Qos, Classification et Contrôle d'admission des flux TCP

Rana Khanafer

École Doctorale
d'Informatique,
Télécommunications
et Électronique de Paris

Thèse

présentée pour obtenir le grade de Docteur

de l'École Nationale Supérieure des Télécommunications

Spécialité : **Informatique et Réseaux**

Rana Khanafer

QoS, Classification et Contrôle d'admission des flux TCP

Soutenue le 29 Mars 2005 devant le jury composé de :

Gérard Hébuterne	**Président**
André-Luc Beylot	**Rapporteurs**
Ken Chen	
Tijani Chahed	**Examinateurs**
Gabriel Dib	
Daniel Kofman	**Directeurs de thèse**
Samir Tohmé	

Remerciements

Cette thèse au long cours s'achève et je tiens ici à remercier toutes les personnes qui m'ont accompagné, soutenu et bien souvent donné d'elles-mêmes durant toutes ces années.

Je suis reconnaissante envers Monsieur Michel Riguidel, directeur du département INFRES, de m'avoir accueilli au sein du département et permis l'achèvement de cette thèse dans les meilleures conditions possibles.

Je ne saurais assez dire ma profonde gratitude envers Professeur Daniel Kofman et Professeur Samir Tohmé auxquels le bon aboutissement de cette thèse doit beaucoup. Ils furent tout d'abord des encadrants qui m'ont marqué par leur ouverture d'esprit, leur enthousiasme et leur compétence, alliant curiosité et rigueur scientifique. Je les remercie également pour leurs qualités humaines certaines, leur disponibilité et leur écoute qui ne se sont jamais démenties, et pour toute l'énergie consacrée.

J'aimerais ensuite remercier Professeur Gérard Hébuterne qui m'a fait l'honneur de présider le jury de ma soutenance.

Je remercie chaleureusement Professeur Ken Chen et Professeur André-Luc Beylot d'avoir accepté la lourde tâche de rapporter ce mémoire. La pertinence de leurs remarques m'a permis d'en améliorer la clarté. Mes remerciements vont également aux examinateurs qui ont eu l'amabilité d'examiner ma thèse. Je pense, plus particulièrement à Tijani Chahed et Gabriel Dib.

Je remercie Professeur Catherine Rosenberg qui m'a proposé de travailler sur le sujet qui a donné lieu au chapitre 3 de cette thèse.

Bien d'autres noms mériteraient d'être cités, notamment les membres du département INFRES. Je pense particulièrement à Olivier Hudry qui a toujours aimablement répondu à mes questions mathématiques.

Je garde un souvenir chaleureux de l'équipe de thésards (qui sont devenus Docteurs) qui m'a accueilli à l'ère primaire de ma thèse. Je remercie particulièrement Hasard pour les diverses discussions scientifiques enrichissantes et Mohamad, Ibrahim, Jacques, Dani, Rani, Rola et Ouahiba pour leur amitié, leur écoute et leurs encouragements durant la période de thèse.

Et puis encore, pour leur amitié, leurs sourires, et nos souvenirs partagés, merci à Myriam, Xiaoyun, mes collègues de bureau qui m'ont bien supporté surtout durant la période de rédaction. Je pense aussi à Syed, Céline, Vincent et tous les doctorants qui ont vécu avec moi les divers moments tristes et joyeux de ma vie de doctorante. Je pense tout particulièrement à Kinda .

Un grand merci à Ejab, Rola, Darina, Samar, Reina, Nada et Lama pour leur soutien moral et leurs encouragements tout au long de cette thèse. Je leur remercie d'avoir su créer autour de moi une bonne ambiance de détente.

Enfin, je songe à la sphère familiale qui, malgré la distance (ou peut-être à cause d'elle), m'a beaucoup soutenu. J'adresse mes profonds remerciements à mes parents, mes deux soeurs, mon frère, mes nièces et mon neveu pour leur amour, leurs encouragements et leur soutien constant.

Un grand Merci à mon mari. Je ne sais pas si la consistance d'un merci tient encore face à tes efforts conjugués pour réaliser cette thèse. Sache que ta présence a été un soutien bien plus puissant que tu ne l'imagines.

Mon frère Rami et mon oncle Abed-el-Nabi ; bien au-delà des distances qui nous séparent, j'aimerais pouvoir vous dire que votre présence me manque plus que jamais. Vous n'êtes plus de ce monde, mais je vois toujours vos beaux sourires et vos yeux si doux ...A vous, je dédie cette thèse.

Merci aussi à tous ceux que j'ai peut être oublié ...

"Les savants des temps passés et des nations révolues n'ont cessé de composer des livres. Ils l'ont fait pour léguer leur savoir à ceux qui les suivent. Ainsi demeurera vive la quête de la vérité"

Al-Khawarizmi

Résumé

La simplicité et l'ubiquité ont permis au réseau Internet d'évoluer très rapidement. L'Internet connecte des centaines de milliers d'utilisateurs à travers le monde. Ce réseau est dit multiservice : il a vocation à transporter un grand nombre de types de service possédant des caractéristiques différenciées et éventuellement des contraintes de Qualité De Service (QoS) différenciées. De nombreux développements sont actuellement en cours, particulièrement sur la gestion de la QoS et sur l'intégration de ces différents services. Les aspects portant sur l'amélioration des performances des flux élastiques ont été quelque peu négligés par la communauté scientifique. Nous tenons à signaler que le trafic TCP constitue plus de 90% du volume du trafic Internet 'en octets' et plus de 80% des flux. La grande majorité des flux élastiques sont des flux de petit volume (appelés courts ou 'souris') tandis qu'une faible minorité des flux sont de volume très important (appelés longs ou 'éléphants'). La plus grande partie du volume de trafic est engendrée par cette minorité de flux. Ainsi, le manque d'une architecture de réseau qui assure une bonne QoS à ce type de trafic freine l'évolution des applications élastiques qui sont en perpétuelle évolution.

Nos travaux portent sur l'évaluation et l'amélioration des performances des applications élastiques (nous ne nous intéressons pas ici aux applications streaming qui restent encore minoritaires en nombre et en volume de ressources consommées). Plus exactement, nos études mettent en avant l'importance d'assurer une bonne qualité de service au trafic élastique. Nous proposons une évolution possible de l'Internet vers une nouvelle architecture plus fiable avec garantie de la qualité de service. Une caractérisation à l'échelle de flux du trafic Internet, nous conduit à proposer une architecture dite orientée flux. Cette architecture est basée sur **la classification, le contrôle d'admission et le traitement préférentiel** appliqués sur les deux types de flux TCP : courts et longs.

La classification des flux a été étudiée dans deux cas de figures : premièrement, entre les flux TCP longs qui ont des RTT différents et deuxièmement entre les flux TCP courts et longs.
Dans un premier temps, nous montrons, en se basant sur une étude théorique et sur des simulations, l'avantage de séparer les flux TCP longs selon le RTT. Dans le cas de deux types de sources, l'avantage se caractérise, d'une part, par un débit plus important pour les connexions générées par la source avec un RTT plus large, et d'autre part, par un système plus prévisible puisque les flux d'un agrégat parviennent à partager équitablement la bande passante qui leur est allouée. Dans le cas

de plusieurs sources, la séparation des flux en deux classes nous permet d'attribuer une bande passante plus petite pour garantir un débit minimal par flux.
Dans un deuxième temps, nous montrons en se basant sur une étude analytique et sur des simulations l'avantage d'allouer une capacité minimale pour les agrégats des flux courts et longs. La séparation des flux obtenue grâce à la politique WFQ permet de protéger les flux courts et d'améliorer leurs performances. En particulier, le fait de diminuer à l'intérieur d'un agrégat la variance de la taille des flux rend le trafic plus prévisible et plus facile à dimensionner.

Le contrôle d'admission est proposé pour les flux TCP. Il prend en compte la caractérisation en flux longs et flux courts ainsi que les contraintes de QoS propres à chaque type de flux. Dans un premier temps, nous nous intéressons au cas où les deux classes de flux sont multiplexées à l'aveugle. Dans un deuxième temps, nous étudions l'intérêt de garantir une bande passante minimale pour les agrégats des flux longs et courts. Afin d'éviter des gaspillages de ressources, dans ce dernier cas, le partage de la bande passante entre agrégats se fait par un mécanisme du type WFQ : si un agrégat n'utilise pas la bande passante minimale qui lui est attribuée, la bande passante résiduelle peut être utilisée par l'autre agrégat.
Un modèle analytique ainsi que des simulations ont été réalisés afin d'évaluer les performances des mécanismes proposés et d'analyser l'impact des seuils d'admission. Nous avons également analysé l'intérêt de faire attendre les flux qui ne peuvent pas être acceptés immédiatement. Les avantages d'utiliser la phase d'attente est d'un côté de mieux utiliser les ressources du réseau, et d'un autre côté, de diminuer le taux de rejet des flux.
Il est important de signaler que, outre l'amélioration des performances dans les cas étudiés, l'approche proposée fourni un outil de dimensionnement du réseau permettant d'atteindre, pour une structure de trafic donnée, les mesures de performances attendues.
Dans l'architecture proposée, nous nous sommes basés sur le type d'application pour reconnaître la nature du flux et ensuite le classifier. Cette méthode de classification peut, quelquefois, introduire des erreurs quant au type de flux. Ceci nous a poussé à proposer une nouvelle architecture basée sur le traitement préférentiel.

Le traitement préférentiel est appliqué aux premiers paquets de chaque connexion, favorisant ainsi les connexions courtes. Un flux arrivant au réseau sera classé comme court. Une fois il dépasse un certain seuil, le flux sera basculé dans une deuxième classe. Nous comptons également sur l'architecture DiffServ proposée dans [BER99] pour classifier les flux aux bordures d'un réseau. Plus spécifiquement, nous maintenons la longueur (en paquets) de chaque flux actif aux routeurs de bordures et l'employons pour classifier les paquets entrants. Cette architecture a la particularité de ne pas nécessiter le maintien en mémoire d'un état par flux au coeur du réseau. Dans ce dernier, nous utilisons la politique de gestion de file d'attente RED [FLO93] avec des seuils différents pour les deux types de classes. Ceci nous permet de réduire le taux de pertes éprouvé par les paquets des flux courts.
Nous montrons, à travers des analyses et des simulations, que le délai de transfert

résultant pour les connexions de courte durée est réduit sans pour autant pénaliser les performances des autres connexions. D'ailleurs, notre méthode de classification traite les premiers paquets d'un flux long comme ceux d'un flux court. Pour cela, les premiers paquets de chaque flux éprouvent moins de pertes (y compris les paquets d'ouverture de connexion qui, autrement, causeraient des retards inutiles pour les sessions). Nous confirmons également que notre modèle peut réaliser une meilleure équité et un temps de réponse plus petit pour les flux courts que les modèles sans traitement préférentiel.
Notre but est de fournir un nouveau service meilleur que le service best effort qui tente à améliorer les performances des flux TCP courts et longs. Un tel service crée alternativement un environnement plus équitable et utilise mieux les ressources du réseau, particulièrement pour le trafic Web qui reste dominant dans le réseau Internet actuel.

Abstract

The carried out work concerns the evaluation and the improvement of the performance of elastic flows. More exactly, our studies put the accent on the importance of ensuring a good quality of service to this type of traffic. Two architectures of QoS were proposed.
The first architecture is based on the classification and admission control applied to the two types of TCP flows : short flows and long flows. The classification of flows enables us to have a system more predictable and easier to dimension since flows of an aggregate manage to share fairly the bandwidth which is allocated to them within the same class. The admission control takes into account the characterization into long flows and short flows as well as the specific QoS constraints of each type of flow. Simulations were carried out and corroborated by an analytical model in order to evaluate the advantages of the suggested architecture and to analyze the impact of the admission thresholds.
The second architecture is based on preferential treatment that was proposed for short connections inside the bottleneck queue, so that short connections experience less packet drop rate than long connections. This was achieved by employing the RED queue management policy which uses different drop functions for different classes of traffic. We also rely on the Differentiated Services (Diffserv) architecture to classify flows into short and long flows at the edge of the network. More specifically, we maintain the length of each active flow (in packets) at the edge routers and use it to classify incoming packets. Through extensive simulations, we confirmed that preferential treatment is necessary to guarantee prompt response to short TCP flows. Moreover, since our threshold-based classification method treats the initial packets from a long flow as packets from a short flow, all flows benefit from experiencing fewer drops of the first packets. We also confirmed that our proposed scheme can achieve a better fairness and response time for short flows than schemes without preferential treatment.

