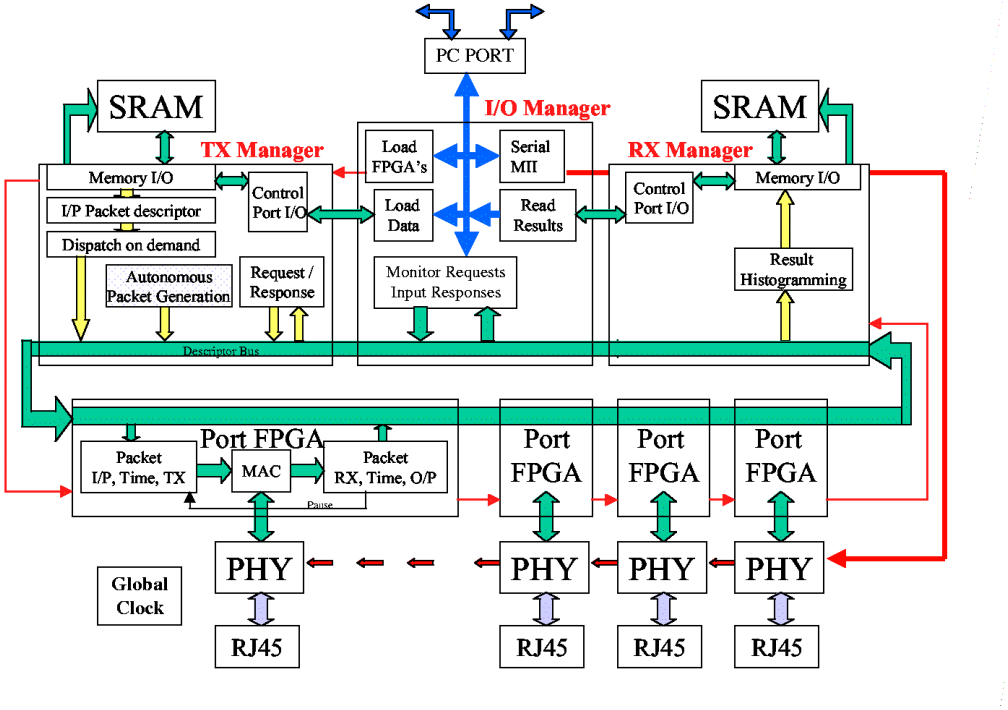


Figure 19 : Architecture du système Enet32.

4.1.1.1 Composantes

La connexion avec le PC hôte est assurée par une connexion parallèle de type IEEE 1284. La gestion de cette connexion sur la carte de test est prise en charge par un FPGA (IoMan) qui envoie des données et commandes depuis le PC hôte vers les différentes composantes de la carte ; il gère également les données destinées au PC hôte. Un anneau à créneaux (« slotted ring » en anglais) à 25 Mhz interconnecte les 32

MACs, IoMan ainsi que deux autres FPGA : le gestionnaire de transmission de paquets (TxMan) et le gestionnaire de réception de paquets (RxMan). L'anneau à créneaux a une largeur de 52 bits, dont 32 sont utilisés pour le transport de données et le reste pour les commandes et signaux d'état, ainsi que l'horloge globale de 25 MHz.

A chaque MAC, on a associé de manière statique un créneau sur l'anneau avec une taille de 5 mots. Pendant l'opération, la fonction de l'anneau à créneaux est de transférer les descripteurs de paquets depuis TxMan vers chacun des MAC, les uns après l'autres. Un descripteur contient suffisamment d'information pour que les MACs puissent générer une trame Ethernet à la sortie (e.g. taille de paquets, adresses source et destination). Mais le contenu exact du descripteur de transmission dépend de la mesure effectuée. TxMan a une mémoire privée statique RAM de 1M mots (avec des mots de 36 bits chacun) qui est utilisée pour stocker les descripteurs de transmission générés par le PC hôte. La largeur de bande fournie par l'anneau à créneaux est suffisante pour permettre à chaque MAC de générer les plus petites trames Ethernet (64 octets) à la capacité maximale de la ligne (100 Mbps).

Les paquets qui arrivent sont traités par les MAC afin de produire des descripteurs de réception (5 mots à 32 bits) qui contiennent l'information essentielle extraite des trames reçues. Par exemple, les timbres-à-date contenus dans les trames reçues sont comparés à l'horloge courante afin de calculer le délai unidirectionnel. Ces descripteurs, placés dans l'anneau à créneaux par chaque MAC, sont reçus par RxMan qui utilise l'information qui lui parvient pour mettre à jour des compteurs et histogrammes de délai, de taille de paquets ou encore d'écart entre paquets, en fonction de la mesure effectuée. Les histogrammes sont stockés dans la mémoire statique privée RAM (1 M mots de 36 bits) avant d'être finalement téléchargés sur le PC hôte à travers la connexion IEEE 1284. La nature des histogrammes accumulés est déterminée par des registres de contrôle implémentés dans RxMan. La nature de l'information extraite par les MACs depuis les trames qui arrivent peut aussi être changée par re-programmation.

Les descripteurs de transmission sont produits par TxMan par une des deux voies suivantes. Les descripteurs peuvent être générés à partir de l'information stockée dans la mémoire de TxMan, en cyclant parmi ces données soit de manière séquentielle soit pseudo aléatoire. Une alternative est la suivante : à partir de descripteurs de réception, RxMan produit des descripteurs de transmission, qui sont ensuite envoyés au TxMan.

Cette méthode permet de reproduire un mécanisme de type requête-réponse et a été utilisée pour générer le comportement des ROBs dans l'expérience ATLAS.

Le traitement des descripteurs de transmission pour générer des trames sortantes est effectué par un processeur implémenté en Handel-C dans les MACs. Pour contrôler le comportement de ce processeur il est nécessaire d'assembler un programme sur le PC hôte, qui est envoyé en tant que données au MACs. Cette procédure est très rapide par comparaison avec le processus de fittage et re-programmation des FPGAs MAC. Toutes les trames sortantes contiennent un timbre-à-date basé sur l'horloge globale du système de 25 MHz, ce qui assure une précision de 40 ns.

Plusieurs systèmes Enet32 peuvent être synchronisés par l'utilisation d'un paradigme de type maître-esclave et une connexion en série entre les systèmes (par des câbles coaxiaux courtes), comme le montré la figure 20. La précision du système synchronisé est supérieure à 0,1 μs.

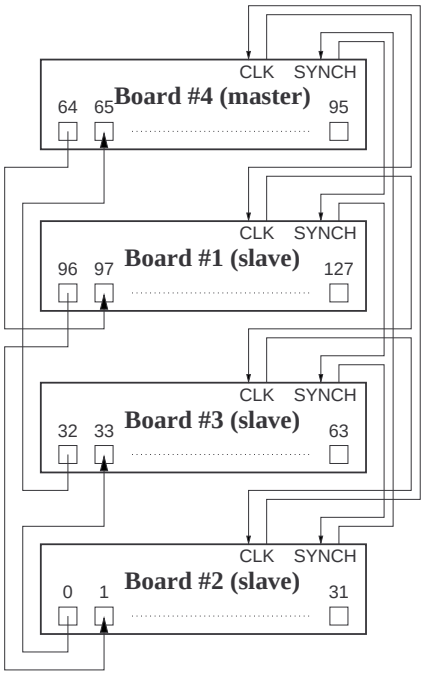


Figure 20 : Synchronisation de quatre systèmes Enet32.

4.1.1.2 Programmation de FPGAs

Handel-C est un langage de programmation matérielle de haut niveau conçu expressément pour générer des configurations pour FPGAs. La syntaxe du langage est très proche de celle du langage C, avec l'addition de variables de taille arbitraire. Le parallélisme est introduit par une syntaxe similaire à celle de Occam [Hoa-88],

incluant des canaux pour la synchronisation des processus parallèles. Le langage en soi est facile à apprendre pour les programmateurs en C, mais son parallélisme nécessite une façon de penser spécifique.

A la différence d'un processeur normal, plus ou moins séquentiel, dans Handel-C, pendant chaque cycle d'horloge, chaque processus exécute une instruction. Programmer un système en Handel-C implique deux étapes. Le compilateur Handel-C produit une liste de configurations qui constitue l'entrée de l'outil de fittage Altera MaxPlus [Altera]. Le code résultant est envoyé par la connexion IEEE 1284 au IoMan. Chacun des FPGAs est ainsi reprogrammable par le PC hôte, par l'intermédiaire de IoMan et d'une chaîne JTAG. IoMan lui-même, dont le comportement change rarement, est reprogrammé par la technologie JTAG avec un câble ByteBlaster. Il a aussi une mémoire reprogrammable de type EEPROM qui contient sa configuration initiale dès la mise sous tension.

4.1.1.3 Génération de trafic

Enet32 peut générer du trafic avec des propriétés variables, en fonction des descripteurs fournis. Voici deux exemples d'histogrammes des valeurs de l'écart entre le temps de réception des paquets pour le trafic généré. Les graphiques ont été obtenus en connectant directement deux ports du testeur, un qui transmet du trafic (paquets de 64 octets) et l'autre qui le réceptionne et calcule les histogrammes.

Il s'agit d'abord d'un taux constant de bits (CBR) avec un espacement de 12 μs entre le début des paquets (équivalent à un taux de 56 Mbps), qui conduit à un histogramme à une seule boîte (« bin » en anglais). Puis nous présentons une distribution exponentielle de l'écart entre le début des paquets, c.-à-d. un trafic de type poissonnien, avec un espacement moyen de 12 μs entre le début des paquets (équivalent toujours à un débit moyen de 56 Mbps).

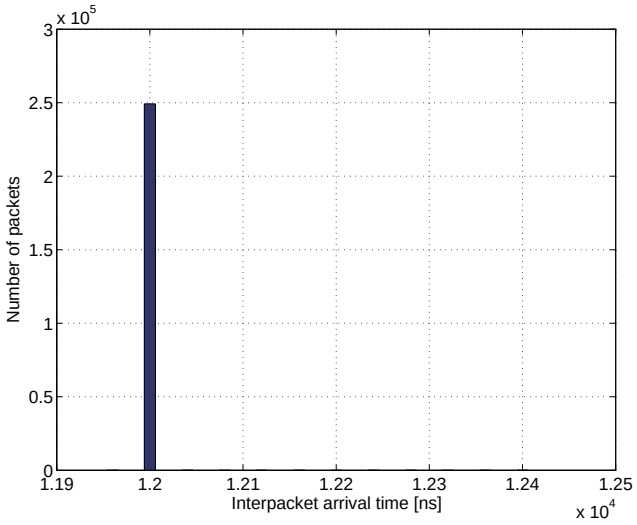


Figure 21 : Distibution CBR de l'écart entre la réception des paquets pour le trafic genéré par Enet32 (paquets de 64 octets, 12 µs espacement entre le début des paquets).

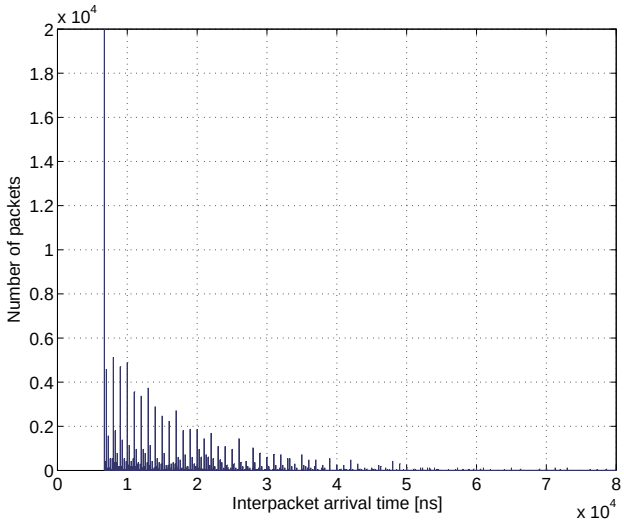


Figure 22 : Distibution de Poisson de l'écart entre la réception des paquets pour le trafic genéré par Enet32 (paquets de 64 octets, 12 µs espacement moyen entre le début des paquets).

4.1.2 Configuration de test

Les tests de ce type ont été effectués en utilisant la configuration indiquée sur la figure 23.

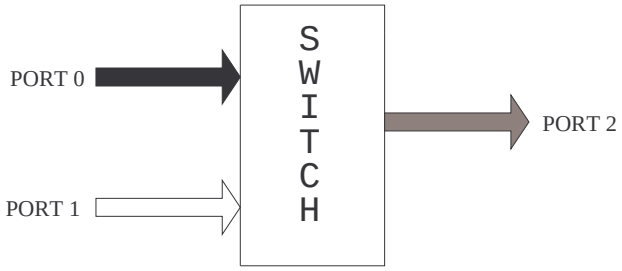


Figure 23 : Configuration de test pour deux priorités.

Les commutateurs FastEthernet testés permettent d'utiliser seulement deux priorités différentes, qu'on va noter « haute » et « basse ». Par conséquent on a deux ports source et un port destination. Chaque port source envoie du trafic avec une priorité différente dans l'étiquette VLAN, de manière qu'il soit classifié comme de priorité « haute » ou « basse » par le commutateur. Le taux de transmission varie de 0 à 100 Mbps et il est égal pour les deux ports qui transmettent. Le trafic des deux sources a la même destination, le port noté PORT 2.

4.1.3 Résultats des tests

Les résultats pour l'un des commutateurs sont présentés ci-dessous. Des paquets de 64 octets ont été utilisés pour obtenir ces résultats.

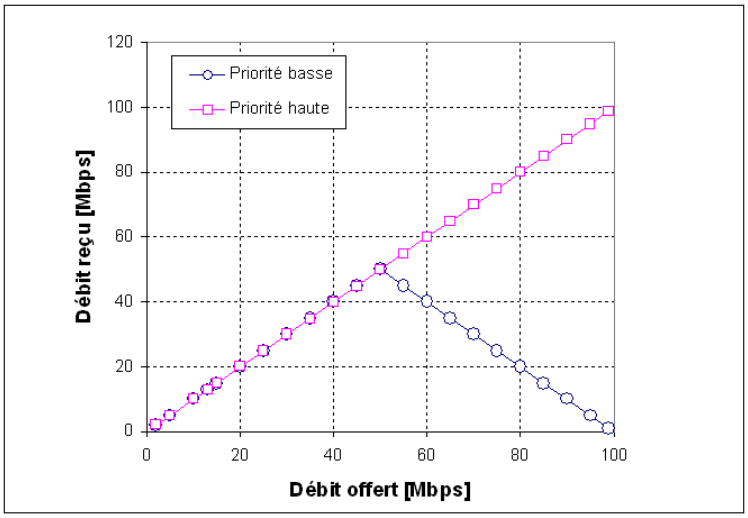


Figure 24 : Le débit reçu en fonction du débit offert pour les deux priorités de trafic.

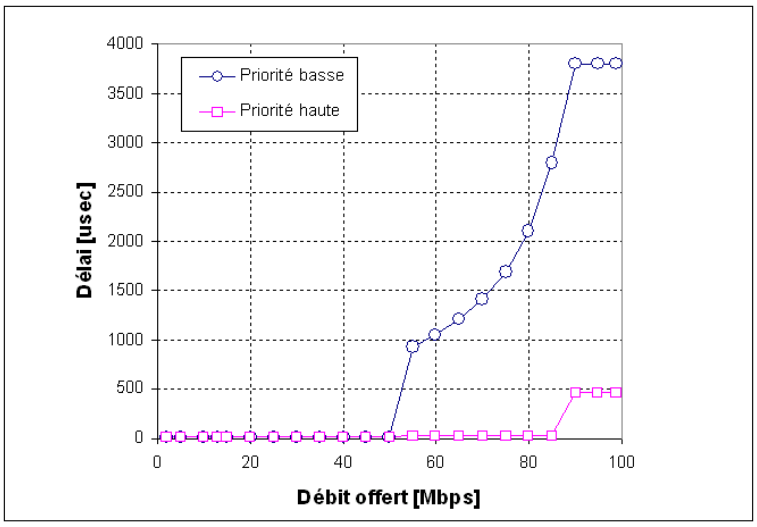


Figure 25 : Le délai unidirectionnel en fonction du débit offert pour les deux priorités de trafic.

On peut observer que, dès que la connexion est saturée (à 50 Mbps par transmetteur) le mécanisme d'ordonnancement de type SP est mis en évidence. Le trafic de priorité haute est transmis au détriment de celui de priorité basse. Le délai pour le trafic de haute priorité est assez réduit et croît seulement à plus de 85 Mbps, arrivant finalement vers 500 μs.

4.2 Système de test à base de cartes réseau programmables

La principale limitation du système décrit en 4.1 est le fait que les ressources du FPGA sont limitées ; par conséquent sa fonctionnalité ne peut pas être étendue de manière illimitée. Un autre système de test a été développé en parallèle pour les tests des réseaux FastEthernet et Gigabit Ethernet en utilisant des cartes réseau programmables Alteon [Alt-97], [Alt-99]. Il a été ultérieurement amélioré pour permettre des tests à niveau IP, ce qui a permis son utilisation pour des tests à grande distance (entre le CERN et Cracovie, Pologne, entre le CERN et Copenhague, Danemark, ou entre le CERN et Edmonton, Canada). De plus, ces cartes réseau ont pû aussi être reprogrammées pour monitorer le trafic réseau dans le cadre de la mesure passive présentée au chapitre 5.

4.2.1 Architecture du système

Le système consiste en un nombre variable de cartes réseau Alteon. Sur chaque PC qui contient de telles cartes s'exécute un logiciel responsable de la configuration et la collecte de résultats de chaque carte de ce PC, nommé *agent*. Sur un autre PC

s’exécute le logiciel central qui coordonne l’activité des agents et assure le contrôle et l’affichage centralisé des résultats. Le plus grand nombre de cartes réseaux qui a été intégré dans ce système est de 32, ce qui permet de tester des commutateurs de taille assez grande.

Il faut noter qu’il est possible d’intégrer dans ce système les cartes à base de FPGAs Enet32. Un agent différent assure dans ce cas la communication avec un numéro variable de cartes Enet32. Le nombre de cartes disponibles est de 4, ce qui conduit à un nombre maximal de 128 sources de trafic. En combinant ces deux systèmes, on réalise en pratique des scénarios très complexes.

4.2.1.1 Cartes réseau Alteon

L’architecture des cartes réseau Alteon est montrée sur la figure 26 [Alt-97]. Les cartes utilisent le microprocesseur Tigon 2 qui contient deux processeurs RISC-MIPS à 86 MHz. La mémoire de la carte (1MB) peut être utilisée par les deux processeurs et contient le code binaire ainsi que les structures de données utilisées pendant l’opération. L’accès à la mémoire de toutes les entités est régulé par un mécanisme de priorité. Les interfaces de réception et transmission des paquets ont la priorité la plus haute. Ensuite se situe l’accès par le PC hôte. Ce n’est pas vrai si il y a des transferts par DMA, mais ces transferts n’ont lieu que pendant le téléchargement du firmware sur la carte ou la lecture des statistiques ; les contrôleurs de DMA sont donc inactifs en temps normal. La carte réseau est contrôlée par le PC hôte par l’intermédiaire d’un driver modifié qui assure un accès transparent à la mémoire pour le code de l’utilisateur.

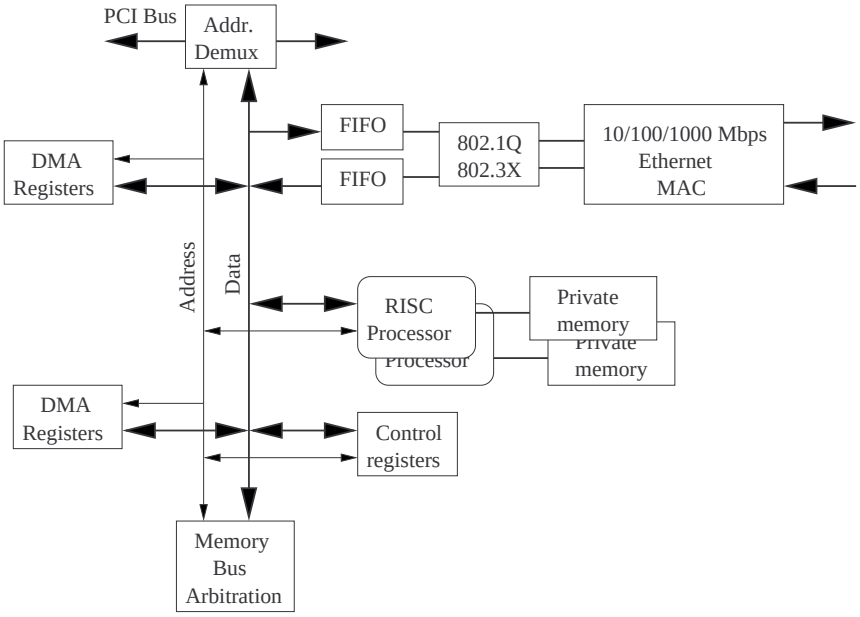


Figure 26 : Architecture interne des cartes réseau programmables Alteon.

4.2.1.2 Communication entre agents et serveur

La communication entre agents et serveur est représentée par l'automate à états suivant, quel que soit le type de l'agent (de contrôle d'une carte Alteon ou Enet32)

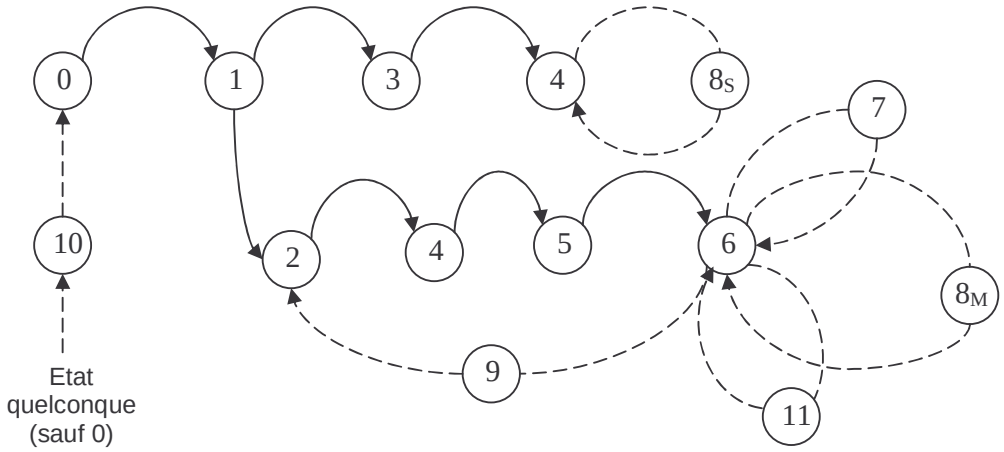


Figure 27 : Machine à états pour la communication entre clients et serveurs.

Chaque flèche représente une transition après la réception d'un message. La signification des états est la suivante :

- 0 = état initial ; le système part de cet état à la mise en route ;
- 1 = état après le message « get_architecture » qui demande à l'agent d'informer le serveur sur les capacités des cartes qu'il contrôle ; un message

réponse est envoyé, qui contient les informations requises et tient lieu de confirmation de réussite ;

- 2 = état après le message « set_master » qui configure l'agent comme « maître » du système de test (un maître peut configurer le test, le démarrer et l'arrêter, et lire les résultats) ; une confirmation est envoyée comme réponse en cas de réussite ;
- 3 = état après le message « set_slave » qui configure l'agent comme « esclave » du système de test (un esclave peut seulement lire les résultats); une confirmation est envoyée comme réponse en cas de réussite ;
- 4 = état après le message « select_ports » qui sélectionne les ports qui seront utilisés pendant le test ; une confirmation est envoyée comme réponse en cas de réussite ;
- 5 = état après le message « configure_test » qui réalise les configurations de la plateforme pour le test suivant; une confirmation est envoyée comme réponse en cas de réussite ;
- 6 = état après le message « start_test » qui démarre le test ; une confirmation est envoyée comme réponse en cas de réussite
- 7 = état temporaire après le message « clear » qui conduit à l'effacement des statistiques courantes ; une confirmation est envoyée comme réponse en cas de réussite et le système passe automatiquement dans l'état 6 ;
- 8_M = état temporaire après le message « update » pour un agent maître ; les statistiques sont envoyées au serveur (depuis une mémoire tampon) comme réponse en cas de réussite et le système passe automatiquement dans l'état 6 ; de plus la lecture (dans la mémoire tampon) des nouvelles statistiques depuis les cartes de test est entamée ;
- 8_S = état temporaire après le message « update » pour un agent esclave ; les statistiques sont envoyées au serveur (depuis la mémoire tampon) comme réponse en cas de réussite et le système passe automatiquement dans l'état 6 ; à noter que la lecture des nouvelles statistiques depuis les cartes de test n'est pas entamée dans ce cas ;

- 9 = état temporaire après le message « stop_test » qui arrête l'expérience courante ; une confirmation est envoyée comme réponse en cas de réussite et le système passe automatiquement dans l'état 2 ;
- 10 = état temporaire après le message « disconnect » qui conduit à l'interruption de la connexion entre l'agent et le serveur ; une confirmation est envoyée comme réponse en cas de réussite et le système passe automatiquement dans l'état 0 ;
- 11 = état temporaire après le message « throttle » qui permet le changement des certains paramètres du système (le taux de transmission, la taille des paquets) pendant le déroulement du test ; une confirmation est envoyée comme réponse en cas de réussite et le système passe automatiquement dans l'état 6.

Les transitions possibles s'effectuent uniquement comme indiqué par les flèches entre états montrées sur la figure 27.

4.3 L'émulateur de « Readout Buffer »

L'Enet32 est un système polyvalent par le fait qu'il est reprogrammable. Pour obtenir les résultats présentés dans la section 4.1 nous avons fait usage d'un trafic soit de type CBR, soit de type poissonnien. Ces types de trafic sont amplement utilisés pour les tests sur commutateurs. Mais le trafic qui sera produit dans l'expérience ATLAS sera de nature différente. C'est pour cette raison que le système a été légèrement modifié pour émuler les systèmes ROB qui font partie du réseau de niveau 2 d'acquisition de données (Trigger/Data Aquisition, TDAQ) [Lev-03]. La même modification a été possible pour les cartes réseau Alteon.

4.3.1 L'architecture de TDAQ

Ce niveau d'acquisition de données contient environ 1600 sources de données appelées mémoire tampon de lecture (Readout Buffer, ROB) qui reçoivent des fragments d'événement depuis le niveau 1 de l'expérience à une fréquence pouvant atteindre 75 kHz, avec 1,4 kB par événement. Un événement est affecté à l'un des membres de la ferme de traitement de niveau 2 (Level 2 Processing Unit, L2PU), qui

demande des fragments d'événement à certains ROBs[1]. Le L2PU applique un critère de sélection s'appuyant sur les fragments reçus et informe le gestionnaire de flux de données (Dataflow Manager, DFM). Si un événement est accepté par le niveau 2, le DFM l'affecte au système SFI (SubFarm Interface) qui collecte l'événement entier. L'acquisition se fait par des requêtes aux ROBs, et l'événement est par la suite envoyé à un élément du filtre d'événements (Event Filter, EF), l'étape suivante dans le processus de décision concernant la sauvegarde de l'événement ou son rejet.

Il n'est pas encore décidé si les ROBs seront connectés indépendamment dans le réseau TDAQ et si les noeuds L2PU et SFI communiqueront directement avec chaque ROB. Une autre version groupe les ROBs dans un système (Readout System, ROS) avec un gestionnaire propre, le contrôleur ROS. Les noeuds L2PU et SFI communiquent dans ce cas avec le contrôleur ROS et les réponses des ROBs sont agrégées dans une réponse ROS expédiée par le contrôleur ROS.

L'émulateur ROS utilise la plateforme du testeur Enet32, les FPGAs étant programmés pour répondre aux requêtes en provenance du système de collecte de données d'ATLAS et en accord avec les protocoles de communication spécifiques. Néanmoins, les données utiles dans les paquets ne sont pas réelles ; seule la taille de la réponse est significative pour ces mesures. Les requêtes sont traitées par un processus dans les FPGAs MAC et transformées en descripteurs sur 32 bits mis dans des files d'attente. Ces descripteurs sont lus par un processus parallèle responsable de la construction des trames de réponse sortantes. Leur contenu est majoritairement fixe, sauf environ 10% qui dépend de la requête qui a été faite.

L'émulateur ROS envoie des paquets de contrôle du flux pour s'assurer que les files d'attente de descripteurs ne débordent pas, et répond aux paquets de contrôle de flux reçus. La réponse de l'émulateur est extrêmement rapide, la transmission de la trame de réponse commençant moins d'une microseconde après la réception de la trame entière de requête, sauf si un message de contrôle de flux de type PAUSE a été reçu. L'émulateur est capable de fonctionner à la capacité maximale d'une connexion FastEthernet.

---

[1] Les ROBs sont contactés en fonction de l'information de niveau 1 associée à cet événement, avec un nombre moyen de ROBs contactés inférieur à 2% du nombre total.

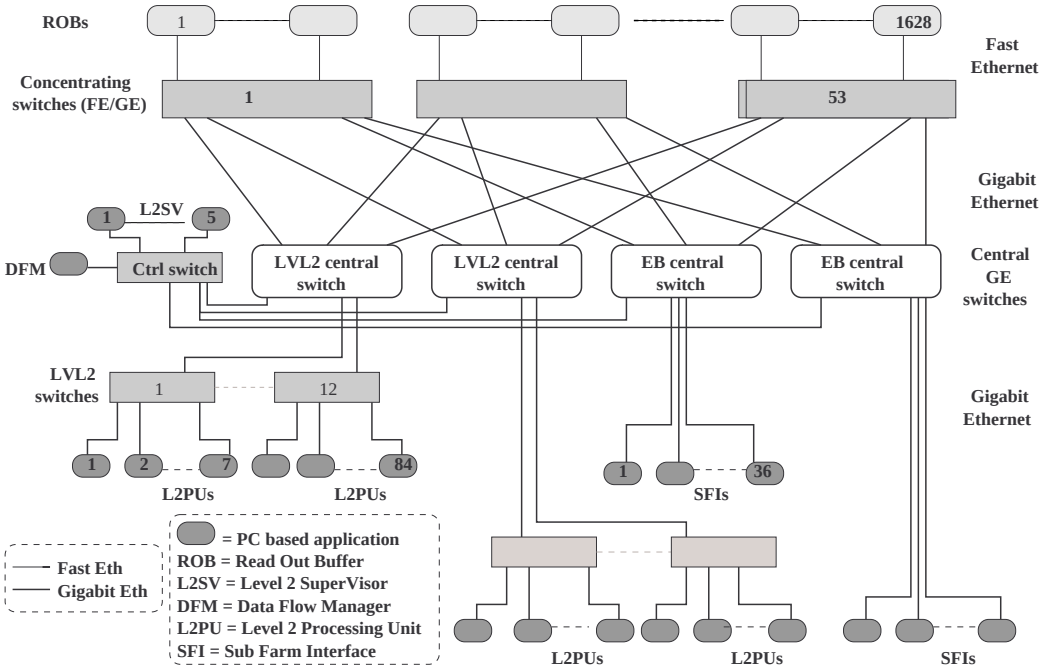


Figure 28 : L'architecture des flux de données du système TDAQ de ATLAS.

4.3.2 Comparaison

Les résultats des tests effectués avec ces systèmes ont été présentés dans [Lev-03]. Voici un graphique en guise d'illustration qui montre les résultats des mesures effectuées dans le cadre des tests TDAQ avec des émulateurs ROB/ROS. La figure 29 représente les résultats obtenus avec 8 cartes Alteon ou 124 FPGAs comme émulateurs et permet une comparaison entre ces deux solutions.

Les émulateurs répondent aux requêtes adressées à 1600 ROB pour des événements de 2,2 MB. L'expérience s'est déroulée en faisant varier le nombre de constructeurs d'événements (SFIs) de 1 à 8. La figure montre le gain en termes de nombre d'événements traités par seconde. Vu la taille d'un événement et la taille des fragments (1400 octets), il en résulte environ 1600 fragments par événement. Pour les émulateurs FPGA, ces requêtes adressées à 1600 ROB sont répandues entre 124 émulateurs (opérant à 100 Mbps chacun), mais seulement entre 8 émulateurs effectifs pour les cartes Alteon (qui opèrent par contre à 1 Gbps chacune). Ceci conduit à une diminution de la performance pour ces dernières à des taux de construction supérieurs à 120 Hz.

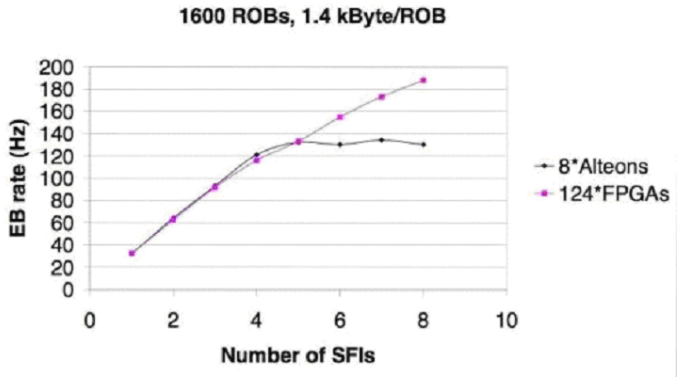


Figure 29 : Le taux de construction d'événements en fonction du nombre de SFIs.

4.4 Résultats expérimentaux

Nous avons effectué une série de tests avec les cartes Alteon en régime de test[1], ayant pour objectif de déterminer les caractéristiques de différenciation de service des éléments de base dans les réseaux informatiques – les commutateurs [Beu-04c]. Nous nous sommes concentrés sur les mécanismes fondamentaux que les commutateurs pourvoient : les algorithmes d'ordonnancement.

Les résultats obtenus dans la mesure de caractéristiques de différenciation de service pour les commutateurs sont influencés par plusieurs facteurs, parmi lesquels un des plus importants est l'architecture interne des commutateurs. Ses propriétés, comme le blocage par tête-de-file[2], interfèrent avec les mécanismes de gestion et d'ordonnancement et peuvent en empêcher leur bon fonctionnement.

Nous avons mené des tests sur six commutateurs Gigabit Ethernet (leur nom n'est pas dévoilé dans ce document pour des raison de confidentialité – on se contentera de leur affecter de numéros). La taille des trames a été en général de 1518 octets, à l'exception de certains tests où nous avons essayé de mettre en évidence les différences qui apparaissent pour les flux de paquets de taille variée.

---

[1] C'est-à-dire dans la configuration décrite dans la section 4.2.

[2] Le blocage par tête-de-file (« head-of-line blocking » en anglais) est le phénomène par lequel certains paquets en tête des files d'attente, dont le port destination est indisponible, empêchent les paquets qui les suivent d'être expédiés, même si leurs destinations sont disponibles. Il a été montré qu'avec une telle architecture, le débit ne peut pas dépasser environ 60% de la capacité de la matrice de commutation.

Pour la différenciation de service, les commutateurs utilisent des files d'attente différentes, le nombre total de files le plus usuel étant quatre ou huit. Les files d'attente sont notées Q_i, où i varie de 0 à 3 ou de 0 à 7. Si le mécanisme d'ordonnancement implique une importance différente des files, la file avec l'indice le plus grand a l'importance la plus haute (par exemple pour SP, Q_3 aura la plus haute prééminence pour un commutateur avec quatre files d'attente).

4.4.1 Conditions expérimentales

Les paquets sont marqués avec une priorité de 0 à 7 dans l'étiquette VLAN. L'association entre VLAN et files d'attente pour les commutateurs à huit files d'attente est faite de manière naturelle : la priorité VLAN i est associée à la file d'attente Q_i, $i = \overline{0,7}$.

Pour les commutateurs qui n'ont que quatre files d'attente, l'association est faite de la manière suivante :

Priorité VLAN	0 & 1	2 & 3	4 & 5	6 & 7
File d'attente	Q_0	Q_1	Q_2	Q_3

Tableau 7 : Association entre prorités VLAN et files d'attente.

Les principaux algorithmes d'ordonnancement étudiés ont été la priorité stricte (SP) et l'algorithme du tourniquet avec poids (WRR). Certains commutateurs permettent de choisir une variante de WRR qui utilise des quanta de 256 octets pendant l'ordonnancement au lieu de paquets, afin de s'approcher de WFQ comme performance pour le trafic avec des paquets de taille différente. Un seul commutateur fournit l'algorithme WFQ, avec poids prédéfinis mais non documentés.

Une autre variante d'ordonnancement présente sur quelques commutateurs est un hybride entre SP et WRR. On va désigner par Hybrid-1 la variante qui utilise SP pour la file de priorité plus haute, Q_3, et WRR pour les files Q_2 à Q_0. La notation Hybrid-2 sera utilisée pour la variante qui utilise SP pour les files de priorité haute, Q_3 et Q_2, et WRR pour les files Q_1 et Q_0.

Les huit sources envoient du trafic de priorité différente vers une destination commune. Le trafic est de type CBR (Constant Bit Rate), avec des taux de transmission de 1 à 60% de la capacité de la connexion (1 Gbps). La configuration de test est montrée sur la figure 30.

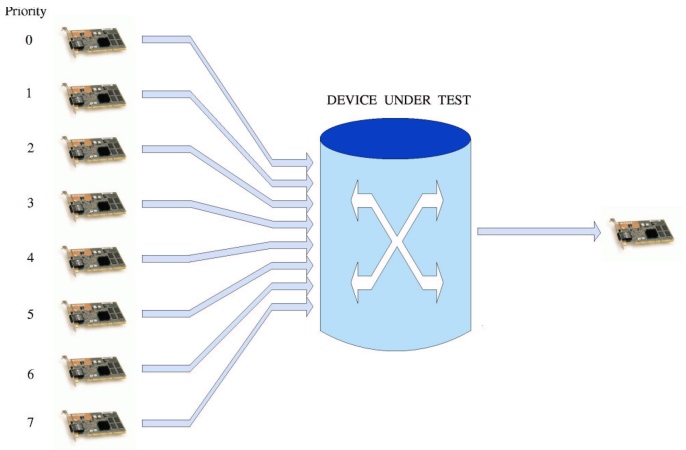


Figure 30 : Configuration de test.

4.4.2 Comportement attendu

La figure 31 montre le comportement attendu, idéal pour l'ordonnancement SP, dans laquelle une file n'est pas traitée tandis que des paquets se trouvent dans les files de priorité plus haute. Nous supposons le cas d'un commutateur Gigabit Ethernet avec quatre files d'attente, l'association entre priorités et files d'attente étant spécifiée dans le tableau 7. Nous précisons de nouveau que si $i > j$, alors Q_i a prééminence sur Q_j.

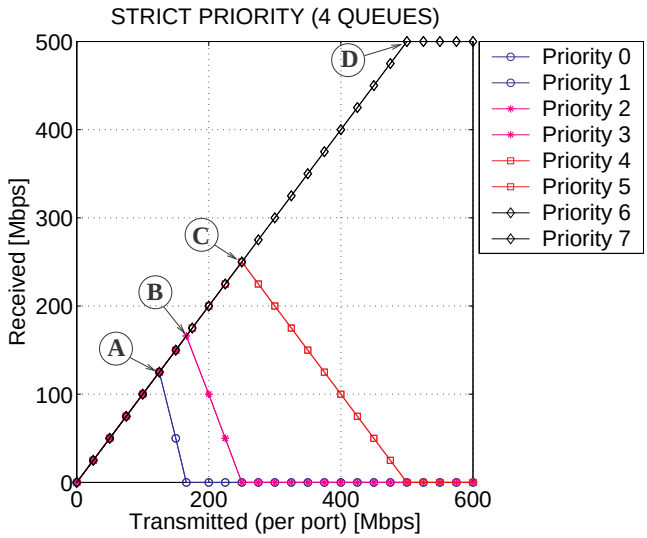


Figure 31 : Comportement idéal de la séléction SP.

Le point Ⓐ (à 125 Mbps de trafic transmis par priorité) représente le moment où le trafic offert pour toutes les files d'attente (de Q_0 à Q_3) dépasse 1 Gbps. Après Ⓐ, Q_0 commence à avoir des pertes. Le point Ⓑ (166 Mbps) indique que la charge totale pour les files d'attente de Q_1 à Q_3 arrive à 1 Gbps. Ensuite Q_1 perd du trafic aussi. Le

point Ⓒ (250 Mbps) montre l'instant où la quantité totale de trafic dans Q_2 et Q_3 est à 1 Gbps. A partir de ce point, Q_2 commence également à perdre des paquets. Dès le point Ⓓ (à 250 Mbps), la file Q_3 est saturée et le trafic est limité à 500 Mbps pour chacune des priorités 6 et 7, le reste étant rejeté.

La figure 32 montre le comportement idéal, attendu de l'ordonnancement WRR (dans l'hypothèse « work-conserving »[1]). Pour ce type d'ordonnancement, chaque file d'attente est traitée de façon cyclique et de manière proportionnelle avec les poids configurés. Ceci permet de donner des garanties sur la bande passante allouée à chaque file d'attente, mais il faut noter que ces garanties peuvent être dépassées, en fonction de l'occupation des autres files d'attente. Sur la figure 32 nous considérons de nouveau le cas d'un commutateur Gigabit Ethernet avec quatre files d'attente et les poids suivants pour WRR : 0,4 pour Q_3, 0,3 pour Q_2, 0,2 pour Q_1 et 0,1 pour Q_0. Avant le point Ⓔ (à 125 Mbps de trafic transmis par priorité) le trafic de toutes les sources arrive à la destination. La saturation est atteinte au point Ⓔ et ensuite la différenciation de service est mise en œuvre. Pour un court intervalle, avant que le trafic cumulé dans Q_3 n'atteigne 400 Mbps (moment indiqué par le point Ⓕ), les files de priorité Q_2 et Q_1 reçoivent plus de bande passante qu'il n'était garanti.

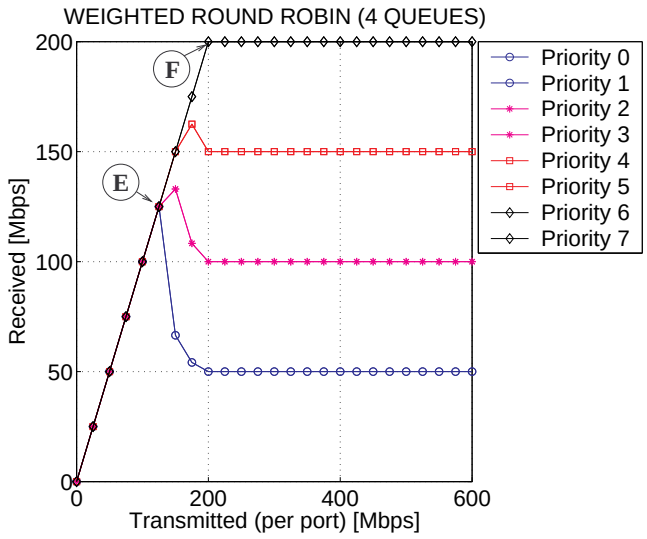


Figure 32 : Comportement idéal de la séléction WRR.

[1] On nomme un système réseau « work-conserving » s'il traite les paquets afin de maximiser le débit à la sortie. Dans le cas contraire le système s'appelle « non work-conserving ».

Le fait que les files de Q_0 à Q_2 envoient plus de trafic qu'en saturation entre Ⓔ et 200 Mbps résulte de ce que les commutateurs opèrent dans un paradigme d'utilisation permanente de la capacité maximale de la liaison (« work conserving » en anglais).

Pour WRR les poids indiquent le rapport entre le débit d'une file d'attente et le débit total du commutateur. On va appeler cette mesure *rapport de débit*.