Table des matières

Table des figures

Liste des tableaux

Introduction

Nos travaux portent sur l'évaluation et l'amélioration des performances des applications élastiques (nous ne nous intéressons pas ici aux applications streaming qui restent encore minoritaires en nombre et en volume de ressources consommées). Plus exactement, nos études mettent en avant l'importance d'assurer une bonne qualité de service au trafic élastique. Nous proposons une évolution possible de l'Internet vers une nouvelle architecture plus fiable avec garantie de la qualité de service. Une caractérisation à l'échelle de flux du trafic Internet, nous conduit à proposer une architecture dite orientée flux. Cette architecture est basée sur **la classification, le contrôle d'admission et le traitement préférentiel** appliqués sur les deux types de flux TCP : courts et longs. Le dernier paragraphe de ce chapitre présente le contenu de la thèse.

Etat de l'art

Ces 20 dernières années ont vu émerger de nouvelles techniques rendant possible l'interconnexion de réseaux différents (internetworking) en les faisant apparaître comme un unique environnement de communication homogène. On désigne ce système d'interconnexion sous le nom d'Internet.

Un réseau de type Internet est dit multiservice : il a vocation à transporter un grand nombre de types de service possédant des caractéristiques différents et éventuellement des contraintes de Qualité de Service (QoS) différenciées. Si la gamme des applications et leur importance sont en perpétuel changement, il est néanmoins toujours possible de répartir le trafic en deux grandes classes : élastique et streaming. Le trafic dit "élastique", ainsi nommé car son débit peut s'adapter à des contraintes extérieures (bande passante insuffisante par exemple) sans pour autant remettre en cause la viabilité du service. Cette classe de trafic est essentiellement engendrée par le transfert d'objets numériques par nature tels que des pages Web (application HTTP), des messages électroniques (e-mail, application SMTP) ou des fichiers de données (application FTP). Le respect de leur intégrité sémantique est indispensable mais les contraintes de délai de transfert sont moins fortes. Cette intégrité sémantique est la plupart du temps assurée par le protocole de transport (TCP) et ne constitue donc pas un élément de performance sur lequel l'opérateur de réseau puisse agir ; en revanche, le maintien d'un certain débit effectif minimum de transfert

des documents est un objectif de QoS. Le trafic de type élastique est actuellement largement majoritaire sur les réseaux IP : on constate couramment des proportions supérieures à 90% en volume (octets) et à 80% en nombre de flux TCP, protocole sous lequel fonctionnent la plupart des applications mentionnées ci-dessus.
Le trafic de type "streaming" est engendré par des applications audio et vidéo dont la durée et le débit ont une réalité intrinsèque bien que variable éventuellement. Le délai de transfert des données de même que sa variation, la gigue, doivent être contrôlables, tandis qu'un certain degré de perte de paquets peut être tolérable. Toutes ces applications s'accommodent mal au modèle "best effort", qui est le seul service offert dans le réseau Internet.

Jusqu'à maintenant, l'Internet était basé sur le best effort, donc le réseau n'offrait aucune garantie sur les paquets. La décentralisation de l'administration, l'émergence de communautés d'utilisateurs avec des exigences différentes en termes de qualité, de sécurité, de fiabilité, . . .et l'introduction des nouvelles applications exigent à présent des nouveaux services qui garantissent la QoS de chaque application.

Deux approches ont été envisagées pour offrir une alternative au modèle de service "best effort" dans les réseaux IP : IntServ et DiffServ.

L'approche Integrated Services (IntServ) représente la première tentative par l'IETF pour améliorer la qualité de service dans l'Internet [RFC2215]. La caractérisation principale de IntServ est de garantir les niveaux de QoS par un mécanisme de réservation de ressources par flux. Deux nouvelles classes de services autre que le Best Effort ont été proposées :
- Service à QoS garantie "Guaranteed Service (GS)" : Elle offre une garantie de délai de bout en bout. Cette classe est destinée aux applications temps-réel qui sont sensibles au délai.
- Service à charge contrôlée "Controlled Load Service (CLS)" : Elle offre aux applications une QoS meilleure que celle offerte par le service "Best Effort". Elle présente une approximation du comportement offert par le service "Best Effort" quand le réseau n'est pas chargé. En effet, dans de telles conditions, un très grand pourcentage de paquets transmis sera bien reçu et les paquets subiront un délai minime qui est celui du délai de transit minimal. Le "CLS" est destiné aux applications qui s'adaptent au délai.

Intserv est basé sur la réservation explicite des ressources requises par les flux dont les paramètres de trafic sont signalés par le protocole RSVP (Ressource ReSerVation Protocol) [RFC2205].
L'approche IntServ (Integrated Services) est trop complexe. Elle présente des problèmes de déploiement à large échelle, et nécessite une réservation de ressources pour chaque flot dans les différents noeuds du réseau. Si les fonctions de signalisation, de réservation et de gestion se réalisent facilement sur de petits réseaux, leur complexité devient vite rebutante au niveau d'un grand réseau.

L'approche DiffServ (Differented Services) [RFC2475] présente des mécanismes plus

souples et plus efficaces pour la gestion des ressources dans le réseau. Elle effectue une agrégation de plusieurs flux dans quelques classes de services. Dans ce modèle de service, les paquets sont classés et marqués aux frontières des réseaux pour recevoir un comportement particulier dans chaque noeud le long de leur trajet.
L'approche Diffserv propose donc d'abandonner le traitement du trafic sous forme de flux pour le caractériser sous forme de classes. Chaque classe est identifiée par une valeur codée dans l'en-tête IP. Cette classification doit se faire au niveau des routeurs de bordures (edge router) à l'entrée du réseau. L'architecture des services différenciés proposée dans le [RFC2475] contient deux types d'éléments fonctionnels :

1. Les éléments de bordures (edge functions) : ils sont responsables de la classification des paquets et du conditionnement du trafic.
2. Les éléments du coeur du réseau (core functions) : ils sont responsables de l'envoi des paquets uniquement.

Le service de meilleur qualité pouvant être offert par un PHB qui porte le nom de **expedited fowarding** ou premium service ; il a pour but de garantir une bande passante avec des taux de perte, de délai et de gigue faible. Ensuite, un ensemble de PHB a été regroupé sous le nom d'**assured forwarding**. Cette famille de PHB est scindée en 4 classes garantissant de fournir une bande passante et un délai minimum. L'avantage de Diffserv est qu'il n'y a plus nécessité de maintenir un état des sources et des destinations dans les routeurs au coeur du réseau, d'où une meilleure scalability. Mais il présente en pratique plusieurs incertitudes quant à la manière d'utiliser le marquage et le traitement par agrégats pour offrir des garantis de qualité de service précises. Un des problèmes de l'architecture DiffServ est l'absence de contrôle des chemins empruntés par les flux, ce qui rend difficile la réalisation effective de réservations. L'introduction de MPLS remédie à ce problème, en fixant les chemins des LSP (Label Switched Path). Cependant, la possibilité de souscrire des contrats, où seul le trafic global entrant ou sortant d'un site est spécifié, complique davantage le problème d'allocation des ressources.

L'absence de consensus sur le choix de modèle de service à QoS a poussé les responsables réseaux de remédier les problèmes de bande passante en sur-dimensionnant le réseau.

L'objectif du surdimensionnement est d'assurer une capacité de réseau suffisante pour écouler le trafic offert avec une bonne qualité de service. Outre le fait que le surdimensionnement peut parfois s'avérer coûteux, il n'offre qu'une solution partielle dans la mesure où il ne permet pas de préserver les performances du réseau en cas d'erreur de prévision ou de panne, par exemple. Des mécanismes supplémentaires pour contrôler les surcharges occasionnelles sont nécessaires.

Afin d'éviter des dégradations importantes de performance en cas de surcharge, c'est à dire un débit quasiment nul pour les transferts en cours et une énorme perte d'efficacité due aux abondons, il est nécessaire de mettre en oeuvre un contrôle préventif : refuser de nouveaux flux pour préserver la qualité des flux en cours. C'est le principe

de contrôle d'admission.
Notons que la motivation évoquée ici pour le contrôle d'admission est différente de celle habituellement évoquée qui s'intéresse uniquement aux flux streaming. Au contraire, dans notre travail, nous nous intéressons aux flux élastiques qui représentent plus de 90% du volume du trafic Internet en Octets et plus de 80% des flux. Les flux UDP restent encore minoritaires en nombre et en volume de ressources consommées.
De nombreuses méthodes de contrôle d'admission ont été étudiées dans ce contexte. Elles s'appuient habituellement sur la signalisation et la réservation plus ou moins explicite des ressources comme dans le modèle IntServ. Ces méthodes s'avèrent inadaptées au contrôle d'admission des flux élastiques car ces derniers sont très nombreux et en majorité très courts. Il semble nécessaire de mettre en oeuvre un contrôle 'implicite' fonctionnant sans signalisation et sans réservation de ressources. Ceci évite les problèmes d'extensibilité caractéristique de IntServ et de délai d'établissement nécessaire pour un échange de signalisation.
Le contrôle d'admission envisagé s'effectue au niveau flux. Nous définissons un flux comme une connexion TCP. Il prend en compte la caractérisation en flux courts et flux longs ainsi que les contraintes de QoS propres à chaque type de flux. Il consiste à limiter le nombre de flux courts et le nombre de flux longs admis simultanément. Nous définissons donc deux seuils qui représentent les maximums des flux courts et longs qui peuvent être admis. L'application d'un contrôle d'admission avec des seuils différents permettent de diminuer le taux de blocage des flux courts qui sont majoritaires mais ils n'introduisent que 15 à 20% de la charge totale. Le contrôle d'admission est basé sur l'identification au vol des flux. Si le flux est accepté/refusé, ses paquets sont transmis/retardés. Nous évaluons l'intérêt de garder des flux en attente au lieu de les rejeter directement. Cette méthode nous permet d'un côté de mieux utiliser les ressources du réseau, et d'un autre côté, de diminuer le taux de rejet des flux.

En complément du contrôle d'admission, il est aussi intéressant de classifier les flux TCP ayant des caractéristiques différentes. Nous pouvons distinguer les flux TCP avec des RTT ou tailles différents. Ces flux ne parviennent pas à se partager équitablement la bande passante. Les flux de petites tailles ou de large RTT sont défavorisés. De plus, les flux courts et longs ne présentent pas les mêmes besoins de QoS. Les premiers sont plus sensibles au temps de séjour tandis que les derniers sont plus sensibles au débit. Ainsi, une classification entre les flux courts et longs permet d'assurer les contraintes de QoS propres à chaque type de flux et de dimensionner plus facilement les seuils d'acceptabilité pour chaque classe. En particulier, le fait de diminuer à l'intérieur d'un agrégat la variance de la taille des flux rend le trafic plus prédictible et plus facile à dimensionner. Le dimensionnement est bien plus difficile à réaliser dans le cas où aucune séparation n'est introduite.
Dans l'architecture proposée, nous nous sommes basés sur le type d'application pour reconnaître la nature du flux et ensuite le classifier. Cette méthode de classification peut, quelquefois, introduire des erreurs quant au type de flux. Ceci nous a poussé à proposer une nouvelle architecture basée sur le traitement préférentiel pour

construire un réseau qui améliore les performances du modèle "best effort" tout en préservant son ubiquité et sa simplicité. Le traitement préférentiel est appliqué aux premiers paquets de chaque connexion, qui sont plus sensibles aux pertes, favorisant ainsi les connexions courtes. Un flux arrivant au réseau sera classé comme court. Une fois il dépasse un certain seuil, le flux sera basculé dans une deuxième classe. Nous comptons également sur l'architecture DiffServ proposée dans [BER99] pour classifier les flux aux bordures d'un réseau. Plus spécifiquement, nous maintenons la longueur (en paquets) de chaque flux actif aux routeurs de bordures et l'employons pour classifier les paquets entrants. Cette architecture a la particularité de ne pas nécessiter le maintien en mémoire d'un état par flux au coeur du réseau. Dans ce dernier, nous utilisons la politique de gestion de file d'attente RED [FLO93] avec des seuils différents pour les deux types de classes. Ceci nous permet de réduire le taux de pertes éprouvé par les paquets des flux courts.
L'objectif de la présente thèse est de démontrer que les deux architectures proposées sont efficaces pour améliorer la QoS dans le réseau Internet et pratiquement réalisables.

Contenu de la thèse

Ce paragraphe décrit, chapitre par chapitre, le contenu de la thèse.

Le chapitre 1 présente l'état de l'art relatif au réseau Internet. Nous nous intéressons ici en particulier à la problématique d'assurer une bonne qualité de service aux flux actifs, dans un réseau où le seul service proposé est le "best-effort". Nous analysons d'abord les propriétés essentielles du trafic Internet à différentes échelles et décrivons les caractéristiques des différentes applications et leur besoin en terme de la QoS. Ensuite, nous présentons les architectures IntServ et DiffServ qui ont cherché à apporter des solutions pour la fourniture de services différenciés dans IP avec des garanties de QoS tout en conservant un service Best Effort bien éprouvé. Nous montrons les insuffisances et les incertitudes de ces modèles. Finalement, nous présentons les algorithmes adaptatifs qui ont été développés aux extrémités, pour améliorer la QoS. Nous nous intéressons principalement au protocole TCP qui réalise l'adaptation aux pertes de paquets et le contrôle de congestion.

Dans le chapitre 2, nous étudions les différentes versions du protocole TCP (Tahoe, Reno, New Reno, ...). Bien que ces versions aient apporté beaucoup d'amélioration à la version TCP Tahoe d'origine ; elles restent incapables, d'une part, d'assurer un partage équitable entre les différents flux transportés, et d'autre part, de garantir un débit minimal par flux. En effet, dans une étude menée sur le partage statistique de la bande passante, nous montrons l'incapacité du protocole TCP d'assurer un partage équitable entre d'un côté, les flux longs qui ont des RTT différents, et d'un autre côté, les flux courts et longs.
Le partage de la bande passante est très défavorable aux flux dont le RTT est large. Ces derniers n'arrivent pas à atteindre le même débit que les flux avec un petit RTT.

De même, les flux courts sont défavorisés quand ils se partagent les mêmes ressources avec les flux longs. Les performances des flux courts sont étroitement liées à la phase Slow-Start du protocole TCP, qui est la phase de démarrage. Par conséquent, leur fenêtre de congestion est souvent petite. Les flux TCP avec des fenêtres de congestion petites sont très sensibles aux pertes car ils ont souvent recours au TO pour détecter une perte. Tandis que, les performances des flux longs dépendent plus de la phase Congestion Avoidance que de la phase Slow Start, ce qui leur permet d'augmenter suffisamment leur fenêtre de congestion. Durant la phase Congestion Avoidance, les flux entrent dans un régime permanent et arrivent à mieux se partager les ressources. Les flux TCP avec des fenêtres de congestion larges sont plus tolérants aux pertes, car ils ont souvent recours aux trois acquittements dupliqués pour détecter une perte.

Les études menées sur la classification des flux, le contrôle d'admission et le traitement préférentiel appliqué aux flux courts constituent les contributions principales de la thèse et font l'objet des chapitres 3, 4 et 5. L'objectif principal de ces études est d'évaluer l'apport des modèles proposés pour assurer un partage équitable entre les flux TCP transportés et une bonne qualité de service dans un réseau Internet multiservice. Le contrôle d'admission que nous proposons est orienté flux et implicite.

Dans le chapitre 3, nous montrons l'intérêt de classifier les flux TCP longs selon le RTT. D'abord, nous étudions le cas de deux types de sources. Nous nous intéressons, en particulier à évaluer l'impact de la classification sur les performances des flux avec large RTT. Ensuite, nous généralisons le modèle pour tenir compte de plusieurs sources avec RTT différent. Dans ce cas, nous cherchons à minimiser la bande passante nécessaire pour garantir un débit minimal par flux. Pour séparer les flux, nous utilisons la politique WFQ. L'objectif des modèles proposés est d'assurer un débit minimal par flux indépendamment de son RTT. Dans un premier temps, une étude analytique a été menée pour montrer l'avantage de séparer les flux selon le RTT. Dans un deuxième temps, nous validons les performances des modèles proposés par une étude quantitative basée sur des simulations.

L'objectif du chapitre 4 est de monter la nécessité du contrôle d'admission pour les flux élastiques. Nous commençons par présenter l'importance de reconnaître les flux comme entité de trafic et nos motivations pour imposer un contrôle d'admission qui tient compte de deux types de flux : courts et longs. Nous abordons ensuite les principes de base de notre proposition : les conditions de rejet d'un flux, l'aspect implicite de la méthode, l'identification des flux et l'admissibilité basée sur le calcul du débit minimal à garantir pour chaque type de flux.
Nous présentons alors l'état de l'art en matière des méthodes de contrôle d'admission proposées. Ces méthodes ne conviennent pas au contrôle d'admission appliqué aux flux élastiques. Ces derniers sont majoritairement courts. Il semble ainsi nécessaire de mettre en oeuvre un contrôle 'implicite' fonctionnant sans signalisation et sans réservation de ressources. Même dans le cas où un contrôle d'admission implicite a été proposé pour les flux élastiques [OUE00, BEN02], ce dernier introduit un grand nombre de trafic supplémentaire puisqu'il est basé sur des mesures. Il s'agit de me-

surer le débit en temps réel d'une connexion TCP permanente. De plus, les auteurs supposent que les flux TCP parviennent à se partager équitablement la bande passante. Ils font pas ainsi la différence entre les flux TCP courts et longs qui ont des contraintes de QoS différentes. Notre approche de contrôle d'admission est différente, elle est appliquée aux deux types de flux TCP. Ainsi, deux seuils d'admission sont alors fixés pour limiter les nombres de flux courts et longs admis simultanément. Notre critère d'admission étant de préserver un débit minimal à garantir par flux tout en maintenant le taux de blocage assez faible, surtout pour les flux courts. Pour ceci, nous utilisons une phase d'attente. Ainsi, un flux arrivant au réseau et ne pouvant pas être admis est dans un premier temps mis en attente. A l'expiration d'un temps limite, le flux est rejeté. Nous étudions ensuite les propriétés de différenciation d'un mécanisme qui alloue la bande passante aux flux courts proportionnellement à un poids. Le poids attribué aux flux courts est basé sur le calcul de la charge moyenne introduite par ces derniers. Nous présentons une étude analytique afin de trouver les seuils d'admission optimaux. Ensuite, une étude basée sur des simulations a été menée, dans laquelle des hypothèses plus réelles sont considérées.
Dans l'architecture proposée, nous nous sommes basés sur le type d'application pour reconnaître la nature du flux. Cette méthode peut, quelquefois, introduire des erreurs quant au type de flux. Ceci nous a poussé à proposer une nouvelle architecture basée sur le traitement préférentiel. Elle fait l'objet du chapitre 5.

Dans cette architecture, nous proposons d'assurer un traitement préférentiel aux premiers paquets de chaque connexion, qui sont plus sensibles aux pertes, favorisant ainsi les connexions courtes. Un flux arrivant au réseau sera classé comme court. Une fois il dépasse un certain seuil, le flux sera basculé dans une deuxième classe. Nous comptons également sur l'architecture DiffServ proposée dans [BER99] pour classifier les flux aux bordures d'un réseau. Plus spécifiquement, nous maintenons la longueur (en paquets) de chaque flux actif aux routeurs de bordures et l'employons pour classifier les paquets entrants. Cette architecture a la particularité de ne pas nécessiter le maintien en mémoire d'un état par flux au coeur du réseau. Dans ce dernier, nous utilisons la politique de gestion de file d'attente RED [FLO93] avec des seuils différents pour les deux types de classes. Ceci nous permet de réduire le taux de pertes éprouvé par les paquets des flux courts. Dans un premier temps, nous comparons l'architecture proposée à un modèle classique où aucun traitement préférentiel n'a été envisagé pour la classe des flux courts. Nous constatons que le traitement préférentiel est nécessaire pour garantir une réponse rapide aux flux courts et pour améliorer l'équité entre ces deux types de flux. Dans un deuxième temps, nous comparons cette architecture à un modèle qui consiste à appliquer une priorité absolue aux paquets des flux courts [GUO02a], [GUO02b] et [AVR04].

Chapitre 1

Qualité de service dans le réseau Internet

Ce chapitre présente l'état de l'art relatif au réseau Internet. Nous nous intéressons ici en particulier à la problématique d'assurer une bonne qualité de service aux flux actifs, dans un réseau où le seul service proposé est le "best-effort". Introduire la qualité de service et les services différenciés nécessaires à ses usagers, dans une infrastructure globale telle qu'Internet, reste un défi majeur et encore largement ouvert. Dans l'ingénierie de trafic, il est essentiel de comprendre les propriétés intrinsèques d'un trafic pour pouvoir en maîtriser les effets sur un réseau, et en particulier bien dimensionner les ressources de ce réseau.
Le but de ce chapitre est d'abord d'analyser les propriétés essentielles du trafic Internet à différentes échelles. Ensuite, nous décrivons les caractéristiques des différentes applications et leur besoin en terme de la QoS. Finalement, nous résumons les différentes propositions qui ont été faites pour améliorer la QoS ainsi que leurs avantages et leurs inconvénients.

1.1 La caractérisation à l'échelle de paquet

Les paquets forment l'entité de trafic la plus fine que l'on considère dans les réseaux de données, le paquet étant l'unité élémentaire traitée par les éléments du réseau. Ils sont à priori de longueur variable dans un réseau IP et surtout leur processus d'apparition est très complexe, en raison notamment de la superposition de services de natures très diverses et de l'interaction des couches protocolaires.
Le processus décrivant l'arrivée des paquets possède la caractéristique d'auto-similarité [PAX94, PAX95, FEL00], ce qui rend très difficile l'évaluation des performances du réseau au niveau paquet. Cette propriété statistique, connue aussi sous les noms de dépendance à long terme, persistance ou encore effet de Hurst, a suscité un grand intérêt dans les études de trafic des réseaux de données.
Un trafic auto-similaire présente deux propriétés :
– Une structure complexe à des petites échelles arbitraires. Si on amplifie une partie de sa représentation, on continue à voir des combinaisons complexes de points

séparés par des écarts de tailles différentes.
– Une structure répétitive, une structure auto-similaire contient des répliques d'elle-même à toutes les échelles de temps.

II existe plusieurs expressions mathématiques décrivant un processus auto-similaire. Nous citons la plus standard et la plus générique [RIN00].
Un processus continu dans le temps

$$Y = y(t),\ t \in T$$

est auto-similaire (avec un paramètre d'auto-similarité H appelé Hurst) s'il vérifie :

$$Y(t) = a^{-H} y(at),\ \forall t \in T,\ \forall a > 0,\ 0 \leq H < 1$$

L'idée derrière cette définition est de dire que le processus auto-similaire apparaît être le même à n'importe quelle échelle temporelle. Il se reproduit de manière similaire quelle que soit la finesse temporelle avec laquelle il est représenté. La valeur de H varie entre 0.5 (absence d'auto-similarité) et 1 (auto-similarité élevée).
La cause principale de l'auto-similarité du trafic IP est l'extrême variabilité de la taille des documents transférés. Il est en fait établi que la distribution de la taille des fichiers transférés utilisant le protocole FTP est la plus fidèlement approchée par une loi à queue lourde. La loi de Pareto, plus couramment utilisée pour caractériser les distributions à décroissante lente, est caractérisée par l'expression de densité de probabilité suivante :

$$f(x) = \beta \frac{k^\beta}{x^{\beta+1}}, \quad x \geq k \tag{1.1}$$

La loi de Pareto possède une queue de distribution proportionnelle à $x^\beta,\ \beta > 0$. Cette expression justifie le terme de " loi de distribution à décroissance lente " (plus lente que l'exponentielle). De plus, pour $1 < \beta < 2$, la moyenne est finie, mais la variance ne l'est pas. Ce fait signifie une très grande variabilité de la variable aléatoire considérée.
Ainsi, nous constatons que la caractérisation du trafic Internet en terme de flux, plutôt qu'en terme de paquet, s'avère plus facile à gérer.

1.2 La caractérisation à l'échelle des flux

La notion de flux est particulièrement difficile à définir dans le contexte de l'Internet du fait qu'il s'agit d'un réseau en mode non connecté. Les définitions utilisées sont donc liées soit à l'espacement entre paquets ayant de caractéristiques communes (adresses source et destination, numéros de port UDP/TCP, etc.), avec un espacement maximal de quelques secondes entre paquets successifs, soit à des notions applicatives.
Les flux peuvent être classés en deux catégories [ROB00] : d'une part, les flux dits stream, qui correspondent généralement à des applications de type audio ou vidéo et qui ont un débit intrinsèque que le réseau doit préserver et, d'autre part, les flux

dits élastiques, qui peuvent s'adapter à la bande passante disponible (typiquement les flux générés par le transfert de fichiers). En général, les flux de type stream sont basés sur le protocole UDP, tandis que, les flux élastiques sont gérés par le protocole TCP. Il est important de noter que le trafic élastique représente la grande majorité du trafic Internet actuel, même si le trafic UDP est en augmentation. Les mesures reportées dans http ://ipmon.sprintlabs.com montrent que le trafic TCP représente plus de 90% du volume du trafic Internet 'en octets' et plus de 80% des flux (voir section 1.4).
Plusieurs travaux ont modélisé l'arrivée des flux par une loi Poissonienne [BEN01, AVR03, LIA01]. Cette représentation est vraie dans des cas très particuliers, au coeur du réseau par exemple, mais sa justification au niveau d'un lien d'accès reste difficile. Dans [FED00, NUZ00, PAX95], les auteurs constatent que le processus d'arrivée des flux a aussi des propriétés d'auto-similarité, à un degré moindre que le processus d'arrivée des paquets. Ainsi, pour décrire le processus d'arrivée des flux élastiques, il est probablement nécessaire d'analyser de près le comportement de l'utilisateur durant une session Web, par exemple, dans le but de se rapprocher des périodes d'activité des utilisateurs.
Dans notre travail, nous avons utilisé le mot flux pour désigner une connexion TCP suivie d'une période d'inactivité. Nous nous intéressons uniquement aux flux élastiques (nous ne nous intéressons pas ici aux flux UDP qui restent encore minoritaires en nombre et en volume de ressources consommées).