Pour l'ordonnancement WRR on peut aussi déterminer les rapports qui existent entre les délais moyens pour chaque file d'attente. Supposons qu'on attribue les poids *a*, *b*, *c* et *d* aux files d'attente Q_0, Q_1, Q_2 et Q_3 respectivement (avec la condition usuelle $a+b+c+d=1$). Les délais correspondants (dans un modèle qui présume la prééminence de Q_3) sont donnés par les formules suivantes :

$$\begin{aligned} D_0 &= \frac{N}{aV}+\frac{N}{bV}+\frac{N}{cV}+\frac{N}{dV}, \\ D_1 &= \frac{N}{bV}+\frac{N}{cV}+\frac{N}{dV}, \\ D_2 &= \frac{N}{cV}+\frac{N}{dV}, \\ D_3 &= \frac{N}{dV}, \end{aligned} \qquad (4.1)$$

où *N* est la taille des files (pour simplicité on suppose qu'elles ont toutes la même taille) et *V* est le taux de service.

Si on calcule le rapport entre ces délais et *D*, leur somme, on obtient les rapports de délais qui donnent une indication sur les délais moyens que vont subir les paquets dans chaque file d'attente.

D'autres suppositions également valables peuvent être faites, qui conduisent à des résultats différents pour le délai, tout en respectant les rapports de débit. Ceci constitue une sorte d'imprédictibilité des implémentations de WRR, dans le sens que on ne peut pas savoir *a priori* quels seront ces rapports pour une implémentation quelconque.

4.4.3 Commutateur #1

Le commutateur #1 a quatre files d'attente et permet d'utiliser l'un des algorithmes d'ordonnancement SP, WRR (par paquet ou quanta de 256 octets), Hybrid-1 et Hybrid-2.

4.4.3.1 SP

Sur les figures 33 et 34 sont représentés les résultats de l'ordonnancement SP. La file Q_2 n'est pas complètement bloquée, même lorsque la charge sur Q_3 attend 1 Gbps, un écart par rapport au comportement attendu.

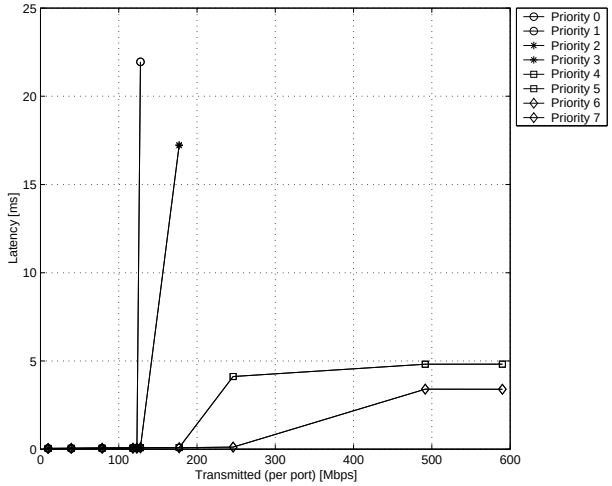


Figure 33 : Délai par priorité en fonction du trafic transmis (SP).

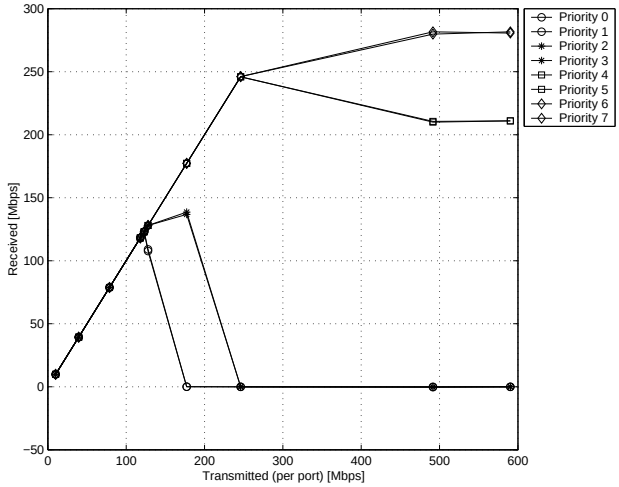


Figure 34 : Le trafic reçu par priorité en fonction du trafic transmis (SP).

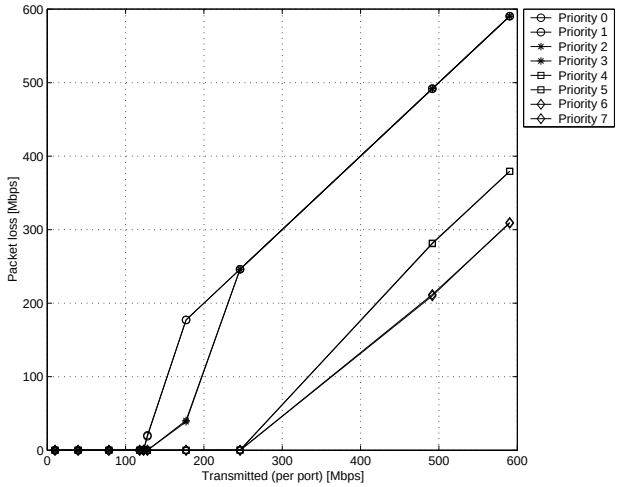


Figure 35 : Le trafic perdu par priorité en fonction du trafic transmis (SP).

En dépit du traitement préférentiel de la file Q_3 (avec le plus haut débit et un délai limité à 3 ms), cette file ne reçoit pas le niveau de service qu'elle aurait dû avoir en conformité avec l'ordonnancement SP.

Pour les autres cas nous ne donnerons pas les graphiques représentant le trafic perdu car ils ne contiennent pas plus d'information que dans le graphique du trafic reçu, puisque les pertes sont calculées comme la différence entre le trafic transmis et celui qui est réceptionné.

4.4.3.2 WRR

Les poids pour WRR sont spécifiés comme pourcentages relatifs, soit en termes de nombre de paquets, soit en quanta de 256 octets. Les poids assignés dans nos expériences ont été de 0,1 pour Q_0, 0,2 pour Q_1, 0, 3 pour Q_2 et 0,4 pour Q_3 dans les deux cas.

4.4.3.2.1 WRR par paquets, même taille de paquets

Quand on utilise WRR au niveau des paquets, les propriétés sont bonnes tant que les paquets dans toutes les files d'attente ont la même taille. Les poids observés pour le débit sont en bon accord avec les prévisions (voir la figure 37). La relation entre les poids et le délai est aussi assez bien respectée (voir le tableau 8).

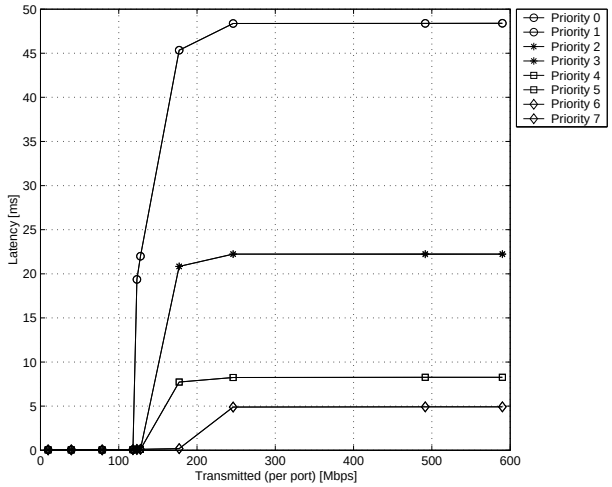


Figure 36 : Délai par priorité en fonction du trafic transmis (WRR par paquets, même taille).

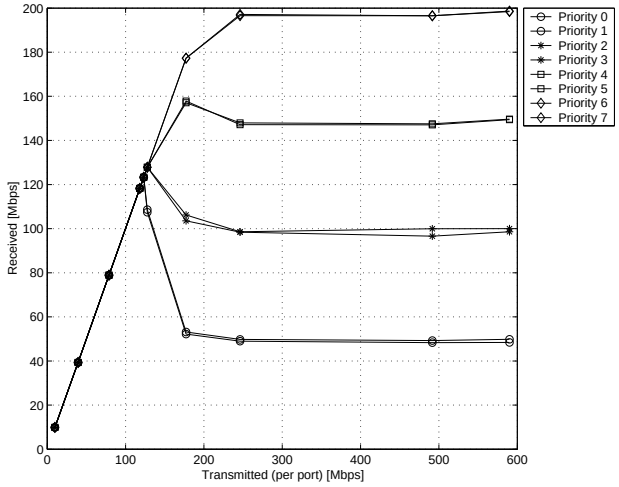


Figure 37 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par paquets, même taille).

File d'attente	***Rapport de délai attendu***	***Rapport de délai mesuré***	***Rapport de débit attendu***	***Rapport de débit mesuré***
Q_0	0,520	0,578	0,1	0,097
Q_1	0,270	0,265	0,2	0,202
Q_2	0,145	0,096	0,3	0,299
Q_3	0,062	0,060	0,4	0,401

Tableau 8 : Les rapports de délai et débit, attendu et mesuré (WRR par paquets, même taille).

4.4.3.2.2 WRR par paquets, taille différente de paquets

Par contre, quand les paquets ont des tailles différentes, le WRR à base de paquets n'est plus équitable du point de vue de débit (voir la figure 39). Les délais changent aussi. Les tailles de paquets utilisées sont précisées dans le tableau suivant.

Priorité VLAN	0 & 1	2 & 3	4 & 5	6 & 7
Taille des paquets [octets]	1518	1024	512	256

Tableau 9 : Les tailles de paquets pour chaque prorité VLAN.

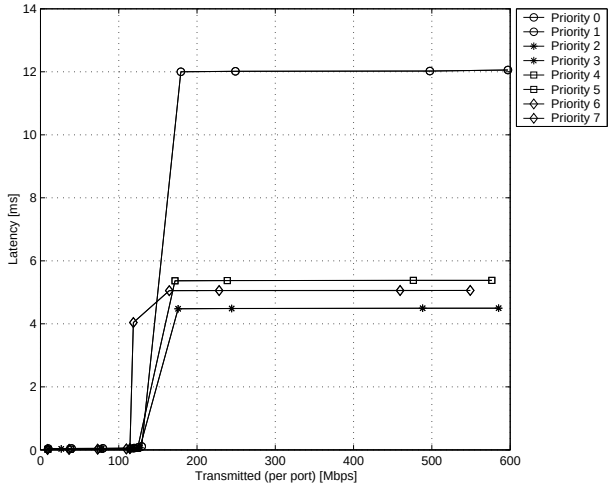


Figure 38 : Délai par priorité en fonction du trafic transmis (WRR par paquets, taille différente).

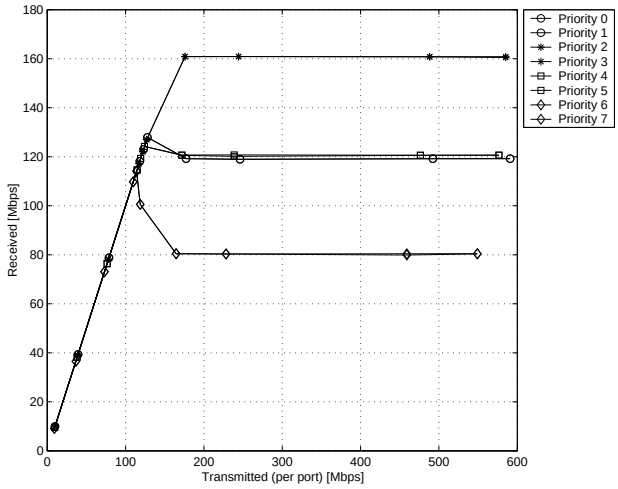


Figure 39 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par paquets, taille différente).

Dans le tableau 10 nous montrons d'abord les rapports de débit attendu et mesuré, qui sont en total désaccord. Etant donné que l'ordonnancement se fait par paquets, et que

les paquets ont des tailles différentes, nous avons calculé le rapport de débit qui correspond aux files d'attente si, en plus du poids, on prend aussi en considération la taille des paquets dans la file correspondante et donc la quantité de données traitée dans chaque reprise d'ordonnancement.

File d'attente	*Taux de débit attendu*	*Taux de débit mesuré*	*Quantité de données [octets]*	*Taux de débit attendu corrigé*
Q_0	0,1	0,245	1518	0,247
Q_1	0,2	0,334	2048	0,334
Q_2	0,3	0,251	1536	0,250
Q_3	0,4	0,168	1024	0,167

Tableau 10 : Les taux de débit attendu et mesuré et le taux de débit attendu corrigé (WRR par paquets, taille différente).

On voit que il y a un bon accord avec le rapport corrigé, mais les résultats de l'ordonnancement WRR par paquets ne sont pas intuitifs dans ce cas (et ils sont imprévisibles si les paquets d'une même file n'ont pas tous la même taille).

4.4.3.2.3 WRR par quanta, à même taille de paquets

Pour résoudre ce problème on peut utiliser une version de WRR qui emploie des quanta de 256 octets pour l'ordonnancement. Dans la figure 41 ce mécanisme est illustré pour des trames de même taille. Les résultats sont similaires à ceux obtenu pour le WRR à base de paquets.

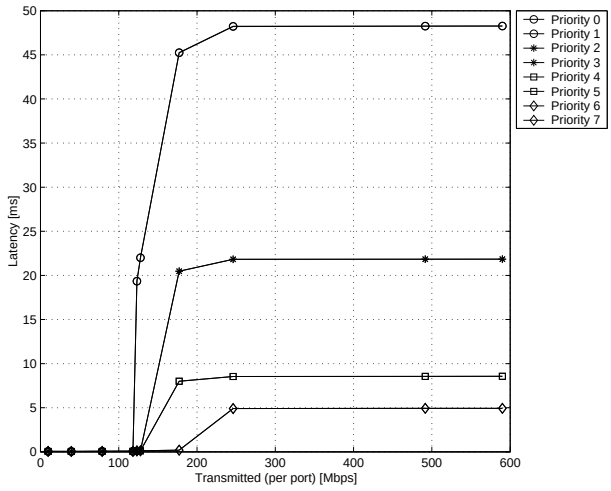


Figure 40 : Délai par priorité en fonction du trafic transmis (WRR par quanta, même taille de paquets).

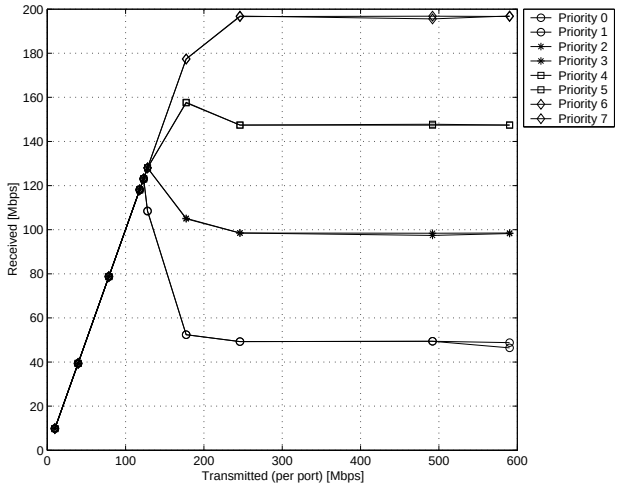


Figure 41 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par quanta, même taille de paquets).

4.4.3.2.4 WRR par quanta, à taille différente de paquets

Les figures 42 et 43 montrent les résultats obtenus quand on envoie des paquets de taille différente (en accord avec le tableau 9) et que l'on utilise le WRR à base de quanta de 256 octets. Les rapports de débit ne sont pas en très bon accord avec ceux attendus (voir le tableau 11). Nous pensons que c'est dû à la surcharge de traitement de paquets : pour atteindre le même débit, environ 6 fois plus de paquets doivent être traités s'ils ont 256 octets au lieu d'avoir 1518 octets.

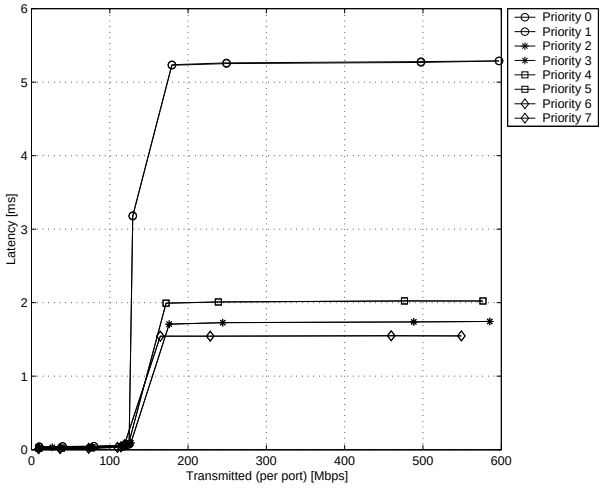


Figure 42 : Délai par priorité en fonction du trafic transmis (WRR par quanta, taille différente de paquets).

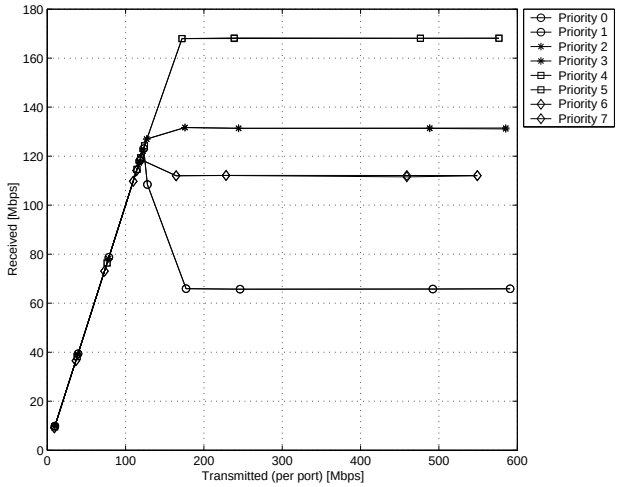


Figure 43 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par quanta, taille différente de paquets).

File d'attente	***Rapport de débit attendu***	***Rapport de débit mesuré***
Q_0	0,1	0,137
Q_1	0,2	0,274
Q_2	0,3	0,351
Q_3	0,4	0,237

Tableau 11 : Les taux de délais attendu et mesuré (WRR par quanta, taille différente de paquets).

4.4.3.3 Ordonnancement hybride

Dans l'ordonnancement hybride Hybrid-1, Q_3 est traité de manière SP et les trois autres files d'attente de manière WRR. Les poids sont des pourcentages relatifs d'utilisation de la bande passante, soit en termes de paquets, soit en termes de quanta de 256 octets. Nous avons utilisé les poids en termes de paquets comme suit : 0,1 pour Q_0, 0,3 pour Q_1 et 0,6 pour Q_2.

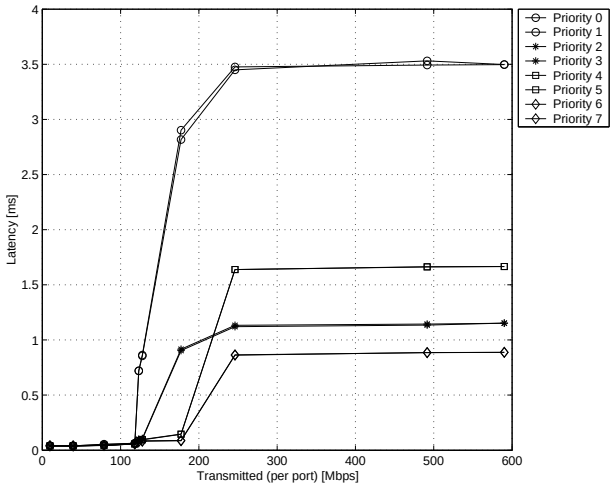


Figure 44 : Délai par priorité en fonction du trafic transmis (Hybrid-1).

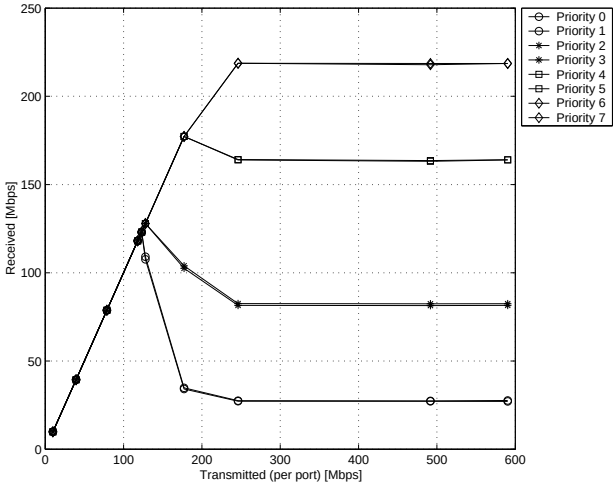


Figure 45 : Le trafic reçu par priorité en fonction du trafic transmis (Hybrid-1).

Dans l'ordonnancement hybride Hybrid-2, Q_3 et Q_2 sont traités de manière SP et les deux autres files d'attente de manière WRR, de façon similaire à Hybrid-1. Nous avons utilisé les poids en termes de paquets comme suit : 0,4 pour Q_0 et 0,6 pour Q_1.

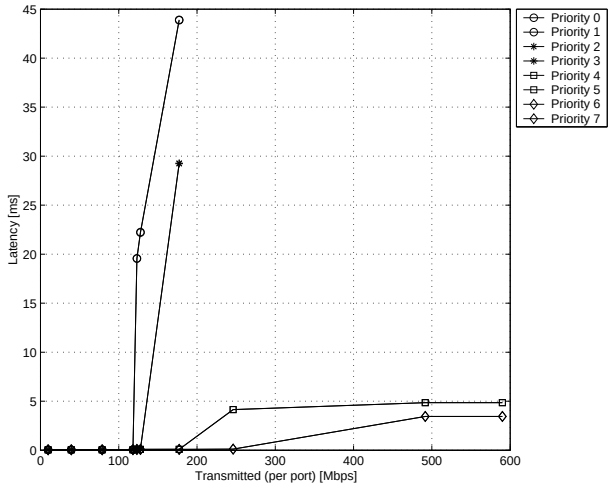


Figure 46 : Délai par priorité en fonction du trafic transmis (Hybrid-2).

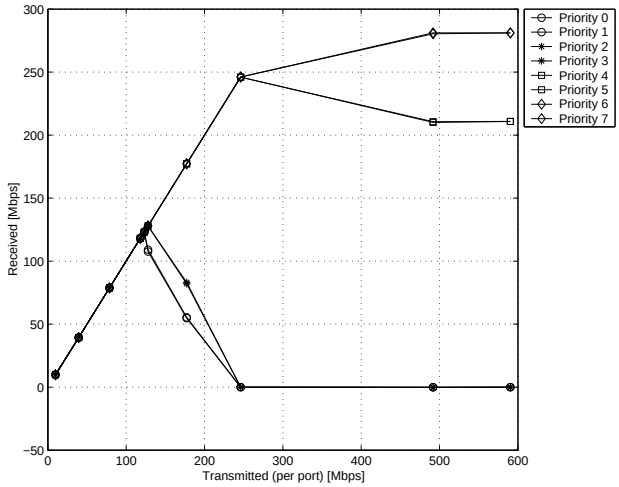


Figure 47 : Le trafic reçu par priorité en fonction du trafic transmis (Hybrid-2).

Les mêmes problèmes que pour SP sont observés : dans les deux cas Q_3 ne reçoit pas un service exclusif lorsque le débit atteint 1 Gbps.

4.4.3.4 Analyse

Pour ce commutateur il y a une série de problèmes concernant la fidélité du comportement attendu par rapport à celui qui a été observé. L'ordonnancement SP ne se comporte pas bien pour des taux de transmission supérieurs à 500 Mbps. L'ordonnancement WRR par paquets est satisfaisant pour les paquets de même taille, mais défaillante pour les paquets de taille différente. WRR à base de quanta ne produit pas finalement des résultats satisfaisants non plus, quand les tailles des

paquets sont différents. Une explication possible est la surcharge donnée par le nombre plus grand de paquets dans Q_3 par rapport aux files Q_2 à Q_0.

4.4.4 Commutateur #2

Le commutateur #2 a quatre files d'attente et permet d'utiliser un des algorithmes d'ordonnancement SP, WRR (par paquet ou quanta de 256 octets), Hybrid-1 et Hybrid-2.

4.4.4.1 SP

L'ordonnancement par priorité stricte, s'il n'est pas combiné avec des mécanismes de conditionnement et régulation de trafic, peut conduire au blocage des flux de priorité basse, comme on peut le voir sur les figures 48 et 49. Mais le comportement observé est différent de celui attendu, car la file Q_2 n'est pas complètement bloquée même quand la charge sur Q_3 attend 1 Gbps.

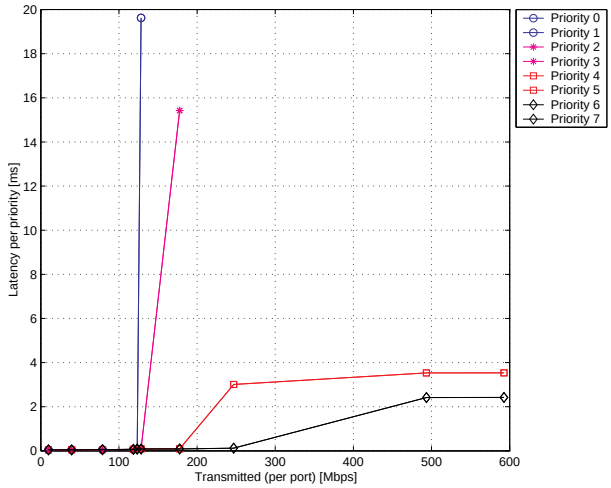


Figure 48 : Délai par priorité en fonction du trafic transmis (SP).

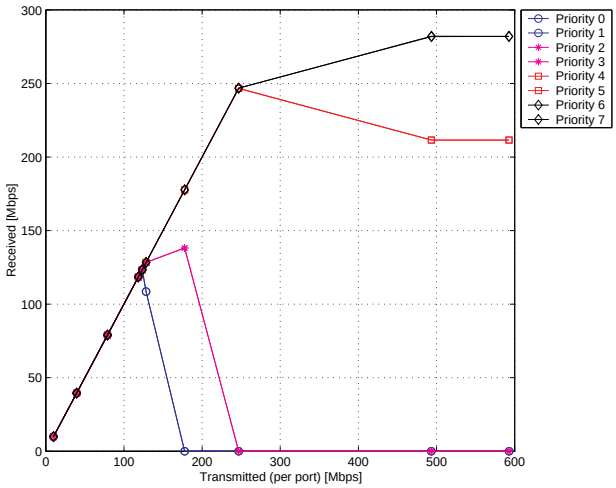


Figure 49 : Le trafic reçu par priorité en fonction du trafic transmis (SP).

4.4.4.2 WRR

Les poids pour WRR sont spécifiés comme pourcentages relatives, soit en termes de nombre de paquets, soit en quanta de 256 octets. Les poids assignés dans nos expériences ont été de 0,1 pour Q_0, 0,2 pour Q_1, 0,3 pour Q_2 et 0,4 pour Q_3 dans les deux cas.

4.4.4.2.1 WRR par paquets, à même taille de paquets

Pour les premiers tests nous avons utilisé la même taille de paquets (1518 octets) pour tous les transmetteurs. Voici les résultats (les figures 50 et 51). Les taux observé pour le débit et le délai sont en bon accord avec les attentes.

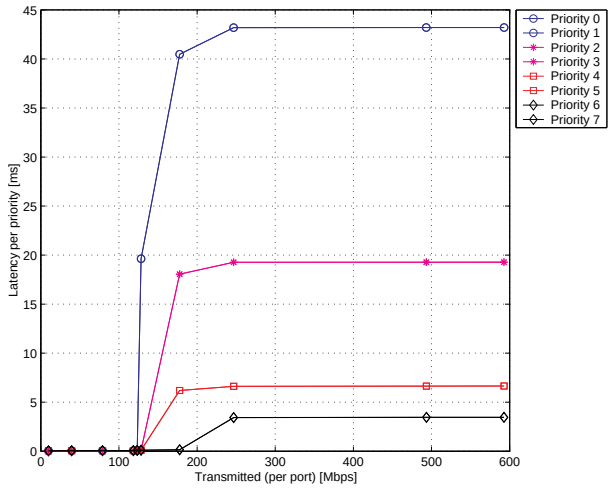


Figure 50 : Délai par priorité en fonction du trafic transmis (WRR par paquets, même taille).

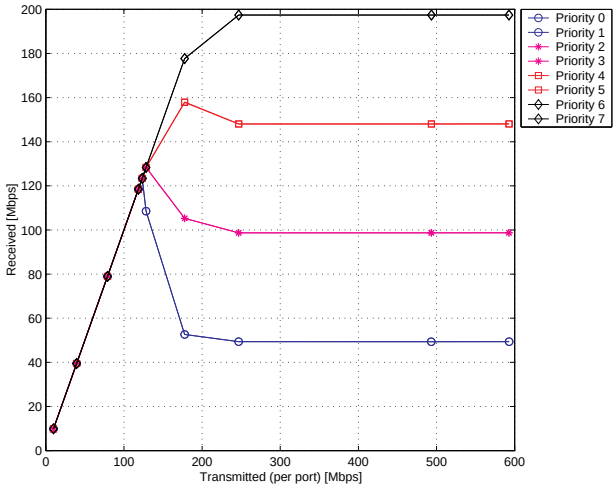


Figure 51 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par paquets, même taille).

4.4.4.2.2 WRR par paquets, à taille différente de paquets

Les tailles de paquets utilisées sont précisées dans le tableau suivant.

Priorité VLAN	0 & 1	2 & 3	4 & 5	6 & 7
Taille des paquets [octets]	1518	1024	512	256

Tableau 12 : Les tailles de paquets pour chaque prorité VLAN.

On voit que, par contre, quand les paquets ont des tailles différentes, le WRR à base de paquets n'est plus équitable de point de vue de débit (voir la figure 53). Les délais changent aussi, comme on peut voir sur la figure 52.

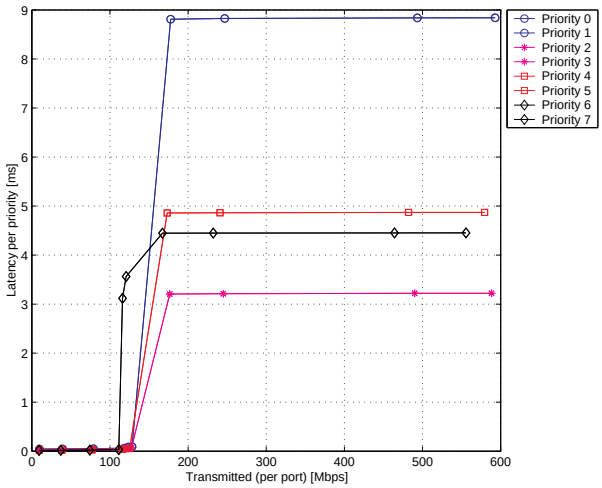


Figure 52 : Délai par priorité en fonction du trafic transmis (WRR par paquets, taille différente).

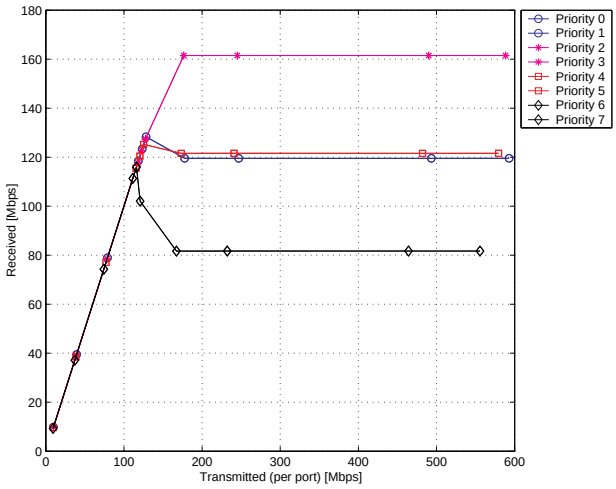


Figure 53 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par paquets, taille différente).

4.4.4.2.3 WRR par quanta, à même taille de paquets

Pour résoudre ce problème on peut utiliser une version de WRR qui utilise des quanta de 256 octets pour l'ordonnancement. Sur la figure 55 on peut voir les effets de l'utilisation de ce mécanisme pour des trames de même taille. Les résultats sont inattendus, car les poids ne sont pas respectés. Pour ce test nous avons utilisé des paquets de 1518 octets pour toutes les priorités.

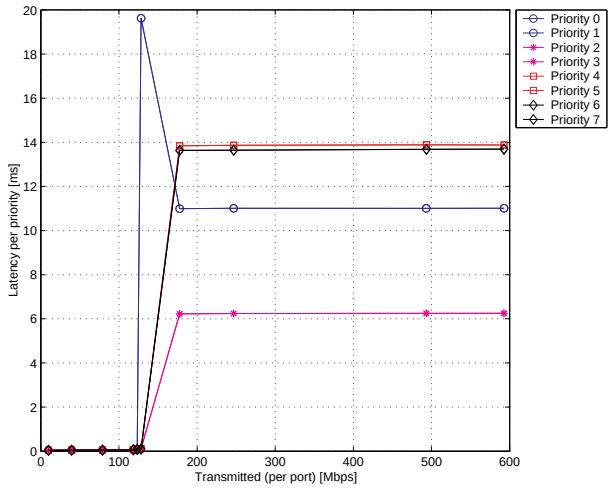


Figure 54 : Délai par priorité en fonction du trafic transmis (WRR par quanta, même taille de paquets).

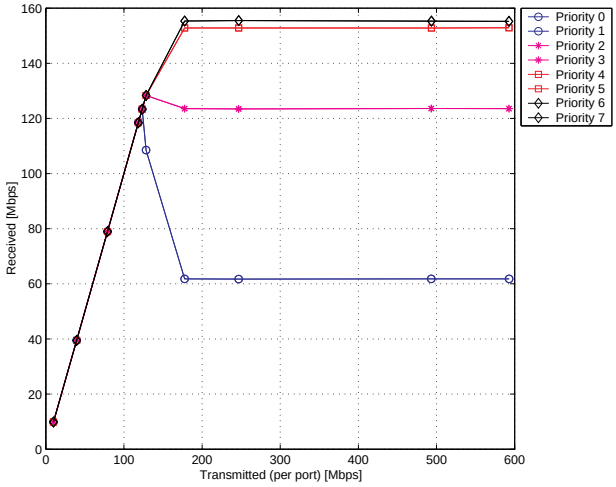


Figure 55 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par quanta, même taille de paquets).

4.4.4.2.4 WRR par quanta, à taille différente de paquets

La figure 56 montre les résultats obtenus quand on envoie des paquets de taille différente et que l'on utilise le WRR à base de quanta de 256 octets. Les rapports de débit ne sont pas en très bon accord avec les attentes (voir la figure 57). La surcharge de traitement de petits paquets peut en être toujours la cause. Il s'agit ici de la même correspondance entre priorité et taille des paquets transmis que dans 4.4.4.2.2.

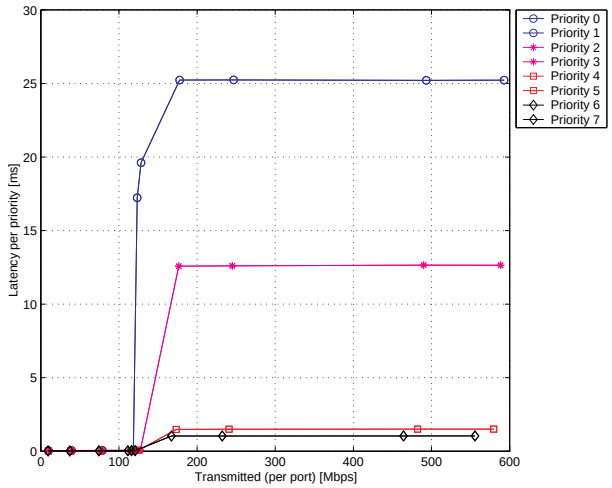


Figure 56 : Délai par priorité en fonction du trafic transmis(WRR par quanta, taille différente de paquets).

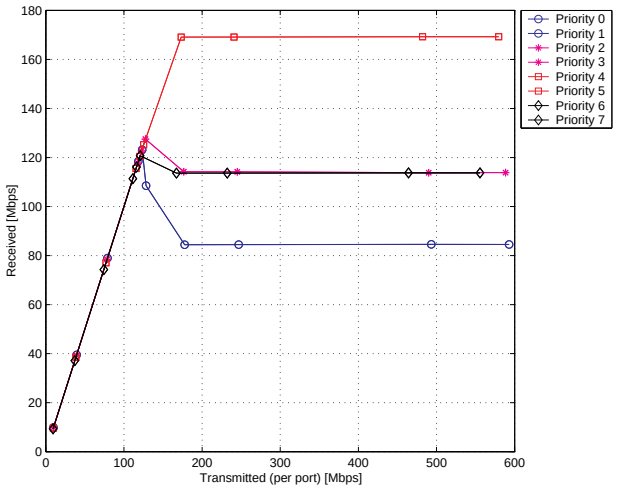


Figure 57 : Le trafic reçu par priorité en fonction du trafic transmis (WRR par quanta, taille différente de paquets).

4.4.4.3 Ordonnancement hybride

Dans l'ordonnancement hybride Hybrid-1, Q_3 est traité de manière SP et les autres 3 files d'attente de manière WRR. Les poids sont spécifiés comme des pourcentages relatifs d'utilisation de la bande passante, soit en termes de paquets, soit en termes de quanta de 256 octets. Nous avons utilisé les poids en terme de paquets comme suit : 0,1 pour Q_0, 0,3 pour Q_1 et 0,6 pour Q_2.

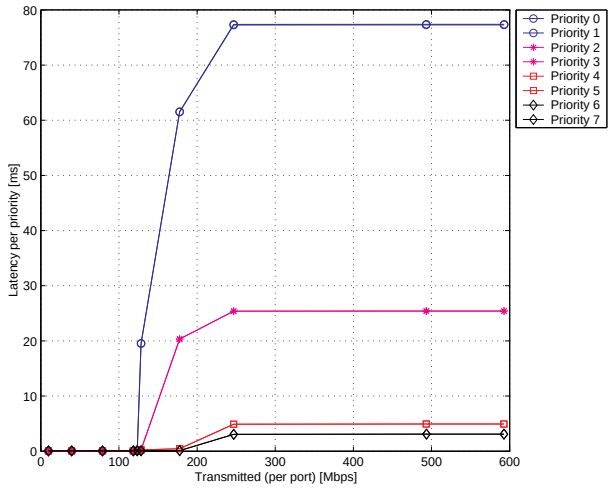


Figure 58 : Délai par priorité en fonction du trafic transmis (Hybrid-1).

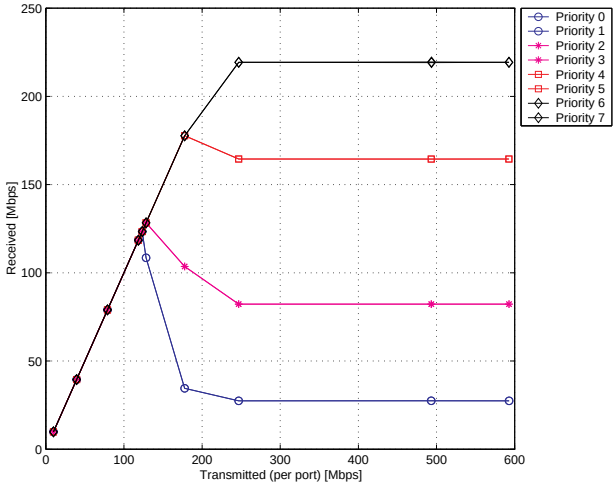


Figure 59 : Le trafic reçu par priorité en fonction du trafic transmis (Hybrid-1).

Dans l'ordonnancement hybride Hybrid-2, Q_3 et Q_2 sont traités de manière SP et les autres 2 files d'attente de manière WRR, de façon similaire à Hybrid-1. Nous avons utilisé poids en terme de paquets comme suit : 0,4 pour Q_0 et 0,6 pour Q_1.

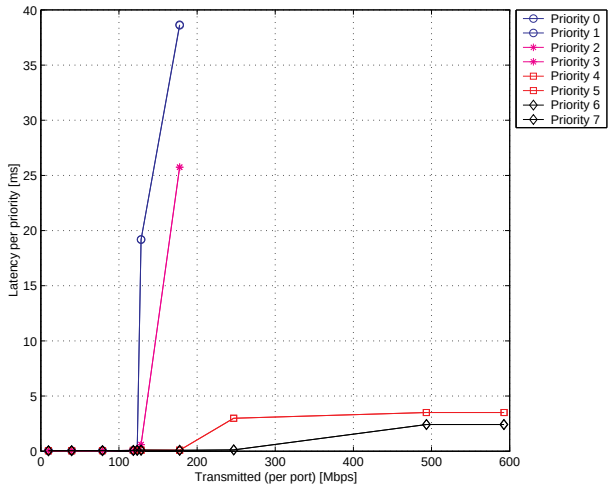


Figure 60 : Délai par priorité en fonction du trafic transmis (Hybrid-2).

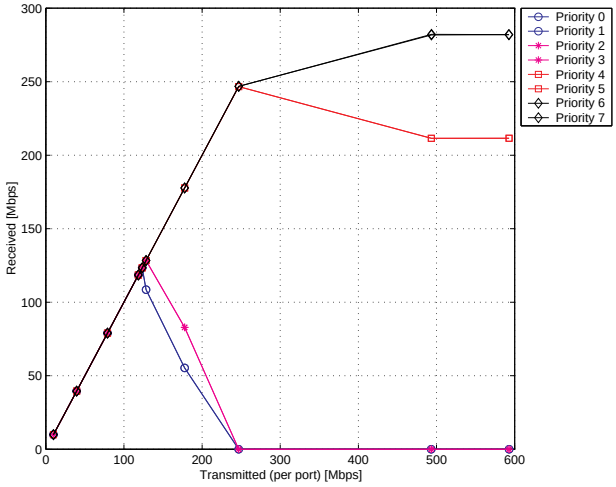


Figure 61 : Le trafic reçu par priorité en fonction du trafic transmis (Hybrid-2).

4.4.4.4 Analyse

Les résultats pour ce commutateur sont assez similaires à ceux obtenus pour le commutateur #1, chose normale puisqu'ils ont le même fabricant. La différence principale tient à ce que l'ordonnancement WRR par quanta est moins efficace.

4.4.5 Commutateur #3

Le commutateur #3 a quatre files d'attente et permet d'utiliser l'un des algorithmes d'ordonnancement SP ou WRR. On peut aussi introduire des limitations de débit.

4.4.5.1 SP

Pour la SP ce commutateur alloue 96% de la capacité à la priorité la plus haute (Q_3) et 4% à la priorité haute (Q_2). Les résultats obtenus sont présentés sur les figures 62 et 63.

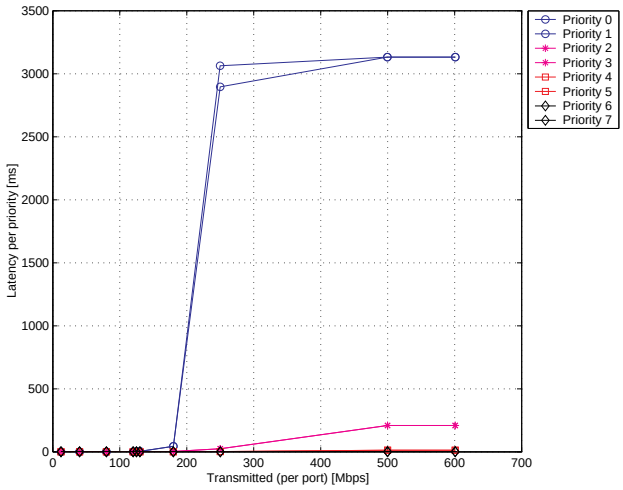


Figure 62 : Délai par priorité en fonction du trafic transmis (SP).