1.3 La caractérisation à l'échelle des sessions

En caractérisant le trafic à l'échelle de session, nous essayons de se rapprocher le plus des périodes d'activité des utilisateurs. Une session peut être définie comme un ensemble de flux avec des attributs communs.
La structure d'une session dépend de la nature de l'application. Des exemples de sessions sont :
- Toutes les pages Web téléchargées à partir d'un serveur donné par un usager pendant la durée d'une connexion.
- Les transferts de fichier suivies des périodes d'inactivité durant une connexion FTP.

La structure d'une session est complexe. Une session est définie par la distribution du nombre de flux transférés, et la suite des durées entre le transfert de deux flux (durée qualifiée de période de réflexion ou d'inactivité de la session). La distribution du volume des documents transférés est encore une distribution à queue lourde [BAR94]. Les statistiques du trafic observé à l'échelle des sessions montrent que le processus d'arrivée des sessions est bien approchée par un processus de Poisson [ROB01].
Dans notre travail, une session a été réduite à une connexion TCP suivie d'une période d'inactivité. Nous nous approchons ainsi de la définition utilisée pour un flux (Section 1.2).

1.4 Profils de trafic par protocole

Les applications TCP comptent pour une grande majorité du trafic Internet d'aujourd'hui. Une campagne de mesure [THO97], conduite sur plusieurs liens dorsale en 1997, montre que plus de 90% du trafic est portée par le protocole TCP, dont 75% est généré par les applications HTTP. Le protocole UDP ne transporte encore que très peu de trafic. Ces mesures ont depuis été validées par d'autres études comme [FRA01], qui confirment la prépondérance du trafic TCP dans l'Internet. Malgré le fait que la proportion du trafic utilisant UDP augmentent, grâce aux applications multimédia qui prennent de plus en plus d'ampleur, on s'attend toujours à ce que le trafic de TCP reste dominant dans un futur proche grâce aux applications Peer-To-Peer.
Dans le cadre du projet Ipmon, des observations de trafic ont été réalisées dans le but d'établir un ensemble de mesure pour les réseaux IP capable de collecter les statistiques détaillées du trafic au niveau paquet, flux et type d'applications, aussi bien que les délais, pertes et autres informations sur les performances du réseau. Les mesures sont faites sur un lien de 2.5Gigabits/s, en entrée du réseau dorsal sprint. La sonde est placée en sortie d'un noeud de concentration sur un lien vers un noeud régional du réseau IP. Le trafic a été observé, dans les deux sens de transmission, durant une période de 3 heures. Les résultats de cette campagne d'observations de trafic nous ont permis de confirmer une fois de plus les statistiques données dans les articles cités plus haut. Ces résultas fournissent quelques éléments quantitatifs comme le profil de trafic par protocole et par type d'applications. Les tableaux 1.1 et 1.2[1] rassemblent les proportions de trafic par protocole de transport, exprimées en nombre de paquets et octets, en Juin 2003 et Février 2004 respectivement. Nous remarquons que les proportions du protocole TCP ont pris de plus en plus d'ampleur, c'est surtout dû aux applications "File sharing" comme le peer to peer.

les protocoles observés					
Proto		Packets/%		Mbytes/%	
tcp	6	31918667	88.38%	4124.97	93.14%
udp	17	3787097	10.49%	277.49	6.27%
icmp	1	402573	1.11%	25.64	0.58%
ipv6	41	7080	0.02%	0.67	0.02%
ip	0	631	0.00%	0.03	0.00%

TAB. 1.1 – Proportions de trafic par protocoles (Juin 2003)

Le profil de trafic par application reporté sur le site du projet montrent que la grande majorité des flux TCP sont des flux de petit volume (appelés courts ou "souris") tandis qu'une faible minorité des flux ont un volume très important (appelés longs

[1]Les tableaux sont donnés sur le site du projet Ipmon ipmon.sprintlabs.com

les protocoles observés					
Proto		Packets/%		Mbytes/%	
tcp	6	202172070	91.42%	110942.05	97.61%
udp	17	16781073	7.59%	2149.01	1.89%
esp	50	754414	0.34%	373.12	0.33%
icmp	1	1183027	0.53%	109.74	0.10%
gre	47	245391	0.11%	88.11	0.08%
ipv6	41	154	0.00%	0.05	0.00%
pim	103	541	0.00%	0.02	0.00%
rsvp	46	92	0.00%	0.01	0.00%
ip	0	151	0.00%	0.01	0.00%
narp	54	64	0.00%	0.00	0.00%

TAB. 1.2 – Proportions de trafic par protocoles (Février 2004)

ou "éléphants"). La plus grande partie du volume du trafic est engendrée par cette minorité de flux. Les flux courts se sont, en général, les flux générés par les applications HTTP, SMTP ··· tandis que les flux longs sont en général générés par les applications FTP, File sharing (peer to peer)··· Le Peer to Peer est une nouvelle technologie qui permet aux utilisateurs d'échanger leur données avec d'autres utilisateurs en leur donnant le droit d'accès. Ce type d'application prend de plus en plus d'ampleur.

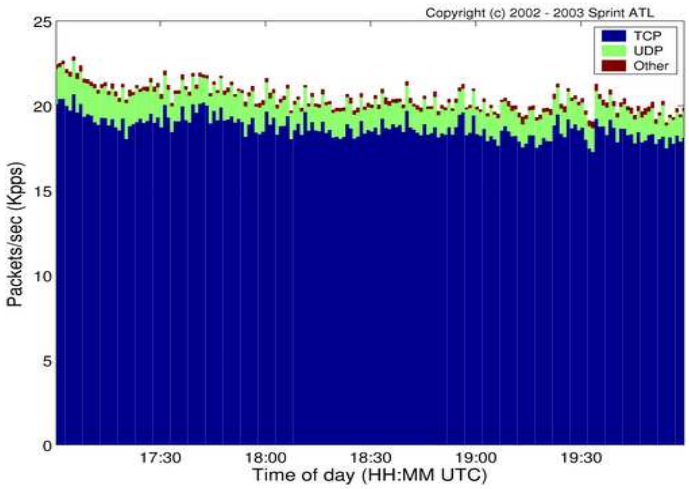

Figure 1.1 – Profils de trafic TCP et UDP en paquets/s

1.5 Besoins de Qualité de service des applications

Pour déterminer les besoins des applications en termes de qualité de service, nous pouvons s'appuyer sur une classification qui permet de cerner, pour chaque type d'application, les dimensions critiques. La classification des applications de l'Internet définie dans le cadre du modèle IntServ [RFC1633] distingue les applications élastiques des applications rigides. Les applications rigides génèrent des flots de données dont la livraison à débit constant (CBR) est prévisible. Les applications élastiques, génèrent des rafales avec une livraison imprévisible de blocs de données à débit variable (VBR). Les applications telles que le transfert de fichiers envoient des données en masse ce qui augmente le débit de la source et peut utiliser toute la bande passante disponible (il n'y a pas de borne supérieure ni en délai, ni en débit). Une autre typologie [WU98] distingue les applications personne-personne des applications personne-serveur. Il existe aussi des applications serveur-serveur. Lorsque deux personnes sont en communication, comme dans le cas des applications coopératives, les contraintes temporelles sont très fortes, les délais doivent être courts et invariables. En effet, le contenu numérisé des applications multimédia temps-réel échantillonné et émis à intervalle de temps régulier, doit être transmis de manière isochrone afin d'être restitué correctement. Pour ces applications personne-personne, et en particulier pour la voix numérisée, le cas idéal est celui dans lequel le réseau introduit un délai faible et constant, c'est à dire une gigue nulle. Un délai trop important a un impact néfaste dans une communication interactive telle qu'une conversation téléphonique.
Dans le domaine de la QoS, l'usage est de considérer les besoins des applications en terme de délai car ce sont les besoins les plus exigeants et difficiles à honorer dans un réseau partagé. C'est ainsi que l'on assimile souvent la notion de qualité de service à cette dimension temporelle. Cependant, les réseaux informatiques, et Internet en particulier, sont aujourd'hui utilisés pour des applications de plus en plus variées. Les applications communicantes couvrent un large spectre allant de la messagerie textuelle, à la distribution de logiciel, à la vidéo et l'audio de loisir, à la vidéo conférence, l'e-commerce, les jeux temps-réel multi-utilisateur, la visualisation scientifique et l'imagerie médicale. Il en résulte une très grande diversité des caractéristiques de trafic et des besoins en performances des applications existantes, ainsi qu'une profonde incertitude face aux applications futures. L'hétérogénéité et la fluctuation des besoins est vaste et recouvre plusieurs formes.
Dans la section suivante, nous déterminons les caractéristiques et les besoins de la qualité de service pour les applications streaming et élastiques.

1.5.1 Les applications élastiques

Dans cette section, nous décrivons en détail les caractéristiques et les besoins de trois applications élastiques qui sont Telnet, HTTP et FTP. Elles représentent chacune le groupe de l'interactivité élevée, l'interactivité moyenne et l'interactivité faible respectivement. Les deux dernières sont utilisées dans les études de simulation présentées dans cette thèse.

1.5.1.1 Telnet

Nous présentons ici les caractéristiques de l'application Telnet en termes de modélisation et quantité du trafic produit, ainsi que, ses exigences en termes de délai et de perte.

Caractéristiques

Telnet est un Protocole d'émulation de terminal qui permet à l'internaute d'entrer en communication avec un hôte Internet à partir de son propre ordinateur pour utiliser des programmes ou consulter des données qui y sont stockées. Le protocole Telnet permet de consulter, par exemple, des banques de données et des catalogues en ligne de bibliothèques. Les paquets générés par une telle application sont souvent très petits (en général plus petit que 50 Octets).

Les inter-arrivés des paquets Telnet suivent une distribution à queue lourde (heavy tail), résultant du trafic rafale [PAX95]. En effet, le temps d'inter-paquet est normalement limité par la vitesse de dactylographie des humains, qui est généralement plus lente que 5 caractères par seconde [RFC1144], donnant ainsi à un temps moyen d'inter-paquet minimum de 200msec et à un taux de données plus bas que 16 kb/s, quand le pire cas, de 1 en-tête TCP/IP (40 Octets) par caractère, est considéré [RFC1144].

Plusieurs campagnes de mesure ont été faites sur des vrais réseaux ; ces dernières ont fourni des informations sur la modélisation actuelle de Telnet. Ainsi, le processus d'arrivée des connexions Telnet s'est avéré être bien modélisé par un processus de Poisson [[PAX95]. Une connexion type dure quelques minutes, s'étendant entre 1.5 et 50 minutes [CAC91], avec une durée moyenne entre 2 et 4 minutes [PAX94]. Le nombre d'octets envoyés par le client et le serveur ont des distributions dites à queue lourde. La première s'est avérée suivre une distribution appelée log-extreme, alors que la seconde suit une distribution log-normal, comme indiqué dans le tableau 1.1 [PAX94].

Exigences

Comme indiqué ci-dessus, Telnet est une application fortement interactive et a donc des contraintes strictes de délai sur les paquets individuels. Les études subjectives de qualité ont constaté que les délais des écho commencent à être gênant au moment où ils excèdent 100 ms, et en général, un délai de 200 ms est la limite au delà de laquelle l'utilisateur commence à en souffrir. Par conséquent, le trafic de Telnet est particulièrement sensible au délai d'attente dans les files et aux pertes de paquet, puisque les procédures de retransmission de TCP introduisent souvent des délais qui excèdent le délai maximal acceptable de l'écho. Pour cette raison, la perte de paquet dans Telnet doit être maintenue à un minimum pour que les performances soient bonnes.

1.5.1.2 Web

Le World Wide Web a été la force primaire derrière la croissance rapide de l'Internet pendant les 10 dernières années. Aujourd'hui, le WWW constitue l'application la plus importante dans le réseau Internet, puisqu'il constitue actuellement la grande majorité des flux Internet (plus de 90%).
HTTP [RFC1945], le protocole employé pour transporter le contenu du Web, est un protocole basé sur un paradigme requête/réponse. Un client établit une connexion vers un serveur et lui envoie une requête sous la forme d'une méthode, d'une URI, du numéro de version, suivi d'un message de type MIME contenant les modificateurs de la requête, les informations sur le client, et éventuellement un corps. Le serveur répond par une ligne d'état, incluant la version de protocole et un message de succès ou d'erreur, suivi d'un message de type MIME contenant l'information sur le serveur et le corps éventuel.
La plupart des communications HTTP sont initiées par un utilisateur et consistent en une requête qui devait être appliquée (il s'agit d'une méthode) sur des ressources dans un serveur.
Dans Internet, les communications HTTP s'appuient principalement sur le protocole de connexion TCP/IP. Le port utilisé par défaut est le port TCP 80, d'autres ports pouvant être utilisés. Ceci n'exclut pas que HTTP puisse être implémenté au dessus d'un autre protocole d'Internet, ou sur d'autres types de réseau. HTTP nécessite seulement un transport fiabilisé ; tout protocole garantissant la sécurité de transmission peut être utilisé pour supporter HTTP.
Exceptées les applications expérimentales, la pratique courante spécifie qu'une connexion doit être initiée par un client avant transmission de la requête, et refermée par le serveur après délivrance de la réponse. Les deux côtés, client et serveur, doivent être préparés à ce que la connexion soit coupée prématurément, suite à une action de l'utilisateur, une temporisation automatique, ou une faute logicielle, et doivent apporter une réponse prévisible à cette situation. Dans tous les cas, la fermeture d'une connexion qu'elle qu'en soit la raison est assimilable à la conclusion de la requête, quel qu'en soit son état.

Caractéristiques

Les mesures faites sur le trafic HTTP tel que [HEI97a, MAH97], ont prouvé que la plupart des requêtes ont des tailles inférieures à 500 octets, et peuvent donc être transmises dans un seul segment typique de TCP. D'autre part, la taille moyenne d'une réponse (portant un composant d'une page) est typiquement entre 10 et 20 Koctets. Au cours des études sur le trafic Web [PIT98], un volume moyen de fichier HTML d'environ 5 Koctets et une taille moyenne d'image de 14 Koctets sont énumérés. Ces chiffres tendent à augmenter avec l'amélioration de l'infrastructure du réseau et les utilisateurs peuvent ainsi télécharger de plus grands fichiers. La distribution de la taille des transferts est caractérisée par une distribution à queue lourde. Cette caractérisation a été confirmée par une étude de mesure sur le trafic web prise pendant 4 heures chargées [CRO97]. Cette étude prouve également que

le trafic web peut générer de dépendance à long terme (auto-similarité) quand la charge du réseau est assez forte.

Exigences

Une latence faible est exigée pour le téléchargement des pages web, cette condition est primordiale pour ces applications. Les études sur le comportement humain indiquent que ces derniers considèrent que les performances sont bonnes tant que le temps de téléchargement ne dépasse pas les 5 secondes. Si celui-ci est entre 5 et 10 secondes, les performances restent acceptables, tandis que des périodes plus grandes que 10 secondes donnent des performances très faibles [BHA00, BOU00, MOG95]. En outre, la variance du temps entre le téléchargement des différentes pages est aussi un facteur de performances pour les applications web.

1.5.1.3 FTP

Le protocole de transfert de fichiers (FTP) est la méthode la plus courante pour transférer des fichiers d'un ordinateur sur Internet. Lorsqu'un utilisateur importe des fichiers d'un serveur FTP vers son ordinateur local (client), il s'agit de téléchargement en aval ; le processus inverse s'appelle téléchargement en amont. FTP permet de transférer des fichiers de tout type tel que du texte (.txt, .html) ou des fichiers binaires (.doc, .pdf, .gif, .jpg) il n'est pas nécessaire que le fichier soit en format HTML ou dans un format graphique précis. FTP permet également de transférer des fichiers entre différents types d'ordinateurs ; par exemple, entre un serveur UNIX et un PC. C'est une des raisons pour lesquelles cette application est si populaire. Sur Internet, les applications FTP s'appuient principalement sur le protocole de connexion TCP/IP. Le port utilisé par défaut est le port TCP 20, d'autres ports pouvant être utilisés.

Caractéristiques

Des campagnes de mesures ont montré que le processus d'arrivée des connexions FTP peut être modélisé par un processus Poissonien [PAX95, PAX94]. D'autres études plus récentes [REG02] indiquent que cette approximation n'est pas toujours valable et que le processus d'arrivée des connexions FTP suit une distribution à queue lourde. Il a été ainsi observé que les données générées par les connexions FTP tendent à se produire dans des rafales, reflétant des opérations étroitement séparées comme dans le transfert multiple généré par des multiples messages "get". Nous tenons aussi à noter, que la quantité de données transférées dans une connexion FTP ou dans une session FTP (se composant de plusieurs transferts de données individuels et successifs entre deux hôtes) suit une distribution à queue lourde [FRA01].

Exigences

La qualité perçue lors d'un transfert FTP est déterminée par son débit, d'où le

besoin de lui garantir un débit moyen minimal. En revanche, les connexions FTP sont moins sensibles à la variation du délai vu que ce type de trafic appartient au groupe de l'interactivité très faible.
Dans notre travail, nous nous intéressons uniquement aux flux FTP et HTTP, qui représentent respectivement la classe des flux longs et celle des flux courts. Nous définissons un flux long comme un flux qui a un très grand nombre de paquets à envoyer, tandis qu'un flux court est un flux qui a très peu de paquets à envoyer et qui est en général d'une taille moyenne égale à 20 ko [LIA01, AVR03].

1.5.2 Les applications streaming

Contrairement aux applications élastiques, ce qui caractérise les applications temps réel (voix et vidéo) est qu'elles n'exigent pas un transfert fiable à 100%.
Ces applications ont typiquement des exigences au niveau du délai, excluant ainsi l'utilisation du protocole TCP comme protocole de transport. Par conséquent, la plupart des applications streaming emploie le protocole UDP, et n'implémentent pas ainsi le mécanisme de contrôle de congestion.
Les deux applications streaming, les plus répandues sont la voix et la vidéo. Ces deux types d'applications font intervenir un temps de transport en temps réel et demandent un débit constant avec une contrainte de temps de bout en bout.
Si les applications streaming ne sont pas interactives, un temps de retard bien plus important est toléré : regarder un film avec un retard de 10 secondes sur l'instant d'émission, écouter une radio avec un retard de 15 secondes après son émission ou regarder à la télévision des séquences filmées il y a 20 secondes n'est en général absolument pas gênant pour l'utilisateur final. Dans ce cas, la resynchronisation des données est assez simple à effectuer.
En conclusion, les applications streaming sont difficiles à acheminer dès qu'une interaction en temps réel est nécessaire. Cette contrainte demande une qualité de service de la part du réseau ; elle constitue ainsi une caractéristique importante des réseaux.
Dans notre travail, nous nous sommes intéressés uniquement aux applications élastiques, vu que, les applications Streaming restent encore limiter en nombre de flux et en nombre d'octets (voir section 1.4)

1.6 Problématique de la Qualité de Service

Les approches traditionnelles de Qualité de Service (QoS) pour les flux temps-réel sont fondées sur le "mode circuit" et la réservation stricte de ressources. Ainsi le réseau téléphonique analogique ou numérique (ISDN) offert-il un circuit physique dédié tout au long de la conversation ou bien le réseau de télévision utilise une bande de fréquence non partagée pour la diffusion de vidéo. La technologie ATM s'est inspirée des avantages du mode circuit en proposant un mécanisme de multiplexage temporel asynchrone basé sur la commutation rapide de cellules pour résoudre le problème de la qualité de service dans les réseaux de commutation de données. La

technologie ATM a été inventée pour apporter une solution au réseau numérique à intégration de Service large bande et conçue pour offrir une couche réseau multiservices. Son utilisation à large échelle par les applications multimédia aurait pu être une solution universelle au problème de la qualité de service. Ce rêve s'est avéré impossible à réaliser parce qu'aucune "prise ATM" directe n'a pu remplacer la "prise TCP/IP" fournie en standard dans les systèmes d'exploitation des milliers d'usagers d'Internet et que le mécanisme de commutation rapide de petites cellules limite les performances. Différentes approches ont été étudiées ces dernières années pour assurer ou améliorer la qualité de service (QoS) dans Internet. La réservation de ressources pour offrir une garantie stricte de service aux flux individuels dans l'approche IntServ [RFC1633], ou un traitement différentié des flux par une priorisation dans leur acheminement dans l'approche DiffServ [RFC2475] sont les deux alternatives proposées, analysées et déployées dans des plate-formes expérimentales et commencent à peine à arriver sur les réseaux de production. Parallèlement, le développement des applications multimédia s'est poursuivit même en l'absence de garanties strictes fournies au niveau du réseau. C'est ainsi que sont apparues de nombreuses techniques adaptatives qui cherchent à masquer aux utilisateurs les faiblesses du réseau. La question de la fourniture de qualité de service dans Internet demeure un problème largement ouvert. Aucune solution miracle n'a été trouvée ni du côté du réseau, ni du côté des applications. C'est donc dans ce contexte que nous avons développé nos travaux de recherche sur le concept de réseau sensible aux flux (chapitre 3-4-5).

1.6.1 Paramètres de performances d'un réseau

Le fournisseur de service réseau requiert un ensemble d'outils et de services pour contrôler les paramètres de performances du réseau afin de fournir un service meilleur et plus prévisible. Un modèle de performance de réseau s'appuie sur cinq métriques principales.
- **le débit**, noté r,
- **le délai**, noté d,
- **la variation de délai** ou gigue, noté j,
- **le taux de pertes**, noté p,
- **le taux d'erreurs**, noté e.

Le groupe IPPM (IP Performance Supervision) de l'IETF a décrit un cadre pour les métriques de performances IP dans la RFC2330. Ces métriques peuvent être définie dans une direction (aller simple) ou dans les deux directions (aller-retour). Chacune de ces métriques peut être définies de la manière suivante :

1.6.1.1 Paramètres de débit

Le débit binaire, r, ou, par abus de langage, la bande passante, entre deux systèmes communicants est le nombre de bits que le réseau est capable d'accepter ou de délivrer par unité de temps. C'est le taux de transfert maximum pouvant être maintenu entre deux points. Le débit utile dépend du niveau auquel on se place dans

la hiérarchie protocolaire. Par exemple, la bande passante d'un lien réseau, métrique analytique définie par le groupe IPPM, représente la capacité de transport d'un lien réseau, mesurée en bits par seconde, dans laquelle les données n'incluent pas les bits nécessaires pour les entêtes de niveau 2. Lorsque l'on se place à un niveau supérieur à la couche réseau, on considère la capacité du lien (throughput) qui correspond au volume effectif de données application transmis. La capacité utile du lien (goodput) est égale au nombre total de bits issus de l'application et correctement transmis par unité de temps. Par exemple pour un flux TCP, on ne comptabilise que les bons paquets reçus et on n'inclue pas dans les statistiques les paquets retransmis. C'est cette capacité utile qui est le paramètre pertinent pour l'application. Comme il existe différentes implémentations du protocole TCP pour acheminer l'information, cette métrique est empirique et délicate à manipuler. L'IPPM a défini un cadre pour la mesure de cette capacité de transport définie par la formule V/T où V est le volume de données reçues et T le temps écoulé.

1.6.1.2 Paramètres de délai

Délai de transit

Le délai de transit de bout en bout, d, est le temps écoulé entre l'envoi d'un paquet par un émetteur et sa réception par le destinataire. C'est une des caractéristiques principales de la QoS. Le délai de transit d est composé d'une partie fixe df et d'une partie variable dv.
On a $d = df + dv$ avec $df = dp + ds + dt$ et $dv = dq + dj$ où
- dp est le délai de propagation (5 à 6 micros seconde par km sur une liaison filaire),
- ds est le temps de transmission qui est fonction du débit binaire et de la taille des paquets émis,
- dt est le temps de traitement qui correspond par exemple à la traversée d'un codeur-décodeur,
- dq est le délai cumulé dans les files d'attente des routeurs,
- dj est le délai introduit par le buffer de compensation de la gigue pour assurer la synchronisation.