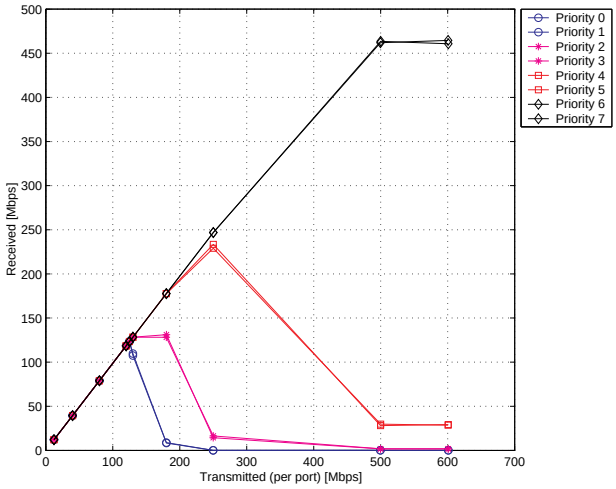


Figure 63 : Le trafic reçu par priorité en fonction du trafic transmis (SP).

4.4.5.2 WRR

Pour WRR le commutateur a été configuré avec les poids 0,1, 0,1, 0,3 et 0,5 pour les files d'attente Q_0 à Q_3, respectivement. Apparemment, seuls certains poids sont

permis, car les poids calculés par le commutateur ont été 0,08, 0,08, 0,34 et 0,5 respectivement.

4.4.5.2.1 WRR, même taille de paquets

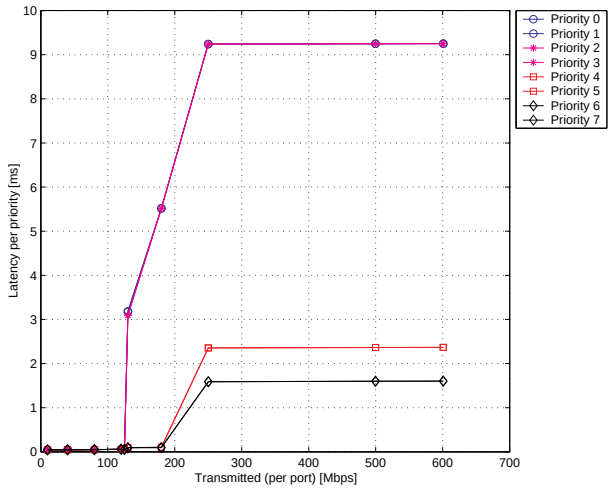


Figure 64 : Délai par priorité en fonction du trafic transmis (WRR, même taille de paquets).

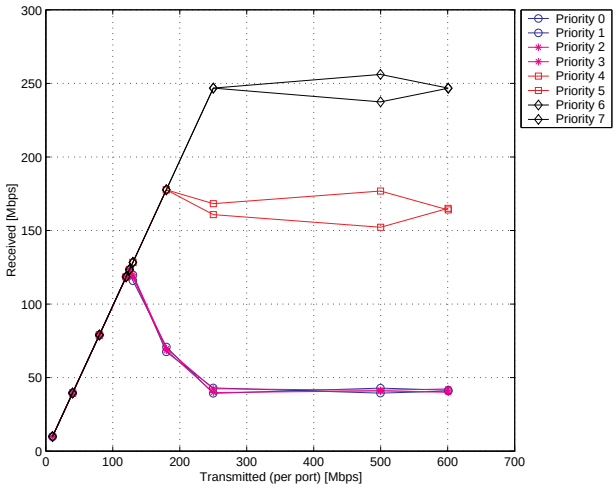


Figure 65 : Le trafic reçu par priorité en fonction du trafic transmis (WRR, même taille de paquets).

4.4.5.2.2 WRR, taille différente de paquets

Les tailles de paquets utilisées sont précisées dans le tableau 13.

Priorité VLAN	0 & 1	2 & 3	4 & 5	6 & 7
Taille des paquets [octets]	1518	1024	512	256

Tableau 13 : Les tailles de paquets pour chaque prorité VLAN.

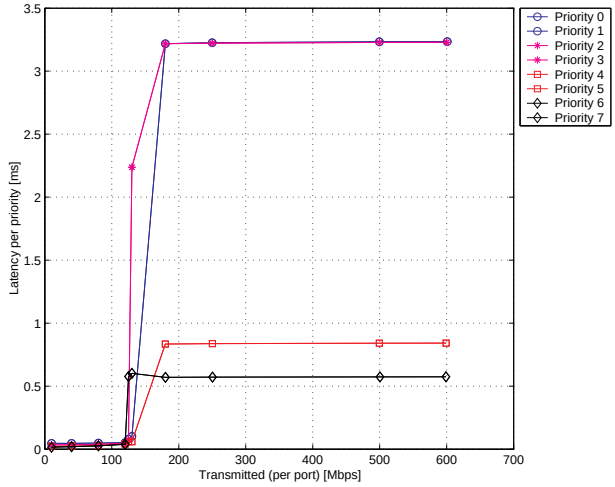


Figure 66 : Délai par priorité en fonction du trafic transmis (WRR, taille différente de paquets).

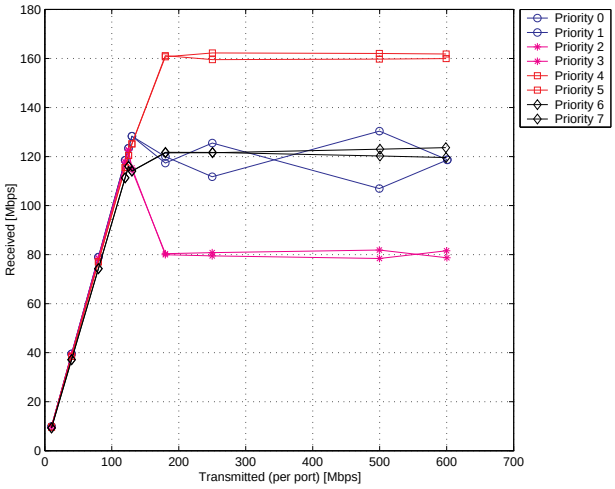


Figure 67 : Le trafic reçu par priorité en fonction du trafic transmis (WRR, taille différente de paquets).

4.4.5.3 Limitation de débit

La limitation de débit est configurée au niveau des ports. Les ports correspondants aux priorités VLAN 4 et 5 ont été limités à 100 Mbps chacun, et les ports correspondant aux priorités VLAN 6 et 7 à 200 Mbps. L'algorithme d'ordonnancement pour ces tests a été SP.

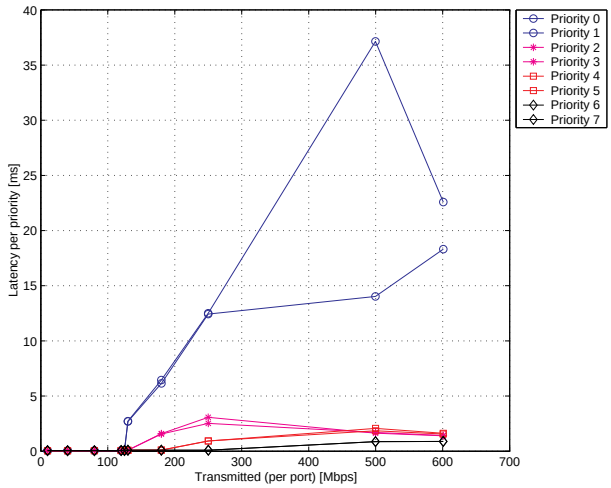


Figure 68 : Délai par priorité en fonction du trafic transmis (limitation de débit).

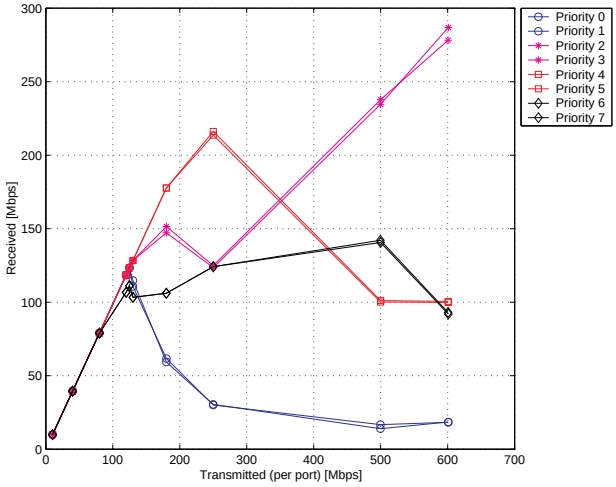


Figure 69 : Le trafic reçu par priorité en fonction du trafic transmis (limitation de débit).

4.4.5.4 Analyse

Le commutateur a un bon comportement, à la fois pour SP et WRR.

Pour SP, les priorités basses ne sont pas complètement bloquées (pour les priorités 0 et 1, environ 9 paquets par seconde sont reçus pour 48000 transmis), ce qui conduit à de très grandes valeurs du délai.

Le comportement de WRR est bon pour les paquets de même taille, le taux de débit étant en accord avec les poids configurés. Il y a par contre un problème d'équité entre les flux qui partagent la même file d'attente, visible surtout pour les priorités hautes.

Un problème plus important peut être observé pour le mécanisme de limitation du débit. Les limites configurées ne sont pas respectées de manière adéquate. La limite de 100 Mbps est dépassée même avec 100%, alors que celle de 200 Mbps n'est jamais atteinte, et il s'en fait de jusqu'à 50%.

4.4.6 Commutateur #4

Le commutateur #4 a quatre files d'attente et permet d'utiliser un des algorithmes d'ordonnancement SP ou WRR. On peut aussi introduire des limitations de débit au niveau des ports.

4.4.6.1 SP

Pour la SP ce commutateur alloue 96% de la capacité à la priorité la plus haute (Q_3) et 4% à la priorité haute (Q_2). Les résultats obtenus sont présentés sur les figures 70 et 71.

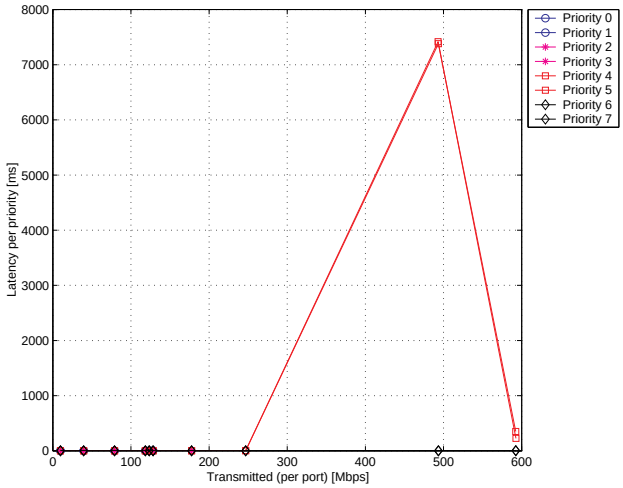


Figure 70 : Délai par priorité en fonction du trafic transmis (SP).

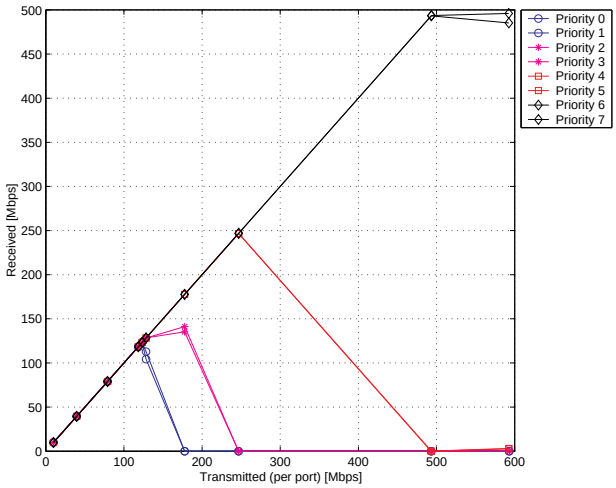


Figure 71 : Le trafic reçu par priorité en fonction du trafic transmis (SP).

4.4.6.2 WRR

Pour WRR le commutateur a été configuré avec les poids 0,1 pour Q_0, 0,1 pour Q_1, 0,3 pour Q_0 et 0,5 pour Q_3. Les poids calculés par le commutateur ont été les mêmes que ceux configurés.

4.4.6.2.1 WRR, même taille de paquets

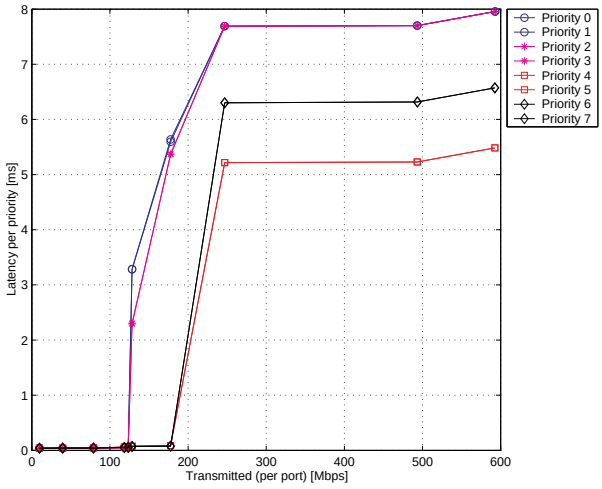


Figure 72 : Délai par priorité en fonction du trafic transmis (WRR, même taille de paquets).

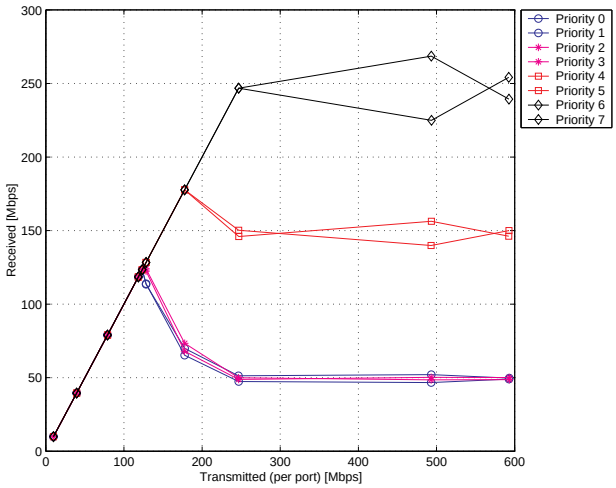


Figure 73 : Le trafic reçu par priorité en fonction du trafic transmis (WRR, même taille de paquets).

4.4.6.2.2 WRR avec taille différente des paquets

Les taille de paquets utilisées sont précisées dans :

Priorité VLAN	0 & 1	2 & 3	4 & 5	6 & 7
Taille des paquets [octets]	1518	1024	512	256

Tableau 14 : Les taille de paquets pour chaque priorité VLAN.

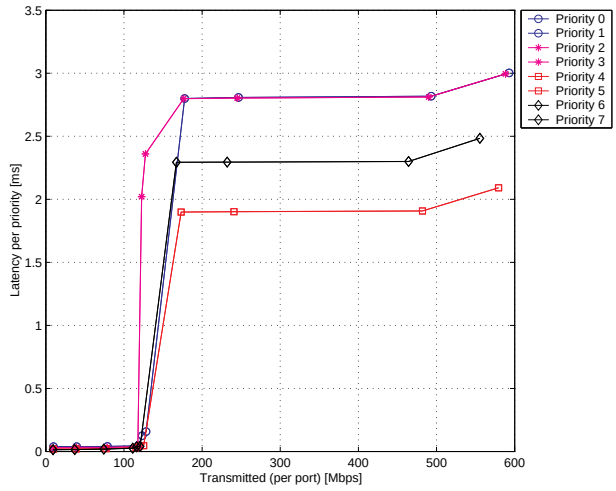


Figure 74 : Délai par priorité en fonction du trafic transmis (WRR, taille différente de paquets).

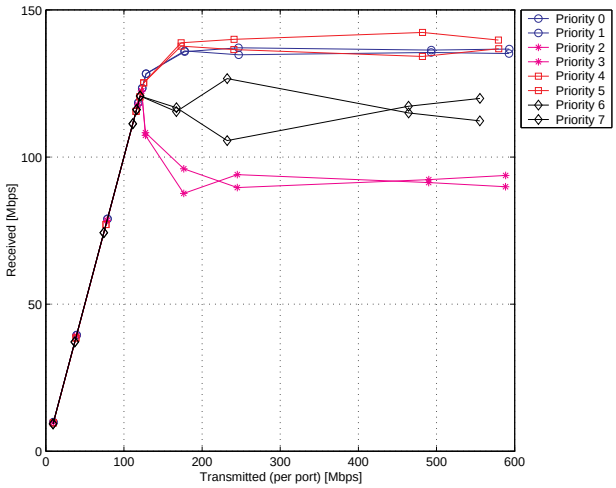


Figure 75 : Le trafic reçu par priorité en fonction du trafic transmis (WRR, taille différente de paquets).

4.4.6.3 Limitation de débit

Les limitations de débit au niveau des ports ont été configurées comme suit : 100 Mbps pour les ports des priorités VLAN 4 et 5, et 200 Mbps pour les ports des priorités VLAN 6 et 7. L'algorithme d'ordonnancement pour ces tests a été SP.

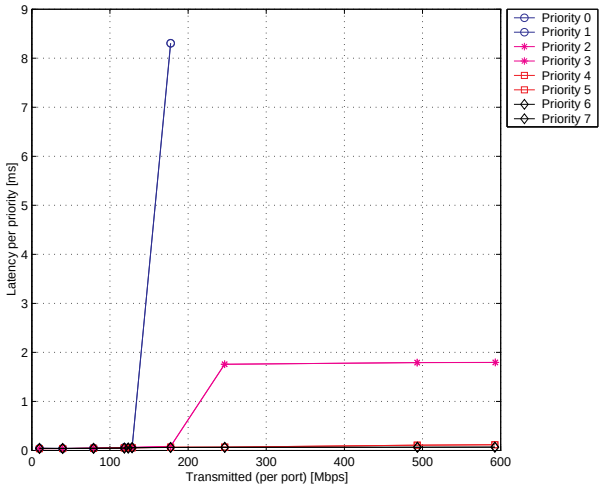


Figure 76 : Délai par priorité en fonction du trafic transmis (limitation de débit).

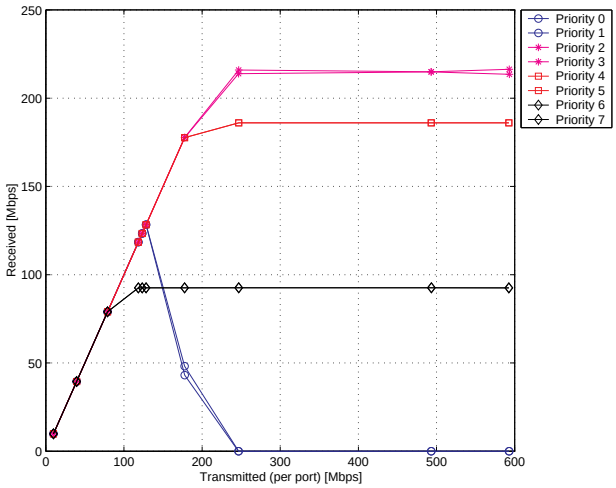


Figure 77 : Le trafic reçu par priorité en fonction du trafic transmis (limitation de débit).

4.4.6.4 Analyse

Le commutateur a un bon comportement global. Il y a tout de même des petites différences par rapport au comportement attendu, comme nous allons le détailler.

Pour l'ordonnancement SP, en dépit du fait que le poids calculé par le commutateur pour les priorités 4 et 5 (Q_2) est de 6%, ces priorités ont été bloquées quant le trafic dans Q_3 est arrivé à 1 Gbps. Mais du point de vue du comportement SP idéal, il y a conformité aux attentes.

Pour WRR nous observons que le délai pour Q_3 est supérieur à celui pour Q_2 (voir la figure 72), contrairement à nos attentes. Il y a un bon accord entre les pourcentages de bande passante alloués et les seuils configurés si on utilise la même taille de paquets. Pour des tailles différentes, les désavantages du WRR sont mis en évidence.

Le mécanisme de limitation de débit marche assez bien. La bande passante allouée est légèrement inférieure à celle configurée (90 Mbps au lieu de 100 Mbps et 180 Mbps au lieu de 200 Mbps). Comme attendu, les flux dans Q_1 utilisent la bande passante qui reste et ceux dans Q_0 sont bloqués (le mécanisme d'ordonnancement a été SP).

4.4.7 Commutateur #5

Le commutateur #5 n'a qu'un algorithme d'ordonnancement de type WFQ, fait déduit par l'intermédiaire des résultats des tests, avec des poids fixes et non documentés. Le fabricant ne nous a pas fourni les quanta non plus, on peut supposer que

l'ordonnancement se fait au niveau des octets. Par contre, à la différence des commutateurs précédents, ce-ci permet d'utiliser huit files d'attente différentes.

Le commutateur a une structure modulaire, avec 12 ports par module. Les modules sont connectés entre eux par une architecture de type matrice de commutation totalement interconnectée (« crossbar » en anglais). Cette structure permet de tester les différences entre la commutation dans le même module (c.-à-d. quand les sources et la destination sont dans le même module) et la commutation entre modules différents (les sources sont dans un module et la destination est dans un module différent).

4.4.7.1 Tests dans le même module

Les premiers tests ont été effectués avec tous les ports source et destination dans le même module de 12 ports.

4.4.7.1.1 Tests avec la même taille de paquets

Les résultats des tests avec la même taille de paquets (1518 octets) sont présentés sur les figures 78 et 79.

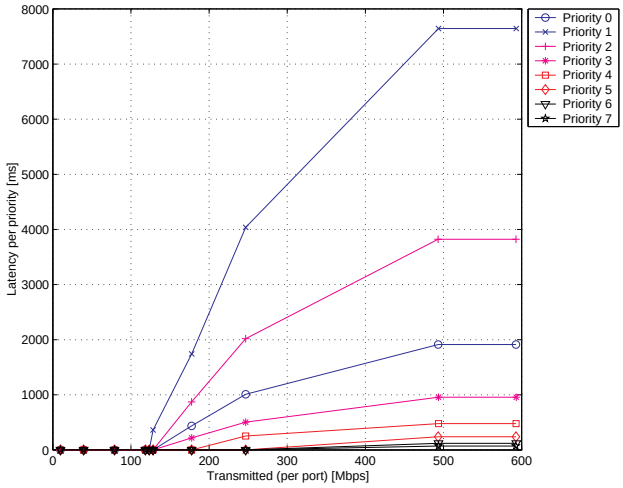


Figure 78 : Délai par priorité en fonction du trafic transmis (même module, même taille de paquets).

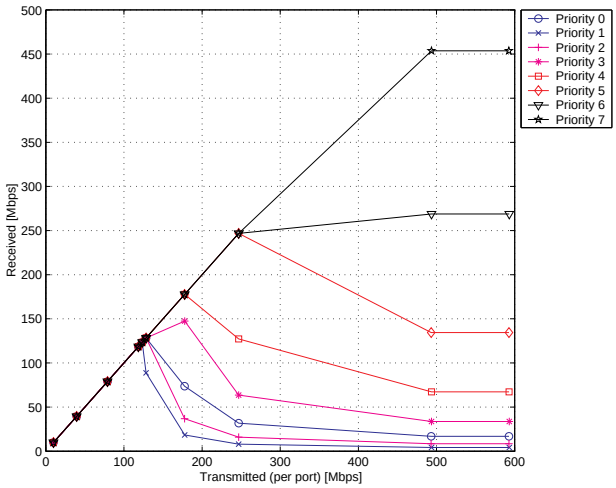


Figure 79 : Le trafic reçu par priorité en fonction du trafic transmis (même module, même taille de paquets).

4.4.7.1.2 Tests avec taille différente des paquets

Pour chaque source/priorité nous avons choisi une taille différente des paquets, comme indiqué dans le tableau 15.

Priorité VLAN	0	1	2	3	4	5	6	7
Taille des paquets [octets]	1280	1152	1024	896	768	512	384	256

Tableau 15 : Les tailles des paquets pour chaque priorité VLAN.

Les résultats obtenus sont présentés ci-dessous.

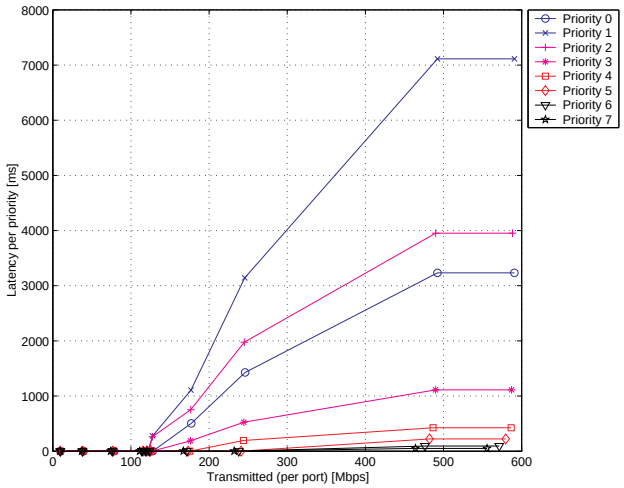


Figure 80 : Délai par priorité en fonction du trafic transmis (même module, taille différente de paquets).

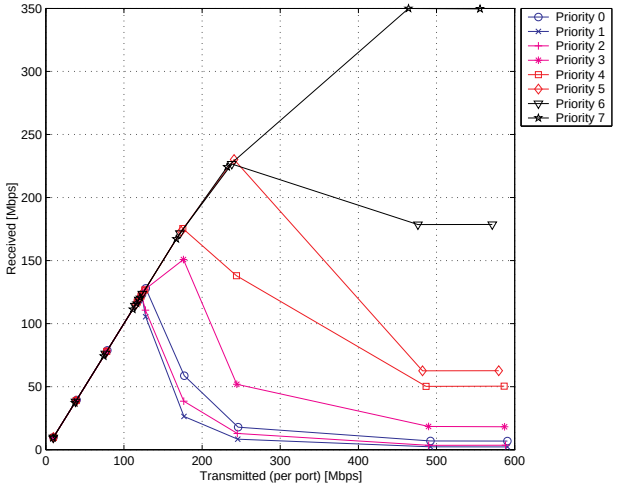


Figure 81 : Le trafic reçu par priorité en fonction du trafic transmis (même module, taille différente de paquets).

4.4.7.2 Tests entre modules

Les tests entre modules (avec des paquets de 1518 octets) permettent de voir si la structure interne du commutateur influence ses performances. Les résultats suivent.

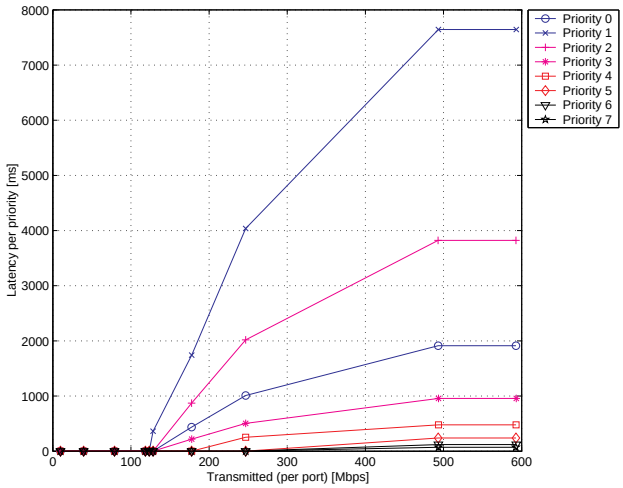


Figure 82 : Délai par priorité en fonction du trafic transmis (entre modules, même taille de paquets).

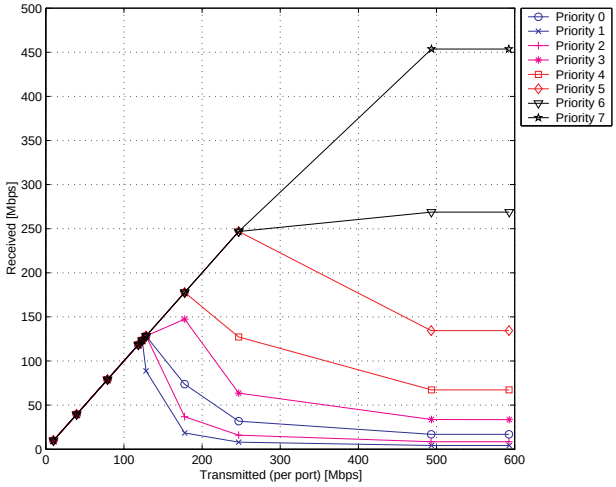


Figure 83 : Le trafic reçu par priorité en fonction du trafic transmis (entre modules, même taille de paquets).

4.4.7.3 Analyse

A cause de l'impossibilité de configurer ce commutateur, nous avons pu tester uniquement l'algorithme d'ordonnancement de type WFQ par défaut.

Les résultats sont légèrement différents pour les tests dans le même module entre les cas de la même taille et des tailles différentes des paquets. Même si les délais unidirectionnels sont similaires, il y a des différences entre la bande passante pour chaque flux. Par exemple, la priorité 7 reçoit environ 450 Mbps pour une taille de

1518 octets et seulement 350 Mbps quand les paquets du flux ont 256 octets. Une cause possible est la surcharge du traitement de 6 fois plus de paquets pour un taux équivalent. Le quantum de l'algorithme, s'il est trop grand, peut aussi constituer une explication.

En comparant les résultats pour les tests dans le même module et entre modules on ne constate pas de différences significatives. Le délai est assez grand pour les flux de priorité basse, montrant que le commutateur possède une large mémoire tampon.

4.4.8 Commutateur #6

Pour ce commutateur il n'est pas possible de changer l'algorithme d'ordonnancement, qui d'ailleurs n'est pas précisé. Il permet d'utiliser huit files d'attente.

L'outil pour introduire la différenciation de service est constitué par les limitations de débit. On peut configurer une limite inférieure garantie pour une certaine file d'attente et une limite supérieure, qui ne peut pas être dépassée par la file d'attente correspondante. Les valeurs représentent des pourcentages de la capacité maximale, mais seules certaines valeurs sont permises. En plus la somme des limites inférieures ne peut pas dépasser 90%.

4.4.8.1 Test sans limitations de débit

Nous avons commencé par un test avec la configuration par défaut: une limite inférieure de 0% et une limite supérieure de 100% pour chaque priorité, avec des paquets de 1518 octets. Les résultats indiquent que l'ordonnancement se fait de manière SP.

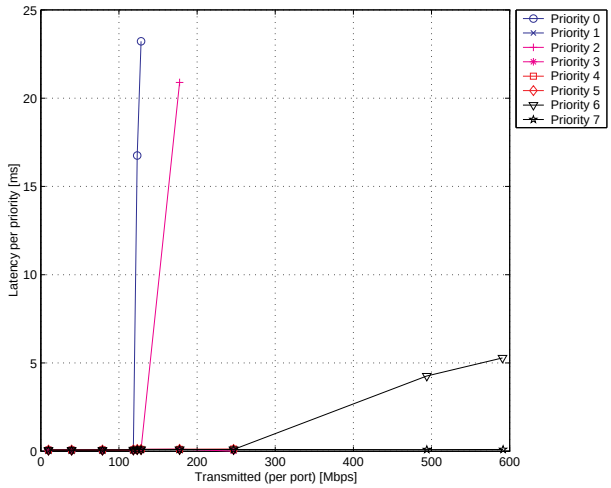


Figure 84 : Délai par priorité en fonction du trafic transmis (sans limitation de bande passante).

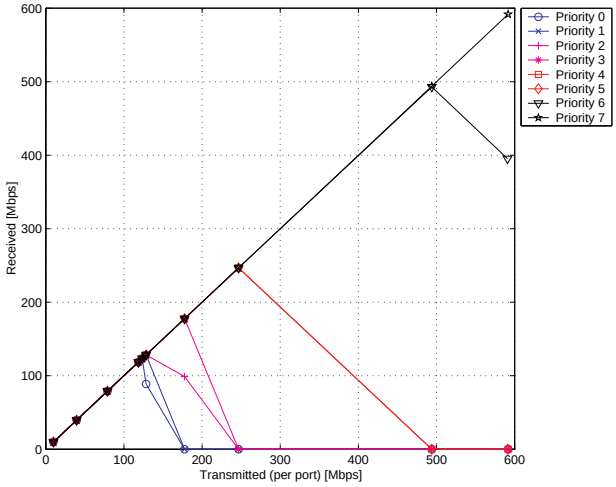


Figure 85 : Le trafic reçu par priorité en fonction du trafic transmis (sans limitation de bande passante).

4.4.8.2 Tests avec limitations de débit

Les tests suivants ont été menés avec des configurations différentes des limites pour chaque priorité, comme suit :

File d'attente	Q_0	Q_1	Q_2	Q_3	Q_4	Q_5	Q_6	Q_7
Limite inférieure [%]	2	3	5	8	10	11	15	20
Limite supérieure [%]	3	5	8	10	11	15	20	25

Tableau 16 : Les limites inférieures et supérieures sur le débit pour chaque file d'attente.

Ces configurations conduisent à une utilisation maximale de la connexion de 97%. L'utilisation observée n'est que de 96%.

4.4.8.2.1 Test avec la même taille de paquets

D'abord tous les ports ont été configurés pour transmettre des paquets de 1518 octets. Les résultats obtenus dans ce cas sont présentés sur les figures 86 et 87.

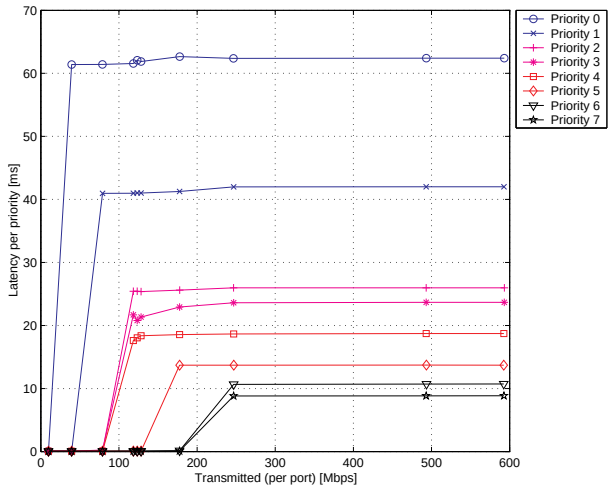


Figure 86 : Délai par priorité en fonction du trafic transmis (avec limitation de bande passante, même taille de paquets).

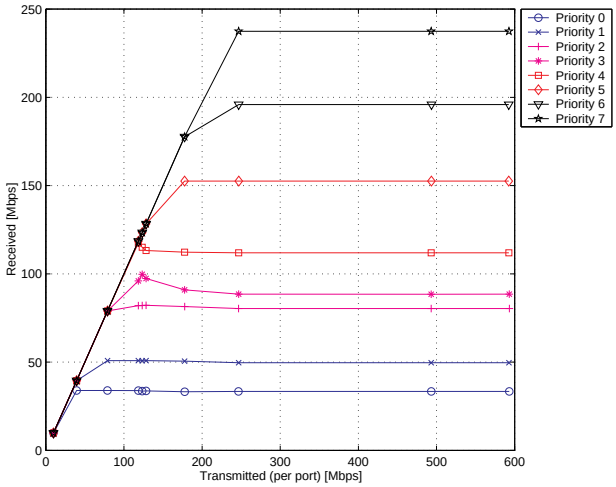


Figure 87 : Le trafic reçu par priorité en fonction du trafic transmis (avec limitation de bande passante, même taille de paquets).

4.4.8.2.2 Tests avec taille différente de paquets

Pour le deuxième test nous avons affecté des tailles différentes pour les paquets transmis par chaque port/priorité comme suit :

Priorité VLAN	0	1	2	3	4	5	6	7
Taille des paquets [octets]	1280	1152	1024	896	768	512	384	256

Tableau 17 : Les tailles des paquets pour chaque priorité VLAN.

Même si les limitations de débit permettaient théoriquement une utilisation de 97% de la capacité de la connexion, l'utilisation observée a atteint seulement 76% ; 21% a été donc gaspillé. Les résultats sont présentés sur les figures 88 et 89.

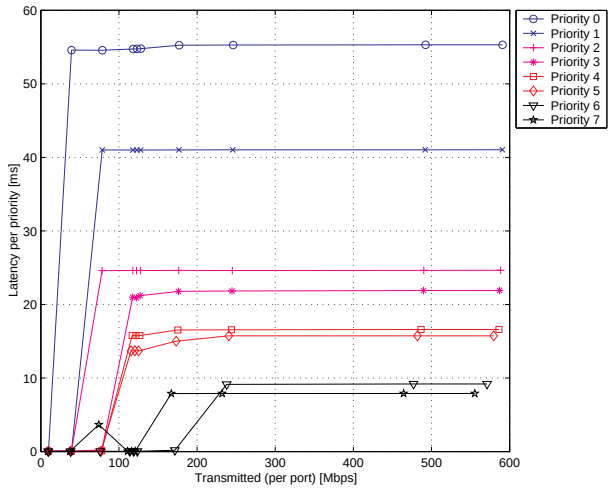


Figure 88 : Délai par priorité en fonction du trafic transmis (avec limitation de bande passante, taille différente de paquets).

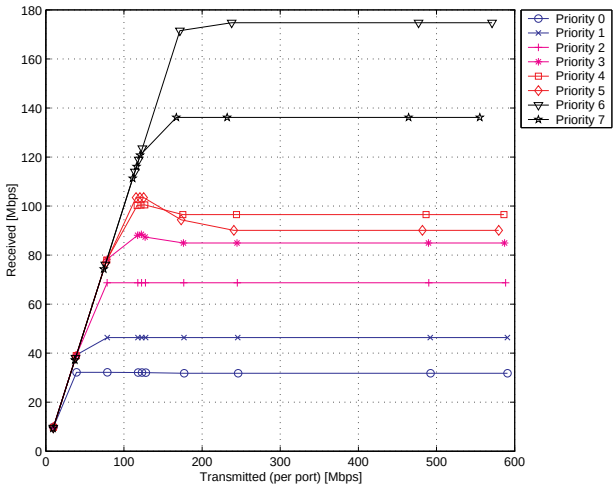


Figure 89 : Le trafic reçu par priorité en fonction du trafic transmis (avec limitation de bande passante, taille différente de paquets).

4.4.8.3 Analyse

Le comportement de ce commutateur ne peut être configuré que par l'intermédiaire des limitations de débit. L'algorithme d'ordonnancement semble être SP, dont on observe le comportement caractéristique, à savoir le blocage des priorités basses, pour lesquelles le délai devient très grand (voir la figure 84).

Les résultats obtenus quand on utilise des limitations sont assez près de ceux attendus quand la taille des paquets est la même. Les différences les plus grandes apparaissent pour les priorités 3 et 7, qui reçoivent environ 10 Mbps de moins que les limites configurées (voir la figure 87).

Par contre, l'utilisation des tailles différentes de paquets induit une différence importante entre le comportement attendu et celui observé par rapport à la limitation de débit. En plus, presque 21% de la capacité reste inutilisée. Une explication est la surcharge du traitement supplémentaire nécessaire pour traiter plus de paquets de petite taille pour la même bande passante.

4.4.9 Discussion

Une façon d'évaluer la performance d'un algorithme d'ordonnancement est de le comparer avec le comportement idéal. Nous avons montré dans 4.4.2 quel doit être le comportement pour l'ordonnancement par SP et WRR.

Pour ces deux algorithmes d'ordonnancement, nous avons comparé le comportement attendu avec celui observé dans nos tests. A noter que la performance d'un commutateur dépend considérablement de l'efficacité des implémentations des mécanismes de différenciation et de leur fidélité par rapport à l'algorithme. Par ailleurs, l'architecture interne du commutateur influence fortement les performances.

Il y a deux paradigmes principaux pour l'architecture interne des commutateurs : mémoire partagée et matrice de commutation totalement interconnectée. Un commutateur avec mémoire partagée a beaucoup de mémoire (des centaines de MB), utilisée en commun par tous les ports pour le stockage des paquets. Par conséquent chaque port a une connaissance globale du trafic arrivant à tous les ports et les décisions d'ordonnancement peuvent être prises en tenant compte de cette information. On peut donc s'attendre à une performance optimale dans ce cas, mais le coût peut être prohibitif à cause des exigences élevées en bande passante d'accès à la mémoire).

Dans un commutateur ayant une architecture de matrice de commutation totalement interconnectée, les ports communiquent entre eux par l'intermédiaire d'une structure globale. Un protocole de communication est nécessaire et la prise de décisions est un processus plus complexe. Cette solution est parfois préférée à cause de son coût plus réduit (car chaque port doit avoir accès seulement à sa propre mémoire), mais la performance peut en souffrir.

Une variante de l'architecture de matrice de commutation totalement interconnectée, avec les ports organisés dans une structure de type arbre, porte le nom d'architecture Clos, d'après le nom de son concepteur. Elle permet d'avoir un commutateur « non bloquant »[1] avec un grand nombre de ports, mais en ayant moins de points de connexion qu'une architecture de matrice de commutation totalement interconnectée. Cependant, dans ces circonstances, le contrôle de la QoS devient encore plus difficile à cause des interactions plus complexes à l'intérieur du commutateur.

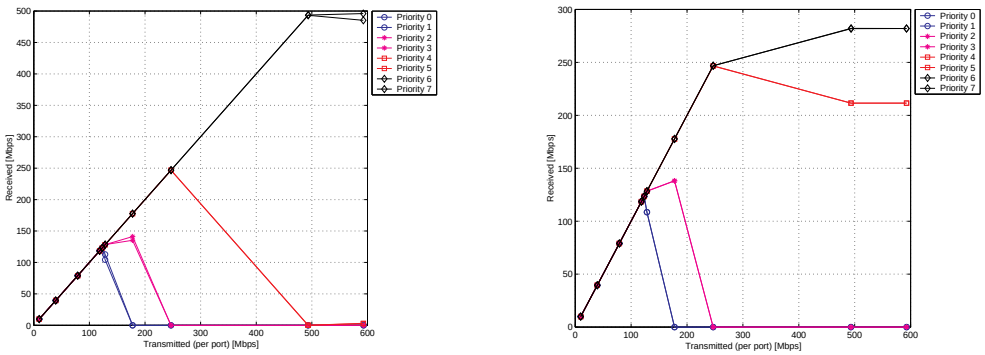
[1] Un commutateur est appelé « non bloquant » si le trafic peut circuler sans perte de n'importe quel port d'entrée à n'importe quel port de sortie, indépendamment des autres flux.

4.4.9.1 Comparaison qualitative

Les graphiques que nous avons présentés véhiculent une représentation synthétique, mais qualitative du comportement observé. Sur les figures 90 et 91 nous avons juxtaposé les graphiques de deux comportements, l'un bon et l'autre mauvais, pour deux mécanismes d'ordonnancement, choisis parmi les résultats des six commutateurs présentés auparavant.

Sur la figure 90(a) on voit un comportement très proche de l'idéal. Les priorités basses sont bloquées dès que le trafic de la priorité la plus haute occupe toute la capacité (i.e. 1 Gbps). Le traitement des deux priorités qui sont assignées dans la même file d'attente est en général équitable, sauf pour Q_3, où une différence d'environ 2% est observée.

La figure 90(b) montre les résultats d'un comportement mauvais, car la file Q_2 n'est pas bloquée pour un taux excédant 500 Mbps ; en conséquence le trafic dans Q_3 ne peut pas occuper toute la capacité et perd des paquets (220 Mbps par transmetteur pour 500 Mbps transmis).