Le temps de traitement, dt, est composé de plusieurs facteurs : le délai système, le délai de traversée des interfaces matérielles au niveau de l'émetteur et du récepteur. Typiquement, pour une information de type voix numérisée sur un PC on a $dt = 20\ ms$.
Le délai dq induit par la mise en file d'attente des paquets dans les systèmes intermédiaires dépend de la taille des tampons mémoire dans les routeurs et de leur encombrement. Un délai de file dq de 50 ms sur un lien à 25 $Mbps$ indique qu'il y a environ 15 *koctets* en attente dans les routeurs. Le délai dépend donc de la distance entre deux points, du nombre d'équipements intermédiaires et du délai d'attente dans ces équipements. Dans les réseaux IP, on considère aussi souvent le délai d'aller-retour ou Round Trip Time, RTT. Celui-ci est facile à calculer en utilisant la méthode du ping qui consiste à émettre des paquets de contrôle de type *echo* du protocole ICMP et de calculer le délai entre l'émission et le retour de l'écho. Un certain nombre de

travaux récents [TAS02] ont montré les limites de l'approche *aller-retour* par une mise en évidence de l'asymétrie des performances sur les liens Internet. Ainsi de nouvelles techniques de mesures du délai sont apparues. Elles nécessitent la synchronisation des horloges de l'émetteur et du récepteur par un système GPS (Global Positionning System) ou un protocole de type NTP [RFC1305].

gigue

La gigue est la variation du délai de bout en bout notée j. La gigue du délai pour un flux de paquets est définie comme étant la différence maximum de délai expérimentée par n'importe quels paquets pris deux à deux dans le flux. Si d_0 est le délai de référence et d_k le délai du k^{ime} paquet alors $j_k = d_k - d_0$.
A cause de la gigue introduite par le réseau, les flux audio/vidéo capturés au même instant peuvent ne pas arriver simultanément chez le récepteur. Au niveau du récepteur, les applications temps réel doivent stocker les données pour enlever la gigue ajoutée par le réseau et retrouver les relations temporelles originales.
L'IETF a aussi défini formellement la notion de gigue instantanée : variation de délai instantanée (instantaneous packet delay variation : IPDV). C'est la différence du délai de transmission entre deux paquets k et $k+1$ consécutifs. Cette gigue instantanée reflète l'évolution de l'état de congestion du lien. Si elle est stable, la charge du lien est constante. Si par contre elle augmente, elle indique la dérivation vers un état de congestion. Elle est utilisée dans certaines implémentations de TCP (TCP vegas) pour anticiper les pertes de paquets, donc les congestions et mettre en oeuvre les mécanismes d'évitement de congestion plus rapidement. La gigue ne permet d'anticiper un état de congestion que si le délai de bout en bout est relativement important. Cette contrainte limite donc son usage dérivé.

1.6.1.3 Paramètres de fiabilité

Taux de pertes

Le taux de pertes p correspond au rapport du nombre de paquets non arrivés sur le nombre total de paquets transmis. Les pertes dans Internet sont causées par la congestion, l'instabilité du routage, les défaillances de liens et la non fiabilité des liaisons téléphoniques ou sans fil. La congestion est la principale cause des pertes.
La perte de paquets peut se produire soit par dépassement de capacité des files d'attente dans les routeurs ou dans les systèmes d'extrémité soit par violation de la limite des délais bornés. La distribution des pertes est aussi une métrique très importante pour les protocoles adaptatifs tels que TCP. Un lien réseau peut être caractérisé par son taux d'erreur e qui est calculé à des intervalles de temps relativement longs. Il correspond au nombre de bits reçus erronés sur le nombre total de bits reçus ou le nombre de paquets erronés sur le nombre total de paquets reçus. Dans les liaisons filaires actuelles, ce taux d'erreur résiduel est très faible. Cependant, dans les réseaux sans fil, le taux d'erreur n'est plus négligeable.
Dans les chapitres 3, 4 et 5, nous proposons de nouvelles approches pour améliorer la qualité de service perçue par les applications élastiques. Nous focalisons nos efforts

sur l'évaluation des paramètres de performance suivants : le débit moyen obtenu par chaque flux (volume transmis divisé par le temps total de transmission), le temps moyen de séjour (le temps total de transmission de tout le flux). Ces mesures caractérisent la QoS vue au niveau flux et ne doivent pas être confondues avec les mesures au niveau paquet.
Quand un modèle de contrôle d'admission est proposé pour limiter le nombre de flux admis, nous évaluons le taux de blocage représenté par le nombre de flux rejeté divisé par le nombre total de flux qui arrivent au réseau.
Dans la section suivante, nous exposons les différentes architectures qui ont été proposées par de nombreux chercheurs pour améliorer la QoS. Nous montrons ainsi que le besoin en qualité de service n'est pas toujours le même et qu'il dépend du type d'application utilisé.

1.7 Solutions proposées pour améliorer la QoS

Généralement, on oppose deux approches architecturales pour résoudre le problème de la qualité de service réseau : l'approche **dans le réseau** et l'approche **aux extrémités**. Dans l'approche **aux extrémités**, la qualité de service peut être traitée au niveau utilisateur (a) [BUS95], au niveau application (b), au niveau système (c) ou au niveau de la couche transport (d) [STE95]. Les applications adaptatives traitent la QoS au niveau (b) en mettant en oeuvre des algorithmes de compensation d'erreurs ou des mécanismes d'adaptation pour pallier le manque de qualité de service du réseau. Une activité importante est mené au niveau (c) avec les architectures de QoS et des interfaces de programmation, API de QoS [AUR97]. La couche transport est la première couche susceptible de réaliser l'adaptation de bout en bout de niveau (d). La couche AAL, Adaptation Application Layer, apporte la flexibilité du modèle ATM. TCP dans le modèle IP est un algorithme adaptatif.
Dans l'approche **dans le réseau**, il existe des solutions à tous les étages du modèle en couches. Les architectures IntServ et DiffServ proposent des solutions de QoS pour la couche réseau IP. Pour la couche 2, l'IETF a défini le protocole RSVP pour la réservation de la bande passante dans Ethernet et d'autres réseaux de la famille 802. IEEE a proposé aussi d'étendre Ethernet pour ajouter 8 niveaux de priorité dans la norme 802.1p qui est exploitée aussi dans le cas des réseaux mobiles (norme 802.11). La commutation en longueur d'onde (lambda switching), proposera des longueurs d'onde dédiées offrant des capacités distinctes et garanties aux utilisateurs.
L'opposition d'approche *dans ou hors réseau* est l'objet de débats récurrents, preuve qu'aucune solution unique n'existe pour répondre à l'ensemble des questions. Selon la philosophie IP, le coeur de réseau devrait rester le plus simple possible et les services doivent être assurés par les extrémités [CLA88]. Au contraire, dans la technologie ATM, la qualité de service est gérée dans le réseau, mais sa mise en oeuvre de bout en bout s'est avérée très complexe.
Dans la section suivante, nous étudions les avantages et les limites des solutions proposées dans l'approche dite **dans le réseau**. Dans l'approche dite **aux extrémités**, seul le mécanisme de contrôle de congestion TCP sera détaillé.

L'explication de ces deux approches permet de mieux comprendre les modèles proposés dans les chapitres 3, 4 et 5.

1.7.1 Solutions réseaux

1.7.1.1 Mécanismes de base

Le surdimensionnement

Les défenseurs de l'IP Best effort proposent la solution du surdimensionnement. Le principe est d'augmenter toujours la bande passante disponible pour éviter la congestion, les délais de file d'attente et les pertes de paquets. Ainsi, s'il y a toujours suffisamment de capacité réseau, les garanties de service peuvent être offertes. L'évolution des technologies optiques, telle que le *DWDM* délivrant une capacité croissante en augmentant le nombre de longueurs d'onde, permet la construction de réseaux multiterabit. Il suffit de faire confiance aux progrès des technologies optiques des supports de transmission et de ne rien changer au niveau de la couche IP. Cependant le surdimensionnement ne peut être qu'une solution partielle à la fourniture de qualité de service et ce pour plusieurs raisons :

- ce n'est pas une solution de bout en bout. S'il est possible de sur-approvisionner une épine dorsale et d'offrir de la QoS à ce niveau du réseau, les transactions Internet, traversant deux ou plusieurs domaines administratifs n'ont aucune garantie. On ne peut en rien assurer que deux réseaux surdimensionnés individuellement auront des capacités de couplage, *peering*, suffisantes pour écouler le trafic entre eux. Ainsi, des campagnes de mesures Internet ont montré qu'une large part des problèmes de performances (délais et pertes) se produisent aux points d'accès réseaux publics qui interconnectent plusieurs réseaux.
- pour réduire la congestion et maintenir des délais de bout en bout faibles, un réseau peut avoir à faire du contrôle de trafic en écartant des paquets à ces points d'entrée du réseau. Puisque ces pertes se produisent aux points d'interconnexion, un opérateur peut toujours blâmer les autres réseaux tout en affichant un faible taux de pertes à l'intérieur de son propre réseau.
- on donne la même qualité de service aux paquets de tout type d'application. Et, en cas de surcharge occasionnelle, les paquets sensibles au délai seront plus pénalisés que d'autres par une mise en file d'attente tandis que ceux sensibles aux pertes seront plus pénalisés par les *drops*.
- le dernier kilomètre est traditionnellement le moins bien dimensionné. D'aucune manière un surdimensionnement du coeur de l'Internet ne pourra permettre d'obtenir une qualité vidéo suffisante à partir d'une connexion modem. Il est donc tout aussi important d'approvisionner correctement ce dernier kilomètre. Mais l'émergence des communications sans fil avec leur faibles capacités de communication et de traitement, laisse présager que l'hétérogénéité des connexions d'accès sera toujours considérable.
- finalement, le sur-approvisionnement est une solution inefficace en termes économiques et de gestion de ressources. Tout le trafic reçoit la même QoS très élevée, même si toutes les applications n'en ont pas besoin.

Prévenir et contrôler les congestions

Puisque le principal problème du Best Effort, qui est le responsable de la faible qualité de service dans l'Internet, est la congestion. Une autre solution qui consiste à prévenir les congestions avant qu'elles ne se produisent et à les contrôler si elles arrivent a été proposée. Une étude détaillée de ces mécanismes de contrôle de congestion pour les flux unicast et multicast est présentée dans [DRA00]. Ces mécanismes de gestion active des files d'attente tels que RED [FLO93] tendent à améliorer le service Best Effort en ayant plus de contrôle sur les congestions. Mais ils ne permettent pas d'offrir des garanties ni des services différentiés. Des mécanismes de plus en plus sophistiqués pour le contrôle du trafic et des ressources ont été progressivement ajoutés à ces mécanismes de gestion de files dans les routeurs IP. Ces mécanismes sont les briques de base de la conception des architectures de Qualité de Service de l'Internet. On distingue généralement ceux qui opèrent dans le plan contrôle de ceux du plan données. La table 1.3 ci-dessous répertorie les mécanismes proposés dans les différents plans. Ces mécanismes sont détaillés ci-dessous.

	Plan	Architecture cible
Classification MF	données	Intserv-DiffServ (bordure)
Classification BA	données	DiffServ (coeur)
Mesure	données	Intserv-DiffServ (bordure)
Lissage	données	Intserv-DiffServ (bordure)
Ecartement	données	Intserv-DiffServ
Marquage	données	DiffServ (bordure)
Gestion de file	données	Intserv-DiffServ (coeur)
Ordonnancement	données	Intserv-DiffServ (coeur)
Contrôle d'admission	contrôle	Intserv-DiffServ (bordure)
Contrôle de politique	contrôle	DiffServ (bordure)
Gestion de trafic	contrôle	DiffServ (bordure)
Réservation de ressources	contrôle	Intserv

TAB. 1.3 – Récapitulatif des mécanismes de QoS

Plan données

Les mécanismes du plan données implémentent les actions que les routeurs doivent entreprendre sur chaque paquet pour offrir différents niveaux de service. On distingue :
- les mécanismes de conditionnement : classification, marquage, mesure, contrôle et lissage.
- les mécanismes de gestion de files.
- les mécanismes d'ordonnancement.

Mécanismes de conditionnement

Lorsqu'un paquet est reçu, le classifieur détermine à quel flux ou à quelle classe il appartient en fonction du contenu d'une certaine portion de l'entête et selon certaines règles. Il existe une classification générale qui associe une signature avec des informations de niveau transport (port) et des champs de niveau IP (adresses). Cette opération est coûteuse. Elle est effectuée dans tout routeur IntServ et dans les routeurs de frontière de DiffServ (multifield classification : MF). Une classification basée sur un schéma bit ordonne les paquets selon un seul champ de l'entête IP. C'est une opération simple et très rapide. Dans DiffServ c'est la *behavior aggregate* classification (BA), elle est employée uniquement dans les routeurs de coeur. Après la classification, le paquets est passé à une instance logique de conditionnement qui peut marquer un champ, mesurer les propriétés temporelles d'un flux et les comparer à un certain profil pour décider si le paquet est "dans" ou "hors" profil. Un *shaper*, basé sur un seau percé, *leaky bucket* ou sur un seau à jetons *token bucket* [FER98], peut retarder certains ou tous les paquets pour rendre le flux conforme au profil. C'est ce qu'on appelle le lissage de flux. Certain paquets peuvent être éliminés par un *dropper*.

Mécanismes de gestion de file

Un des buts principaux de la QoS IP est de contrôler la perte de paquets. L'espace mémoire alloué aux files d'attente dans les routeurs est conçu pour absorber les rafales de données de courte durée. Limiter la taille de ces files peut aider à réduire les bornes de délai. Traditionnellement, les paquets sont jetés lorsque les files débordent. Ce sont soit les derniers arrivés (tail drop) soit les plus anciens dans la file (front drop), soit des paquets choisi au hasard qui sont éliminés. Il y a deux inconvénient à cette technique : le lock-out ou monopolisation de la file par certains flux et le problème des files pleines (full queue) qui allongent le délai. Pour éviter ces deux problèmes, des mécanismes sophistiqués de gestion active de file d'attente ont été préconisés et introduits dans les routeurs IP. Ainsi le mécanisme RED (Random Early Detection) [FLO93] est un algorithme de gestion active de file d'attente, de plus en plus populaire et recommandé par l'IETF.
Le principe de RED est de surveiller l'évolution de la file d'attente et d'éliminer des paquets aléatoirement pour prévenir les congestions. Ce mécanisme écarte les paquets de la file de transmission d'un lien avec une probabilité p calculée comme une fonction croissante de la moyenne courante de la taille q de cette file, $p = H(q)$. RED offre plus de flexibilité que le mécanisme classique de Tail Drop dans le choix du comportement du routeur en cas de congestion. Un certain nombre de travaux ont montré les inconvénients et les limites du mécanisme RED simple [MAY99]. Dans certains routeurs le mécanisme WRED (weighted random early detection) est proposé comme alternative à RED. L'écartement se fait non plus de manière purement aléatoire mais est basée sur la priorité du paquet représentant le poids du flux ; cette priorité est marquée dans le sous-champ IP precedence du champ type de service de l'entête IP. RIO [CLA98] raffine le concept RED en introduisant les bits "in" et "out" selon que les paquets sont dans ou hors profil. Ainsi les paquets

"out" sont éliminés préférentiellement. Le mécanisme ECN, Explicit Congestion Notification quant à lui permet d'envoyer un signal à la couche de transport pour prévenir de la congestion en positionnant un bit spécifique de l'entête du paquet [RFC2481]. C'est un mécanisme inspiré de la notification FECN (Forward Explicit Congestion Notification) défini dans le relais de trame. Son apport au niveau des performances de bout en bout observé dans TCP, par exemple, est encore en cours d'évaluation [FLO01].

Mécanismes d'ordonnancement

La fonction d'un ordonnanceur est de sélectionner le paquet à transmettre au cycle suivant, pour chaque interface de sortie d'un routeur et parmi les paquets disponibles et appartenant aux flux partageant la même file de sortie. L'ordonnancement joue une part importante dans le contrôle du délai. Les disciplines de services doivent être simples pour permettre une analyse aisée ainsi qu'une implémentation efficace. Les ordonnanceurs font l'allocation effective des ressources telle que la bande passante (quels paquets doivent être transmis). Ils gérent aussi la rapidité (quand ces paquets doivent-ils être transmis) et l'espace mémoire (quels paquets sont écartés). Ces disciplines de service affectent donc trois paramètres de performance qui sont : le débit, le délai et le taux de perte.
Dans [ZHA95], Zhang présente une étude très détaillée des disciplines de services disponibles pour garantir les performances dans un réseau à commutation de paquets. Les disciplines de service peuvent être classifiées selon qu'elles soient *work conserving*, un serveur n'est jamais inactif lorsqu'il y a un paquet à transmettre ou *non work conserving*, à chaque paquet une date d'éligibilité est affectée. Même lorsque le serveur est inoccupé, si aucun paquet n'est éligible, aucun paquet ne sera transmis. Zhang a montré, que le type de la discipline de service (work conserving ou non) affecte le délai de bout en bout, les besoins en espace mémoire (bufferisation) et les caractéristiques de la gigue.
Il existe une grande variété d'algorithmes d'ordonnancement qui sont analysés dans [GUE99, STO98]. Les plus courants sont :

- FIFO : First In First Out qui est la politique d'ordonnancement la plus simple. Pas de différentiation de flux ou de classe, pas de garantie de délai ou de débit. Cet ordonnanceur a été optimisé pour le Best effort.
- PQ : Priority Queing insère les paquets prioritaires en tête de file. Ce type de discipline permet de fournir le service premium [FER00] de l'architecture DiffServ. Ce mécanisme d'ordonnancement prend en compte les différentes classes de service. La file d'attente correspondante à la classe de priorité la plus élevée (priorité 1) est servie en premier tant qu'elle n'est pas vide. Une fois vide, l'algorithme d'ordonnancement sert la file d'attente de priorité 2 et ainsi de suite. Pendant le temps de service de la file d'attente de priorité 2, si un paquet arrive dans la file d'attente de priorité 1 (priorité supérieure), il sera servi puis la classe 2 reprend la main.
  Dans chaque file d'attente, plusieurs flux de même classe peuvent être agrégés. Les paquets d'une même classe sont servis suivant une politique FIFO.

Les avantages de cet algorithme d'ordonnancement sont les suivants :
- PQ est simple à implémenter et ne nécessite pas beaucoup de temps de traitement.
- Il permet de gérer plusieurs classes et de fournir des traitements différents pour les paquets des différentes classes.

Ses inconvénients sont :
- si le trafic de plus haute priorité n'est pas contrôlé à l'entrée du réseau, les paquets des autres priorités peuvent subir des délais très élevés.
- Tous les flux d'une même classe sont agrégés dans une même file d'attente. Par conséquent, un seul flux peut perturber le fonctionnement des autres flux de la même classe.

- CBQ : Class Based Queing : fournit une file séparée pour chaque classe. En général, chaque file est gérée en FIFO. Aucune garantie de délai et de débit ne peut être fournie aux flux individuels.
- WRR : Weighted Round Robin est un algorithme classique d'ordonnancement de type tourniquet qui permet de retirer les paquets dans différentes files selon une certaine pondération par file.
- WFQ : Weighted Fair Queuing est une variante de WRR dans laquelle les poids sont couplés aux débits réservés. Cet ordonnanceur peut fournir une garantie de délai aux flux individuels, mais il ne peut pas séparer la garantie de débit de la garantie du délai. Le problème résultant est qu'un flux qui a une petite bande passante allouée peut obtenir un délai de bout en bout important. Un nombre important de variantes ont été proposées.
- GPS : Generalized Processor Sharing est défini comme le modèle fluide de trafic et sert de modèle théorique de référence.
- EDF : Earliest Deadline First est une forme dynamique d'ordonnancement par priorité. A chaque paquet on assigne un délai limite qui est la somme de la date d'arrivée et de la garantie de délai. Couplé avec un trafic shaper, EDF peut séparer la garantie de délai et de débit.

Un certain nombre de critères peuvent être identifiés pour classifier ces mécanismes d'ordonnancement et de gestion de file. On distingue les propriétés suivantes :
- isolation : protection vis à vis du trafic des autres utilisateurs,
- équité : accès des différents flux à la capacité,
- complexité : en terme de surcoût d'implémentation et de contrôle.

Un ordonnanceur doit optimiser un certain nombre de critères, (délai, débit, etc), mais doit rester très simple. En effet, il est censé opérer à la vitesse du lien, ce qui l'oblige, pour un débit OC-48, à prendre une décision en 100ns pour chaque paquet. Dans un ordonnanceur à classement par priorité, une variable globale, le temps virtuel, est associée à chaque lien de sortie de l'équipement. Une estampille, calculée comme fonction de cette variable, est associée à chaque paquet du système. Les paquets sont classés selon leur estampille et sont transmis dans cet ordre. La complexité d'une implémentation de tout algorithme basé sur la priorité dépend de la complexité de calcul de l'estampille associée à chaque paquet.

Plan contrôle

Ce sont les mécanismes qui concernent la configuration des routeurs pour assurer que certains paquets reçoivent un traitement spécial et qu'un certain nombre de règles concernant l'utilisation des ressources sont bien appliquées. Trois types de mécanismes sont identifiés :
– le contrôle d'admission
– le contrôle de la politique
– la gestion des ressources

Le contrôle d'admission

Le contrôle d'admission doit décider s'il y a assez de ressources dans le réseau pour accepter une nouvelle connexion. Il s'agit de calculer correctement une région d'admission pour ne pas refuser inutilement l'accès à certains flux tout en évitant les violations de QoS. Il existe trois approches pour contrôler l'admission d'un nouveau trafic : l'approche déterministe, l'approche statistique et l'approche basée sur la mesure. Les deux premières utilisent une estimation *à priori* tandis que la dernière est basée sur des mesures courantes de certains paramètres. L'approche déterministe effectue un calcul du pire des cas pour éviter toute violation de QoS. Cette technique, si elle est efficace pour les trafics fluides, l'est beaucoup moins pour les trafics en rafale et conduit à une sous-utilisation des ressources. Les deux autres approches autorisent une faible probabilité de violation de la QoS pour optimiser l'utilisation des ressources.

Le contrôle de la politique

La politique de QoS spécifie les règles d'accès aux ressources et aux services selon des critères administratifs. Elle définit, dans un domaine administratif donné, si tel utilisateur, site ou application peut accéder aux ressources et dans quelles conditions. Depuis quelques années, des architectures basées sur les politiques ont été définies, avec des modèles à trois tiers ou à deux tiers, un point de mise en oeuvre de la politique, *Policy Enforcement Point PEP*, un point de prise de décision, *Policy Decision Point PDP*, qui peuvent être combinés en une seule entité, et un dépôt des politiques. Pour échanger les informations entre ces entités, des protocoles standards sont nécessaires. *COPS, Common Open Policy Service*, [RFC2748] permet les échanges entre PDP et PEP. *LDAP, Light Weight Access Protocol* [YEO95] permet l'accès aux politiques stockées dans des *PIB : Policy Information Base*. Un nombre important de travaux autour des réseaux basés sur les politiques, *Policy Based Network*, ont vu le jour ces dernières années. On commence à voir apparaître aussi des propositions de fusion des aspects politiques de sécurité et politiques de QoS.

La gestion des ressources

La gestion des ressources entre domaines administratifs est un problème très important et a une grande influence sur la qualité de service de bout en bout. Dans

l'architecture DiffServ, un courtier de bande passante, le *Bandwidth Broker BB*, a, dans chaque domaine administratif, la mission de contrôler les opérations au niveau des routeurs de bordure. Il inclut des fonctions de PDP tandis que le routeur de bordure sert de PEP.
L'architecture d'un *BB* a quelques similarités avec le protocole de routage *BGP4* qui sert de protocole standard pour le routage inter-domaine. Le routage et la QoS sont souvent présentés de manière indépendante, mais certains proposent une fusion du routage et de la QoS pour résoudre le problème de la QoS [ROB01].

Dans notre travail, plusieurs mécanismes de l'approche dite dans le réseau ont été utilisés, comme le WFQ, RED et le contrôle d'admission. L'avantage de ces mécanismes est qu'ils sont implémentés dans le coeur du réseau, ce qui les rend transparent vis à vis du client.

1.7.1.2 Architectures de QoS-IP standard

Depuis la fin des années 90, ces mécanismes sophistiqués du plan contrôle et du plan données se développent dans les équipements. Ainsi, avec un tel soutien des fabricants et des organismes de standardisation, on peut penser que des services évolués vont se déployer à large échelle dans les années futures. Mais le choix de l'architecture et des modèles de services appropriés à Internet reste encore une question ouverte. Différentes architectures ont été proposées dans Internet. Nous les présentons selon le niveau de garantie offert. Cette présentation correspond aussi à leur chronologie d'apparition.

IntServ : Garanties strictes et réservations de ressources

Plusieurs modèles de gestion de qualité de service adoptent la solution de réservation des ressources afin de garantir des services aux utilisateurs. Une telle approche nécessite un protocole de réservation qui se fait généralement par une signalisation explicite dans le plan contrôle. Les seuils minimum de qualité sont déterminés par les programmeurs d'application ou par les utilisateurs puis transmis au gestionnaire de QoS. Une phase de négociation est entamée entre le gestionnaire et l'application pour déterminer si ces seuils peuvent être garantis. On détermine alors les besoins en ressources système et réseau, ces ressources sont ensuite réservées pour garantir le service.
Ainsi dans un réseau basé sur la technologie ATM, on réserve au préalable un canal virtuel avec une bande passante et des caractéristiques de débit (CBR, VBR) fixes. C'est au gestionnaire du réseau (fournisseur) d'assurer que le service offert l'est bien avec les caractéristiques spécifiées et négociées.
L'architecture **IntServ** est la première à avoir été élaborée pour fournir à l'Internet un paradigme prenant en considération les services temps-réels. Le but est de fournir un lien de communication à **qualité constante** en terme de débit, délai et taux d'erreur dans Internet. En fait, la plus grande force de cette architecture réside dans l'approche systématique et rigoureuse qu'elle propose au niveau de :
– l'identification des services intéressants

– l'analyse des spécifications réseaux nécessaires pour les supporter
– la proposition d'un cadre de référence pour l'implémentation

L'architecture Intserv est donc un cadre de services. Les principaux standards ont été développés entre 1995 et 1998 [RFC2212, RFC2215]. Les services qui ont été identifiés sont le service garanti et le service à charge contrôlée.
Le **service garanti, GS**, est défini pour émuler au mieux un circuit virtuel dédié. Il fournit des bornes, que l'on peut prouver mathématiquement, sur les délais de file d'attente de bout en bout. Le **service à charge contrôlée, CLS**, est équivalent à un service Best Effort sous des conditions de faible charge. Ainsi, il est mieux que le Best Effort mais ne fournit pas un service garanti borné comme le promet le service garanti.
Les composants de l'architecture IntServ sont le protocole de réservation de ressource, le contrôle d'admission et les mécanismes d'ordonnancement des paquets dans les routeurs. Le protocole RSVP [RFC2210] à états dynamiques, soft state, permet l'acheminement d'une requête de réservation ainsi que la réservation effective des ressources le long d'un chemin. Ce protocole de réservation, retenu par le groupe IntServ, présente cependant plusieurs faiblesses qui en rendent l'implémentation complexe. Certaines évaluations d'implémentation du protocole RSVP ont montré que le surcoût induit par le traitement d'un flux augmente linéairement avec le nombre de flux. Cela pose un problème de passage à l'échelle d'une part, dans le plan donnée pour le traitement des paquets, et d'autre part, dans le plan contrôle pour la signalisation. Les principaux problèmes d'IntServ sont donc l'extensibilité, le maintien des états de contrôle et d'ordonnancement pour chaque flux dans tous les routeurs, les fonctions de gestion et de comptabilité ainsi que l'interface application-réseau requise. Ces inconvénients majeurs ont arrêté le déploiement d'Intserv dans l'Internet.