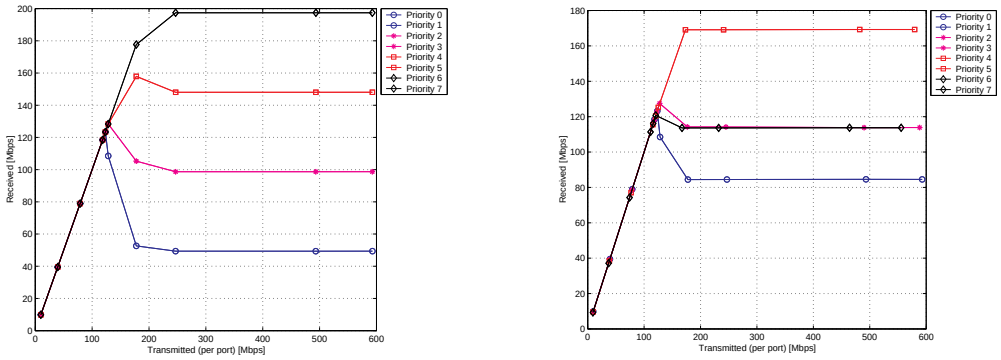
(a) (b)

Figure 90 : Comportement observé bon (a) et mauvais (b) pour SP.

La figure 91(a) présente un bon comportement WRR. Le seul problème est que la somme du trafic réceptionné est seulement de 990 Mbps ; ainsi environ 1% de la capacité est gaspillé, probablement à cause de la complexité supérieure du WRR par rapport à SP. Les poids ont été : 0,4 pour Q_3, 0,3 pour Q_2, 0,2 pour Q_1 et 0,1 pour Q_0. Dans les tests que nous avons effectués, lorsque le trafic consiste en des paquets de même taille, toutes les implémentations de WRR se comportent bien. Par contre, avec des tailles différentes, vu l'ordonnancement par paquets, il n'existe pas un contrôle du

débit comme taux de données mais seulement comme taux de paquets. La solution, utiliser des quanta, permet le contrôle du débit (avec une granularité donnée par la taille des quanta).

Mais nous avons établi que des problèmes d'allocation de la bande passante apparaissent même avec l'utilisation des quanta. Sur la figure 91(b) nous montrons un résultat de ces tests. Les tailles des paquets ont été 256, 512, 1024 et 1518 bytes pour les files d'attente Q_3, Q_2, Q_1 et Q_0 respectivement. Le comportement est évidement loin de l'idéal. La bande passante totale occupée est de 963 Mbps, donc 37 Mbps, presque 4% de la capacité est gaspillée. La cause probable en est le coût de calcul plus grand occasionné par la nécessite de traiter plus de petits paquets que de grands pour arriver au même débit.



(a) (b)

Figure 91 : Comportement observé bon (a) et mauvais (b) pour WRR.

4.4.9.2 Comparaison quantitative

Une évaluation plus précise, quantitative, doit être effectuée afin de permettre la comparaison plus détaillée, plus exacte, des différents commutateurs. Nous proposons la comparaison des valeurs attendues avec celles mesurées, éventuellement par le calcul des rapports de débit et délai pour WRR. Le rapport de débit représente le rapport entre la bande passante utilisée par une file d'attente et la bande passante totale utilisée. Le rapport de délai représente le rapport entre le délai moyen pour une file d'attente et le délai moyen total du trafic.

La comparaison des valeurs mesurées/calculées avec les valeurs attendues, dérivées de modélisation et paramétrées par les poids configurés fournit une information précieuse sur le degré de proximité entre le comportement observé et celui attendu.

Le tableau 18 montre le débit moyen (incluant la partie gaspillée) et le délai moyen pour l'ordonnancement SP, pour les quatre commutateurs ayant quatre files d'attente. Les valeurs sont obtenues quand chaque transmetteur a un débit de 600 Mbps ; le commutateur est donc au delà de la saturation même pour la file la plus prééminente (Q_3). Les valeurs attendues pour le débit et le délai sont incluses. Nous avons noté $\Delta(\#n)$ la valeur dépendante de commutateur du délai moyen pour Q_3 dans le cas de chaque commutateur #*n*, avec *n* de 1 à 4.

Commutateur	***Débit [Mbps]***					***Délai [ms]***			
	Q_0	Q_1	Q_2	Q_3	*Gaspillé*	Q_0	Q_1	Q_2	Q_3
Attendu	**0**	**0**	**0**	**1000**	**0**	∞	∞	∞	**Δ(#*n*)**
#1	0	0	420	566	14	∞	∞	4,9	3,5
#2	0	0	422	564	14	∞	∞	3,5	2,4
#3	0,2	3,4	56	922	18,4	3133	209	13,1	0,9
#4	0	0	5,4	972	22,6	∞	∞	351	2,4

Tableau 18 : Comparaison pour l'ordonnancement SP (paquets de 1518 octets, 600 Mbps par transmetteur).

Dans le tableau 18 nous observons que les deux premiers commutateurs n'arrivent pas à fournir le meilleur service pour la file d'attente la plus prééminente, car les priorités 4 & 5 (Q_2) utilisent 42% de la capacité. Le commutateur #4 a le meilleur comportement, même si les priorités 4 & 5 ne sont pas complètement bloquées, mais 22,6 Mbps sont gaspillés.

Le tableau 19 montre le rapport de débit moyen, la bande passante gaspillée et le rapport de délai moyen pour l'ordonnancement WRR. Le traitement se fait par paquets et les flux ont des paquets de 1518 octets. Les valeurs présentées sont obtenues pour un taux de transmission de 600 Mbps par port, le commutateur est donc au delà de la saturation pour toutes les files d'attente. Les valeurs attendues sont calculées en fonction des poids WRR (0,1, 0,2, 0,3 et 0,4 pour les commutateurs #1 et #2, et 0,1, 0,1, 0,3 et 0,5 pour les commutateurs #3 et #4). Nous avons supposé que les

files d'attente ont toutes la même taille, une hypothèse bien fondée en accord avec notre expérience.

Commutateur	***Rapport de débit***				***Gaspillé [Mbps]***	***Rapport de délai***			
	Q_0	Q_1	Q_2	Q_3		Q_0	Q_1	Q_2	Q_3
Attendu	**0,10**	**0,20**	**0,30**	**0,40**	**0**	**0,52**	**0,27**	**0,15**	**0,06**
#1	0,10	0,20	0,30	0,40	14	0,58	0,26	0,10	0,06
#2	0,10	0,20	0,30	0,40	14	0,60	0,27	0,09	0,05
Attendu	**0,10**	**0,10**	**0,30**	**0,50**	**0**	**0,53**	**0,32**	**0,11**	**0,04**
#3	0,08	0,08	0,33	0,50	14	0,41	0,41	0,11	0,07
#4	0,10	0,10	0,30	0,50	10	0,28	0,28	0,20	0,23

Tableau 19 : Comparaison pour l'ordonnancement WRR par paquet (paquets de 1518 octets, 600 Mbps par transmetteur).

A noter le bon accord qui existe entre les valeurs attendues et observées des rapports de débit. Le commutateur #3 montre une légère déviation, mais elle est expliquée par les particularités du commutateur (voir 4.4.5). Tous les commutateurs gaspillent environ 1,5% de la capacité.

L'accord concernant les valeurs attendues et mesurées du rapport de délai est moins bon. Les commutateurs #1 et #2 ont des rapports de délai plus grands (de 15%) pour la file Q_0, signifiant qu'elle reçoit un traitement moins bon que celui attendu du point de vue du délai. D'un autre côté, le commutateur #3 offre un meilleur service à Q_0 et un moindre service à Q_1. Pour le commutateur #4 les rapports de délai sont presque égaux, même si les rapports de débit sont différents. Même si une explication basée sur l'architecture du commutateur peut exister, ce fait démontre que les délais moyens pour WRR sont imprédictibles et dépendants des commutateurs (et de l'implémentation interne du mécanisme d'ordonnancement). Ce fait est une preuve de plus que les mécanismes courants de QoS, même s'ils introduisent une évidente différenciation de service, n'offre pas pour autant la prédictibilité indispensable au déploiement des applications multiples sur la même connexion, dans l'esprit de la convergence.

5 Mesure passive de la qualité

Ce chapitre présente les mesures passives effectuées au moyen du système de test que nous avons conçu.

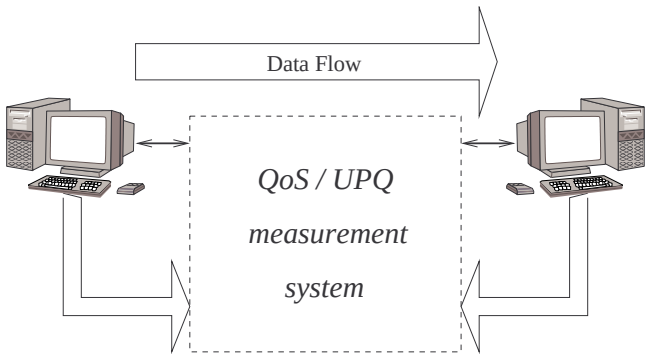


Figure 92 : Presentation simplifiée du système de test.

Les résultats présentés sont obtenus pour deux applications différentes, qui tournent sur deux PC : le transfert de fichiers et le VoIP. Les applications sont exécutées sur les PC, générant des flux de données. Les données sont envoyées à travers un émulateur réseau, qui introduit de manière artificielle une perte de paquets et un délai variable configurés, permettant l'étude de comportement des applications dans des états du réseau variés.

5.1 Le système de test

La figure 93 montre le système que nous avons conçu dans une configuration de test habituelle. Nous utilisons deux diviseurs FastEthernet pour refléter le trafic circulant sur les connexions entre les deux PC (avec le système d'exploitation Linux) sur lesquels tournent les applications étudiées. Le trafic est saisi par quatre cartes réseau programmables Alteon, deux pour chaque diviseur pour monitorer le trafic full-duplex. Depuis chaque paquet, l'information nécessaire pour calculer les paramètres QoS est extraite et stockée dans la mémoire locale, sous la forme d'un descripteur de paquet. Les PC hôtes, qui contrôlent les cartes réseau programmables, collectent périodiquement cette information et la stockent dans des fichiers de descripteurs. Ces données sont utilisées par le logiciel « QoS Meter » pour le calcul ultérieur des paramètres QoS : délai unidirectionnel et gigue, perte de paquets et débit. Nous pouvons calculer des valeurs instantanées, valeurs moyennes et des histogrammes.

Des cartes PCI spéciales sont employées pour synchroniser les mesures de temps entre les deux points de test.

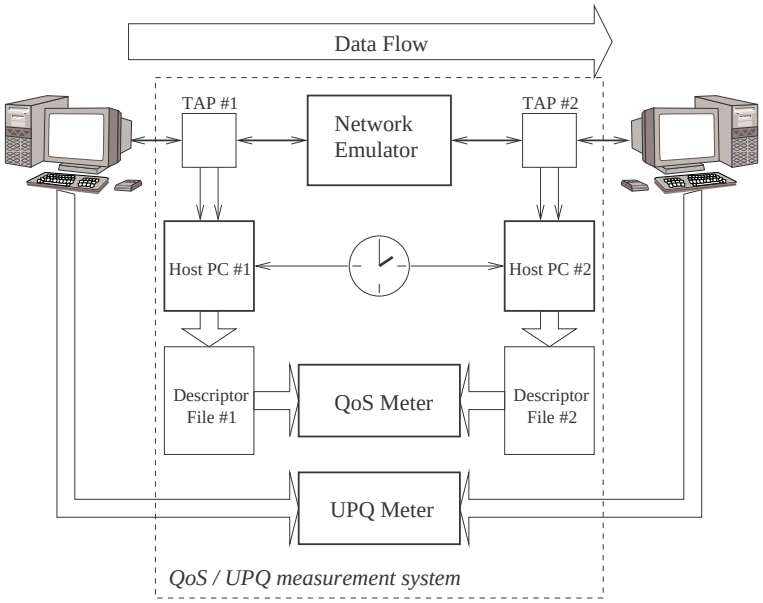


Figure 93 : La configuration du système de mesure.

Un processus additionnel, qui dépend de l'application étudiée, prend place pendant le test. Par exemple, quand on étudie le VoIP, l'application crée un fichier équivalent à la sortie audio, qui nous permet d'utiliser une métrique spécifique, le score PESQ, pour évaluer la qualité vocale pour la communication VoIP (dans le module « UPQ Meter »).

La partie la plus importante est la corrélation des paramètres QoS qui ont été calculés avec l'UPQ évaluée pour l'application étudiée. Cette corrélation permet de tester des connexions réseau avant de déployer les applications mêmes et prédire l'UPQ attendue pour ces applications.

Dans nos expériences nous utilisons un émulateur réseau, NIST Net. Cet émulateur peut dégrader la QoS du réseau par l'introduction contrôlée et artificielle du délai, gigue, perte de paquets et limitation de bande passante. Nous avons opté pour une telle solution afin de pouvoir analyser une large gamme de conditions réseau contrôlables, et dans le même temps utiliser des applications réelles. Cela n'aurait pas été possible en utilisant des réseaux réels ou des simulateurs.

Nous avons dû évaluer les fonctionnalités de NIST Net et voir si nos attentes par rapport aux types de dégradation et la précision correspondante sont satisfaites.

La même remarque vaut pour Speak Freely, l'application VoIP, qui a dû être modifiée afin de fournir toutes les fonctionnalités requises (e.g. la sauvegarde d'un fichier équivalent au signal audio réceptionné).

Pour le transfert des fichiers nous utilisons un client et un serveur de FTP standard pour Linux.

Nous présentons aussi une série de tests préliminaires effectués pour calibrer le système, les détails des configurations expérimentales et quelques questions liées à la représentation des résultats.

5.1.1 Mesure de la QoS

Cette section décrit avec davantage de détails les composants et procédés utilisés pour mesurer la QoS.

5.1.1.1 Diviseurs passifs

Les diviseurs (« tap » en anglais) sont des dispositifs réseau passifs qui peuvent être utilisés pour monitorer une connexion full-duplex, dans notre cas une connexion FastEthernet.

La configuration décrite inclut deux diviseurs FastEthernet produits par NetOptics, qui reflètent le trafic circulant dans les deux directions et le dirigent vers des cartes réseau programmables. Ces connecteurs sont des diviseurs passifs de signaux qui permettent le renvoi des signaux, avec aiguillage d'une partie d'entre eux vers les ports de monitoring.

Les diviseurs permettent de se brancher sur le réseau pour effectuer la surveillance, l'analyse et le dépannage. Ils représentent la meilleure solution pour monitorer des connexions full-duplex. Utiliser l'option de mirorisation de ports (« port mirroring » en anglais) des commutateurs a l'inconvénient d'introduire des délais dépendant de la charge sur le commutateur. On peut envisager l'utilisation de concentrateurs, mais ils ne permettent pas le monitoring full-duplex à cause des collisions possibles entre le trafic dans les deux directions.

5.1.1.2 Cartes réseau Alteon

Les autres composants utilisés sont les cartes réseau Alteon Fast et Gigabit Ethernet. Les détails de leur architecture ont été déjà présentés en 4.2.1.1. Le PC hôte

communique avec le NIC par un segment de mémoire partagée et des structures de contrôle. Le NIC effectue tous les traitements nécessaires au niveau physique et transport. Les NICs ont été programmées pour monitorer des connexions de 100 Mbps opérant à plein débit. La mémoire de 1 MB est utilisée pour stocker le programme et les descripteurs de paquet extraits du trafic reflété.

5.1.1.3 Procédé de mesure

Le système de mesure de la QoS utilise l'information collectée par les NICs pour calculer les paramètres QoS pour chaque connexion réseau. Un NIC est nécessaire pour chaque direction de trafic ; de la sorte, quatre NICs sont nécessaires pour monitorer deux connexions full-duplex. Ces NICs produisent pour chaque paquet un descripteur avec les champs suivants :

- Timbre-à-date (ou estampille, « timestamp » en anglais) – 32 bits ;
- Identificateur de paquet – 32 bits ;
- Taille du paquet – 16 bits ;
- Numéro de protocole – 8 bits.

Le timbre-à-date représente l'instant de réception du paquet, incluant le temps nécessaire pour stocker le paquet dans la mémoire tampon de réception. L'identificateur de paquet est une clé unique. Il est obtenu à partir d'informations contenues dans le paquet, tel que le numéro de séquence pour les en-têtes RTP et TCP, la somme de contrôle etc. La taille du paquet contient la dimension du paquet, en octets, incluant la somme de contrôle (Cyclic Redundancy Check, CRC) de 4 octets. Le numéro de protocole permet de distinguer différents protocoles et de filtrer les paquets présentant un intérêt.

5.1.1.3.1 Synchronisation temporelle

La synchronisation entre NICs est réalisée par l'utilisation d'un système de temps global, formé par une carte horloge « maître » et des cartes horloge « esclave ». La carte maître envoie un signal d'horloge rectangulaire à toutes les cartes esclave. Pour une robustesse accrue, la valeur courante de l'horloge maître est envoyée périodiquement (128 fois par seconde) aux cartes esclave pour la mise à jour. L'horloge globale a une fréquence de 66,67 MHz, dérivée de l'horloge PCI de 33,33

MHz, et est utilisée pour stocker une information de temps valide pour tout le système.

Par contre, les NICs utilisent des horloges processeur avec une fréquence de 86 MHz pour toutes les opérations internes. Les timbres-à-date sont obtenus par la transformation de la valeur d'horloge locale en une valeur globale, en utilisant des tableaux de conversion qui sont générés 128 fois par seconde. L'erreur globale de nos mesures de délai est déterminée par les facteurs suivants :

- L'horloge globale provient de l'horloge PCI ;
- L'horloge locale de 88 MHz est à base d'un oscillateur quartz à 22 MHz ;
- Des opérations sur des entiers sont effectuées pour la conversion de l'horloge locale à l'horloge globale ;
- Les NICs utilisent des transferts DMA pour lire les valeurs de l'horloge globale depuis les cartes horloge ;
- Par la conception même des NICs, la différence de temps entre la réception effective d'un paquet et le début de son traitement est variable.

L'erreur totale pour la mesure de délai par notre système est limitée à 900 ns (voir 5.1.1.4), étant en principe une superposition de gaussiennes.

Les signaux d'horloge et de mise à jour étant envoyés par des câbles coaxiaux courts, ce mécanisme de synchronisation ne peut pas être appliqué pour des localisations éloignées. Pour des tests à longue distance, la synchronisation peut être réalisée par l'utilisation de systèmes GPS, comme décrit dans [Kor-03].

5.1.1.4 L'évaluation du système de monitoring

Avant de commencer nos expériences nous avons évalué le système de mesure de la QoS dans la configuration particulière présentée sur la figure 94. Les diviseurs étant connectés l'un à l'autre, le temps passé par un paquet entre les deux diviseurs est constant. Nous avons mesuré la différence entre les instants de réception d'un paquet aux deux PC hôtes. Nos résultats sont présentés sur la figure 95 sous la forme d'un histogramme.

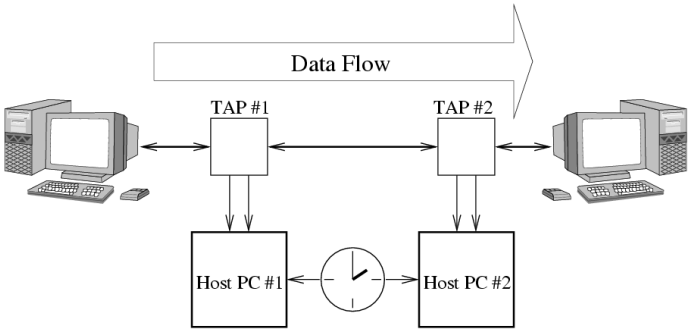


Figure 94 : Configuration spéciale pour tester le système de monitoring.

L'histogramme des différences de temps aurait dû, en principe, consister en une seule classe de valeur égale au temps de propagation entre les deux diviseurs. L'étendue de 900 ns de l'histogramme et une largeur à mi-hauteur (Full Width at Half Maximum) de 400 ns sont assez petites pour permettre des mesures de délai de l'ordre de quelques dizaines de microsecondes, une précision satisfaisante pour nos expériences.

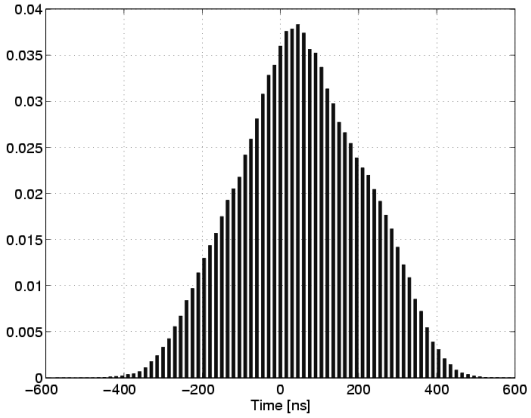


Figure 95 : Histogramme des différences de temps entre les deux diviseurs.

5.1.1.5 Métriques QoS

Sur la base de données collectées par le système de mesure de la QoS, nous calculons les paramètres QoS suivants : le délai unidirectionnel moyen, la gigue, le débit moyen et la perte de paquets suivant les standards de UIT-T et ceux de IETF. Pour la gigue, le délai de référence a été choisi de trois manières : le délai du premier paquet [ITU-380], le délai moyen et le délai du paquet précédent [RFC-3393]. La figure 96 montre une comparaison de ces méthodes pour le calcul de la gigue moyenne sur un flux de plus de 6000 paquets de 1518 octets.

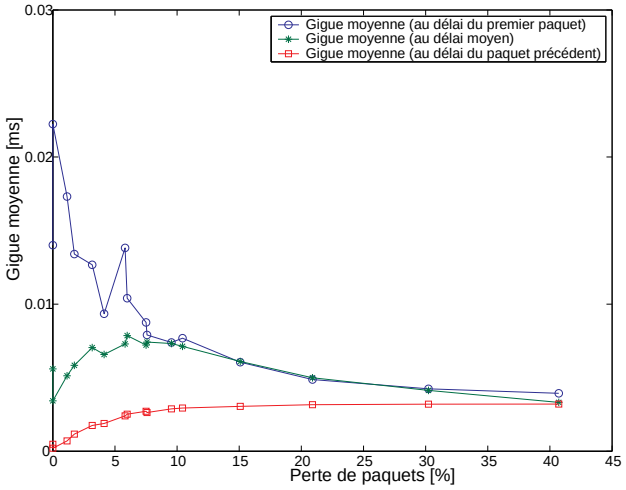


Figure 96 : Gigue moyenne calculée en fonction du délai du premier paquet, du délai moyen et du délai du paquet précédent.

La gigue moyenne du point de vue d'une application est donnée par les variations entre paquets successifs. Nous considérons donc que ce calcul en fonction de délai du paquet précédent est le plus approprié et nous allons utiliser ce manière de calcul pour tous les résultats. En plus, la figure 96 montre qu'elle a une variation plus « lisse » en fonction de la perte de paquets : elle peut donc être utilisée avec plus de confiance.

La perte de paquets est déterminée à base d'identificateurs de paquet dans les descripteurs. Un paquet est considéré comme perdu si son identificateur, qui apparaît dans le fichier de descripteurs au premier point de mesure, n'apparaît pas dans le fichier de descripteurs au second point de mesure.

Nous pouvons aussi calculer des valeurs instantanées (e.g. pour le débit) et des histogrammes variés (e.g. l'histogramme de l'espacement entre les temps de réception des paquets).

5.1.2 Mesure de l'UPQ

Pour les expériences de transfert des fichiers aucune étape supplémentaire n'est requise. Les paramètres UPQ peuvent être calculés à partir des mêmes données que les paramètres QoS.

Notre application VoIP produit un fichier de sortie qui permet la reconstruction du signal vocal réceptionné. A partir de ce fichier et du fichier initial envoyé par l'application VoIP, le logiciel « UPQ Meter » estime la qualité perçue pour la

communication VoIP, en utilisant le score PESQ (voir 5.2.2.4 pour plus de détails sur l'application VoIP utilisée, Speak Freely).

5.1.3 L'émulateur réseau NIST Net

Il y a quatre voies possibles pour réaliser des études de réseau. La méthode la plus abstraite est analytique [Whi-02] ; des modèles numériques du comportement des éléments réseau sont utilisés pour calculer directement les résultats. La deuxième est la simulation, quand une représentation du réseau réel est créée et puis un moteur de simulation utilise les modèles associés aux éléments de réseau simulés et les connexions entre eux pour calculer les résultats estimés. La troisième approche consiste à utiliser des applications réelles qui s'exécutent sur des réseaux réels. Des mesures précises de ce qui se passe dans un scénario réel peuvent être ainsi effectuées. Le problème dans ce cas est le fait qu'un réseau réel ne peut pas être contrôlé avec grande précision ; par la suite, la gamme de scénarios qui peut être testée est limitée.

On peut amalgamer les deux dernières voies mentionnées, en combinant leurs avantages. Cette approche met en jeu un émulateur réseau, un outil qui reproduit les effets des réseaux réels sur le trafic, par l'introduction d'une dégradation artificielle de la QoS. Le trafic utilisé est du trafic réel ; on peut par conséquent toujours analyser les effets des conditions dans le réseau sur des applications réelles. Pourtant, étant donné que la dégradation introduite est contrôlable, une large gamme de conditions réseau peut être étudiée.

5.1.3.1 Caractéristiques

L'émulateur réseau NIST Net [NIST] est un « gratuiciel » de l'Institut National des Standards et Technologies de USA qui émule dynamiquement les conditions réseau. Cet émulateur est implémenté comme un module de noyau Linux qui fait usage d'un module spécial de temps réel pour obtenir une précision supérieure. L'émulateur permet l'expérimentation contrôlée et reproductible avec des applications qui sont sensibles et/ou adaptatives aux performances du réseau. Opérant au niveau IP, NIST Net peut émuler les caractéristiques de performance de bout en bout imposées par des conditions réseau variées. En l'utilisant, nous avons pu introduire des pertes de

paquets et délais pour des débits jusqu'à 100 Mbps avec des paquets de taille minimale.

Le taux de perte envisagé est donné à NIST Net comme le pourcentage de paquets perdus. On peut utiliser une corrélation entre les événements de perte successifs, c.-à-d. qu'il y a une probabilité non négligeable d'avoir plusieurs paquets perdus à la suite. Nous avons décidé de ne pas activer cette corrélation car les conditions de ce type sont difficiles à évaluer et notre but était de produire une étude de base, i.e. étudier le scénario le plus élémentaire. Par la suite, dans tous nos tests, le paramètre correspondant a été mis à zéro ; il n'y a donc pas eu de corrélation.

Pour le délai il faut fournir le délai moyen envisagé et sa déviation standard (une mesure de la gigue). On peut également utiliser une corrélation entre valeurs successives du délai afin qu'ils ne différent pas significativement. Notre scénario de base suppose que les valeurs du délai ne sont pas corrélées ; par suite, dans tous nos tests, ce paramètre a été mis à zéro.

L'émulateur permet aussi de configurer une limitation de bande passante, mais nous ne l'avons pas utilisée dans nos expériences, car cette fonctionnalité ne nous a pas été nécessaire.

5.1.3.2 Evaluation

Utilisant le système de mesure de la QoS/UPQ décrit dans 5.1.1, nous avons mesuré les valeurs réelles des paramètres QoS après la dégradation introduite par NIST Net, avec une précision de 1 μs pour la mesure du délai. Tous les résultats présentés dans ce mémoire sont basés sur les valeurs réelles de paramètres QoS, et non sur celles envisagées, tels qu'elles sont fournies à NIST Net. Par conséquent nous avons une description très précise des flux de trafic. Un test a été tout de même fait pour déterminer si l'émulateur réseau peut traiter des taux de transmission au niveau de la capacité maximale (100 Mbps) avec une précision suffisante pour la dégradation artificielle introduite – perte de paquets et gigue.

Pour commencer, voici les résultats d'une série de tests effectués pour estimer la fidélité du délai introduit par rapport à celui qui a été configuré. Environ 5000 paquets de 64 octets ont été envoyés à travers l'émulateur. La gigue a été configurée à 0 pour

tous les tests. L'accord observé est assez bon. L'écart constaté sera compensé dans nos expériences par le fait que nous mesurons le délai réel avec une grande précision.

Délai configuré [ms]	1	2	3	4	5	10
Délai mesuré [ms]	1,061	2,008	3,144	4,119	5,282	10,317
Ecart-type	0,008	0,043	0,033	0,019	0,077	0,022

Tableau 20 : Le délai moyen configuré, le délai moyen mesuré et sa deviation standard pour une gigue nulle.

Un test similaire a été effectué pour évaluer le comportement du point de vue de la perte de paquets. Le tableau 21 indique les résultats obtenus, les statistiques étant réalisées sur 100 tests. Nous avons fait varier le nombre de paquets envoyés dans chaque test pour voir dans quelle mesure la précision dépend du nombre de paquets.

Nombre de paquets	1000	10000	100000
Taux de perte configuré [%]	10	10	10
Taux de perte mesuré [%]	9,9730	10,0011	10,0003
Ecart-type	0,9012	0,0870	0,0010

Tableau 21 : Le taux de perte moyen configuré, le taux de perte moyen observé et son écart-type pour un nombre variable de paquets dans un test.

On peut conclure que la précision augmente avec le nombre de paquets dans un test. Par contre dans nos expériences nous allons utiliser un nombre relativement réduit de paquets ; ainsi les résultats de la première colonne s'appliquent plutôt. De nouveau cet inconvénient est compensé par une mesure de la perte réelle dans chaque expérience.

Nous avons également effectué des tests pour observer la distribution du délai. Selon la documentation de NIST Net et le code source, l'émulateur utilise par défaut une distribution observée de manière expérimentale[1]. Elle est représentée sur la figure 97 pour un délai moyen de 300 ms et un écart-type du délai de 20 ms (équivalent à la gigue moyenne). A noter la longue queue de la distribution et le fait que les valeurs du délai sont limitées supérieurement à 380 ms, d'où le petit pic dans la partie droite.

[1] La distribution est basée sur 25000 RTT de messages *ping* vers un point éloigné du réseau (www.ntt.co.jp).

Cette gamme des valeurs, donnée par la formule qui suit, provient de la conception de l'algorithme d'introduction de délai artificiel du NIST Net :

$$D \in \left[\overline{D} - 4\sigma, \overline{D} + 4\sigma\right]. \qquad (5.1)$$

où D est une valeur introduite du délai, $\overline{D}$ est le délai moyen envisagé et σ est l'écart-type envisagé.

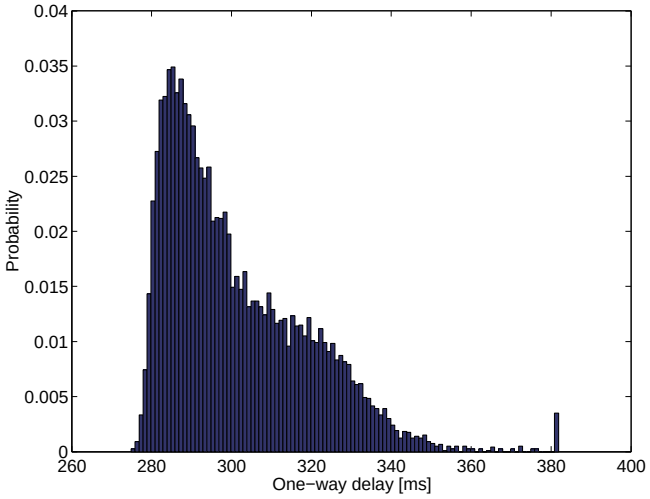


Figure 97 : Distribution du délai.

La figure 98 montre les même données dans une représentation différente, utilisant la fonction de distribution cumulative (FDC), comme 1-*FDC*(délai) sur une échelle logarithmique[1]. Ceci permet de déterminer le nombre des paquets qui ont un délai supérieur à une certaine limite (e.g. dans notre cas 40% des paquets ont des délais excédant 300 ms).

[1] Cela signifie que les valeurs sur l'axe *y* (1, 0,9, 0,8 etc.) ne sont pas représentées sur une échelle linéaire, mais en accord avec leur logarithme.

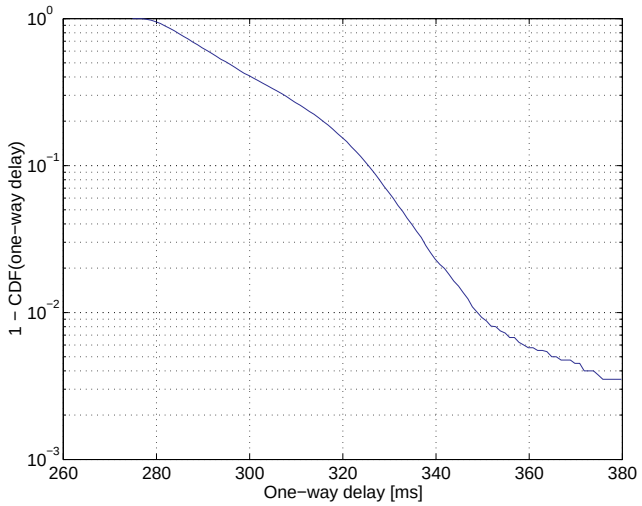


Figure 98 : 1-*FDC*(délai).

Nous avons calculé la gigue en utilisant la définition de (2.9) ; son histogramme est représenté sur la figure 99. Pour mieux souligner les propriétés de la gigue introduite, nous avons calculé son FDC et représenté 1-*FDC*(gigue) sur une échelle logarithmique (voir la figure 100). De ce graphique on déduit, par exemple, qu'environ 10% des valeurs instantanées de la gigue sont en dessus de 40 ms et 0,6% en dessus de 80 ms. Ainsi, une compensation d'une gigue allant jusqu'à 80 ms (en utilisant une mémoire tampon) va conduire au fait que 99,4% des paquets sont joués normalement (i.e. dans le bon ordre et sans intervalles de silence) ; les 0,6% restants vont causer des effets négatifs, notamment des intervalles de silence, à cause de leur indisponibilité lors de la restitution du son.

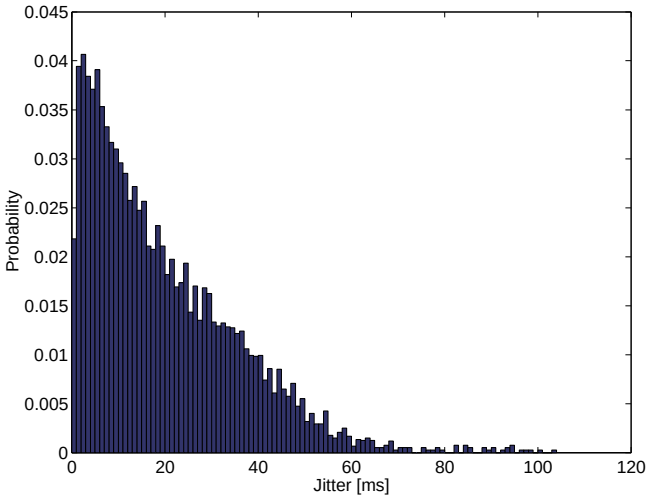


Figure 99 : Distribution de la gigue.

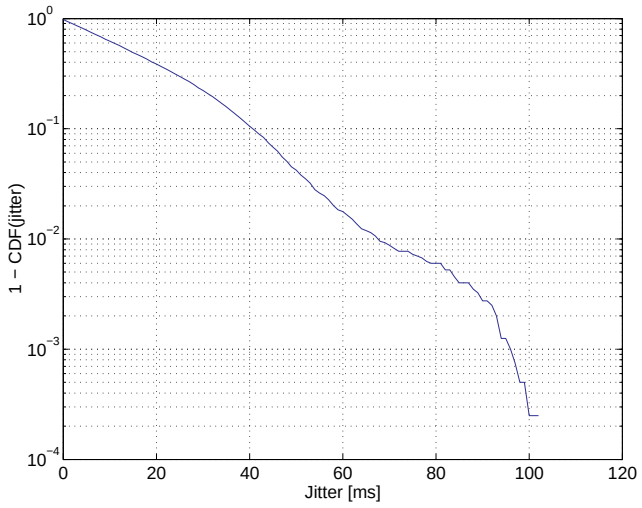


Figure 100 : 1-*FDC*(gigue).

Même si l'émulateur offre une modalité de configuration des limitations de débit, les tests ont montré que le comportement correspondant n'est pas celui attendu. Quand une telle limite est configurée, NIST Net réduit le débit de sortie, mais stocke les paquets qui ne peuvent pas être expédiés dans une mémoire tampon. Une fois que cette mémoire est pleine, tous les paquets sont expédiés, déterminant un débit de sortie qui dépasse momentanément celui d'entrée. Ce fait est en contradiction avec nos attentes par rapport au débit fixe, qui doit s'accompagner de perte si le débit à l'entrée est supérieur au débit configuré pour la sortie. A noter que pour VoIP les exigences de bande passante sont connues et spécifiques à chaque codeur. Pour la suite, nous n'avons pas besoin d'utiliser cette caractéristique pour la communication vocale et on peut se limiter à l'étude de l'influence de la gigue et de la perte de paquets. Pour TCP nous avons utilisé la limitation intrinsèque à 100 Mbps de la connexion FastEthernet que nous avons employé.

Le problème principal rencontré avec NIST Net est que l'introduction d'un délai variable conduit potentiellement à un changement d'ordre. Intuitivement, si les paquets ont au transmetteur un espacement de 40 ms il est évident que l'introduction d'une certaine gigue, qui est une variation du délai unidirectionnel, va conduire à un changement d'ordre au récepteur (à partir d'un certain seuil pour la gigue). Une démonstration plus rigoureuse se trouve dans [Beu-04a]. Pour VoIP, jouer les paquets dans le mauvais ordre a des effets négatifs sur la qualité perçue, car les échantillons de voix ne sont pas rendus dans l'ordre dans lequel ils ont été produits.

Une étape supplémentaire est donc nécessaire, et cette opération peut se faire à l'aide des timbres-à-date inclus dans l'en-tête RTP des paquets VoIP. Puisque notre application VoIP (voir 5.2.2.4) produit un fichier qui contient tous les paquets reçus, nous avons décidé de traiter les paquets en mauvais ordre dans le cadre du processus de compensation de gigue (voir 5.2.2).

5.2 Applications étudiées

Le système que nous avons conçu et implémenté permet la mesure des paramètres QoS du réseau. Ce système permet l'évaluation objective des exigences des applications réseau pour qu'elles délivrent une qualité acceptable par les utilisateurs. Nous avons employé ce système pour l'évaluation de la performance et l'étude du comportement des applications réseau réelles, telles que le transfert de fichiers et la téléphonie IP. Pour ces applications des métriques de la qualité perçue par l'utilisateur ont été définies/sélectionnées pour déterminer leurs exigences en termes de QoS.

Etant donné que nous mesurons simultanément la QoS du réseau et l'UPQ de l'application, nous pouvons les corréler. Déterminer les exigences des applications a deux usages principaux :

- Prédire l'UPQ attendue pour une application tournant sur un réseau donné, prenant en compte les paramètres QoS mesurés, et comprendre les causes de défaillance d'applications ;
- Concevoir/configurer un réseau pour fournir les conditions nécessaires afin qu'une application réseau tourne au niveau UPQ désiré.

Les applications que nous avons sélectionnées pour notre étude sont utilisées de manière extensive dans les réseaux locaux et dans l'Internet. Elles différent dans leurs comportements :

- Le transfert des fichiers est une application élastique, basée sur TCP. Le TCP essaye d'occuper le plus de bande passante disponible. Il adapte aussi son taux de transmission aux conditions réseau courantes – pour des taux de perte élevés, il réduit le taux de transmission ;
- VoIP est une application inélastique basée sur UDP. UDP utilise une quantité fixe de bande passante et n'a pas de mécanismes inhérents de récupération en cas d'erreur ; ainsi il ne peut pas s'adapter aux conditions réseau courantes.

Dans les expériences que nous avons effectuées, perte de paquets et gigue ont été introduits artificiellement par l'utilisation d'un émulateur réseau. Pour les transferts de fichiers, la perte a été introduite dans les deux directions.

5.2.1 Transfert des fichiers

Le transfert de fichiers est une des applications réseau de base. Il est utilisé dans le simple but de transférer de données entre deux points en utilisant le protocole FTP (File Transfert Protocol), mais son usage le plus important est dans le cadre de la navigation sur Internet par le protocole HTTP (Hyper Text Transfer Protocol) Nous avons étudié le FTP, qui est utilisé de manière extensive dans le réseaux locaux ainsi que dans l'Internet. D'autres recherches sont centrées sur HTTP, comme par exemple dans [Pad-01].

Comme mentionné auparavant, le transfert de fichiers par FTP ou HTTP est une application élastique à base de TCP. Par conséquent il s'adapte aux conditions du réseau et fournit un transfert de données sûr par son mécanisme de retransmission en cas des pertes.

Nous avons choisi cette application pour montrer comment notre système de test peut être utilisé pour évaluer les propriétés d'une implémentation du TCP quelconque (dans notre cas celle qui est disponible sur le système d'exploitation Linux). Elle est largement répandue et son étude offre donc une vue d'ensemble sur les performances du transfert de fichiers pour les utilisateurs réguliers. Nous ne nous sommes pas intéressés aux améliorations de l'algorithme de base de Jacobson [Jac-88], car ces améliorations sont toujours plus ou moins expérimentales (voir, par exemple, [FAST-02]), et les implémentations de TCP courantes sont principalement basées sur l'algorithme initial.

5.2.1.1 FTP

Les applications FTP utilisées dans nos tests sont :

- Un client FTP ftp-0.17-7 sur une machine Linux avec un noyau 2.4.6 (la fenêtre maximale du TCP a été de 64 kB) ;
- Un serveur FTP, wu-ftpd-2.6.1-20, sur une machine Linux avec un noyau 2.4.9 (la fenêtre maximale du TCP a été de 64 kB).

Ces implémentations sont standard dans l'environnement Linux et nous avons aussi utilisé la configuration standard (e.g. pour la fenêtre maximale du TCP). Ces applications étant d'usage continu, une évaluation particulière n'a pas été nécessaire. D'ailleurs aucune modification n'a été nécessaire pour effectuer nos expériences sur ces applications.

L'utilisation d'un client FTP est assez basique, il faut lui indiquer le nom de l'ordinateur sur lequel le serveur tourne. Apres la connexion, des commandes de transfert de fichiers sont données. Dans tous nos calculs sur le transfert de fichiers nous n'avons pas pris en compte le temps nécessaire pour établir la connexion principale, mais seulement le temps écoulé entre l'initiation d'un transfert et son aboutissement.

5.2.2 Téléphonie IP

Nous nous sommes concentrés sur VoIP car il s'agit d'une des applications les plus utilisées dans laquelle la perception humaine joue un rôle fondamental. Les applications VoIP ne « stressent » pas le réseau, du point de vue de la bande passante, car les exigences de la transmission de la voix sont réduites (e.g. 64 kbps de donnée voix ou même moins) Par contre, le délai – et sa variation –, et la perte des paquets sont très importants car ils affectent directement la qualité perçue et, par la suite, la satisfaction des utilisateurs. Il résulte que VoIP est une des applications qui bénéficierait amplement de l'usage des techniques de contrôle de la QoS et la différenciation de services. Un des buts de notre recherche est de fournir l'information nécessaire pour le réglage afin que les réseaux offrent aux utilisateurs la qualité désirée au niveau de l'application.

La relation entre la QoS et l'UPQ pour applications VoIP est quantifiée par la mesure simultanée de la QoS dans le réseau et l'évaluation de l'UPQ pour la communication VoIP par l'utilisation du score PESQ [ITU-862].

Quand nous avons entamé l'étude de VoIP [Beu-04a], nous avons découvert que certaines questions de base ne sont pas suffisamment documentées dans la littérature et une analyse plus détaillée a été nécessaire. A titre d'exemple : tout le monde mesure la gigue, mais celle-ci est définie de façon très variable, comme il a été montré déjà dans la section 2.2, et elles doivent être spécifiées quand on présente et utilise les résultats des tests. La compensation de gigue (« dejittering » en anglais) est aussi une

tâche potentiellement complexe, et le nombre d'algorithmes est si grand que sélectionner un algorithme de base n'est pas facile. Un autre type de variabilité réside dans la manière très diverse de gérer les paquets qui arrivent dans le mauvais ordre (« out-of-order » en anglais) constituent un problème qui peut être abordé de différentes façons.

5.2.2.1 Questions de délai

Le délai de bouche à l'oreille (de bout en bout) est un paramètre important pour la communication VoIP du point de vue de la perception humaine, mais n'affecte pas par lui-même la qualité du signal vocal. De ce fait nous n'avons pas inclus une étude précise de l'influence que le délai unidirectionnel a sur la qualité vocale perçue dans notre recherche.

A noter toutefois qu'il a plusieurs résultats et recommandations UIT-T concernant l'influence du délai de bout en bout sur l'UPQ de VoIP [Reijs], [ITU-1541]. Ainsi, un délai entre 0 et 100-150 ms (en fonction de la source citée) assure une haute interactivité, et un délai entre 100-150 ms et 400 ms fournit un niveau acceptable d'interactivité. Les délais excédant 400 ms ne sont pas acceptables pour la communication par VoIP. La seule métrique UIT-T qui prend en compte le délai dans la mesure de la qualité de la communication par VoIP est le modèle E et son facteur R, mais les autres – incluant celle que nous avons utilisée – évaluent seulement la qualité de l'écoute (voir la section 3.4.2 pour plus de détails). Par la suite, les recommandations mentionnées ci-dessus doivent être utilisées en complément pour estimer le degré d'interactivité pour toutes les métriques sauf le facteur R.

D'un autre côté, la gigue, c.-à-d. la variation du délai, est un facteur très important qui a une influence directe sur la qualité de la VoIP. La gigue affecte les paquets par le changement de la distribution des instants auxquels les paquets arrivent au récepteur par rapport à la distribution qu'ils ont eu au transmetteur[1]. Pour atténuer les effets de la gigue, toutes les applications VoIP utilisent un tampon de compensation de gigue afin de restaurer la distribution initiale, au coût de l'ajout d'un délai supplémentaire à la restitution du son.

---

[1] Pour VoIP, cette distribution est déterministe, les paquets étant expédiés avec un espacement constant.

Il faut mentionner quelle est la définition de la gigue que nous avons utilisée dans nos calculs. Entre la définition de UIT-T dans [ITU-380] et celle de IETF dans le contexte de métriques de performance IP [RFC-3393], nous préférons la deuxième. La raison est qu'elle est plus en accord avec notre point de vue sur l'influence de la gigue sur les applications. La formule correspondante (2.9) mesure la variation des délais unidirectionnels des paquets consécutifs. Etant donné que les paquets sont expédiés avec un taux constant, la formule (2.9) mesure aussi la variation des moments auxquels les paquets arrivent au récepteur (étant donné que les paquets sont également espacés au transmetteur, la variation de l'espacement est identique à la variation des délais unidirectionnels correspondants). De manière imagée, on peut dire que c'est ce qu'une application « perçoit » comme gigue, car elle n'a pas de connaissance directe de la variation des délais unidirectionnels, mais elle est affectée par la variation de l'espacement entre paquets.

5.2.2.1.1 Questions de gigue

Comme mentionné auparavant, pour limiter les effets de la gigue sur la qualité, toutes les applications VoIP utilisent un délai de restitution. Dans nos tests préliminaires (voir 5.2.3) nous avons également choisi un délai de restitution de 0 ms pour illustrer les effets de la gigue quand il n'y a pas de compensation à l'extrémité réceptrice.

Le processus de compensation de gigue est intégré dans nos tests dans le logiciel de post-traitement « process_voice_data » ; par conséquent on peut utiliser exactement les mêmes données de réseau et changer simplement les paramètres du tampon de compensation pour examiner leur influence sur l'UPQ.

Pour le trafic VoIP, le flux de données a un taux constant de bits, i.e. les paquets sont expédiés avec un taux fixe et un espacement fixe. Appliquer une gigue à un tel flux signifie varier les délais des paquets afin qu'ils arrivent au récepteur avec un espacement variable. Cela a été fait dans nos expériences par l'usage du l'émulateur réseau NIST Net [NIST].

Le plus souvent dans la pratique on rencontre un délai qui croît et décroît successivement pendant une communication VoIP, à cause des variations dans le réseau (e.g. l'occupation des files d'attente change). Ceci signifie que les différences entre les valeurs successives du délai seront à la fois positives et négatives. Dans certains cas, quand le système n'est pas encore arrivé dans un état stationnaire, il est

possible d'observer des valeurs du délai qui grandissent ou diminuent constamment pendant un certain laps de temps (i.e. les différences de délai sont toujours positives ou négatives). Cet événement ne peut cependant pas durer pour des périodes trop longues (comparables à la durée de la communication), donc nous allons nous situer dans les conditions de la première hypothèse.

On peut démontrer (voir [Beu-04a]) que dans ce cas il est possible que des paquets arrivent dans le mauvais ordre si la gigue moyenne dépasse un certain seuil (qui dépend de l'espacement entre paquets et de la distribution de la gigue). Il est donc indispensable de réordonner les paquets, car dans nos tests l'espacement a été de 40 ou 80 ms, tandis que la gigue moyenne est allée jusqu'à 75 ms. Même si, en principe, l'ordre des paquets est respecté dans les réseaux, ils peuvent changer d'ordre aussi à cause de l'utilisation des routes alternatives (quand des mécanismes pour équilibrer les routes sont employés) ou par des équipements de routage ayant de multiples voies internes entre l'entrée et la sortie (e.g. certaines routeurs Juniper ont 4 voies d'un port d'entrée à un port de sortie).

Pour réordonner les paquets, il y a deux alternatives :

1) Traiter les paquets dans le mauvais ordre avant la compensation de gigue. Ceci oblige à forcer les données à arriver en ordre dans le module de compensation en interchangeant le contenu des paquets avant l'émulation de la compensation de gigue et en gardant les temps d'arrivée. Cette solution artificielle peut être utilisée uniquement dans nos types d'expériences, étant donné que nous sauvegardons les données reçues pour le post-traitement. En plus, cette solution convient seulement pour une gigue réduite, sinon la gigue qui en résulte est limitée par l'espacement des paquets au transmetteur ;

2) Traiter les paquets dans le mauvais ordre dans le cadre de la compensation de gigue. Les paquets sont ainsi réordonnés simultanément avec l'opération de compensation, en utilisant les timbres-à-date des paquets quand ceux qui sont à jouer sont sélectionnés. Cette méthode peut être utilisée pour n'importe quelle valeur de la gigue.

A cause des limitations de la première méthode et de son manque de réalisme, nous avons utilisé dans nos expérimentations la seconde solution (voir aussi la section suivante pour plus de détails).

5.2.2.1.2 Mémoire tampon de compensation de gigue

L'approche la plus simple à l'arrivage des paquets VoIP consiste à jouer les paquets disponibles lors de leur temps de restitution. La qualité perçue dans ce cas est très réduite, car les paquets arrivent souvent soit trop tôt, soit trop tard par rapport aux moments auxquels ils doivent être joués (voir la figure 103 pour une quantification de cet effet ; à noter que ne pas faire une compensation est équivalent avec un retard de restitution de 0 ms). Il est nécessaire de mettre en œuvre une compensation de gigue, qui consiste à retarder la restitution du son afin que les paquets soient joués avec le taux fixe avec lequel ils ont été expédiés.

Il y a deux classes d'algorithmes de compensation de gigue : statiques et dynamiques. La compensation statique utilise un retard fixe de restitution, qui peut être configuré au début de la communication, mais reste fixe tout au long. Certains considèrent que la performance peut être améliorée par la variation du retard de restitution. Cependant cette approche fait la supposition que le comportement passé est une indication du comportement futur, ce qui n'est pas toujours le cas.

La compensation dynamique utilise un retard de restitution variable, afin d'optimiser la qualité par une adaptation aux conditions du réseau [Moo-98]. Un certain nombre d'algorithmes on été conçus pour la compensation de gigue dynamique. En principe, le retard de restitution est ajusté de manière adaptative pour être le plus petit délai qui maximise la qualité du son. Quantifier cette qualité est une variable de chaque algorithme, mais d'habitude cela revient à l'évitement d'une perte de paquets excessive à cause de l'arrivée des paquets au récepteur après leur moment de restitution (cas où ils doivent être rejetés). Ces algorithmes observent l'arrivée des paquets récents et utilisent cette information pour prédire l'arrivée des paquets suivants, en utilisant diverses techniques d'estimation.

5.2.2.2 Questions de perte de paquets

Certaines questions sont liées à la perte de paquets. L'une d'entre elles est la façon dont les paquets sont rejetés dans NIST Net. Actuellement, les événements de perte ne sont pas corrélés et l'influence d'une corrélation possible n'a pas été étudiée. En pratique la perte est un résultat de la congestion et en général c'est toute une séquence de paquets qui est perdue. Mais l'évaluation de telles conditions est difficile et n'a pas fait partie de cette recherche.

Une autre question est la dissimulation de la perte de paquets (Paquet Loss Concealment, PLC). Ce terme désigne les actions prises lorsque un ou plusieurs paquets sont perdus et n'arrivent pas au récepteur. La méthode la plus simple dans ce cas est d'introduire une période de silence à la place du paquet perdu, avec une durée équivalente. Une meilleure solution consiste à créer artificiellement des échantillons audio en se basant sur les échantillons réels de voix qui précédent (et potentiellement suivent) le ou les paquets perdus et qui sont déjà présents dans la mémoire tampon de compensation de gigue. En plus des méthodes sans redondance, il y a des techniques qui ajoutent une information supplémentaire aux paquets, ce qui permet de reconstruire les paquets qui manquent éventuellement.

Il est évident que l'utilisation d'un signal plus proche de celui qui a été perdu au lieu d'une période de silence va améliorer la qualité, mais ce principe va à l'encontre de la décision d'étudier un comportement de base. Il existe une grande variété d'algorithmes de type PLC, la plupart basés sur des techniques de prédiction linéaires, comme [Güz-01]. Une analyse des différentes approches se trouve dans [Wah-01]. Approfondir ces solutions demandait le choix et l'implémentation d'un algorithme de ce type, ce qui est en dehors de notre recherche. On peut citer le fait que, en accord avec [Güz-01], l'amélioration obtenue par divers algorithmes est située entre 0,5 et 1,5 sur une échelle de type MOS, en fonction du taux de perte et de la taille des paquets (évidemment perdre un paquet plus petit est plus facile et mieux masqué, car il contient moins d'information audio). L'amélioration obtenue pour des paquets de 40 ms, comme ceux utilisés dans nos tests, ne dépasse pas 0,6 sur l'échelle MOS.

5.2.2.3 Algorithme de restitution de son et compensation de gigue

Notre travail est, en résumé, une étude de base permettant, d'une manière déterministe, de comprendre le comportement du système dans le pire cas. Des situations plus complexes peuvent être analysées ensuite et des améliorations potentielles évaluées. En conséquence, la stratégie de compensation de gigue que nous avons utilisée est la plus simple possible. Le moment de restitution sonore de chaque paquet qui arrive est retardé d'un certain intervalle constant de temps, le délai de restitution, (*PLAYBACK_DELAY*). Le paquet qui va être joué est sélectionné parmi les paquets de la mémoire tampon en fonction de leur timbre-à-date. Si le paquet correspondant au moment de restitution courant est absent, alors une période de

silence ayant la durée d'un paquet est insérée. A noter que le mauvais ordre possible des paquets est pris en compte par l'utilisation des timbres-à-date.

Tous les paquets qui arrivent sont placés dans la mémoire tampon, indépendamment de leur temps d'arrivée ou restitution, mais les paquets dont le moment de restitution est passé sont rejetés par la suite. Cet effet est appelé « perte de compensation de gigue » et c'est la façon dont la gigue affecte la qualité du parler. L'algorithme de restitution de son et compensation de gigue est décrit ci-dessous en pseudo-code:

```
1) Initialiser le premier moment de restitution (en fonction
   du paramètre PLAYBACK_DELAY)
2) Répéter jusqu'au vidage définitif de la mémoire tampon
   a) Sélectionner le paquet dans le tampon de compensation
      dont le timbre-à-date est égal au moment de
      restitution courant
      i) Si ce paquet existe, le jouer
     ii) Si le paquet n'est pas trouvé, insérer une durée de
         silence équivalente
   b) Calculer le moment de restitution suivant.
```

Figure 101 : L'algorithme de restitution de son et compensation de gigue.

Le paramètre d'entrée de cet algorithme est le *PLAYBACK_DELAY*, le retard de restitution de chaque paquet pour que l'ensemble des paquets soit joué de manière « lisse ». La compensation est statique, donc la valeur de *PLAYBACK_DELAY* reste inchangée pour toute la durée de chaque expérience. La valeur utilisée a été choisie en se basant sur une série de tests préliminaires.

Il y a plusieurs façons de faire évoluer cet algorithme simple. D'abord, on peut utiliser un algorithme de compensation dynamique, comme discuté auparavant.

Il existe aussi une solution alternative au rejet de paquets qui sont arrivés trop tard. Cette solution implique l'accélération de la restitution pour que les données audio jouées prennent moins de temps et par conséquent qu'il n'y ait pas un déplacement global en temps lorsque la séquence audio sera jouée intégralement. Il y a aussi la

possibilité de jouer les paquets à un taux réduit avant un paquet manquant, en minimisant ainsi la durée des intervalles de silence.

L'évaluation des améliorations obtenues de cette manière peut être faite au moyen du score PESQ, mais elle n'était pas dans le cadre de notre recherche. Nos tests fournissent une étude de base de la performance de VoIP et ces améliorations ne constituent de notre point de vue que des ajouts. Les résultats obtenus par notre approche de base représentent le comportement de référence.

5.2.2.4 Speak Freely

L'application VoIP que nous avons utilisé pour nos tests est Speak Freely, la version 7.6a [Wiles], un outil gratuit sous Linux. L'application envoie de la voix par le réseau en utilisant un certain codage, et assure le décodage et la restitution du son à l'autre bout (voir aussi 3.4 pour les principes de VoIP).

Les données vocales sont codées par des codeurs qui compriment les données audio en provenance d'une source d'entrée (e.g. microphone ou fichier d'entrée). Au décodage, les données sont décompressées pour pouvoir être jouées sur la carte son. Les codeurs sont nécessaires car les données audio doivent être empaquetées puis transformées en une représentation au niveau du réseau. Ceci implique l'échantillonnage à un certain taux (d'habitude 8 kHz) et une certaine dimension d'échantillon (d'habitude 1 octet). Un codage basique est élémentaire, tel que *A-law* ou *μ-law*, ou des formes plus avancées de compression sont utilisées. Chaque codeur a ses caractéristiques spécifiques par rapport au niveau de compression et à la qualité perçue qu'il permet obtenir, qui sont en général un compromis entre l'efficacité d'utilisation du réseau et la qualité sonore.

5.2.2.4.1 Caractéristiques

Speak Freely implémente une série de codeurs qui sont disponibles pendant l'utilisation du protocole interne : G.711, G.726, GSM, LPC, LPC-10, CELP. Mais certains manquent pendant l'utilisation des protocoles RTP ou VAT, parce que ceux-ci on des standards concernant le codage des données empaquetées. L'information liée au RTP est donnée dans [RFC-3551] ; ce RFC définit les identificateurs pour une série de codeurs. Pour garder la compatibilité avec d'autres applications utilisant le même protocole, les modes de compression non spécifiés dans ces standards ne sont

pas disponibles avec RTP ou VAT. C'est le cas pour les codeurs LPC-10 et CELP, pour lesquels l'information n'a pas pu être fournie dans les tableaux 22 à 24, sauf pour l'utilisation du protocole interne de Speak Freely.

L'application permet aussi d'utiliser une méthode supplémentaire très simple de compression, qui réalise un sous-échantillonnage par deux fois. Elle peut s'appliquer à tout codeur et est notée par 2X (e.g. GSM+2X si on l'applique au codeur GSM).

Même s'il n'est pas disponible dans Speak Freely, G.729 [ITU-729] est un codeur intéressant, car il a un taux de compression élevé (8 kbps de données vocales) avec un score PESQ relativement élevé, d'environ 3,5 dans nos tests. Comme ce codeur n'a pas été inclus dans Speak Freely, nous avons obtenu l'implémentation de référence de UIT-T pour la version G.729AB, qui l'est une des plus utilisées. Nous avons intégré ce codeur dans la phase de post-traitement de nos expériences. Les tests effectués avec ce codeur utilisent à l'entrée de l'application VoIP un fichier déjà compressé, et la taille des paquets est ajustée en accord avec le niveau de compression de G.729. Seul le protocole RTP a été utilisé ; par suite les données concernant G.729 ne sont pas fournies dans les tableaux 22 à 24 pour les autres protocoles.

Dans les premières expériences, nous avons essayé d'utiliser directement la sortie de la carte son vers les haut-parleurs pour évaluer le UPQ de VoIP par l'enregistrement du signal dans un fichier de son. La qualité assez basse de l'enregistrement (déterminée par la qualité basse des voies du signal audio, qui incluent des conversions numérique à analogique et analogique à numérique) a rendu impossible une évaluation pertinente. La solution trouvée a été l'émulation logicielle des actions prises après réception d'un paquet par l'application VoIP, dans notre cas la compensation de gigue. Par conséquent Speak Freely a été modifié pour écrire un fichier de données qui nous permet la création d'un fichier son équivalent au signal vocal que l'application VoIP aurait produit aux haut-parleurs du PC. Cette fonctionnalité additionnelle nous permet de comparer différents paramètres pour la compensation de gigue au récepteur en utilisant exactement le même trafic réseau (voir 5.2.3).

Les changements effectués sont les suivants. Pour chaque paquet VoIP reçu, la version modifiée de Speak Freely écrit dans un fichier de sortie les données audio décompressées, conjointement avec son timbre-à-date (le moment auquel le paquet

devrait être joué) et son temps d'arrivée au PC récepteur[1]. Le fichier obtenu ainsi est l'entrée d'un logiciel que nous avons écrit, « process_voice_data », qui émule la restitution de la voix et la compensation de gigue comme décrit dans 5.2.2.

5.2.2.4.2 Evaluation

Les caractéristiques de Speak Freely sont résumées dans les tableaux suivants. Elles ont été déterminées en s'appuyant sur la documentation de l'application, le code source, ainsi que plusieurs tests. Nous avons noté SFP le protocole spécifique de communication de Speak Freely. Tous les taux sont donnés pour une connexion simplex. Ils doivent être doublés pour une opération normale full-duplex, quand les nœuds communiquent de manière bidirectionnelle. A noter que les codeurs utilisés plus tard dans nos expériences à pleine échelle sont présentés dans la première partie des tableaux.

Le tableau 22 montre le taux de transmission (taux de données VoIP, taux au niveau du réseau et taux de paquets), ainsi que la taille des paquets, pour les différents codeurs en utilisant RTP comme protocole de transport.

Codeur	*Taux de données [kbps]*	*Taille de paquets [octets]*	*Taux au niveau du réseau [kbps]*	*Taux de paquets [pps]*
G.711	64	378	75.6	25
G.726	32	382	38.2	12.5
GSM	13	190	19	12.5
G.729	8	170	17	12.5
LPC	5.3	116	10.6	11.4
LPC-10	*N.D.*	*N.D.*	*N.D.*	*N.D.*
CELP	*N.D.*	*N.D.*	*N.D.*	*N.D.*

Tableau 22 : Les taux de transmission et la taille des paquets pour les codeurs disponibles avec RTP (surcharge de l'en-tête = 58 octets).

[1] Le timbre-à-date est obtenu par l'en-tête RTP. Le temps d'arrivée est déterminé sur le PC récepteur en utilisant la fonction « gettimeofday », une fonction standard de bibliothèque C, qui a une précision de l'ordre de quelques millisecondes. Cette précision est suffisante étant donné que les paquets VoIP contiennent dans nos tests au moins 40 ms de données.

Comme on peut l'observer sur le tableau 22, chaque codeur a des caractéristiques différentes concernant les taux de données et de paquets qu'il produit. Ils sont corrélés avec le niveau de compression de chaque codeur (voir 5.4 pour plus de détails sur les codeurs étudiés).

Les tableaux 23 et 24 ci-dessous comparent les protocoles disponibles et tous les codeurs qui peuvent être utilisés avec ces protocoles.

Codeur	***SFP données [octets]***	***RTP données [octets]***	***VAT données [octets]***	***SFP audio [échantillons]***	***RTP audio [échantillons]***	***VAT audio [échantillons]***
G.711	488	320	320	488	320	320
G.726	489	324	324	972	640	640
GSM	332	132	132	1600	640	640
G.729	*N.D.*	112	*N.D.*	*N.D.*	640	*N.D.*
LPC	142	58	58	1600	640	640
LPC-10	70	*N.D.*	*N.D.*	1800	*N.D.*	*N.D.*
CELP	146	*N.D.*	*N.D.*	1920	*N.D.*	*N.D.*

Tableau 23 : Taille brute des données, par paquet, pour chaque codeur et chacun des protocoles disponibles.

A noter par exemple sur le tableau 23 que, même si la quantité de données envoyée en réseau dans un paquet est presque la même (e.g. pour G.711 et G.726), la quantité de données audio en échantillons varie à cause des différents niveaux de compression. Ce fait est aussi mis en évidence sur le tableau 24, où la quantité de données audio est fournie en millisecondes.

La compression utilisée par les codeurs de voix est avec perte. Par conséquent les données audio reconstruites après décompression ne coïncident pas avec les données pourvues à l'entrée du codeur. Par suite, une dégradation de la qualité perçue apparaît, qui croît d'habitude avec le niveau de compression. Ce fait est illustré par la dernière colonne du tableau 24, qui montre (comme mesure de l'UPQ pour VoIP) les scores PESQ maximaux, obtenus dans nos tests pour les codeurs correspondants. A noter qu'à cette dégradation intrinsèque s'ajoute la dégradation causée par les variations des conditions de QoS dans le réseau, dégradation que nous avons étudié par la suite.

Codeur	*SFP audio [ms]*	*RTP audio [ms]*	*VAT audio [ms]*	*Score PESQ maximum*
G.711	61	40	40	4,3
G.726	122	80	80	3,8
GSM	200	80	80	3,4
G.729	*N.D.*	80	*N.D.*	3,5
LPC	200	80	80	1,9
LPC-10	225	*N.D.*	*N.D.*	2,5
CELP	240	*N.D.*	*N.D.*	3,2

Tableau 24 : Taille brute des données audio (en millisecondes), par paquet, et score PESQ maximum correspondant à chaque codeur pour chacun des protocoles disponibles.

Le tableau 25 montre la variation obtenue pour le codeur GSM si on applique le sous-échantillonnage par deux fois (le protocole SFP est utilisé). Néanmoins nous n'allons pas utiliser ce type non standard de compression dans nos tests.

Codeur	*Taux de données [kbps]*	*Données par paquet [octets]*	*Audio par paquet [ms]*	*Score PESQ maximal*
GSM	13	332	200	3,4
GSM + 2X	6,6	336	400	2,7

Tableau 25 : Comparaison entre GSM et GSM+2X (en utilisant le protocole SFP).

RTP étant plus répandu pour la communication VoIP, nous l'avons sélectionné comme protocole de transfert dans nos tests. En fonction de leur disponibilité et des scores PESQ maximaux correspondants (proches de la qualité PSTN), les codeurs suivants ont été choisis pour une étude approfondie : G.711, G.726 et GSM. En plus, G.729 a été intégré et utilisé comme mentionné auparavant. La section 5.4 présente les résultats obtenus, avec une partie séparée pour chaque codeur étudié, suivis par une comparaison des codeurs. La comparaison peut être utilisée pour sélectionner le codeur offrant la meilleure qualité dans certaines conditions réseau.

5.2.2.4.3 Détection du silence

La détection du silence est une caractéristique présente dans certaines applications VoIP. A la base, elle fonctionne par sélection d'un seuil d'activité en dessous duquel aucun son n'est transmis. Cette fonction est désignée par le terme anglais « Voice Activity Detection » (VAD). G.729AB intègre cette fonction dans le codeur même. Par une série de tests nous avons pu déterminer que la baisse du score PESQ lors de l'utilisation de VAD et relativement petite, inférieure à 0,1 dans tous les cas. L'effet beaucoup plus notable est celui de la décroissance, de l'ordre de 40%, de l'intensité du trafic réseau (voir le tableau 26).

VAD	*Score PESQ*	*Taille comprimée [octets]*	*Taux de compression*
Désactivée	3,48	33600	5,71
Activée	3,42	20224	9,49

Tableau 26 : Les scores PESQ, la taille de données codées par G.729 et le taux de compression avec ou sans l'option VAD (pour l'enregistremment "female_all" de 192000 octets – voir aussi 5.2.3).

En général la détection du silence est effectuée par l'application VoIP ; les autres codeurs n'ont pas un tel mécanisme ; en conséquence cette forme de VAD dépend de chaque application, car chaque application utilise son propre algorithme de détection. En plus cette option n'est pas d'habitude activée par défaut dans les applications VoIP, probablement car les utilisateurs sont sceptiques quant à son effet sur la qualité. Pour toutes ces raisons nous avons décidé de ne pas inclure VAD dans nos tests sur aucun des codeurs, pour fournir une étude de base sur leur comportement.

5.2.3 Tests préliminaires pour VoIP

Nous avons effectué quelques types de tests préliminaires afin de déterminer les conditions de base pour mener nos expériences. Pour obtenir les résultats présentés ici nous avons utilisé le protocole RTP, comme dans les expériences.

Le codeur utilisé dans tous les tests préliminaires a été G.711. Aucune perte de paquets n'a eu lieu pendant les expériences. Le délai moyen artificiel introduit a été de 300 ms et la gigue moyenne a varié en fonction du type de test effectué.

5.2.3.1 Calibrage du délai de restitution

Plusieurs tests ont été effectués pour calibrer le délai de restitution. Ces expériences ont porté sur un trafic de gigue moyenne de 20 ms, considérée comme représentative des conditions normales dans un réseau sous stress modéré. Par conséquent la gigue envisagée a été configurée à 20 ms dans NIST Net. La gigue mesurée au niveau du réseau, calculée à base de l'information fournie par le système de monitoring, s'est située aux alentours de 20 ms aussi. Le délai de restitution (« playback » ou « playout » en anglais), *PLAYBACK_DELAY*, a varié de 0 à 200 ms par incréments de 20 ms. Une valeur nulle est équivalente à une absence de compensation de gigue.

Les expériences ont été effectuées avec des enregistrements de voix d'homme et de femme à la fois (obtenus avec le standard UIT-T P.862 [ITU-862]), comme suit :

- « female1 », « female2 » et « female3 » : trois enregistrements différents avec la même voix de femme et une durée de 8 s chacun ;
- « male1 », « male2 » et « male3 » : trois enregistrements différents avec la même voix d'homme et une durée de 8 s chacun ;
- « female_all » / « male_all » : les enregistrements de 24 s obtenus par la concaténation des trois enregistrements avec une voix de femme / homme mentionnés auparavant.

Comme exemple nous fournissons ici la transcription du fichier « female_all » :

> *You are the perfect hostess. Are you going to be nice to me? You know my outlook on life. I jumped at least two feet. He took out his pipe and lit up. It was the same in the public bar.*

Figure 102 : La transcription de l'enregistrement « female_all ».

Un total de 20 tests a été effectué pour chaque fichier d'entrée et des moyennes ont été associées à chaque enregistrement. Les résultats marqués « female123 » ou « male123 » sont les moyennes des résultats moyens obtenus pour « female1 », « female2 » et « female3 », ou « male1 », « male2 » et « male3 » respectivement.

Les figures 103 et 104 montrent les résultats obtenus pour les enregistrements avec une voix de femme – les scores PESQ moyens et leur déviation standard. On note que les graphiques sont similaires pour les différents enregistrements.

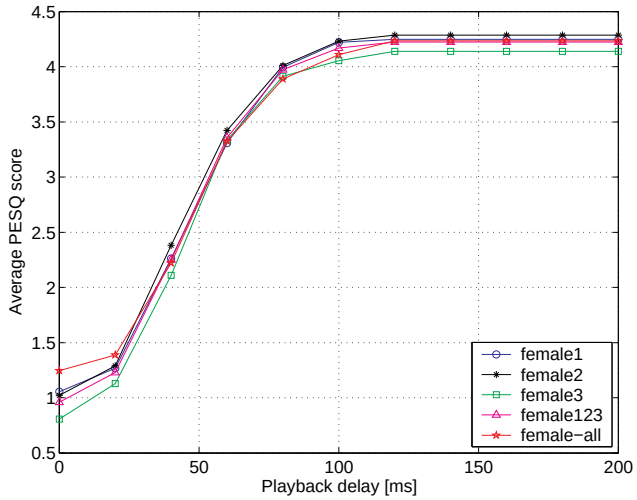


Figure 103 : Scores PESQ moyens pour des enregistrements de voix de femme (gigue moyenne = 20 ms).

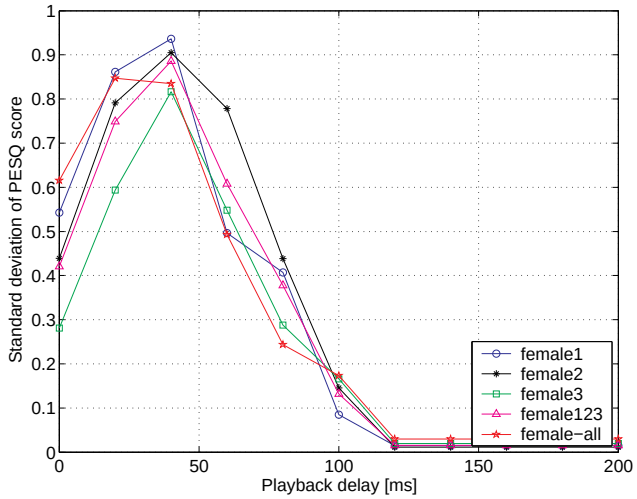


Figure 104 : Ecart-type des scores PESQ moyens pour des enregistrements de voix de femme (gigue moyenne = 20 ms).

Les figures 105 et 106 montrent les scores PESQ et leur écart-type pour les enregistrements de voix d'homme. De nouveau les graphiques sont similaires pour les différents fichiers.

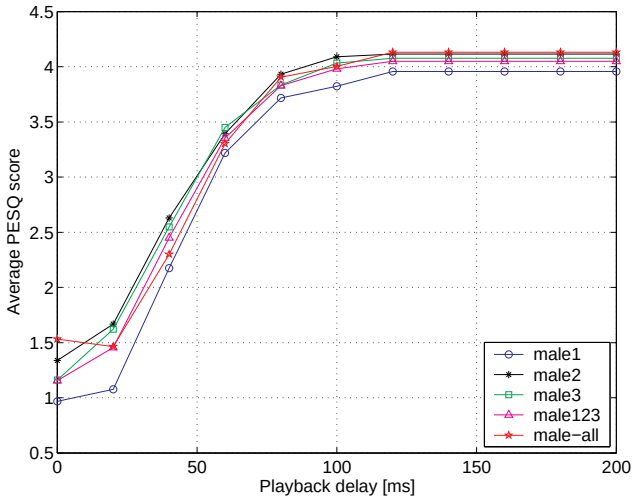


Figure 105 : Scores PESQ moyens pour des enregistrements de voix d'homme (gigue moyenne = 20 ms).

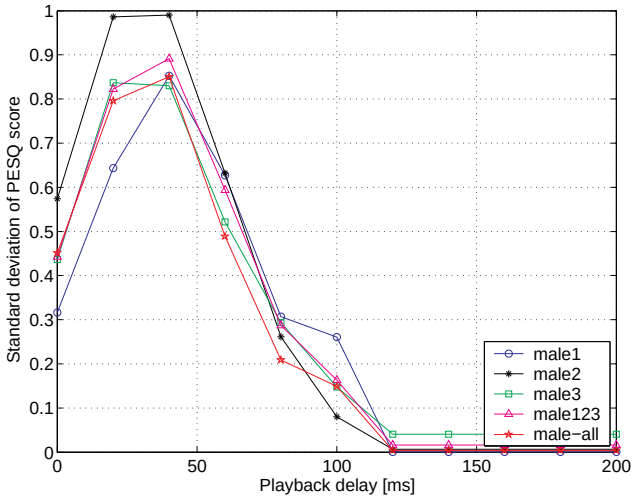


Figure 106 : Ecart-type des scores PESQ moyens pour des enregistrements de voix d'homme (gigue moyenne = 20 ms).

Les graphiques présentés ci-dessus montrent clairement que les différences entre les résultats obtenus avec les différents enregistrements en fonction du locuteur sont mineures. Dans nos tests, les scores PESQ obtenus pour les enregistrements de voix de femme sont supérieurs d'environ 0,2 à ceux obtenus avec les enregistrements de voix d'homme.

La comparaison des résultats obtenus pour les enregistrements de 24 s avec ceux obtenus pour les trois enregistrements originaux de 8 s permet d'affirmer qu'il n'y a pas de différence significative persistante entre les scores PESQ moyens.

Ensuite nous avons décidé de poursuivre nos tests uniquement sur le fichier « female_all » à cause du nombre plus grand de paquets impliqués (ce qui génère des statistiques plus fiables) et aussi à cause du score supérieur à celui de l'enregistrement « male_all ».

En ce qui concerne le tampon de compensation de gigue, nous notons que pour un délai de restitution de 80 ms, qui est équivalent à deux paquets VoIP quand on utilise G.711, le score PESQ obtenu pour « female_all » est assez près du maximum (4,0 au lieu de 4,25 – cf. la figure 103). Le score obtenu n'est pas le maximum car, bien que la gigue moyenne soit seulement de 20 ms, certaines valeurs instantanées de la gigue dépassent 80 ms (cf. la figure 99), ce qui empêche ce délai de restitution de compenser entièrement les effets de la gigue.

5.2.3.1.1 La relation entre le délai de restitution et la gigue

Pour analyser avec davantage de détails les effets du délai de restitution sur la qualité, nous avons aussi effectué des tests supplémentaires avec des valeurs différentes de la gigue. L'enregistrement utilisé a été « female_all ». *PLAYBACK_DELAY* a reçu des valeurs entre 0 et 200 ms avec incréments de 40 ms[1]. Une valeur nulle signifie l'absence de compensation de gigue. La gigue moyenne a varié entre 0 et 40 ms avec un pas de 10 ms.

Le graphique sur la figure 107 montre que la qualité perçue croît avec le délai de restitution, car le tampon de compensation de gigue compense les effets négatifs de la gigue. En même temps, en comparant les graphiques obtenus pour des valeurs différentes de gigue on en déduit que la valeur du délai de restitution qui assure une qualité optimale dépend du niveau de gigue (et, en fait, de la partie de la « queue » de son histogramme qui dépasse la valeur du délai de restitution).

[1] Le délai maximal uni-directionnel qui assure une bonne interactivité étant de 150 ms [Reijs], [ITU-1541], il n'a pas été nécessaire d'étudier des délais plus importants.

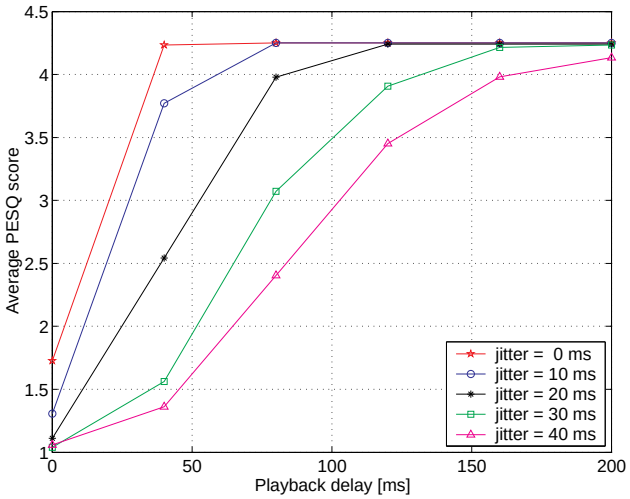


Figure 107 : Le score PESQ en fonction du delai de restitution pour différentes valeurs de la gigue moyenne.

A noter qu'une gigue envisagée de 0 ms conduit en effet à une très petite gigue mesurée, inférieure à 0,02 ms dans tous les cas. Par contre, cette gigue, même très petite, affecte fortement la qualité si le délai de restitution est zéro. Comme on peut le voir sur la figure 107 (la ligne correspondant à une gigue de 0 ms et un délai de restitution de 0 ms) le score PESQ est seulement d'environ 1,7. La cause est que si un paquet n'est pas déjà arrivé au moment de la restitution, il est remplacé par une silence et rejeté finalement s'il finit par arriver : la qualité décroît donc significativement.

Les mêmes données, représentées d'une autre manière (la figure 108), mettent en évidence la relation entre les scores PESQ et la gigue. On peut voir que pour un délai de restitution plus grand la qualité commence se dégrader à une valeur de la gigue supérieure.

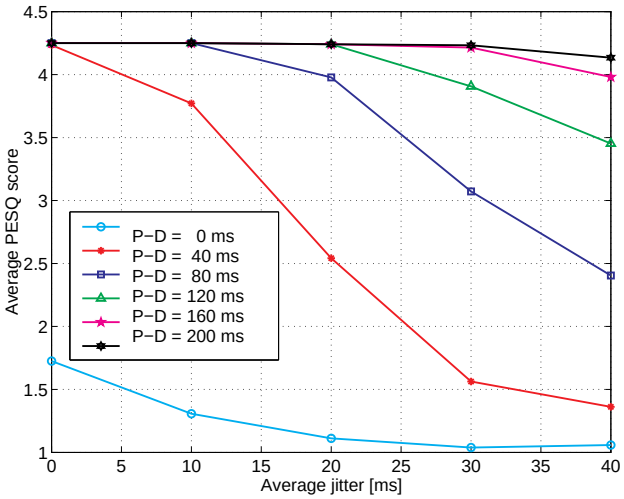


Figure 108 : Le score PESQ en fonction de la gigue moyenne pour différentes valeurs du délai de restitution (noté par P_D).

5.2.3.1.2 Valeur du délai de restitution

Vu les tests effectués nous avons décidé d'employer dans tous les tests présentés en section 5.4 un délai de restitution (*PLAYBACK_DELAY*) de 80 ms. La contribution d'un délai de 80 ms au délai total de bout en bout lui permet d'être toujours en dessous de seuil d'interactivité de 150 ms, si le délai dans le réseau ne dépasse pas 70 ms.

5.2.3.2 Perte de compensation de gigue

La perte de compensation de gigue est la perte entraînée par le mécanisme de compensation de gigue lorsqu'un paquet arrive trop tard par rapport à son temps de restitution et doit être rejeté. Nous avons quantifié la perte de compensation de gigue afin de fournir une meilleure compréhension de la manière dont la gigue influe sur la qualité. Le graphique de la figure 109 montre la dépendance entre la perte de compensation de gigue (donc la perte induite dans le cadre du mécanisme de compensation de gigue) pour tous les codeurs étudiés. Nous avons utilisé les mêmes conditions que pour tous les tests, i.e. *PLAYBACK_DELAY* a été 80 ms et le délai unidirectionnel moyen 300 ms. La perte de paquets au niveau du réseau a été nulle.

L'influence exacte du codeur sur la dépendance entre la perte de compensation de gigue et la gigue moyenne n'est pas connue. L'examiner avec davantage de précision n'a pas été considéré dans notre recherche. A part les paramètres du mécanisme de compensation, nous supposons qu'il peut exister une influence du nombre des paquets

et la quantité de voix qu'ils représentent. Pour G.711 il y a un nombre double de paquets par rapport aux autres codeurs, car chaque paquet représente 40 ms, tandis que pour G.726, GSM et G.729 les paquets contiennent 80 ms de voix. Dans le premier cas, les paquets sont moins espacés et ceci pourrait expliquer pourquoi la même gigue induit une perte supérieure pour G.711.

Par contre nous ne pouvons pas encore expliquer pourquoi G.729 semble mieux se comporter que les autres codeurs ayant les mêmes caractéristiques en termes de nombre de paquets et quantité de données. Une possibilité est que, à cause de la façon dont nous avons mené les tests, pour tous les codeurs sauf G.729, le décodage a eu lieu en temps réel et qu'il est pris en compte par le temps d'arrivée des paquets au récepteur. Pour G.729 les données vocales sont écrites comprimées dans le fichier intermédiaire et sont seulement décompressées lors de la compensation de gigue.

Ceci soulève un nouveau problème. En effet nous avons toujours supposé que la décompression prend un temps fixe et réduit, donc n'introduit pas une gigue supplémentaire ; or il se peut que ce ne soit pas le cas en pratique. Une évaluation plus précise de ce fait n'est pas triviale et n'a pas été effectuée.

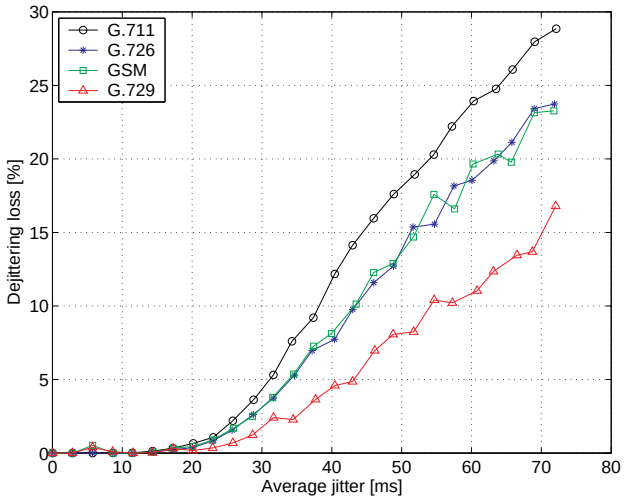


Figure 109 : Perte de compensation de gigue en fonction de la gigue moyenne (délai de restitution = 80 ms).

5.2.3.3 Evaluation des codeurs

L'utilisation de codeurs pour envoyer de la voix sur un canal de communication induit, par le mécanisme de compression des données, il y a une baisse de la qualité perçue ; par conséquent pour chaque codeur il y a un score PESQ maximal qui peut

être obtenu. Il est évident que sa performance quand des dégradations apparaissent dans le réseau doit seulement être estimée par rapport à ce score maximum.

Même si le MOS moyen de chaque codeur est connu (voir le tableau 5), nous avons dû vérifier ces résultats pour notre système et nos enregistrements. Les scores PESQ moyens obtenus dans notre configuration sont présentés sur le tableau 24 pour les codeurs disponibles dans Speak Freely. Tous les codeurs utilisés dans nos tests (G.711, G.726, GSM et G.729) ont une qualité intrinsèque au dessus du niveau d'acceptabilité, et deux d'entre eux (G.711 et G.726) ont même une qualité supérieure au niveau PSTN.

Le tableau 27 montre les variations possibles dans l'utilisation d'un même codeur (dans ce cas G.711) entre des enregistrements différents de voix d'homme et de femme.

Enregistrement	***PESQ score***	***Enregistrement***	***PESQ score***
female1	4,25	male1	3,96
female2	4,29	male2	4,12
female3	4,15	male3	4,11
female123	4,23	male123	4,06
female_all	4,25	male_all	4,14

Tableau 27 : Les scores PESQ maximales obtenu pour divers enregistrements dans notre configuration expérimentale (en utilisant le codeur G.711).

Ces tests montrent qu'il y a de petites différences entre les résultats obtenus pour les divers enregistrements. Nous avons déjà remarqué que les scores PESQ obtenus pour les enregistrements de voix de femme sont supérieurs par environ 0,2 aux ceux obtenus pour voix d'homme. Par contre, pour le même locuteur, la variation des scores entre les enregistrements est remarquablement petite, en dessous de 0,05 par rapport à la moyenne.

5.2.4 Déroulement de tests pour VoIP

Cette section présente les configurations utilisées pour effectuer nos expériences à pleine échelle. Nous avons configuré NIST Net pour introduire une dégradation artificielle de la QoS. A la lumière de tests préliminaires, nous avons décidé de nous

concentrer sur des taux de perte de 0 à 15% avec un pas de 0,5%. La gigue moyenne envisagée a varié de 0 à 75 ms avec un pas de 3 ms.

Pour nos expériences nous avons aussi configuré un délai unidirectionnel moyen de 300 ms. Il permet une variation autour de la moyenne allant jusqu'à 4 fois la valeur moyenne maximale de la gigue (i.e. 75 ms), en accord avec le fonctionnement de NIST Net – voir l'équation (5.1) –, sans générer de valeurs négatives du délai. A noter que pour de larges valeurs du délai, les paquets peuvent être mis dans le mauvais ordre par NIST Net. Cet effet est compensé dans le cadre de l'algorithme de compensation de gigue décrit dans 5.2.2, qui utilise un délai de restitution du son de 80 ms.

Un test consiste en 806 expériences, une pour chaque combinaison de taux de perte et gigue mentionné plus haut. Pour chaque expérience nous avons calculé les valeurs obtenues pour le taux de perte et la gigue moyenne, déterminées par la formule (2.9). Ce calcul est fait sur la base des descripteurs de trafic produits par le système de monitoring. Nous calculons aussi le score PESQ par comparaison du fichier de son de sortie avec un fichier de son de référence. Le fichier de référence est un enregistrement de voix de femme, « female_all », sélectionné sur la base de tests préliminaires présentés dans la section précédente. L'enregistrement a une durée de 24 s et a été obtenu par la concaténation des trois enregistrements de test fournis avec la recommandation UIT-T P.862 [ITU-862].

Un total de 5 tests a été mené pour chaque codeur afin d'obtenir les résultats présentés dans 5.4. Le signal d'entrée de l'application VoIP est obtenu par le codage PCM du fichier de son de référence à 64 kbps (fréquence d'échantillonnage de 8 kHz, 8 bits par échantillon, variante de codage *μ-law*). Le signal d'entrée est stocké dans un fichier de 192000 octets. Envoyer les données avec le codeur G.711 ne demande aucune compression additionnelle ; pour tous les autres codeurs, ce signal représente l'entrée de l'algorithme de compression correspondant.

5.2.4.1 Représentation des résultats

Pour chaque codeur nous avons obtenu 5 ensembles de points (un pour chaque déroulement de test), points distribués dans l'espace défini par le taux de perte et la gigue. Etant donné que les valeurs réelles obtenues dans chaque expérience pour les paramètres QoS varient légèrement par rapport à celles envisagées configurées dans

NIST Net, les points ne se situent pas exactement sur une maille régulière ; des étapes additionnelles sont nécessaires pour représenter les données. Il y a deux questions liées à la représentation des résultats de nos tests qui seront discutées dans cette section : la représentation des surfaces et la modélisation des surfaces.

A noter que nous allons toujours représenter sur l'axe de la gigue la valeur moyenne de la gigue calculée dans l'expérience correspondante, même si le tampon de compensation va compenser une partie de la gigue (en dessous de 20 ms, comme on peut voir de nos tests préliminaires). Pour étudier les effets de la gigue on peut aussi utiliser la métrique de « perte de compensation de gigue », telle qu'elle a été définie dans 5.2.3.2.

5.2.4.1.1 Représentation de surfaces

Les données brutes obtenues dans nos expériences sont des points dans l'espace de perte de paquets, gigue et score PESQ. Ces points peuvent être représentés en tant que surface dans cet espace tridimensionnel, avec la perte et la gigue comme axes des abscisses et ordonnées, et le score PESQ sur l'axe *z*. Afin de construire la surface qui contient tous les points de test, une opération nommée triangulation doit être effectuée, par exemple par la méthode de Delaunay.

La figure 110 montre les résultats de 5 tests réalisés dans la configuration de test pour l'enregistrement « female_all » utilisant le codeur G.711, un délai moyen de 300 ms et un délai de restitution de 80 ms. Les mêmes données sont utilisées pour les graphiques dans 5.4, la section dédiée au codeur G.711.

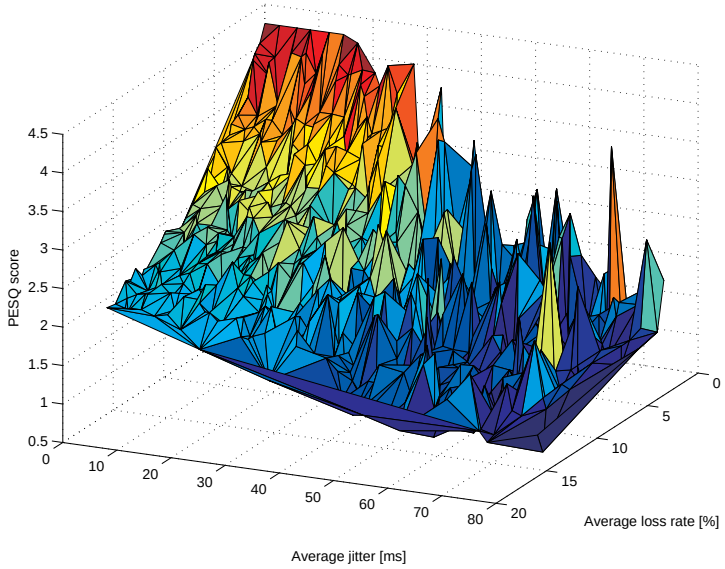


Figure 110 : Représentation par triangulation Delaunay des résultats bruts.

La représentation dont nous venons discuter est la plus élémentaire et constitue une visualisation des données brutes. Nous observons qu'il y a du « bruit » dans les résultats PESQ, la surface n'est pas lisse comme attendu, car on suppose que la qualité varie de façon continue. La raison probable est que la perte d'un paquet, selon qu'il contient de la voix ou du silence, a une influence différente sur la qualité perçue. Nous concluons sur la base de la figure 110 qu'un algorithme de lissage doit être utilisé avant la représentation finale. Plusieurs alternatives sont présentées par la suite. A noter qu'ils produisent le lissage sur une maille rectangulaire ; la triangulation n'est alors plus nécessaire.

La première méthode essayée est l'interpolation sur un maillage rectangulaire avec une distribution uniforme dans les gammes étudiées des paramètres. Le graphique est présenté sur la figure 111. L'interpolation n'a pas produit un résultat satisfaisant à cause du « bruit » initial des données à interpoler.

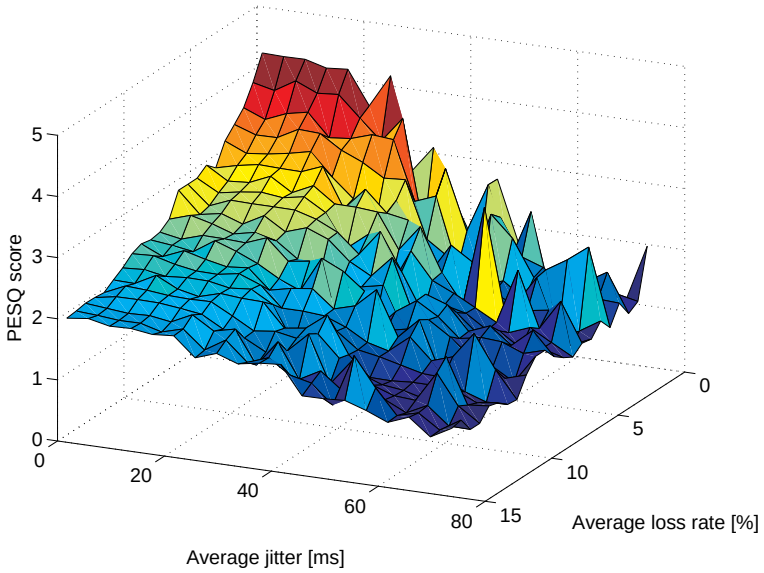


Figure 111 : Représentation par interpolation basée sur les résultats bruts.

La seconde méthode utilisée est la suivante : nous faisons glisser une fenêtre avec recouvrement (« overlapping » en anglais) de taille fixe dans l'espace perte-gigue et calculons la moyenne des scores PESQ pour tous les points contenus dans cette fenêtre. Le résultat est associé au centre de la fenêtre et utilisé pour la représentation de résultats montrée sur la figure 112. Les valeurs optimales pour la taille de la fenêtre ont été déterminées en comparant les graphiques obtenus avec différentes tailles afin qu'une surface relativement lisse soit produite et qu'un niveau suffisamment haut de détail soit maintenu. Ces valeurs sont de 4,5 ms sur l'axe de la gigue et 0,75% sur l'axe de la perte de paquets. Le glissement se fait avec un pas d'une moitié de la taille de la fenêtre, c.-à-d. 2,25 ms sur l'axe de la gigue et 0,375% sur celle de la perte.

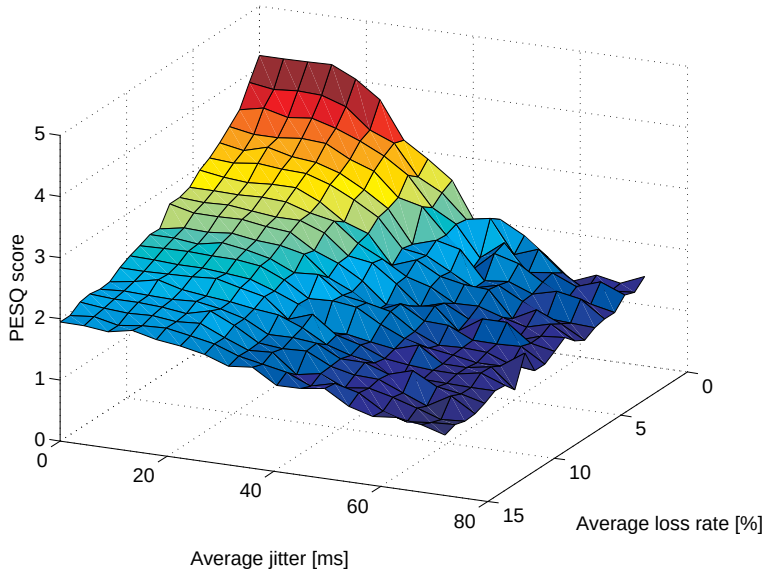


Figure 112 : Représentation par lissage avec une fenêtre glissante basée sur les résultats bruts.

Une autre façon de lisser la surface est de calculer le score PESQ moyen pour les 5 points obtenus dans chaque test pour tous les couples taux de perte-gigue. Cette moyenne est ensuite reportée au point de coordonnées de perte et gigue envisagés, et utilisée pour la représentation de la surface 3D (la figure 113).

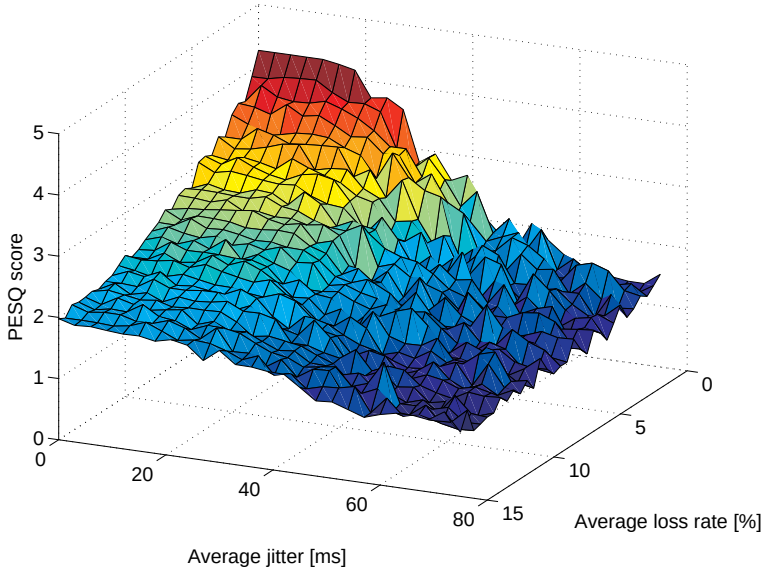


Figure 113 : Représentation par moyennage direct des résultats bruts.

Il est évident que ces trois méthodes de visualisation produisent des surfaces ayant une forme similaire, avec une qualité meilleure pour les deux dernières méthodes basées sur des moyennes. Parmi les quatre possibilités de visualisation nous choisissons la troisième pour les raisons suivantes :

- La représentation des données brutes (la méthode 1) contient trop de bruit pour être utile par la suite ;
- L'interpolation (la méthode 2) ne fournit pas des résultats satisfaisants si les points représentent des données avec bruit et ils ne sont pas distribués de manière uniforme dans l'espace étudié ;
- Faire la moyenne des résultats des cinq tests (la méthode 4) et les représenter dans le système de coordonnées envisagées n'est pas précis, car nous savons que les valeurs obtenues diffèrent de celles envisagées, et ce sont les valeurs mesurées qui influencent les résultats.

Le seul inconvénient de la méthode 3 est que la technique de la fenêtre glissante rend les transitions lisses, et peut donc cacher certains détails. Mais les valeurs mentionnées auparavant, utilisées pour la taille de la fenêtre, sont à notre sens un bon compromis et seront utilisées par la suite (dans 5.4) pour toutes les représentations de résultats des tests à pleine échelle.

5.2.4.1.2 Modélisation des surfaces

Une description mathématique de la surface de PESQ sera utile pour réduire la quantité d'information stockée pour représenter la surface. L'avantage principal, par contre, est qu'elle permet d'avoir un aperçu sur la dépendance entre les conditions dans le réseau et le score PESQ. A une première approximation, cette dépendance pourrait être considérée comme une relation linéaire, très utile pour produire des estimations grossières. Nous n'avons pas inclus le développement de cette question dans nos recherches.

5.3 Résultats pour le transfert des fichiers

Nous avons effectué des tests employant la configuration décrite dans 5.1 avec différentes tailles de fichiers transférés. Les résultats présentés ci-dessous sont des moyennes obtenues pour 100 transferts à chaque taux de perte envisagé[1]. Nous avons

[1] A noter que le taux de perte est le même dans les deux directions ; il affecte donc également les confirmations du TCP, et pas seulement les paquets de données.

réalisé deux séries de tests, une avec un RTT de 0,8 ms (émulant un LAN) et l'autre avec un RTT de 60 ms (émulant un WAN).

Vu la formule (3.1), on s'attend à une performance très bonne pour le RTT de 0,8 ms, car la taille optimale de la fenêtre de transmission est en ce cas 10 kB (*BP* = 100 Mbps), inférieure à celle qui a été configurée de 64 kB. Par contre pour un RTT de 60 ms, la taille optimale est de 750 kB, largement supérieure à celle configurée ; la performance sera donc assez basse. Un « réglage » est possible, mais le but de nos tests est de mesurer la performance observée par un utilisateur habituel de l'application de transfert de fichiers.

5.3.1 Résultats instantanés

Nous présentons d'abord deux graphiques qui montrent les moments de réception des paquets et le débit. Les écarts entre points sur la figure 114 correspondent aux retards dûs à la perte d'un ou plusieurs paquets, ce qui active le mécanisme de retransmission. Il en résulte naturellement une diminution momentanée du débit instantané[1] (voir la relation avec la figure 115). Les résultats ont été obtenus pour le transfert d'un fichier de 1 MB pour une connexion avec un RTT de 0,8 ms et un taux de perte de 1%. Ce type de graphiques permet notamment de comprendre l'importance du type du paquet perdu et de saisir la corrélation entre débit et pertes. Il donne aussi de l'information sur l'évolution du débit dans le temps pendant le transfert, information qui disparaît si l'on n'utilise que les moyennes.

[1] Le graphique de débit instantané a été réalisé avec la technique de la fenêtre glissante. Le glissement se fait sur les événements de réception de paquets, avec un pas de déplacement de 1 ms et une taille de la fenêtre de 2 ms. Chaque point représente le débit calculé dans cette fenêtre glissante temporelle.

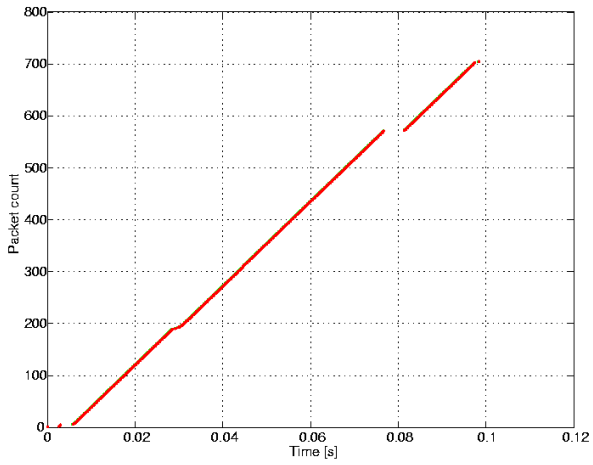


Figure 114 : Le moment d'arrivée de chaque paquet au recepteur (taille du fichier transféré = 1 MB, RTT = 0,8 ms, taux de perte = 1%).

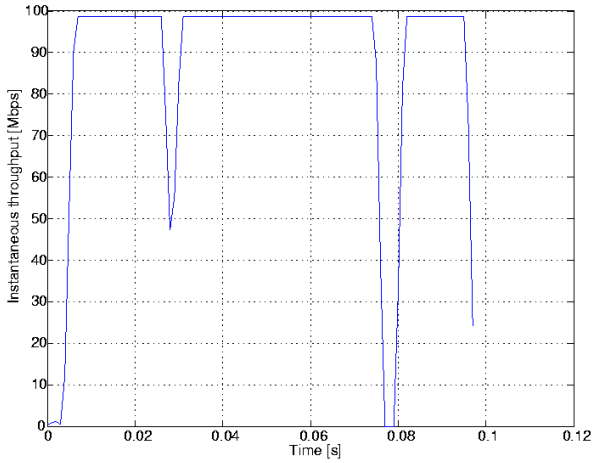


Figure 115 : Le débit instantané au recepteur (taille du fichier transféré = 1 MB, RTT = 0,8 ms, taux de perte = 1%).

5.3.2 Paramètres UPQ

Le tableau 28 montre les valeurs du *TTP* obtenues pour une perte nulle, avec les deux RTT et différentes tailles de fichiers transférés. On constate que l'efficacité temporelle du transfert croît avec la taille des fichiers, car la surcharge du fait d'établir et terminer la connexion devient moins significative comparée au temps même de transfert du fichier. La variation du *TTP* entre les deux RTT est d'un ordre de magnitude, comme on peut l'attendre étant données les considérations sur la taille optimale de la fenêtre TCP faites au début de la section.

Taille du fichier		*10 kB*	*100 kB*	*1 MB*	*10 MB*
TTP	*RTT = 0,8 ms*	0,0219	0,1741	0,9058	0,9361
	RTT = 60 ms	0,0029	0,0141	0,0559	0,0791

Tableau 28 : La performance du transfert en tant que fonction du RTT et de la taille de fichiers (pour des pertes nulles).

Nous présentons maintenant les résultats obtenus pour un fichier de 10 kB, la taille typique des fichiers dans l'Internet [Arl-96]. Pour de plus larges tailles, les graphiques de débit utile et *TTP* ont une forme similaire. Les valeurs de *TTP* s'approchent de 1 pour des fichiers de large taille et des petits RTT (voir le tableau 28), ce qui montre qu'il est plus efficace d'envoyer de larges quantités de données dans un large transfert que dans des transferts plus petits. Le taux de perte a varié de 0 à 25% dans les deux directions de communication.

Le débit utile (la figure 116) décroît de façon presque linéaire avec la perte de paquets, montrant la diminution de l'efficacité de l'utilisation de la connexion. Comme attendu, le *RTT* n'a aucune influence sur le débit utile, car D_{utile} ne dépend pas du temps. Par conséquent le débit utile n'est pas un indicateur suffisant en lui-même et doit être corrélé avec le *TTP*.

La performance de transfert (voir la figure 117) montre que le temps de transfert dépend du taux de perte de façon importante. Pour ce cas, la valeur maximale de *TTP* est 0,0219 (pour le RTT de 0,8 ms) à cause des durées supplémentaires des processus d'établissement et clôture de la connexion, qui représente environ 96% du temps de transfert pour un fichier de 10 kB.

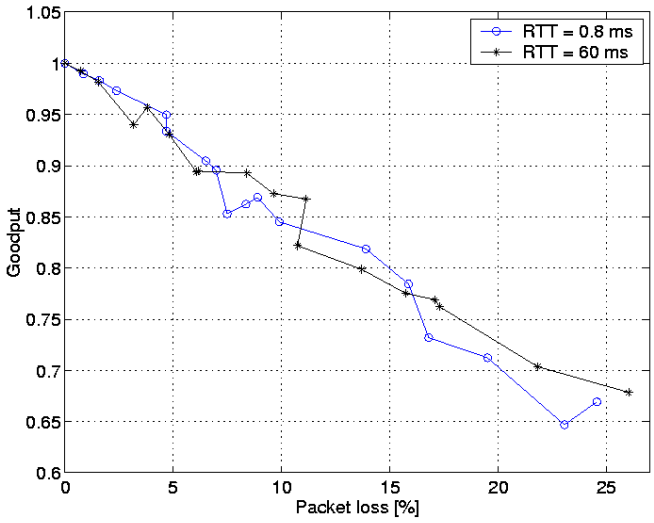


Figure 116 : Le débit utile en fonction du taux de perte de paquets pour deux RTT.

La figure 117 montre aussi que pour un RTT de 0.8 ms la valeur du *TTP* décroît 15 fois pour un taux de perte de 5% comparée à la valeur pour une perte nulle. Ceci équivaut à une augmentation de la durée du transfert d'environ 15 fois, donc une dégradation importante de l'UPQ. Pour des taux de perte supérieurs à 10% la performance est dégradée de centaines de fois. Pour un RTT de 60 ms le *TTP* est dès le début plus petit que pour un RTT de 0.8 ms et les pertes ont une influence moins dramatique.

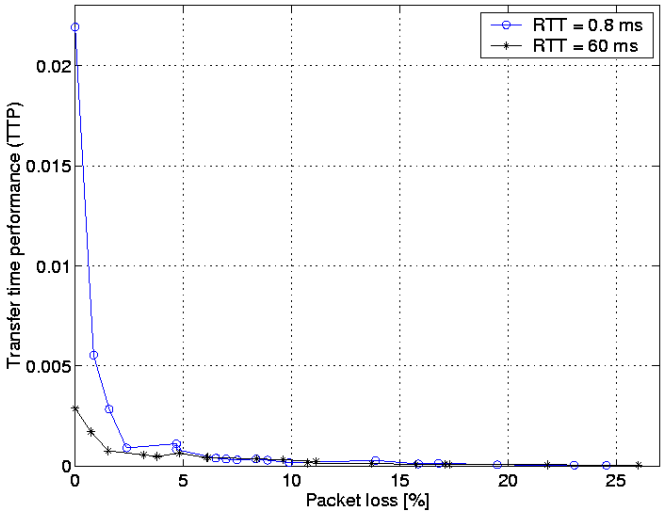


Figure 117 : La performance de transfert en fonction du taux de perte de paquets pour deux RTT.

Nous avons observé que l'influence de la perte des paquets sur la performance du TCP dépend du type des paquets perdus : la perte d'un paquet de données est facilement cachée par le mécanisme de retransmission du TCP, mais la perte d'un

paquet de contrôle (pour établir ou clôturer la connexion) a une influence plus grande à cause de temps d'expiration relativement grands. Pour un fichier de 10 kB, la durée du transfert a augmenté par un ordre de magnitude dans un cas pareil.

Les deux figures qui suivent montrent les résultats pour d'autres tailles de fichiers, groupés pour une comparaison plus facile. Il s'agit de trois tailles de fichiers plus grandes (100 kB, 1 MB et 10 MB), plus proches des tailles de fichiers transférés effectivement par FTP. Le RTT pour ces résultats a été de 0,8 ms et le taux de perte a varié entre 0 et 40% dans les deux directions de communication. A noter les similarités entre les graphiques de débit utile (la figure 118) et le fait que ce paramètre décroît moins rapidement que pour les fichiers de 10 kB (par exemple, à 10% perte, le débit utile est 0,9 dans la figure 118 et seulement 0,85 dans la figure 116).

La performance temporelle plus élevée pour des taux de perte réduits est mise en évidence par l'intermédiaire du TTP (la figure 119) – on voit qu'à 0% de perte la performance dépasse 0,9 pour des fichiers aux alentours de 1MB.

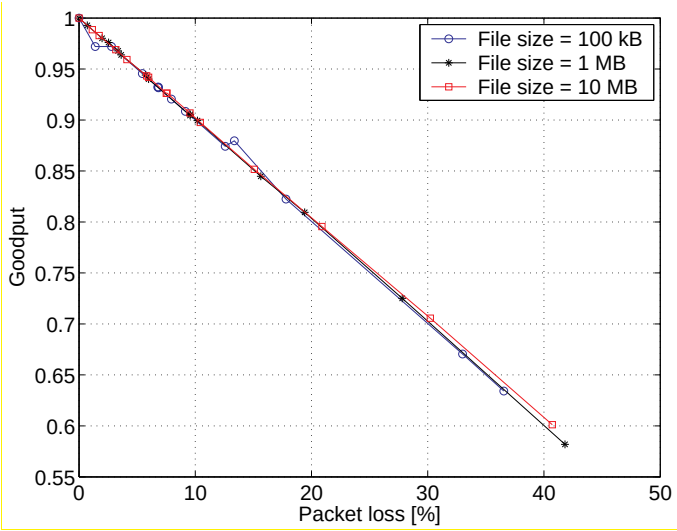


Figure 118 : Le débit utile pour trois tailles de fichiers transférés (RTT = 0,8 ms).

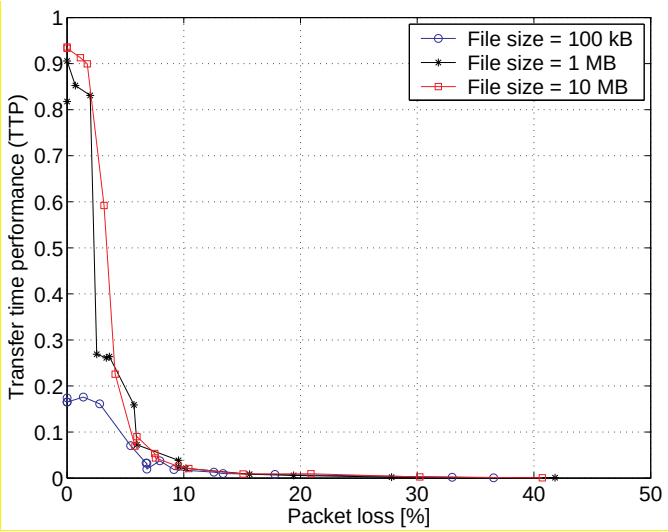


Figure 119 : La performance du transfert pour trois tailles de fichiers transférés (RTT = 0,8 ms).

5.3.3 Discussion

Ces travaux ont permis, par une mesure précise des paramètres QoS et de l'évaluation en parallèle de l'UPQ pour le transfert des fichiers, de quantifier la relation qui les lie et d'identifier les exigences du FTP dans une configuration standard.

Il y a des différences entre les résultats par rapport à la taille du fichier transféré. Nous allons discuter ici ceux obtenus pour le fichier de 10 kB, taille moyenne des fichiers dans l'Internet [Arl-96].

Conformément aux attentes, le débit utile décroît avec le taux de perte. La dépendance est linéaire et la diminution n'est pas très grande dans la gamme des taux de perte de 0 à 5%. Si on considère la valeur 0,95 comme seuil d'acceptabilité pour le débit utile, et donc pour l'efficacité d'utilisation du réseau, nous concluons que le taux de perte ne doit pas dépasser 5%. Pour des taux de perte au dessus de 20%, le débit utile indique une efficacité inférieure à 0,7. Elle approche 0,5 pour des taux de perte de l'ordre de 40%.

Le graphique de la performance du transfert a une forme exponentielle négative, montrant que le temps nécessaire pour le transfert augmente fortement avec le taux de perte. Pour un taux de perte aux environs de 5% et un RTT petit, le TTP est environ dix fois plus petit que la valeur obtenue pour une perte de paquets nulle (sauf pour le fichier de 100 kB, où le rapport est d'environ 6). La dégradation observée est moins importante pour le RTT de 60 ms que pour celui de 0,8 ms. A 25% de taux de perte le

temps de transfert devient des centaines de fois plus grand que celui dans le cas d'une perte inférieure à 5%. Ceci rend la connexion pratiquement inutilisable pour ce type d'application.

Nous concluons que le transfert de fichier exige un taux de perte inférieur à 5% pour maintenir l'efficacité du réseau en dessus de 0,95 et ne pas avoir un temps de transfert plus grand que d'un ordre de magnitude par rapport au temps de transfert pour une perte nulle. Une bonne performance impose des limites encore plus strictes : le taux de perte ne doit pas dépasser 1% pour que l'efficacité d'utilisation du réseau avoisine 0,99 et le temps de transfert n'est pas plus grand que celui qui prévaut dans des conditions de perte nulle que d'un facteur quatre au maximum.

5.4 Les résultats pour VoIP

Les quatre codeurs avec lesquels nous avons fait des expériences sont G.711, G.726, GSM et G.729. Une section séparée est dédiée à chaque codeur et elles sont suivies par une section d'analyse comparative.

5.4.1 Codeur G.711

Le codeur G.711 [ITU-711] envoie des données à 8 kHz avec 8 bits par échantillon, résultant dans un taux de données brutes de 64 kbps. Le son est dans le format PCM, codé dans notre cas avec μ-*law*.

Un nombre de 227410 octets a été transmis. Les paquets de données ont 378 octets, dont 320 représentent l'information vocale. Ceci correspond à 320 échantillons, ce qui est équivalent à 40 ms de signal vocal (pour une fréquence d'échantillonnage de 8 kHz, un échantillon à 125 μs). Le taux de paquets a été de 25 paquets par seconde. La taille des données vocales étant de 192000 octets (voir 5.2.4), la surcharge a été de 35410 octets, c.-à-d. 16% des octets transmis. En utilisant ce codeur nous avons obtenu les résultats présentés dans les figures 120 à 123.

Les résultats de l'ensemble des tests ont été utilisés pour calculer les scores PESQ moyens. Leur dépendance de la gigue et de la perte de paquets à la fois est représentée sur la figure 120 en 3D.

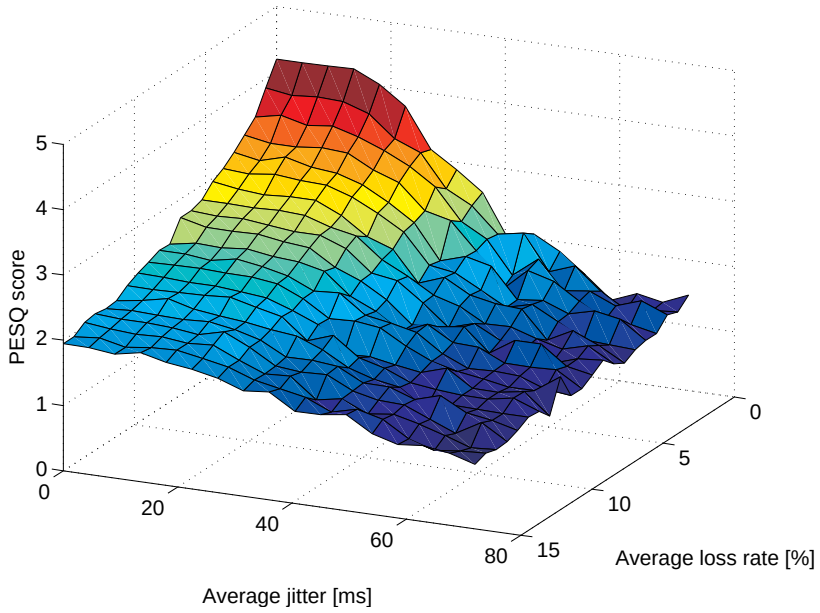


Figure 120 : Le score PESQ moyen en fonction de la gigue et de la perte de paquets.

La figure 121 montre les variations du score PESQ avec le taux de perte pour quelques valeurs de la gigue (0, 25, 50 et 75 ms). Ce graphique représente des sections verticales dans la surface sur la figure 120. On peut établir que pour un taux de perte allant jusqu'à 4%, l'influence sur le score PESQ est observable, mais la qualité reste « bonne[1] » pour de petites valeurs de la gigue ; la qualité devient « basse » pour un taux de perte supérieur, jusqu'à 15%. Si la gigue est élevée, la perte n'influence pas considérablement la qualité perçue, qui en plus est « inacceptable » : la cause est le niveau important de la perte de compensation de gigue.

[1] On rappelle que la signification de qualité « bonne », « basse » ou « inacceptable » est donnée dans 3.4.2.5.

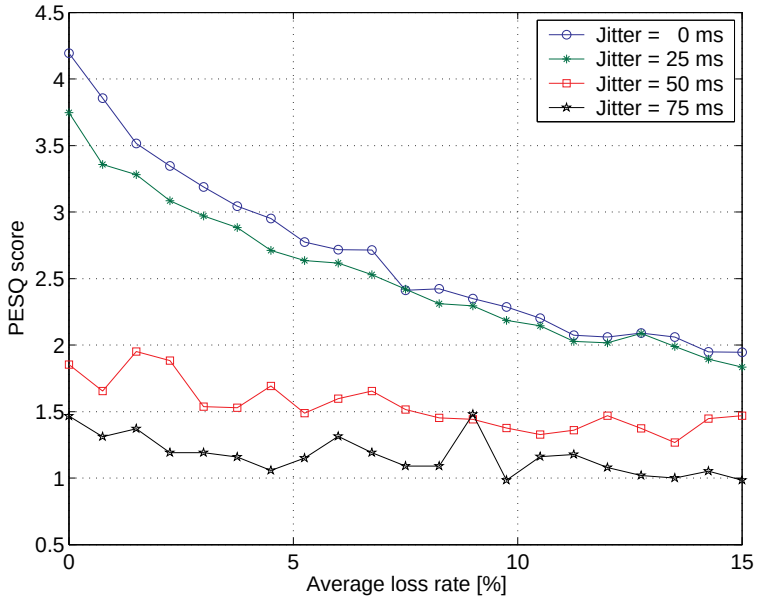


Figure 121 : Le score PESQ moyen en fonction du taux de perte de paquets.

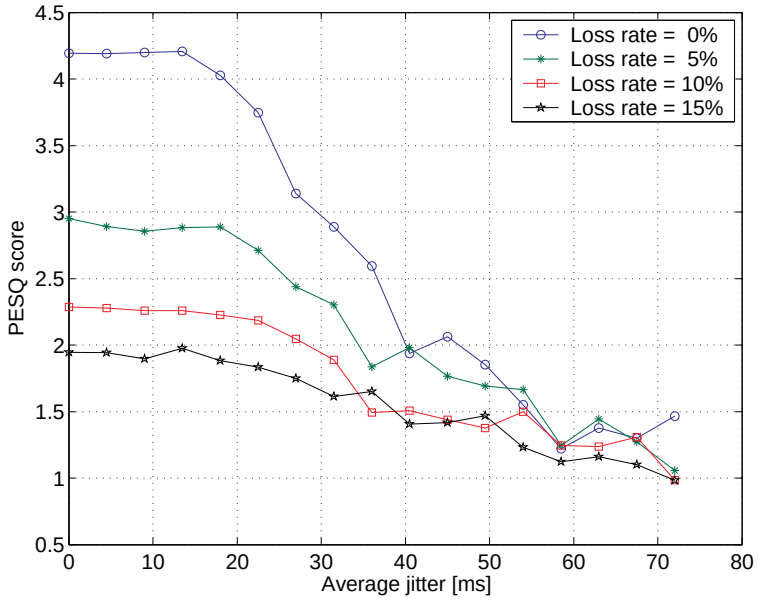


Figure 122 : Le score PESQ moyen en fonction de la gigue.

La figure 122 montre la variation du score PESQ en fonction de la gigue, pour certaines valeurs du taux de perte (0, 5, 10 et 15%), comme des sections verticales dans la surface de la figure 120. Cette figure montre que pour une gigue en dessous de 20 ms, il n'y a presque aucun changement du score PESQ – c'est l'effet attendu d'un délai de compensation de gigue de 80 ms (cf. la figure 100). Plus la gigue est importante, plus son effet est significatif sur la qualité perçue, mais l'amplitude de cet effet décroît avec l'augmentation des pertes de paquets.

Pour donner un aperçu sur la localisation des limites entre qualités « bonne » et « basse », et entre « basse » et « inacceptable », nous avons également opéré des sections horizontales dans la surface représentée sur la figure 120, au niveau des scores PESQ correspondant à ces limites (les valeurs 3 et 2 respectivement). Pour ce codeur on peut aussi déceler le seuil entre la qualité « excellente » et « bonne », le score PESQ de 3,8. G.711 est le seul codeur étudié à atteindre cette qualité – la zone correspondante est trouvée pour un taux de perte de 0 à 1% et une gigue de 0 à 20 ms.

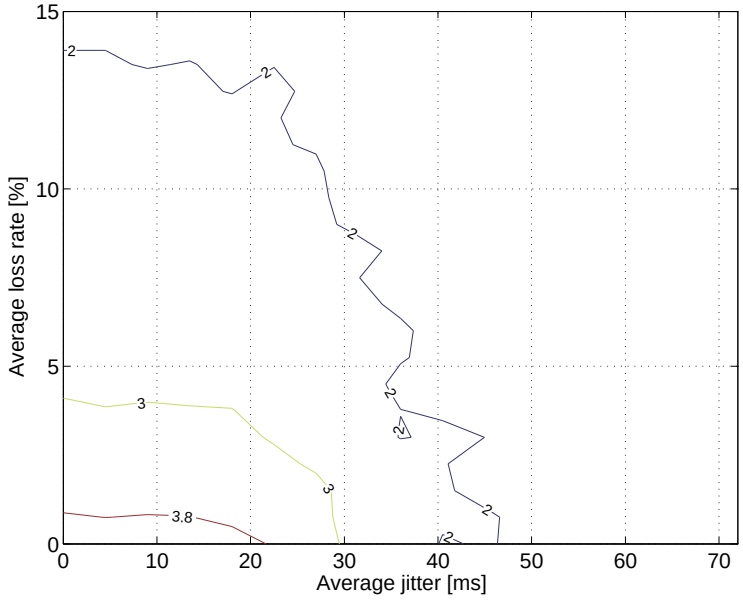


Figure 123 : Les contours des frontières entre différents niveaux de qualité perçue.

5.4.2 Codeur G.726

Le codeur G.726 [ITU-726] convertit un canal PCM de 64 kbps, *μ-law* ou *A-law*, en un canal à 40, 32, 24 ou 16 kbps. Seul le codage à 32 kbps est disponible dans notre application.

Un total de 115210 octets a été transmis. Les paquets de données ont 382 octets, dont 324 octets représentent l'information vocale. Ceci correspond à 648 échantillons, l'équivalent de 80 ms de signal vocal, à un taux de 12,5 paquets par seconde. Etant donné que les données vocales occupent 192000 octets, le taux de compression pour la transmission est de 1,67. En utilisant ce codeur, nous avons obtenu les résultats présentés sur les figures 124 à 127.

La figure 124 met en évidence la dépendance du score PESQ en fonction de la gigue et de la perte de paquets. Les résultats de tous les tests ont été utilisés pour calculer les scores PESQ moyens.

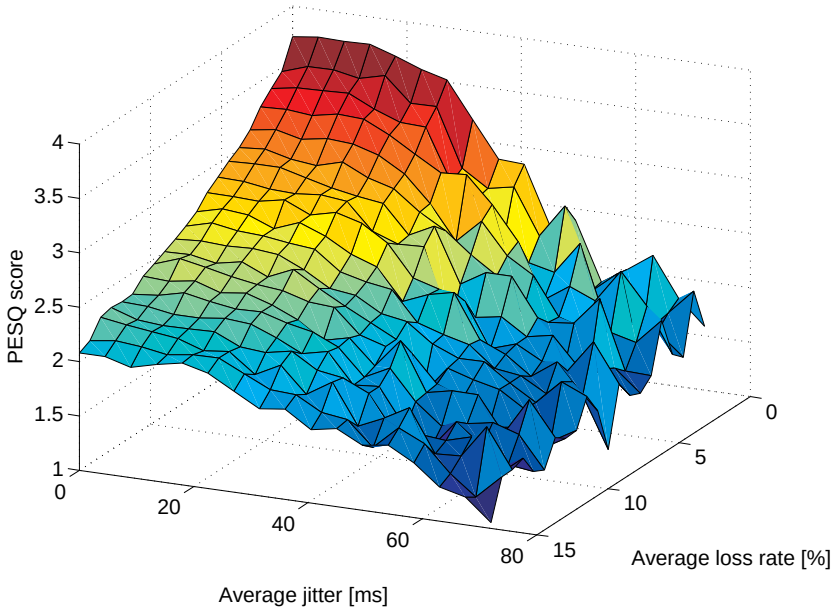


Figure 124 : Le score PESQ moyen en fonction de la gigue et de la perte de paquets.

La figure 125 montre les variations du score PESQ en fonction du taux de perte pour quatre valeurs de la gigue (0, 25, 50 et 75 ms). Ce graphique représente des sections verticales dans la surface sur la figure 124. La forme des résultats est similaire à celle qui a été obtenue pour G.711 et les mêmes limites peuvent être établies par rapport à l'influence des pertes sur la qualité. Une différence est le point de commencement des lignes dans le graphique, qui est plus bas à cause du niveau supérieur de compression. Pour des valeurs élevées de la gigue, les lignes du graphique ne sont pas très lisses.

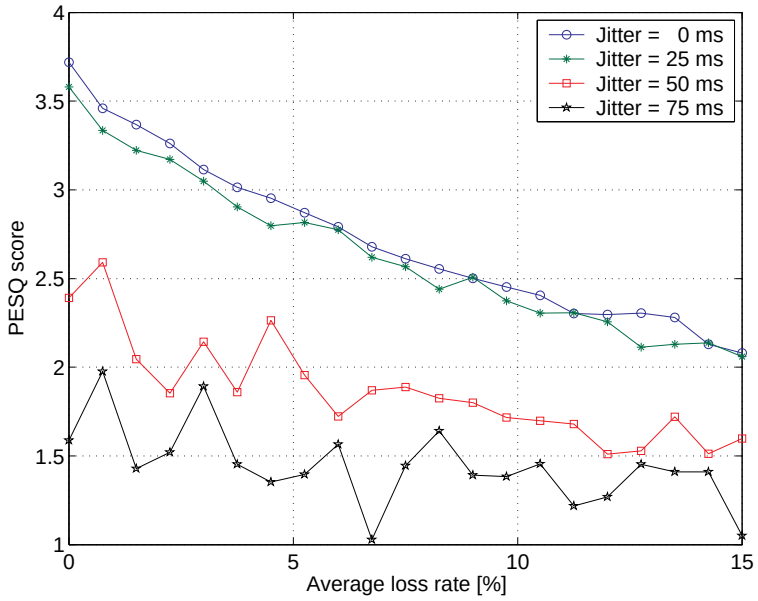


Figure 125 : Le score PESQ moyen en fonction du taux de perte de paquets.

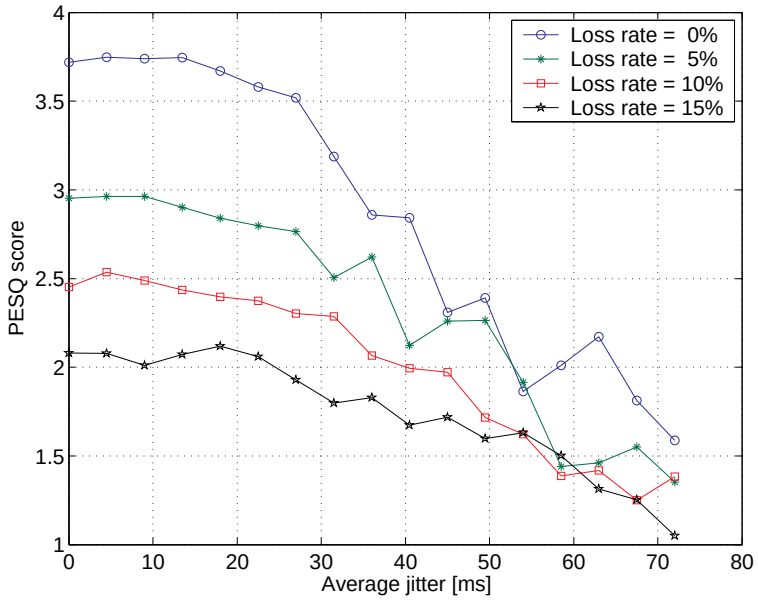


Figure 126 : Le score PESQ moyen en fonction de la gigue.

La figure 126 montre la variation du score PESQ en fonction de la gigue pour quelques valeurs du taux de perte (0, 5, 10 et 15%). Ce graphique représente des sections verticales de la surface sur la figure 124. Tant que la gigue est en dessous de 20 ms, les scores PESQ sont presque constants. Ensuite la qualité décroît et devient « basse » ou « inacceptable » pour une gigue dépassant 40 ms, en fonction du taux de perte.

Des sections horizontales dans la surface sur la figure 124 fournissent une meilleure vue sur les limites entre les qualités « bonne » et « basse » et entre les qualités « basse » et « inacceptable » (voir la figure 127). Ce codeur ne peut pas atteindre une qualité « excellente », même dans des conditions réseau très bonnes. A noter que la ligne de niveau correspondant à score PESQ = 2 n'est pas connectée à cause du bruit résiduel après le lissage.

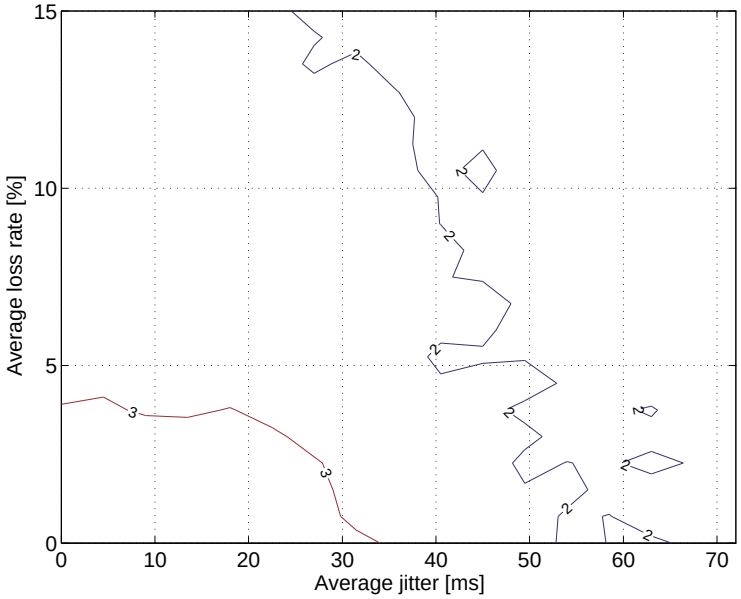


Figure 127 : Les contours des frontières entre différents niveaux de qualité perçue.

5.4.3 Codeur GSM

Le codeur GSM (Global System for Mobile telecommunications) [Rah-93], employé dans la téléphonie mobile, utilise le codage linéaire prédictif pour comprimer un signal vocal à 13 kbps.

Un nombre de 57610 octets a été transmis. Les paquets de données ont 190 octets, dont 132 représentent l'information vocale, ce qui correspond à 649 échantillons, l'équivalent approximatif de 80 ms de signal vocal. Le taux de paquets a été de 12,5 paquets par seconde. Les données vocales constituant 192000 octets, le taux de compression pour la transmission est donc de 3,33. En utilisant ce codeur, nous avons obtenu les résultats présentés dans les figures 128 à 131.

Les résultats obtenus dans tous les tests avec ce codeur ont été utilisés pour calculer les scores PESQ moyennes. Ils sont représentés sur la figure 128 comme un graphique 3D, montrant la dépendance envers la gigue et la perte de paquets.

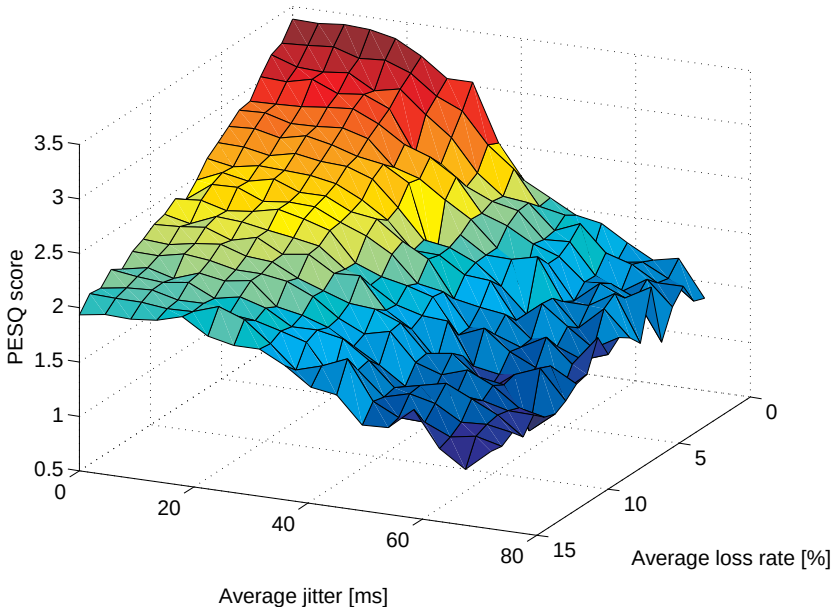


Figure 128 : Le score PESQ moyen en fonction de la gigue et de la perte de paquets.

La figure 129 montre les variations du score PESQ avec le taux de perte pour certaines valeurs de la gigue (0, 25, 50 et 75 ms). Ce graphique représente des sections verticales dans la surface sur la figure 128.

La figure 130 montre la variation du score PESQ par rapport à la gigue pour quelques valeurs du taux de perte (0, 5, 10 et 15%). Ce graphique représente des sections verticales dans la surface sur la figure 128.

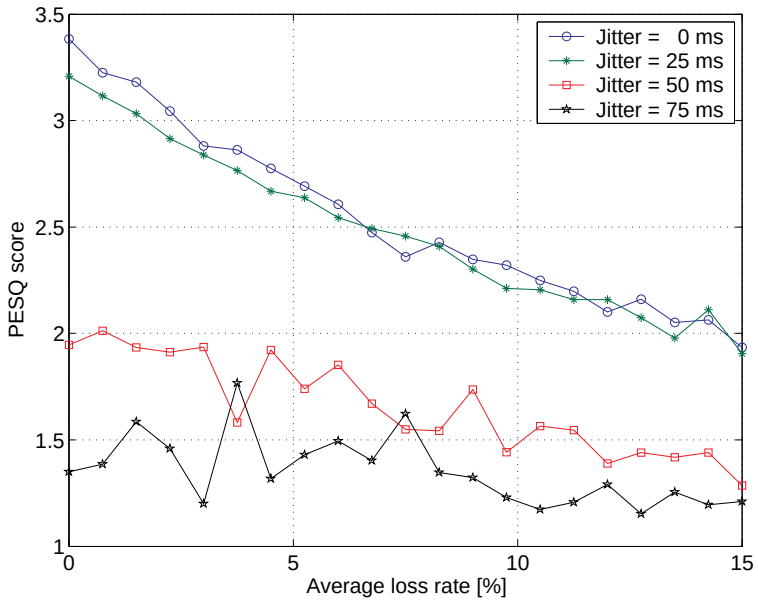


Figure 129 : Le score PESQ moyen en fonction du taux de perte de paquets.

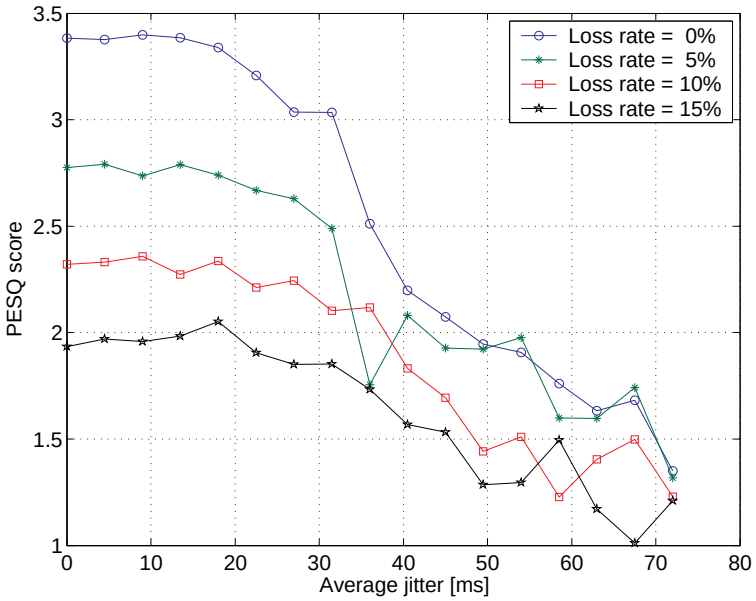


Figure 130 : Le score PESQ moyen en fonction de la gigue.

Les mêmes commentaires que ceux faits pour G.726 peuvent être faits concernant les deux figures précédentes. Pour mieux mettre en évidence les limites entre les différents niveaux de qualité, des sections horizontales dans la surface sur la figure 128 ont été faites et elles sont représentées sur la figure 131.

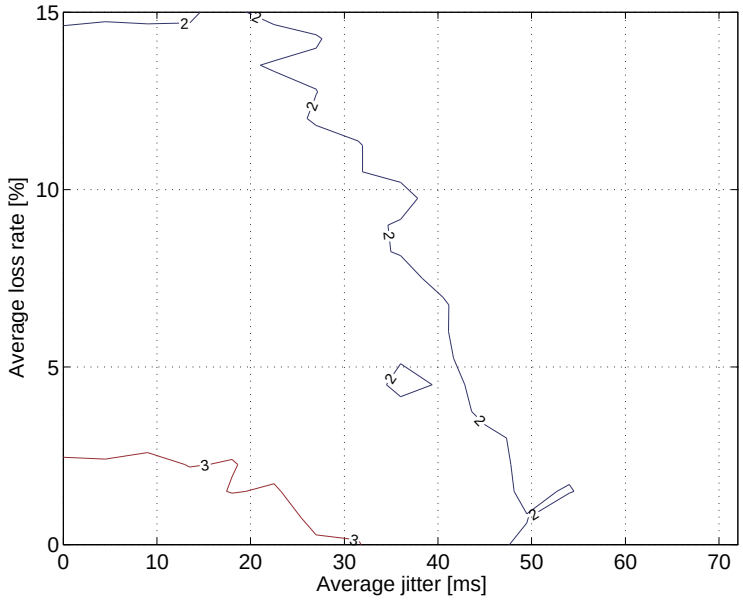


Figure 131 : Les contours des frontières entre différents niveaux de qualité perçue.

5.4.4 Codeur G.729

Le codeur G.729 [ITU-729] est fréquemment utilisé pour la communication par VoIP. Il envoie des données vocales à 8 kbps en utilisant le codage CS-ACELP (Conjugate-Structure Algebraic-Code-Excited Linear Prediction).

Un nombre de 51610 octets a été transmis. Les paquets de données ont 170 octets chacun, dont 112 représentent l'information vocale (plus la surcharge spécifique à ce codeur). Ceci correspond à 640 échantillons, l'équivalent de 80 ms de signal vocal. Le taux de paquets a été de 12,5 paquets par seconde. Etant donné que les données vocales sont 192000 octets, le taux de compression pour la transmission a été de 3,72. En utilisant ce codeur nous avons obtenu les résultats présentés sur les figures 132 à 135.

Les résultats de tous les tests ont été utilisés pour calculer les scores PESQ moyens. Ils sont représentés sur la figure 132 comme un graphique 3D.

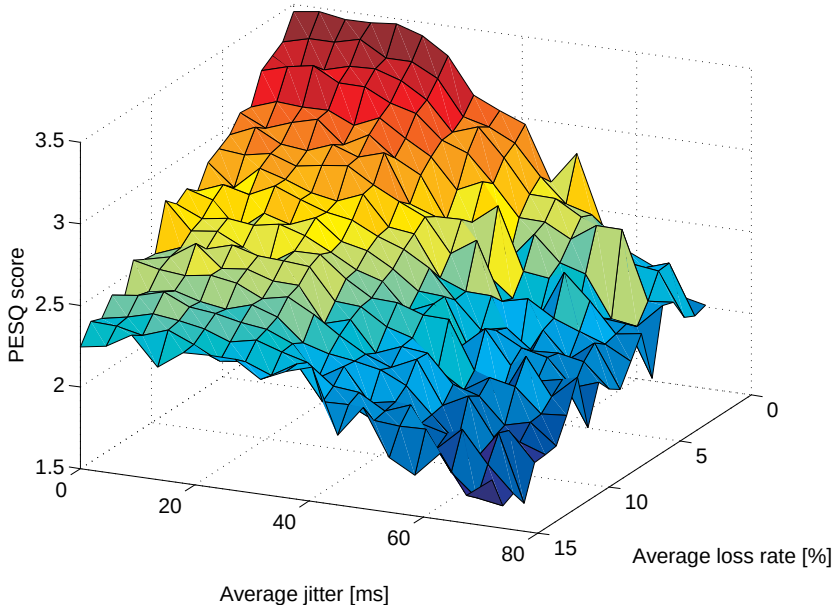


Figure 132 : Le score PESQ moyen en fonction de la gigue et de la perte de paquets.

La figure 133 montre les variations du score PESQ avec le taux de perte pour quatre valeurs de la gigue (0, 25, 50 et 75 ms). Ce graphique représente des sections verticales dans la surface sur la figure 132. L'influence du taux de perte est réduite, la diminution du score PESQ étant inférieure à 1,3 pour une variation du taux de perte de 0 à 15%.

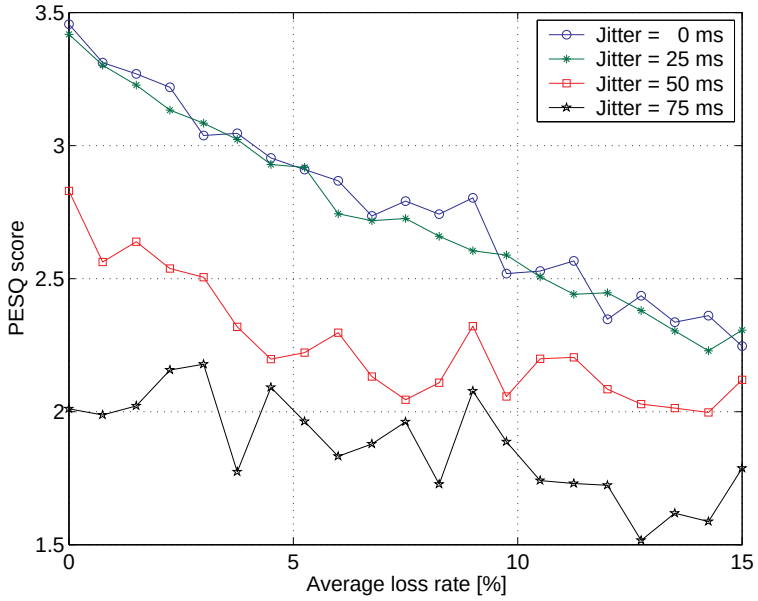


Figure 133 : Le score PESQ moyen en fonction du taux de perte de paquets.

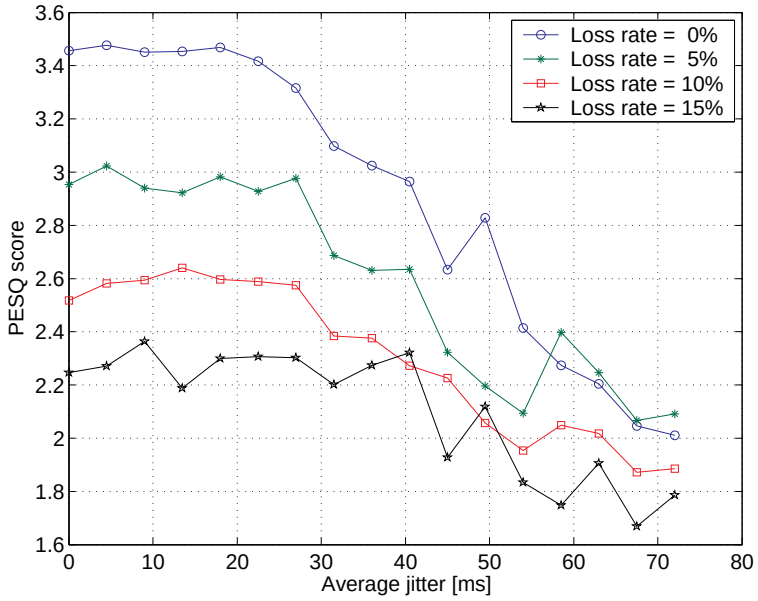


Figure 134 : Le score PESQ moyen en fonction de la gigue.

La figure 134 montre la variation du score PESQ avec la gigue pour certaines valeurs du taux de perte (0, 5, 10 et 15%). Ce graphique représente des sections verticales dans la surface sur la figure 132. L'influence de la gigue aussi est assez réduite : une réduction en dessous de 1,5 est observée pour la variation de la gigue de 0 à 75 ms.

Pour donner un aperçu sur la position des limites entre la qualité « bonne » est « basse » et entre celle « basse » et celle « inacceptable », nous avons effectué des

sections horizontales dans la surface représentée sur la figure 132. A noter la grande zone de qualité « acceptable » obtenu pour ce codeur, qui couvre environ 90% de l'espace étudié.

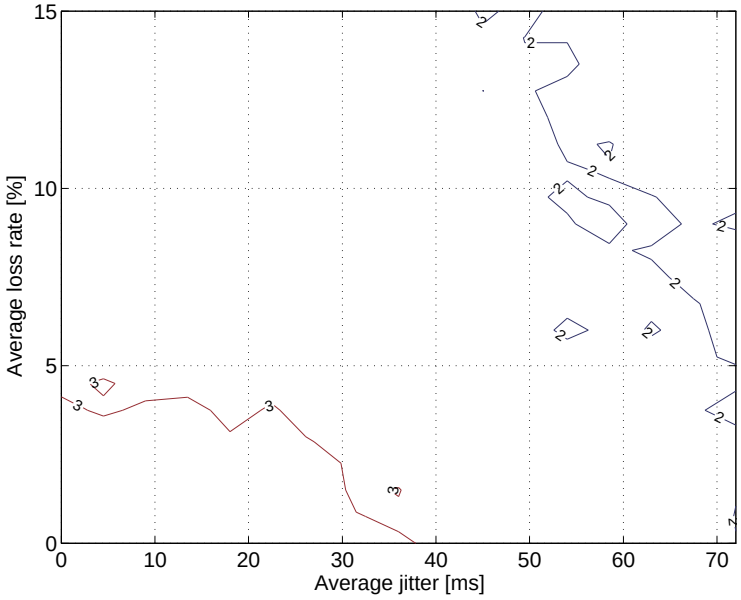


Figure 135 : Les contours des frontières entre différents niveaux de qualité perçue.

5.4.5 Comparaison de codeurs

Nous présentons ici une comparaison entre les codeurs étudiés avec des graphiques construits sur la base des surfaces 3D présentées auparavant. Chaque graphique montre la dépendance du score PESQ par rapport à un des paramètres tout en fixant la valeur pour l'autre. Ces sections verticales superposées donnent une idée sur les différences qui existent entre codeurs en ce qui concerne la dépendance du score PESQ de gigue et perte de paquets. Néanmoins, pour une comparaison plus approfondie les surfaces mêmes doivent être utilisés, conjointement avec les contours.

5.4.5.1 Analyse pour gigue fixe

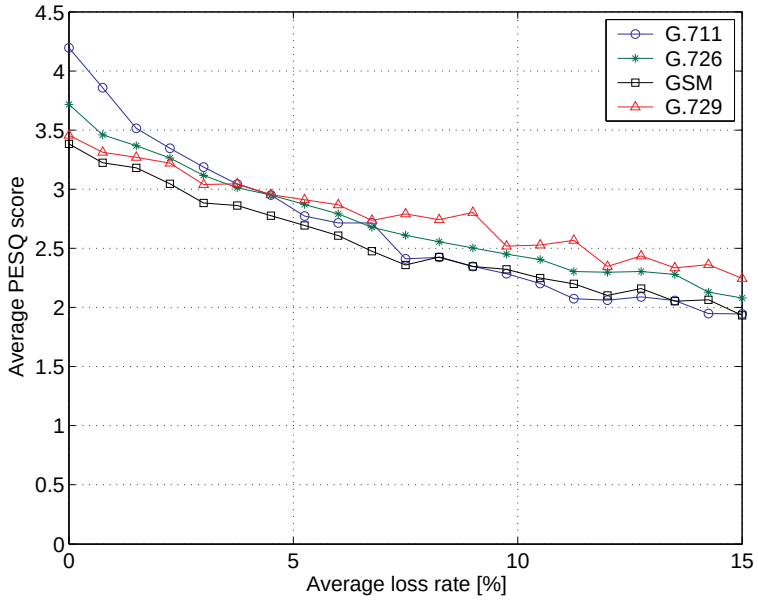


Figure 136 : Le score PESQ moyen en fonction du taux de perte de paquets pour tous les codeurs (gigue = 0 ms).

Pour une gigue de 0 ms (voir la figure 136) G.711 a la meilleure performance pour un taux de perte de jusqu'à 4%, après ce le G.729 qui domine. Le deuxième codeur est toujours G.726. GSM a la pire performance presque tout le temps.

A une gigue de 25 ms (voir la figure 137) G.711 a la meilleure performance seulement jusqu'à un taux de perte de 2%, en suite G.729 est supérieur de nouveau. Comme avant, le deuxième meilleur codeur et toujours G.726. GSM est en général le codeur avec la pire performance, même si pour des taux de perte supérieurs à 10% G.711 est encore un peu plus mauvais.

Pour une gigue en dessus de 50 ms (les figures 138 et 139), G.729 est de loin le meilleur codeur, avec un score PESQ plus grand que pour les autres codeurs de jusqu'à 0,5. G.726 est de nouveau le deuxième meilleur, GSM le troisième et G.711 a la pire performance. Une raison est le fait qu'il utilise des paquets de 40 ms, donc ces hauts niveaux de gigue lui cause le plus de « dégâts ».

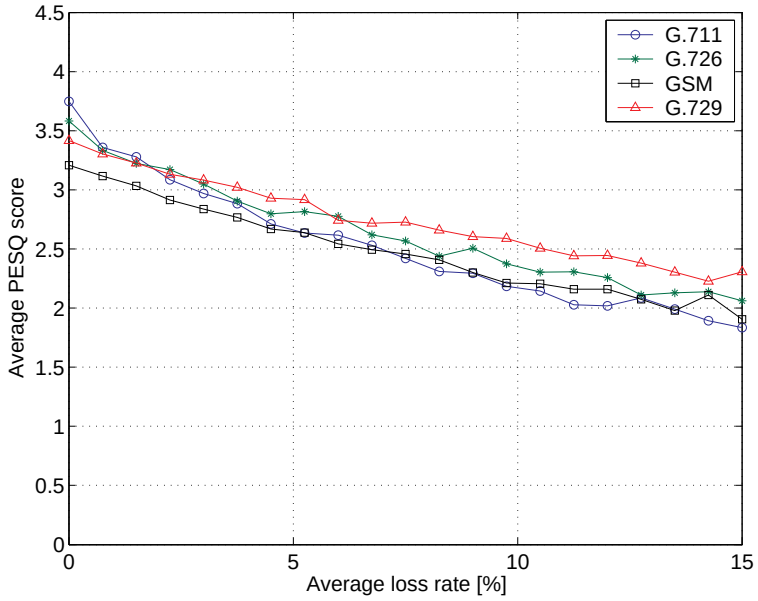


Figure 137 : Le score PESQ moyen en fonction du taux de perte de paquets pour tous les codeurs (gigue = 25 ms).

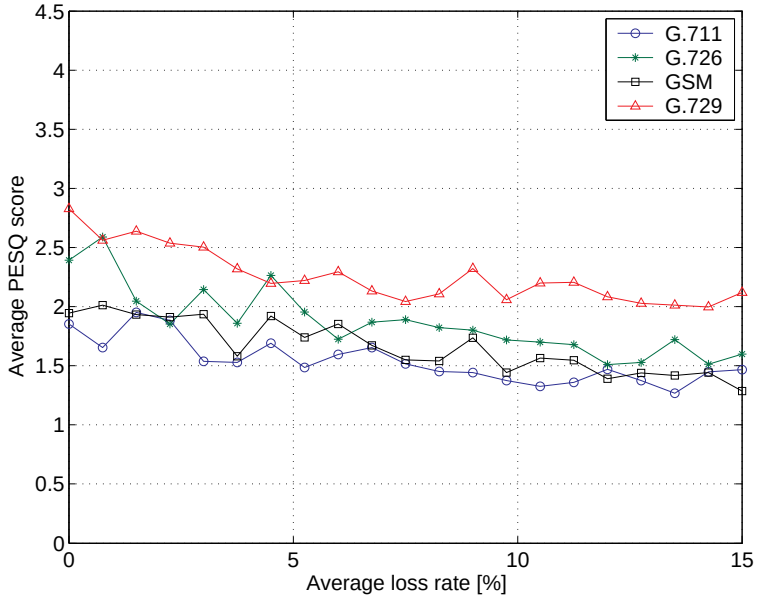


Figure 138 : Le score PESQ moyen en fonction du taux de perte de paquets pour tous les codeurs (gigue = 50 ms).

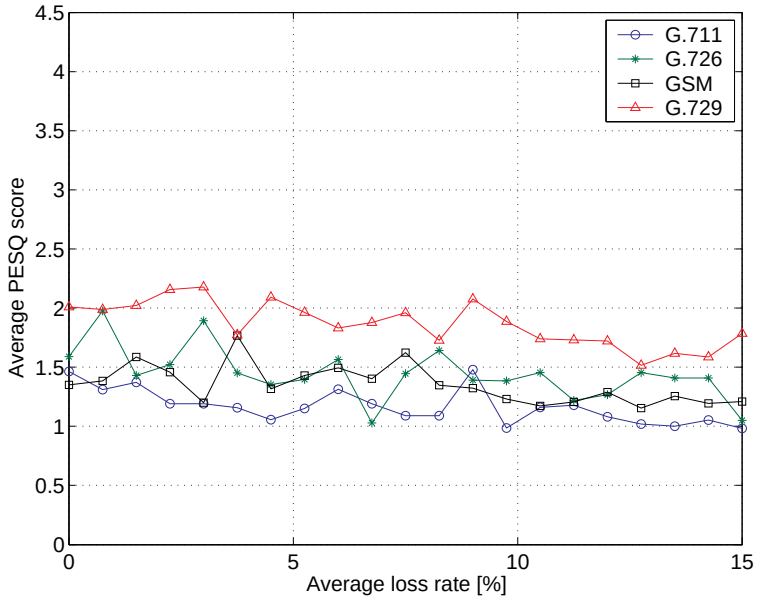


Figure 139 : Le score PESQ moyen en fonction du taux de perte de paquets pour tous les codeurs (gigue = 75 ms).

5.4.5.2 Analyse pour taux de perte fixe

Pour un taux de perte de 0% (la figure 140) la performance de G.711 dépasse celles des autres codeurs tant que la gigue est en dessous de 25 ms. G.726 se situe au deuxième rang, sauf pour l'intervalle 25-32 ms ou il offre la meilleure performance. Dès ce point (une gigue moyenne de 32 ms), G.729 devient le meilleur codeur. GSM a une faible performance est il est seulement supérieur à G.711 pour une gigue excédant 40 ms.

Quand le taux de perte est de 5% (la figure 141) tous les codeurs ont plus ou moins le même comportement pour une gigue jusqu'à 30 ms. G.729 devient après le codeur le plus performant et G.711 le moins performant.

Pour des taux de perte supérieurs à 10%, les graphiques ont le même aspect (les figures 142 et 143). G.729 fournit la meilleure qualité perçue, dans les limites de qualité « basse » pour une gigue inférieure à 50 ms. Tous les autres codeurs ont une performance inférieure, surtout quand la gigue approche 75 ms.

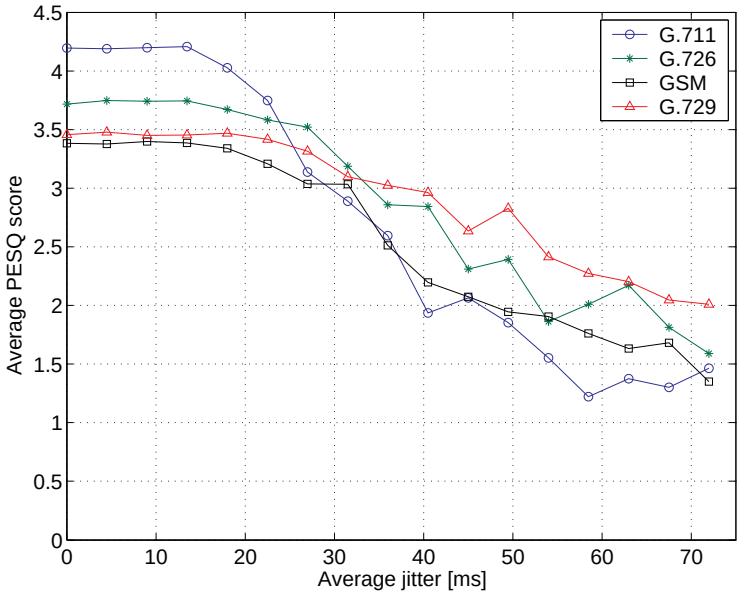


Figure 140 : La dépendance du score PESQ moyen de la gigue moyenne pour tous les codeurs (le taux de perte de paquets = 0%).

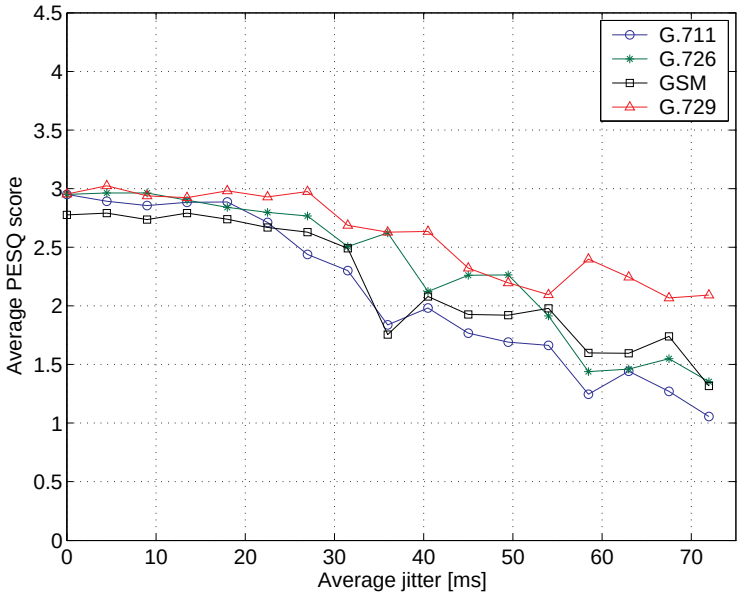


Figure 141 : Le score PESQ moyen en fonction de la gigue moyenne pour tous les codeurs (le taux de perte de paquets = 5%).

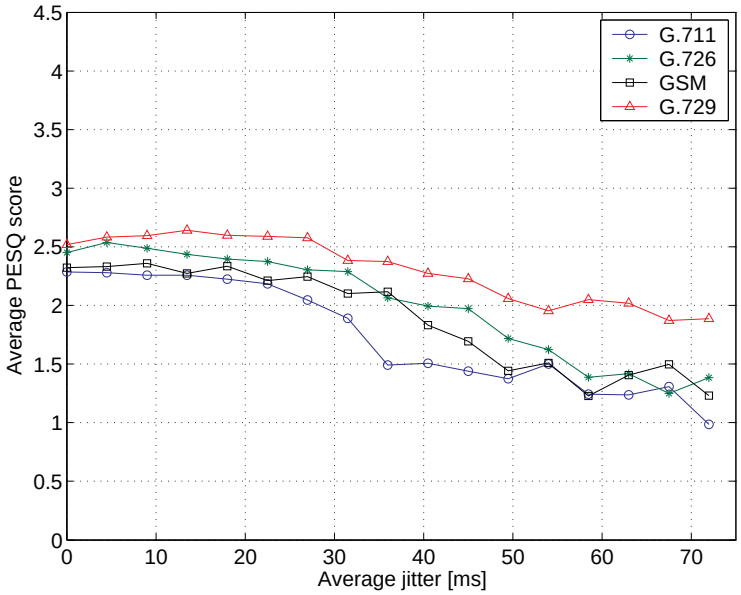


Figure 142 : Le score PESQ moyen en fonction de la gigue moyenne pour tous les codeurs (le taux de perte de paquets = 10%).

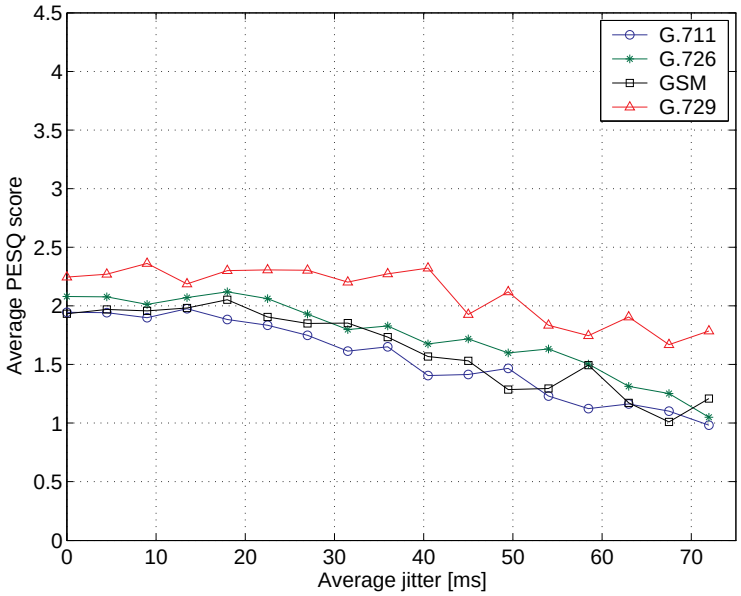


Figure 143 : Le score PESQ moyen en fonction de la gigue moyenne pour tous les codeurs (le taux de perte de paquets = 15%).

5.4.5.3 Sommaire des codeurs

La comparaison est résumée dans le tableau 29. Pour chaque codeur nous montrons la bande passante utilisée et les frontières approximatives sur les paramètres QoS qui doivent être imposées afin d'atteindre un certain niveau de qualité.

Par exemple, G.711 fournit une qualité « bonne » tant que le taux de perte est inférieur à 4% et la que gigue moyenne ne dépasse pas 30 ms. Le même codeur fournit une qualité « basse » mais acceptable pour des taux de perte situés entre 4 et 14% et une gigue entre 30 et 45 ms. En dehors de ces limites la qualité est inacceptable.

Codeur	***Seuil entre qualité « bonne » et « basse »***		***Seuil entre qualité « basse » et « inacceptable »***	
	Perte [%]	*Gigue [ms]*	*Perte [%]*	*Gigue [ms]*
G.711 (64 kbps)	< 4	< 30	4 – 14	30 – 45
G.726 (32 kbps)	< 4	< 35	4 – N.D.	35 – 55
GSM (13 kbps)	< 2,5	< 32	2,5 – 15	32 – 50
G.729 (8 kbps)	< 4	< 38	4 – N.D.	38 – N.D.

Tableau 29 : Comparaison des codeurs par rapport aux seuils entre niveaux de qualité.

A noter tout de même que les régions concernées par le tableau 29 ne sont pas de forme rectangulaire ; par suite, pour établir leurs frontières exactes, les graphiques des figures 120, 124, 128 et 132 doivent être consultés. Le tableau contient seulement les limites des zones rectangulaires englobant les régions d'intérêt. Les figures mentionnées auparavant sont regroupées sur la figure 144 pour faciliter la lecture et donner une image comparative.

Un classement des codeurs peut être fait à partir du recouvrement de la région correspondant à un certain niveau de qualité par rapport à l'espace gigue-perte étudié. Ensuite nous présentons deux tels classements, un pour la région de qualité au moins « bonne » (donc la zone de qualité excellente est incluse dedans, le score PESQ étant supérieur ou égal à 3), et un autre pour la région de qualité au moins « basse » (score PESQ supérieur ou égal à 2) :

- Classement des codeurs basé sur le recouvrement de la qualité « bonne » :

  i) G.729 (10,48%)

  ii) G.726 (9,62%)

  iii) G.711 (9,00%)

  iv) GSM (5,06%)

- Classement des codeurs basé sur le recouvrement de la qualité « basse » :

    i) G.729 (88,16%)

    ii) G.726 (60,33%)

    iii) GSM (51,25%)

    iv) G.711 (41,7%)

A noter que cette classification se fait sur un seul critère, et ne prend pas en compte, par exemple, la bande passante requise par chaque codeur, un critère important pour le compromis entre qualité perçue et efficacité de l'utilisation du réseau. Un classement général qui prend en compte ce paramètre peut être le suivant:

i) G.729

ii) G.726

iii) GSM

iv) G.711

Nous avons pris en considération les classements précédents et le fait que GSM utilise 8 fois moins bande passante que G.711 pour les départager. A noter que le codeur le plus approprié pour une situation quelconque doit toujours être choisi en fonction de sa performance dans certaines conditions réseau, comme montré par exemple dans 5.4.5.4.

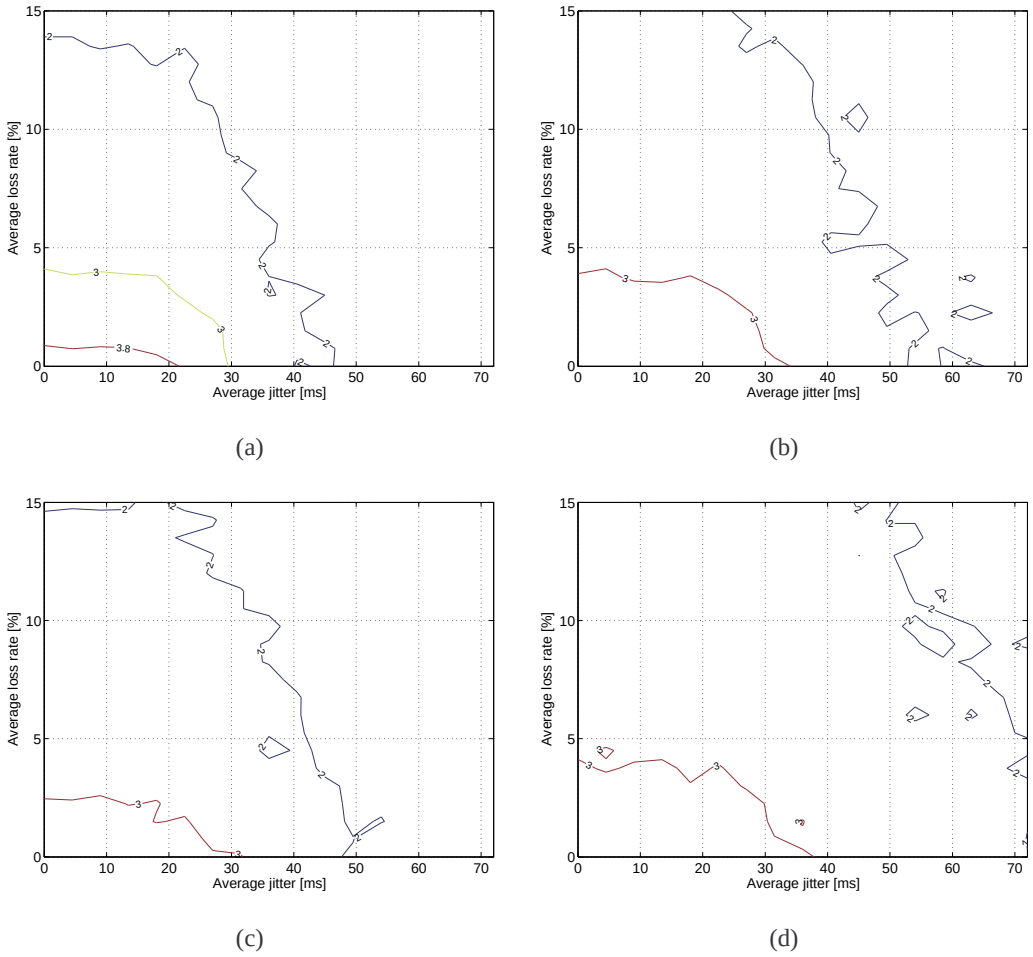


Figure 144 : Les contours des frontières entre différents niveaux de qualité perçue pour les quatre codeurs étudiés: G.711 (a), G.726 (b), GSM (c) et G.729 (d).

5.4.5.4 Discussion

Voici deux exemples de la façon dont les graphiques sur la figure 144 peuvent être interprétés et utilisés en pratique.

Le premier prend le point de vue d'un utilisateur ordinaire, qui n'est pas en mesure de changer le réseau, mais qui peut mesurer le niveau de qualité fournie par ses applications et adapter ces applications. L'autre perspective, discutée dans le second exemple, est celle du fournisseur de services. Au moyen des mécanismes QoS, on peut maîtriser et régler les conditions réseau, de façon à ce que la qualité au niveau de l'application pour les utilisateurs corresponde à ce qui est désiré.

La dépendance que nous avons mis en évidence entre les scores PESQ, la gigue et la perte de paquets peut être utilisée pour choisir le meilleur codeur pour le déploiement

d’une application VoIP. Pour un réseau donné on peut mesurer le taux moyen de perte et la gigue moyenne, puis utiliser les surfaces et contours PESQ comme guides. Si, par exemple, la gigue moyenne est 15 ms et le taux de perte 0,5%, tous les codeurs produisent une « bonne » qualité, mais seul G.711 offre le niveau PSTN. Si la gigue moyenne est 15 ms mais le taux de perte dépasse 3%, le codeur GSM ne peut plus être utilisé pour obtenir une « bonne » qualité (PESQ ≥ 3), mais les autres codeurs restent tous appropriés. Un cas extrême est une situation ayant un taux de perte de 10 à 15% et une gigue dans l’intervalle 40-50 ms ; dans ce cas le seul codeur qui peut être utilisé est G.729, qui offre une qualité acceptable malgré les conditions réseau défavorables. On peut envisager aussi la situation dans laquelle l’application même, de manière active et dynamique, change les codeurs utilisés, pendant l’exécution, à certains moments, afin de maximiser la qualité perçue.

Un autre exemple peut être donné dans une perspective de la QoS. Quand il est possible de contrôler la dégradation dans le réseau au niveau du fournisseur ou même du LAN, une telle approche peut être utilisée pour assurer qu’un certain niveau de qualité est fourni, et implicitement un certain niveau de la qualité perçue est atteint pour une application donnée. Dans le cas de la communication VoIP, nos résultats peuvent être utilisés pour décider quelles sont les limites dans lesquelles la gigue et la perte doivent être maintenues pour obtenir une « bonne » qualité. Par exemple pour le codeur G.729 ces frontières se situent aux environs de 35 ms pour la gigue et 4% pour le taux de perte.

6 Conclusions

Ce travail s'est efforcé de donner une vision globale sur la qualité dans les réseaux informatiques. Dans ce domaine aussi il arrive que les arbres cachent la forêt : l'attention doit se concentrer en premier sur la performance des applications réseau et utiliser dans un deuxième temps cette information pour entraîner les démarches sur le réseau même.

Nous avons eu l'inestimable chance que ce doctorat (en cotutelle à l'Université « POLITEHNICA » de Bucarest et l'Université « Jean Monnet » de St. Etienne) puisse se dérouler au CERN, le Laboratoire Européen pour la physique des particules et l'endroit où le Web a été inventé. Ce cadre et le groupe que m'a accueilli nous ont permis un travail de pointe dans le domaine des réseaux informatiques, à la fois avec des applications spécifiques et générales.

6.1 Résultats principaux

Les principaux résultats de notre recherche, d'abord par la mesure active et ensuite par la mesure passive de la qualité dans les réseaux informatiques, sont les suivants.

6.1.1 La différenciation de service

Nous avons mesuré les propriétés de différenciation de service pour une série de commutateurs. Nous en avons déduit qu'en pratique il y a un écart entre qualité observée et qualité attendue et que cet écart peut être important dans certains cas.

Dans la majorité de commutateurs testés, un trafic différencié partageant la même file d'attente n'est pas traité de manière équitable. Le trafic de haute priorité perd du débit en faveur du trafic de priorité plus basse dans certaines implémentations d'ordonnancement par priorité stricte que nous avons étudié. L'ordonnancement par l'algorithme du tourniquet fonctionne bien en général pour les flux ayant une même taille de paquets, mais est défaillante pour les flux avec tailles différentes des paquets, même si on utilise un ordonnancement plus sophistiqué, à base de quanta. L'utilisation d'équipements produits par des leaders du marché ne garantit donc pas obligatoirement le niveau de performance attendu.

Des tests exhaustifs sont indispensables pour valider les composants utilisés pour construire des réseaux, surtout dans des environnements de haut débit avec des exigences QoS strictes. Il faut aussi noter que pour les réseaux à haut débit aucun des mécanismes de différenciation courants n'offre pas des garanties statistiques et une prédictibilité satisfaisante dans toutes les circonstances. Dans ce contexte, la technologie GoS [U4EA], qui garantit des limites sur les valeurs moyennes des paramètres QoS, est une alternative aux mécanismes d'ordonnancement classiques.

6.1.2 Le transfert de fichiers

Ensuite nous avons montré, par des expériences détaillées et objectives sur le transfert de fichiers et la téléphonie IP, l'influence des conditions dans le réseau sur la qualité perçue au niveau des applications. Quantifier aussi clairement la relation entre les paramètres QoS et l'UPQ est un des aspects novateurs de notre travail.

Pour le transfert de fichiers, nous avons observé la diminution attendue du débit utile avec le taux de perte des paquets. La dépendance mesurée est linéaire et la diminution est seulement de 11% dans la gamme de taux de perte de 0 à 5%. Pour des taux de perte supérieurs à 20%, le débit utile indique une efficacité du transfert inférieure à 0,7.

La performance temporelle de transfert diminue exponentiellement, montrant que le temps de transfert augmente significativement avec la perte de paquets. Pour de taux de perte aux environs de 5% et de *RTT* petites, la valeur du *TTP* est dix fois plus petite que celle qu'on obtient à perte nulle. Pour des taux de perte de 25%, le temps de transfert devient des centaines de fois supérieur à celui qui est obtenu pour des taux de perte inférieurs à 5%.

En conclusion, une performance acceptable du transfert de fichiers nécessite un taux de perte inférieur à 5% (pour maintenir l'efficacité du réseau au dessus de 0,95 et ne pas avoir un temps de transfert plus grand que d'un ordre de magnitude par rapport au temps de transfert à perte nulle). Une bonne performance impose des limites encore plus strictes : le taux de perte ne doit pas dépasser 1% afin que l'efficacité d'utilisation du réseau avoisine 0,99 et le temps de transfert multiplié par quatre fois au maximum par rapport au temps à perte nulle.

6.1.3 La téléphonie IP

Pour VoIP nous avons étudié d'une manière objective la dépendance entre les conditions dans le réseau et la qualité perçue de la communication vocale. Une conclusion générale est que le codeur G.711 a une performance supérieure à celles des autres codeurs, tant que les conditions dans le réseau sont bonnes (taux de perte inférieur à 3% et gigue en dessous de 20 ms). Le codeur G.726 produit de meilleurs résultats que le codeur GSM dans toute la gamme de paramètres étudiés, au prix d'un taux de transmission 2,5 fois plus grand. Leur comportement sous l'influence de la perte et de la gigue est néanmoins similaire. Le codeur G.729 est le codeur le plus robuste dans la gamme de conditions réseau étudiée. Il fournit presque la même qualité perçue (toujours en dessus du seuil de qualité « basse ») dans 90% de l'espace perte-gigue étudié. Une diminution de seulement environ 1,5 sur l'échelle PESQ est constaté entre le point de perte et gigue nulles et celui de perte 15% et gigue 75 ms, une performance bien meilleure que celles des autres codeurs testés.

Les résultats obtenus sur les applications rendent possible la prédiction de l'UPQ d'une application en fonction des paramètres QoS mesurés et la compréhension des causes de défaillance des applications. On peut aussi déterminer les exigences QoS de bout en bout pour qu'une application tourne avec le niveau de UPQ désiré. Associer les exigences au niveau des utilisateurs avec les conditions QoS dans le réseau est un problème clé pour les contrats de prestation.

6.2 Contributions personnelles

L'objectif principal de notre travail est l'analyse de la qualité dans les réseaux informatiques dans une nouvelle perspective, analyse supportée par des mesures effectives réalisées avec des systèmes de test spécialisés.

Nous avons d'abord souligné l'existence de deux aspects de la qualité dans les réseaux informatiques, la qualité de service, QoS, et la qualité perçue par les utilisateurs, UPQ.

6.2.1 Contributions sur la QoS

Nous avons présenté une nouvelle perspective dans laquelle envisager la qualité de service, vue comme la fidélité du comportement mesurable d'un système réseau par rapport aux attentes des utilisateurs en terme de paramètres de performance.

Les trois paramètres de performance, le délai, le débit et la perte de paquets sont liés par une dépendance à deux degrés de liberté. Quantifier ces paramètres représente une mesure de la dégradation induite par le réseau. La qualité ne fait que décroître dans un réseau, tout au long du parcours. Dans notre vision, chaque élément du réseau induit un certain niveau de dégradation, dégradation qui s'accumule. La tâche des mécanismes QoS est d'assurer un partage contrôlé de cette dégradation.

Dans ce contexte, nous avons mis en évidence que la quantification des propriétés de différenciation de service passe nécessairement par une mesure active ; nous en avons défini le cadre et la méthodologie permettant de quantifier avec précision la différenciation de service pour des commutateurs FastEthernet et Gigabit Ethernet. Nous avons indiqué la nécessité d'effectuer les tests surtout en conditions de saturation, un état qui conduit en général à une dégradation importante de la qualité de service.

Nous avons montré qu'il existe toujours des inconvénients liés aux mécanismes QoS courants, avec une analyse plus détaillée sur l'ordonnancement par priorité stricte, SP, et par l'algorithme du tourniquet avec poids, WRR. Nous avons observé des écarts parfois considérables entre les comportements observés et attendus, du point de vue du débit et du délai, mais aussi du point de vue de l'équité de traitement entre les différents flux de trafic qui partagent la même file d'attente.

Nous avons contribué au développement des systèmes de mesure active, à la fois dans les étapes de conception et lors de l'implémentation et de la maintenance.

Pour le système de test Enet32, qui permet des mesures actives sur des commutateurs FastEthernet, nous avons utilisé le langage Handel-C pour étendre la fonctionnalité et améliorer le contrôleur d'accès au media, MAC, afin de permettre la transmission du trafic avec priorités en conformité avec le standard 802.1p/Q. Nous avons modifié le gestionnaire de transmission, TxMan, pour permettre la transmission en accord avec de motifs de trafic téléchargés sur le testeur depuis le PC hôte, ce qui permet de transmettre non seulement du trafic avec un débit constant, mais aussi de type poissonnien, par exemple. Nous avons implémenté le gestionnaire de réception, RxMan, notamment le traitement de l'information sur les paquets reçus et le calcul de résultats (moyennes, histogrammes etc.). Pour le même système Enet32, nous avons conçu et implémenté (dans le langage VHDL) une méthode de synchronisation de

plusieurs cartes de test par envoi de signaux rectangulaires d'horloge entre les cartes dans un paradigme maître-esclave.

Nous avons participé à la conception et implémenté un protocole de communication qui permet de diriger les cartes de test de type Enet32 ou Alteon depuis un système de contrôle centralisé, ce qui rend possibles les tests sur des systèmes réseau de large taille (jusqu'à 128 ports FastEthernet et 32 ports Gigabit Ethernet).

Nous avons aussi collaboré à la conception d'une version du testeur Enet32 qui peut générer du trafic spécifique aux applications qui s'exécutent par l'intermédiaire du réseau du système d'acquisition de données du niveau 2 de l'expérience ATLAS sur l'accélérateur de particules LHC au CERN.

6.2.2 Contributions sur l'UPQ

La qualité perçue par les utilisateurs n'a pas reçu l'attention méritée jusqu'à présent, même si elle est l'aspect le plus important de la qualité, du point de vue de l'utilité, de la valeur d'un réseau. Nous avons établi une méthodologie et contribué au développement d'un système de mesure qui permet de quantifier de manière précise la relation qui existe entre la QoS et l'UPQ pour les applications réseau. Le paradigme consiste dans une approche de mesure passive des paramètres de performance du réseau et une évaluation simultanée de métriques de la qualité perçue pour les applications qui s'exécutent à travers le réseau. Les conditions dans le réseau ont été contrôlées par l'intermédiaire d'un émulateur réseau.

Nous avons contribué à la conception du système de monitoring à base de cartes programmables Alteon qui permet ensuite le calcul de paramètres de performance pour le trafic des applications réelles. Nous avons développé un logiciel qui permet de calculer ces paramètres QoS à base de descripteurs collectés par les cartes Alteon, en conformité avec les standards IETF.

Les applications étudiées en détail sont le transfert de fichiers et la téléphonie IP. Nous avons développé des logiciels pour le calcul de paramètres UPQ pour les deux applications étudiées. Nous avons aussi développé les logiciels pour le traitement des résultats obtenus et pour la mise en évidence de la relation entre les mesures de QoS et celles de UPQ.

Pour le transfert de fichiers, en plus de l'utilisation d'un paramètre UPQ connu, le débit utile, nous avons défini un paramètre nouveau pour la mesure de la qualité perçue pour FTP, la performance temporelle du transfert, qui synthétise les aspects liés à la dépendance de la durée du transfert de fichier par FTP par rapport aux paramètres qui caractérisent la connexion, incluant les paramètres de performance du réseau, mais aussi ceux qui sont spécifiques au TCP, tel que le temps d'aller-retour ou la fenêtre du protocole.

Pour la téléphonie IP, VoIP, nous avons intégré dans notre système de test le calcul du score PESQ, une mesure de l'UPQ pour cette application,. Nous avons aussi conçu et implémenté un algorithme de restitution du son et compensation de gigue qui a été utilisé pour étudier le comportement de référence d'une application VoIP, Speak Freely. Nous avons modifié ce gratuiciel par l'ajout du codeur G.729 parmi ses fonctionnalités. Ce codeur manquant est très utilisé en pratique, donc d'une grande importance. Son étude nous a permis d'établir de manière objective qu'il s'agit réellement du codeur le plus robuste de ceux que nous avons étudiés.

Notre recherche démontre que la mesure précise de la qualité dans les réseaux informatiques, avec ses deux aspects, la qualité de service et la qualité perçue par l'utilisateur, est essentielle afin de pouvoir évaluer les effets que les conditions dans les réseaux ont sur les applications. Les résultats présentés tendent à établir qu'il est possible d'évaluer avec objectivité la qualité et qu'il existe une série de mécanismes qui peuvent être utilisés pour la contrôler. Il est aussi possible de définir avec exactitude les exigences des applications, à la fois en termes de qualité perçue et en termes de paramètres de performance du réseau. Nous pensons que dans ces conditions il est du devoir des utilisateurs d'exprimer leurs exigences auprès des fournisseurs de service et contribuer ainsi à l'amélioration de la qualité fournie par les réseaux.

6.3 Activité future

Dans notre groupe il y a des intérêts sur les approches analytiques et de simulation concernant le comportement de commutateurs, complémentaires à notre travail. Notre système de mesure de la QoS été déjà utilisé pour l'estimation, sur des commutateurs réels, des paramètres utilisés dans les modèles mathématiques (par exemple taille de mémoires tampon dans les ports d'entrée et de sortie). Nous pensons néanmoins

qu'une exploitation plus profonde de cette complémentarité est possible. On peut envisager une série de mesures effectuées à la fois sur de systèmes réels et sur des modèles mathématiques. Ceci permet de améliorer la précision de ces derniers pour ensuite pouvoir les utiliser pour extrapoler les résultats des mesures sur les systèmes réels à des réseaux plus complexes.

La méthodologie que nous avons développé pour la mesure conjointe de la QoS et UPQ est complète, mais il reste certains aspects qui n'ont pas été couverts par des expériences. Pour FTP notamment, nous avons analysé le comportement de base d'une implémentation standard sur le système d'exploitation Linux, dans sa configuration par défaut. Il est possible d'employer notre méthodologie et notre système de mesure pour étudier les effets d'autres configurations de la même implémentation, ou même d'autres implémentations de TCP, sur la performance du transfert de fichiers.

Certaines questions que nous considérons importantes n'ont pas pu recevoir un traitement complet dans notre étude sur la téléphonie IP. L'étude des améliorations qui apparaissent à l'utilisation des algorithmes plus complexes de restitution de son et compensation de gigue (de type dynamique) est une de ces questions. Les techniques de dissimulation de perte de paquets conduisent aussi aux améliorations qui pourront être évaluées. Un autre aspect non approfondi est celui des influences des corrélations, par exemple entre événements de perte de paquets. Une mesure objective de cet aspect, par exemple par les définitions de motifs de perte dans [RFC-3357] permettra l'évaluation de l'impact de ces motifs sur la qualité perçue pour VoIP. Finalement, trouver une façon de modélisation des surfaces PESQ de manière simple et intuitive fournira un outil précieux dans l'estimation des influences des paramètres de performance du réseau sur l'UPQ.

Nous sommes conscients qu'une étude exhaustive d'un domaine aussi large que celui que nous avons abordé dans les activités pratiques sur applications est irréalisable, et de toute façon nos expériences servent seulement à illustrer et démontrer nos propos théoriques sur la qualité. Nous considérons que par notre travail nous avons réussi de quantifier avec précision le comportement de base, sans négliger aucune des questions fondamentales.

Le travail effectué a d'ores et déjà des suites. L'expérience que nous avons eu avec l'émulateur NIST Net nous a conduit au projet de conception d'un émulateur réseau plus performant, appuyé sur des cartes réseau utilisant des FPGAs. L'architecture de ce nouvel émulateur est présentée sur la figure 145. Cet émulateur, sur lequel nous travaillons actuellement en collaboration avec U4EA Technologies et Predictable Network Solutions, va permettre une étude encore plus précise des applications réseau et de l'influence que les conditions dans le réseau ont sur elles.

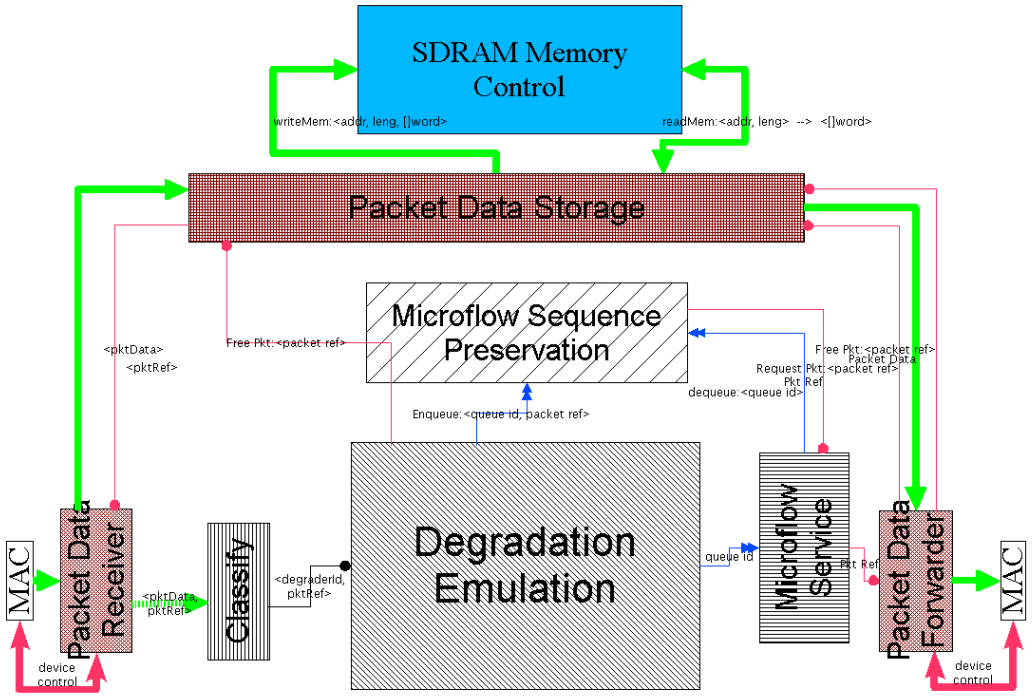


Figure 145 : Architecture d'un nouvel émulateur réseau.

Liste des figures

Liste des tableaux

Abréviations

ADSL	Asymetric Digital Subscriber Line
ATM	Asynchronous Transfer Mode
CBR	Constant Bit Rate
CGO	Classification Gestion Ordonnancement
CRC	Cyclic Redundancy Check
CV-ATM	Connexion Virtuelle ATM
DMA	Direct Memory Access
DWRR	Deficit Weighted Round Robin
FDC	Fonction de Distribution Cumulative
FIFO	First In First Out
FPGA	Field Programmable Gate Array
FQ	Fair Queueing
FTP	File Transfer Protocol
GoS	Guarantee of Service
GPS	Global Positioning System
HDTV	High Definition Television
HTTP	HyperText Transfer Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPDV	IP Packet Delay Variation (version UIT-T)
ipdv	IP Packet Delay Variation (version IETF)
IPER	IP Packet Error Ratio
IPLR	IP Packet Loss Ratio
IPPM	IP Performance Metrics
IPPT	IP Packet Throughput
IPTD	IP Transfer Delay
LAN	Local Area Network
MAC	Medium Access Control
MOS	Mean Opinion Score
MPEG	Moving Pictures Expert Group
MPLS	Multi-Protocol Label Switching

NIC	Network Interface Card
NTSC	National Television System Committee
PAL	Phase Alternation Line
PAMS	Perceptual Analysis/Measurement System
PCI	Peripheral Component Interconnect
PESQ	Perceptual Evaluation of Speech Quality
PQ	Priority Queueing
PSQM	Perceptual Speech Quality Mesurement
PSTN	Public Switch Telephone Network
QoS	Quality of Service
RFC	Request For Comments
RR	Round Robin
RSVP	Resource reSerVation Protocol
RTP	Real Time Protocol
RTT	Round Trip Time
SFP	Speak Freely Protocol
SIPR	Spurious IP Packet Rate
SLA	Service Level Agreement
SP	Strict Priority
TCP	Transmission Control Protocol
TTP	Transfer Time Performance
UDP	User Datagram Protocol
UIT-T	Union Internationale des Télécommunications (division de standardisation des télécommunications)
UPQ	User-Perceived Quality
VAD	Voice Activity Detection
VAT	Video Audio Tool protocol
VHS	Video Home System
VLAN	Virtual LAN
VoIP	Voice over IP
VTC	Video TeleConferencing
WAN	Wide Area Network
WFQ	Weighted Fair Queueing
WRR	Weighted Round Robin

Bibliographie

[All-90] A. Allen, “Probability, statistics, and queueing theory with computer science applications”, Academic Press, 1990.

[Alt-97] Alteon Networks, Inc.,“Tigon/PCI Ethernet Controller”, *révision 1.04*, août 1997.

[Alt-99] Alteon Networks, Inc., “Gigabit Ethernet/PCI Network Interface Card. Host/NIC Software Interface Definition”, *révision 12.4.13*, juillet 1999.

[Altera] Altera, Inc., http://www.altera.com.

[Ari-88] Aristote, “Etica nicomahică”, trad. par Stella Petecel, *Ed. Științifică și Enciclopedică*, Bucarest, 1988.

[Arl-96] M. F. Arlitt, C. L. Williamson, “Web Server Workload Characterizaition : The Search for Invariants”, Proc. *SIGMETRICS*, Philadelphia, PA, USA, avril 1996.

[Arm-00] G. Armitage, “Quality of Service in IP Networks”, *New Riders*, 2000.

[ATLAS] L’expérience ATLAS sur l’accélérateur de particules LHC au CERN, http://atlas.web.cern.ch/Atlas/Welcome.html

[Bar-01] F. R. M. Barnes, R. Beuran, R. W. Dobinson, M. J. LeVine, B. Martin, J. Lokier, C. Meiroșu, “Ethernet Networks for the ATLAS Data Collection System: Emulation and Testing”, *Proc. of the 12th IEEE Real Time Congress on Nuclear and Plasma Sciences*, Valencia, Espagne, juin 2001, pp. 6-10.

[Beu-01] R. Beuran, “Programarea FPGA-urilor în limbajul Handel-C”, *rapport de doctorat*, Université « POLITEHNICA » Bucarest, Roumanie, mai 2001.

[Beu-02] R. Beuran, “Network Quality of Service”, *rapport de doctorat*, Université « POLITEHNICA » Bucarest, Roumanie, mai 2002.

[Beu-03] R. Beuran, M. Ivanovici, B. Dobinson, N. Davies, P. Thompson, “Network Quality of Service Measurement System for Application Requirements Evaluation”, *Int. Symp. on Performance Evaluation of Computer and Telecommunication Systems (SPECTS'03)*, Montreal, Canada, 20-24 juillet, 2003, pp. 380-387.

[Beu-04a] R. Beuran, M. Ivanovici, “User-Perceived Quality Assessment for VoIP Applications”, *rapport technique délivré à U4EA Technologies*, CERN-OPEN-2004-007, CERN, Genève, Suisse, janvier 2004.

[Beu-04b] R. Beuran, M. Ivanovici, V. Buzuloiu, “File Transfer Performance Evaluation”, *accepté pour publication, Scientific Bulletin*, Université « POLITEHNICA » Bucarest, Roumanie, mai 2004.

[Beu-04c] R. Beuran, M. Ivanovici, N. Davies, B. Dobinson, “Evaluation of the Delivery QoS Characteristics of Gigabit Ethernet Switches”, *soumis pour publication, Fifth International Workshop on Quality of Future Internet Services – QofIS*, Barcelone, Espagne, septembre 2004.

[Bla-02] U. Black, S. Waters, “SONET and T1: Architectures for Digital Transport Networks”, *Prentice Hall*, 2002.

[Bou-02] C. Boutremans, G. Iannaccone, C. Diot, “Impact of link failures on VoIP performance”, *Proc. of the 12th Int. Workshop NOSSDAV*, Miami, Florida, USA, mai 2002, pp. 63-71.

[Bou-04] J.-Y. Le Boudec, “Performance Evaluation. Methods, Practice and Theory of the Performance Evaluation of Computer and Communication Systems”, *lecture notes*, Ecole Polytechnique Fédérale de Lausanne, Suisse, mars 2004.

[Bre-00] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, H. Yu, “Advances in Network Simulation”, *IEEE Computer*, 33 (5), pp. 59-67, mai 2000.

[Cam-01] M. Campanella, P. Chivalier, A. Sevasti, N. Simar, “Quality of Service Definition”, *SEQUIN Project*, mars 2001.

[Cam-02] M. Campanella, M. Carboni, P. Chivalier, S. Leinen, J. Rauschenbach, R. Sabatino, N. Simar, “Definition of Quality of Service testbed”, *SEQUIN Project*, avril 2002.

[Cav-02] V. Cavalli, E. Verharen, “TF-STREAM Real Time Multimedia Applications”, *TERENA Technical Report*, mars 2002.

[Celox] Celoxica, Ltd., http://www.celoxica.com.

[Cha-01] H. J. Chao, X. Guo, “Quality of Service in High-speed Networks”, *John Wiley & Sons, Inc.*, 2001.

[Cho-03] G. L. Choudhury, “Analysis of combined voice/data/video operation in cable and DSL access networks : graceful degradation under overload”, *Performance Evaluation, Elsevier Science B.V.*, vol. 52, 2003, pp. 89-103.

[Cis-01] Cisco Systems, “Les enjeux de la convergence voix, données, vidéo en France”, *livre blanc*, 2001.

[Cis-03] Cisco Systems, “Understanding Codecs: Complexity, Hardware Support, MOS, and Negotiation”, *note technique*, 2003.

[Cla-92] D. D. Clark, S. Shenker, L. Zhang, “Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism”, *Proc. ACM SIGCOMM'92*, août 1992.

[Coc-98] P. Cochrane, “Highlights of the 1998 3M lecture”, 1998, http://www.cochrane.ork.uk/opinion/papers/backto.htm.

[Con-02] A. E. Conway, Y. Zhu, “Analyzing Voice-over-IP Subjective Quality as a Function of Network QoS: A Simulation-Based Methodology and Tool”, *Lecture Notes in Computer Science 2324*, Springer-Verlag, 2002, pp. 289-308.

[Dav-99] N. Davies, J. Holyer, P. Thompson, “A Queueing Theory Model that Enables Control of Loss and Delay at a Network Switch” *Technical Report CSTR-99-011, Department of Computer Science, University of Bristol*, novembre 1999.

[Fal-96] K. Fall, S. Floyd, “Simulation-based Comparisons of Tahoe, Reno, and SACK TCP”, *Computer Communication Review*, 26(3), juilllet 1996, pp. 5-21.

[FAST-02] FAST Kernel for Large Data transfers, *California Institute of Technology*, novembre 2002, http://netlab.caltech.edu/FAST.

[Fau-01] F. Le Faucher et al., “MPLS Support of Differentiated Services”, *Internet Draft*, février 2001.

[Fer-00] T. Ferrari, S. Leinen, J. Novak, S. Nybroe, H. Prigent, V. Reijs, R. Sabatino, R. Stoy, “Report on Results of the Quantum Test Programme”, *Quantum Project*, juin 2000.

[Flo-95] S. Floyd, V. Jacobson, “Link-sharing and Resource Management Models for Packet Networks”, *IEEE/ACM Transactions on Networking*, vol. 3, no. 4, pp. 365-386, août 1995.

[Fol-85] H. J. Folse, “The Philosophy of Niels Bohr. The framework of Complementarity”, *Elsevier Science Publishers B.V.*, 1985.

[God-00] D. Goderis (editor), “Functional Architecture Definition and Top Level Design”, *TEQUILA Project*, septembre 2000.

[God-02] D. Goderis *et al.*, “Service Level Specification Semantics and Parameters”, *Internet Draft*, février 2002.

[Gri-00] D. Griffin (editor), “Selection of Simulators, Network Elements and Development Environment and Specification of Enhancements”, *TEQUILA Project*, mai 2000.

[Gup-01] P. Gupta, N. McKeown, “Algorithms for Packet Classification”, *IEEE Network*, mars/avril 2001, pp. 24-32.

[Gün-01] E. Gündüzhan, K. Momtahan, “A Linear Prediction Based Packet Loss Concealment Algorithm for PCM Coded Speech”, *IEEE Trans. On Speech and Audio Processing*, vol. 9, no. 8, novembre 2001, pp. 778-785.

[Hoa-88] C. A. Hoare (editor), “Occam 2 Reference Manual”, Prentice Hall International Series in Computer Science, Cambridge, 1988.

[Hun-02] R. Hunt, “A review of quality of service mechanisms in IP-based networks”, *Computer Communications*, no. 25, 2002, pp. 100-108.

[Int2] Internet2 End-to-End Performance Initiative, http://e2epi.internet2.edu/.

[ITA] The Internet Traffic Archive, http://ita.ee.lbl.gov/index.html.

[ITU-107] ITU-T Recommendation G.107, “The E-model, a computational model for use in transmission planning”, *UIT-T*, mai 2000.

[ITU-123] ITU-T Recommendation Y.123.qos, “A QoS Architecture for Ethernet-based IP Access Network”, *UIT-T*, février 2004.

[ITU-143] ITU-T Recommendation J.143, “User requirements for objective perceptual video quality measurements in digital cable television”, *UIT-T*, mai 2000.

[ITU-350] ITU-T Recommendation I.350, “General Aspects of Quality of Service and Network Performance in Digital Networks, including ISDNs”, *UIT-T*, mars 1993.

[ITU-380] ITU-T Recommendation I.380, “Internet Protocol (IP) Data Communication Service - IP Packet Transfer and Availability Performance Parameters”, *UIT-T*, février 1999.

[ITU-711] ITU-T Recommendation G.711, “Pulse Code Modulation (PCM) of voice frequencies”, *UIT-T*, 1993.

[ITU-726] ITU-T Recommendation G.726, “40, 32, 24, 16 kbit/s Adaptive Differential Pulse Code Modulation (ADPCM)”, *UIT-T*, 1990.

[ITU-729] ITU-T Recommendation G.729, “Coding of speech at 8 kbit/s using Conjugate-Structure Algebraic-Code-Excited Linear-Prediction (CS-ACELP)”, *UIT-T*, mars 1996.

[ITU-800] ITU-T Recommendation P.800, “Methods for subjective determination of transmission quality”, *UIT-T*, août 1996.

[ITU-861] ITU-T Recommendation P.861, “Objective quality measurement of telephone band (300-3400Hz) speech codecs”, *UIT-T*, février 1998.

[ITU-862] ITU-T Recommendation P.862, “Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrow-band telephone networks and codecs”, *UIT-T*, février 2001.

[ITU-1541] ITU-T Recommendation Y.1541, “Network Performance Objectives for IP-Based Services”, *UIT-T, draft*, octombre 2001.

[Jac-88] V. Jacobson, “Congestion Avoidance and Control”, ACM Computer Communication Review, *SIGCOMM’88 Symposium*, Stanford, CA, USA, août 1988.

[Jia-03] W. Jiang, H, Schulzrinne, “Assessment of VoIP Service Availability in the Current Internet”, *Passive and Active Measurement Workshop (PAM)*, La Jolla,California, USA, 6-8 avril, 2003, pp. 151-157.

[Kal-99] S. Kalidindi, M. Zekauskas, “Surveyor : AN Infrastructure for Internet Performance Measurements”, *NET’99*, San Jose, CA, USA, juin 1999.

[Kor-03] K. Korcyl, G. Sladowski, R. Beuran, R. W. Dobinson, C. Meiroşu, M. Ivanovici, M. L. Maia, “Network performance measurements as part of feasibility studies on moving part of the ATLAS Event Filter to off-site insitutes”, *Lecture Notes in Computer Science*, Springer-Verlag Heidelberg, vol. 2970, *Grid Computing: First European Across Grids Conference*, Santiago de Compostela, Spain, 13-14 février, 2003, pp. 206 - 213.

[Lel-94] W. E. Leland, M. S. Taqq, W. Willinger, D. V. Wilson, “On the Self-similar Nature of Ethernet Traffic”, *IEEE/ACM Transactions on Networking*, vol. 2, 1994, pp. 1-15.

[Lev-03] M. J. LeVine, S. Stancu, Ch. Haeberli, L. Tremblet, R. Beuran, C. Meiroşu, R.W. Dobinson, B. Martin, E. Knezo, H.-P. Beck, R. Hauser, D. Botterill, “Validation of the ATLAS Trigger/DAQ Network architecture using hardware data emulators”, *13th IEEE Nuclear & Plasma Sciences Society Real Time Conference*, Montreal, Canada, 18-23 mai 2003.

[Malden] Malden Electronics, Ltd., http://www.malden.co.uk.

[Man-01] D. Manikis (editor), “Overview of the TEQUILA Reference Testbeds”, *TEQUILA Project*, février 2001.

[Mar-02] A. Markopoulou, F. Tobagi, M. Karam, “Assessment of VoIP quality over Internet backbones”, *Proc. of IEEE Infocom*, New York, NY, USA, 23-27 juin, 2002, pp. 150-159.

[Mat-97] M. Mathis, J. Semske, J. Mahdavi, T. Ott, “The macroscopic behavior of the TCP congestion avoidance algorithm”, *Computer Communication Review*, 27(3), juillet 1997.

[Mier] Miercom, Inc., http://www.miercom.com.

[Mil-94] J. S. Mill, “Utilitarismul”, trad. par Valentin Mureșan, *Ed. Alternative*, Bucarest, 1994.

[Mir-02] D. Miras, “A Survey on Network QoS Needs of Advanced Internet Applications”, *Internet2 QoS Working Group*, décembre 2002.

[Mit-03] Mitel Networks Corporation, “IP Telephony Reliability and Availability”, *livre blanc*, avril 2003.

[Moo-98] S. B. Moon, J. Kurose, D. Towsley, “Packet Audio Playout Delay Adjustment: Performance Bounds and Algorithms”, *ACM/Springer Multimedia Systems*, vol. 6, janvier 1998, pp. 17-28.

[NIST] NIST Net network emulator, *National Institute of Standards and Technology*, http://snad.ncsl.nist.gov/itg/nistnet.

[Pad-98] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, “Modeling TCP throughput: a simple model and its empirical validation”, *Computer Communication Review*, 28(4), 1998, pp. 303-314.

[Pad-00] J. Padhye, V. Firoiu, D. F. Towsley, J. Kurose, “Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation”, *IEEE/ACM Transactions on Networking*, 8(2), avril 2000, pp.133-45.

[Pad-01] J. Padhye, S. Floyd, “Identifying the TCP Behavior of Web Servers”, *SIGCOMM 2001*, août 2001.

[PAMS] “PAMS – A Perceptual Analysis/Measurement System”, *Malden Electronics, Ltd.*, http://www.malden.co.uk/products/dsla/pams.htm.

[Par-93] A. K. Parekh, R. G. Gallager, “A generalized processor sharing approach to flow control in integrated services networks : the single node case”, *IEEE/ACM Transactions on Networks*, vol. 1, no. 3, juin 1993, pp. 344-357.

[Par-94] A. K. Parekh, R. G. Gallager, “A generalized processor sharing approach to flow control in integrated services networks : the multiple node case”, *IEEE/ACM Transactions on Networks*, vol. 2, no. 2, avril 1994, pp. 137-150.

[Pet-99] L. Peterson, B. Davie, “Computer Networks: A System Approach”, *Morgan Kaufmann*, San Francisco, 1999.

[Pir-84] R. Pirsig, “Zen and the Art of Motorcycle Maintenance : An Inquiry into Values”, *Bantam Books*, mars 1984.

[Pir-94] R. Pirsig, “Subjects, Objects, Data and Values”, *Einstein meets Magritte conference*, Brussels, Belgique, 29 mai – 3 juin 1994.

[PNSol] Predictable Network Solutions, Inc., http://www.pnsol.com.

[QBone] Qbone, *The QoS project of the Internet2 consortium*, http://qbone.internet2.edu.

[Rah-93] M. Rahnema, "Overview of the GSM system and protocol architecture", *IEEE Communications Magazine*, avril 1993.

[Rai-00] V. Raisanen, G. Grotefeld, "Network performance measurement for periodic streams", *Internet draft*, mars 2000.

[Ree-03] D. Reeve, "A New Blueprint For Network QoS", *thèse de doctorat*, University of Kent at Canterbury, août 2003.

[Reijs] V. Reijs, "Perceived Quantitative Quality of Applications", http://www.heanet.ie/Heanet/projects/nat_infrastruct/perceived.html.

[RFC-1242] S. Bradner, "Benchmarking Terminology for Network Interconnection Devices", *IETF RFC 1242*, juillet 1991.

[RFC-1349] P. Almquist, "Type of Service in the Internet Protocol Suite", *IETF RFC 1349*, juillet 1992.

[RFC-1812] F. Baker, "Requirements for IP Version 4 Routers", *IETF RFC 1812*, juin 1995.

[RFC-1944] S. Bradner, J. McQuaid, "Benchmarking Methodology for Network Interconnect Devices", *IETF RFC 1944*, mars 1999.

[RFC-2105] Y. Rehter, B. Davie, D. Katz, E. Rosen, G. Swallow, "Cisco Systems' Tag Switching Architecture Overview", *IETF RFC 2105*, Février 1997.

[RFC-2178] J. Moy, "OSPF Version 2", *IETF RFC 2178*, juillet 1997.

[RFC-2205] R. Braden, L. Zhang, S. Berson, A. Herzog, S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification", *IETF RFC 2205*, septembre 1997.

[RFC-2210] J. Wroclawski, "The use of RSVP with IETF integrated Services", *IETF RFC 2210*, septembre 1997.

[RFC-2211] J. Wroclawski, "Specification of the Controlled-Load Network Element Service", *IETF RFC 2211*, septembre 1997.

[RFC-2212] S. Shenker, C. Partridge, R. Guerin, "Specification of Guaranteed Quality of Service", *IETF RFC 2212*, septembre 1997.

[RFC-2285] R. Mandeville, "Benchmarking Terminology for LAN Switching Devices", *IETF RFC 2285*, février 1998.

[RFC-2309] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, L. Zhang, "Recommendations on Queue Management and Congestion Avoidance in the Internet", *IETF RFC 2309*, avril 1998.

[RFC-2330] V. Paxson, G. Almes, J. Mahdavi, M. Mathis, "Framework for IP Performance Metrics", *IETF RFC 2330*, mai 1998.

[RFC-2474] K. Nichols, S. Blake, F. Baker, D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", *IETF RFC 2474*, décembre 1998.

[RFC-2475] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, “An Architecture for Differentiated Services”, *IETF RFC 2475*, décembre 1998.

[RFC-2597] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski, “Assured Forwarding PHB Group”, *IETF RFC 2597*, juin 1999.

[RFC-2598] V. Jacobson, K. Nichols, K. Poduri, “An Expedited Forwarding PHB”, *IETF RFC 2598*, juin 1999.

[RFC-2638] K. Nichols, V. Jacobson, L. Zhang, “A Two-bit Differentiated Services Architecture for the Internet”, *IETF RFC 2638*, juillet 1999.

[RFC-2678] J. Mahdavi, V. Paxson, “IPPM Metrics for Measuring Connectivity”, *IETF RFC 2678*, septembre 1999.

[RFC-2679] G. Almes, S. Kalidindi, M. Zekauskas, “A One-way Delay Metric for IPPM”, *IETF RFC 2679*, septembre 1999.

[RFC-2680] G. Almes, S. Kalidindi, M. Zekauskas, “A One-way Packet Loss Metric for IPPM”, *IETF RFC 2680*, septembre 1999.

[RFC-2681] G. Almes, S. Kalidindi, M. Zekauskas, “A Round-trip Delay Metric for IPPM”, *IETF RFC 2681*, septembre 1999.

[RFC-2990] G. Huston, “Next Steps for the IP QoS Architecture”, *IETF RFC 2990*, novembre 2000.

[RFC-2889] R. Mandeville, J. Perser, “Benchmarking Methodology for LAN Switching Devices”, *IETF RFC 2998*, août 2000.

[RFC-2998] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, E. Felstaine, “A Framework for Integrated Services Operation Over DiffServ Networks”, *IETF RFC 2998*, novembre 2000.

[RFC-3031] E. Rosen, A. Viswanathan, R. Callon, “Multiprotocol Label Switching Architecture”, *IETF RFC 3031*, janvier 2001.

[RFC-3036] L. Andersson, P. Doolan, N. Feldman, A. Fredette, B. Thomas, “LDP Specification”, *IETF RFC 3036*, janvier 2001.

[RFC-3148] M. Mathis, M. Allman, “A Framework for Defining Empirical Bulk Transfer Capacity Metrics”, *IETF RFC 3148*, juillet 2001.

[RFC-3357] R. Koodli, R. Ravikanth, “One-way Loss Pattern Sample Metrics”, *IETF RFC 3357*, août 2002.

[RFC-3393] C. Demichelis, P. Chimento, “IP Packet Delay Variation Metric for IP Performance Metrics (IPPM)”, *IETF RFC 3393*, novembre 2002.

[RFC-3551] H. Schulzrinne, S. Casner, “RTP Profile for Audio and Video Conferences with Minimal Control”, *IETF RFC 3551*, juillet 2003.

[RIPE] Réseaux IP Européens (RIPE) Test Traffic Measurements Service (TTM), http://www.ripe.net/ttm.

[Ser-01] V. Servis, “Measuring speech quality over VoIP networks”, *The TOLLY Group*, décembre 2001.

[Surv] Surveyor Project, Advanced Network & Services, http://www.advanced.org/surveyor.

[Tan-97] A. S. Tanenbaum, “Computer Networks”, *Prentice Hall*, 3ème édition, 1996.

[Tei-99] B. Teitelbaum (editor), “QBone Architecture”, *Internet2 QoS Working Group*, août 1999.

[Tirum] A. Tirumala, F. Qin, J. Dugan, J. Ferguson, K. Gibbs, “Iperf”, http://dast.nlanr.net/Projects/Iperf.

[Tolly] The Tolly Group, http://www.tolly.com.

[TFT] TF-TANT, *The Joint DANTE / TERENA Task Force*, http://www.dante.net/tf-tant.

[U4EA] U4EA Technologies, Ltd., http://www.u4eatech.com.

[Wah-00] B. W. Wah, X. Su, D. Lin, “A survey of error-concealment schemes for real-time audio and video transmissions over the Internet”, *Proc. IEEE Int. Symp. Multimedia Software Engineering*, Taipei, Taiwan, décembre 2000, pp. 17-24.

[Wan-01] Z. Wang, “Internet Quality of Service: Architectures and Mechanisms”, *Morgan Kaufmann*, 2001.

[Whi-02] W. Whitt, “Stochastic-Process Limits”, Springer Verlag, January 2002.

[Wiles] B. C. Wiles, J. Walker, “Speak Freely”, http://www.speakfreely.org.

[Wro-01] J. Wroclawski, A. Charny, “Integrated Service Mappings for Differentiated Services Networks”, *Internet Draft*, 2001.