Garanties statistiques : approche DiffServ

Dans l'approche par prioritisation, le trafic est classifié et les ressources sont partagées selon des critères de gestion de la bande passante. Les classifications attribuent un traitement préférentiel aux applications qui ont des demandes plus exigeantes en terme de QoS. On a une signalisation implicite qui s'opère dans le plan donnée (identifiant de classe dans un champ d'entête). Pour garantir un niveau de service requis, généralement on réserve plus de ressources car il est difficile de caractériser précisément le trafic à l'avance. Les mécanismes de prioritisation, dans lesquels les paquets sont labellisés suivant leur priorité et sont traités différemment au niveau des routeurs offrent des solutions à priori plus légères et des garanties de qualité de service non plus absolues au niveau flux mais statistiques. Un réseau qui n'offre pas de garanties de service strictes, est, à priori, moins coûteux à mettre en oeuvre.
Ainsi, pour contourner les faiblesses des propositions faites par le groupe IntServ, en 1998, un nouveau groupe de l'IETF nommé Differentiated Services Group, **DiffServ** a suggéré de mener des investigations dans cette direction : au lieu de se concentrer sur les flux individuels pourquoi ne pas gérer des agrégats de trafic (large ensemble

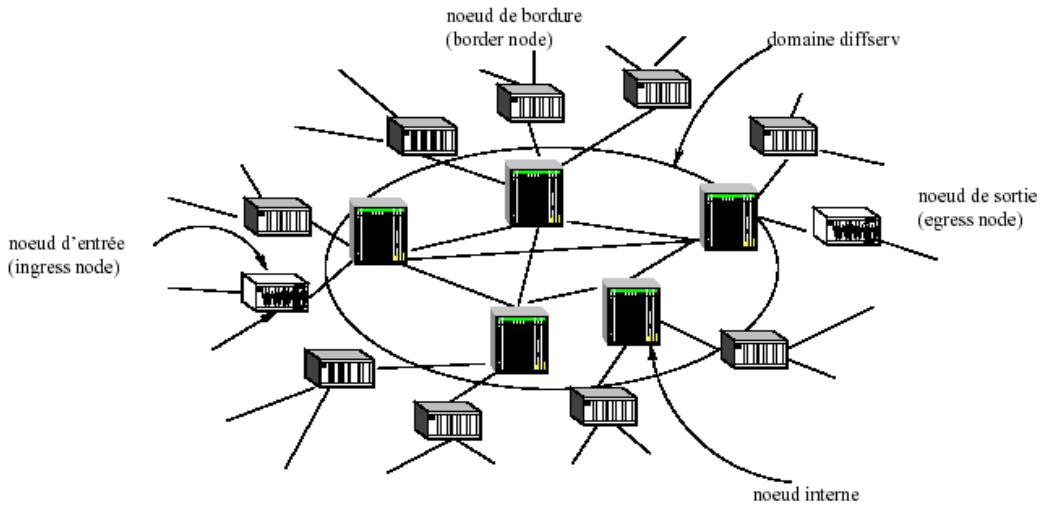

Figure 1.2 – Architecture DiffServ

de flux ayant les mêmes besoins de service) et de discriminer les paquets en fonction de leur priorité (precedence). Cette idée a conduit au concept de services différenciés - à l'opposé des services intégrés - qui ont l'avantage de pouvoir être plus facilement implémentés même dans les réseaux existants [RFC2475]. Diffserv se présente donc comme une architecture plus extensible et plus facile à gérer. Le concept de services différenciés est basé principalement sur un schéma orienté **paquet** et sur le fait que, dans le réseau, la notion de flux d'utilisateur final n'existe pas. Un autre objectif majeur et initial de la différentiation de services est de permettre de facturer différemment les services d'Internet. Plusieurs travaux ont été précurseurs de l'architecture DiffServ et des modèles de services standards.
Le modèle DiffServ repose sur la classification et le marquage des paquets IP. Le comportement sur le chemin PHB (Per Hop Behavior) inclut le traitement différentiel d'un paquet individuel implémenté à l'aide d'une discipline de gestion de file d'attente et/ou d'une discipline de service de file (mécanisme d'ordonnancement). L'architecture DiffServ tout comme l'architecture Intserv définit un cadre de services. Plusieurs services ont été étudiés. Deux ont été standardisés : le service Expedited Forwarding (EF) et le service Assured Forwarding (AF).

Le concept d'**Expedited Forwarding** (EF) [RFC2598] est de créer un sous-réseau à faible latence où les paquets ne sont jamais (ou très peu) détruits. Les conditions d'acheminement des paquets qu'offre ce sous-réseau virtuel sont a priori favorables aux flux temps-réel puisque le temps de traversée du réseau est diminué. Cependant, les flux élastiques y obtiennent évidemment aussi de meilleures performances puisque le chemin est sans perte ni délai de file d'attente.

Le PHB **Assured Forwarding** (AF) [RFC2597], sépare les paquets dans n files qui ont chacune une part de débit déterminé. Au sein d'une file, il existe m sous-

classes dont le taux de perte est plus ou moins élevé. Bien que les valeurs de m et n ne soient pas standardisées, on les trouve souvent instanciées à $n = 4$ et $m = 3$. Certains distinguent les services or, argent ou bronze. En effectuant une différenciation sur le taux de perte, ce service n'est pas vraiment destiné à améliorer le temps de traversée du réseau et profite essentiellement aux flux élastiques.
Le service premium, implémenté à l'aide du PHB " expedited forwarded " a été le premier déployé dans les réseaux expérimentaux. Une première démonstration du service Premium sur Abilène a été réalisée en février 1999. Un constat établit à la fin de l'année 2001 [TEI01] fait état de l'ampleur des difficultés de déploiement de Premium Service dans le Qbone. Dans le projet VTHD, la différentiation de service est aussi en cours d'évaluation. Aucun test en Europe n'est aujourd'hui réellement probant et le déploiement de Premium Service n'est pas encore très large. Ceci devrait cependant évoluer rapidement.

1.7.2 Solutions dites aux extrémités

Pendant que s'élaboraient ces architectures de qualité de service au coeur du réseau, des recherches ont été menées pour trouver des mécanismes capables d'absorber les fluctuations de performances de l'interconnexion. En l'absence de garanties de service strictes en terme de délai ou de taux de perte à l'intérieur du réseau, des algorithmes adaptatifs ont donc été développés aux extrémités.
Traditionnellement c'est la couche transport, première couche de bout en bout qui réalise l'adaptation (définition de la couche transport OSI). Sa complexité dépend de la qualité de service offerte par le réseau sous-jacent. Le modèle ATM, offre aussi plusieurs services d'adaptation à l'application. Dans la couche d'extrémité AAL, application adaptation layer, les paquets d'information sont mis en forme et les informations utiles pour la recomposition de l'information émise chez le récepteur sont insérées. ATM propose 5 couches d'adaptation, de AAL1 à AAL5, correspondant aux différentes classes de services offertes au niveau ATM. Mais ces couches n'implémentent pas d'algorithme spécifique d'adaptation aux variations de performances du réseau puisque les canaux offerts par la couche réseau ATM ont une qualité prévisible.
Dans le modèle Internet, deux principaux protocoles de transport sont disponibles : le protocole UDP qui ne fait que du multiplexage de flux et le protocole fiable très sophistiqué, TCP, qui réalise l'adaptation aux pertes de paquets et le contrôle de congestion. Comme les applications multimédia temps-réel ont plus d'exigence de délai que de fiabilité, TCP ne convient pas et ce sont les applications, qui , pour des raisons de performance, sont elles-mêmes en charge de l'adaptation. Dans le chapitre suivant, nous étudions le protocole TCP plus en détail. Les avantages ainsi que les limitations de ce protocole sont aussi étudiés. Dans notre travail, nous nous sommes intéressés aux applications basées sur le protocole TCP, vu que ces dernières représentent plus que 97% de la charge totale en Octets (voir tableau 1.2) introduite dans le réseau.

1.8 Conclusion

Au niveau réseau IP, des mécanismes de gestion active de file d'attente, tels que RED, ont été développés depuis plus d'une dizaine d'années pour limiter les congestions dans Internet. Ils sont conforment à la philosophie IP et se déploient progressivement. Mais ces mécanismes ne fournissent pas explicitement des services évolués aux applications. Comme le sur-dimensionnement, ils sont une des réponses partielles à l'amélioration des performances globales de l'Internet et au problème de la qualité de service. Les propositions IntServ/RSVP et DiffServ ont cherché à apporter des solutions pour la fourniture de services différenciés dans IP avec des garanties de QoS tout en conservant un service Best Effort bien éprouvé. Dans le plan contrôle, IntServ requiert une signalisation explicite ainsi que le stockage et le traitement d'états par flux. Dans le plan données, IntServ requiert de la classification, de l'ordonnancement et de la gestion de buffer par flux. Cet ensemble de contraintes s'est avéré être un lourd handicap pour le déploiement à large échelle de IntServ. Le modèle DiffServ paru donc dès le début attractif vue sa simplicité. L'IETF n'a pas explicitement recommandé des services de bout en bout comme dans IntServ, mais s'est donné pour objectif de fournir un nombre suffisamment riche de PHB à partir desquels, des services de bout en bout pourraient être construits. Ainsi le groupe DS a standardisé un petit nombre de PHB. Le service Premium est apparu comme celui pouvant le mieux répondre aux besoins des applications temps-réel qui requièrent des bornes précises pour les pertes et la gigue. Un service Premium visant à offrir un service de type liaison spécialisée s'appuie sur trois mécanismes de QoS différents : la différentiation par marquage des paquets au sens DiffServ, le sur-dimmensionnement du service au niveau des routeurs d'entrée et le contrôle d'admission de type Intserv en bordure. Mais le déploiement de Premium Service s'avère lui aussi très complexe et peu concluant. Le modèle d'architecture DiffServ, tel qu'il a été défini au départ, ne semble pas être la solution de QoS de l'Internet du futur. Pour cela, nous avons dirigé nos efforts vers des solutions de QoS plus simples telles que le contrôle d'admission basé sur la classification des flux TCP selon le RTT et la taille d'un flux.
Dans le chapitre suivant, nous expliquons pourquoi et comment nous avons poursuivi notre travail dans cette direction.

Chapitre 2

Partage de la bande passante entre les flux TCP

Pendant une période de temps assez importante, TCP a réussi, en coordination avec le protocole IP, à assurer un moyen de transport solide et efficace pour la plupart des applications existantes sur Internet. Néanmoins, avec la croissance importante de cette dernière en terme de nombre de demandes et en terme de nouveaux services à implémenter, les points faibles dans TCP/IP sont devenus de plus en plus apparents. En fait, les solutions proposées pour améliorer la performance du protocole TCP sont nombreuses, allant du mécanisme "Fast retransmit/Fast recovery" jusqu'à la dernière version du protocole TCP qui est "New Reno". Ces différentes propositions ont permis d'améliorer les performances des applications utilisant le protocole TCP. Mais, elles restent incapables de protéger ces dernières quand le réseau est chargé, ou bien contre l'agressivité des applications " Streaming ".

2.1 Le contrôle de congestion

Le grand avantage que TCP a apporté à l'Internet est son mécanisme qui lui permet de contrôler la congestion au sein du réseau.
L'idée derrière le contrôle de congestion pratiqué par TCP est relativement simple. Elle consiste à ajuster les débits de transmission des sources émettrices en fonction de la charge du réseau ; en d'autres termes, si une source détecte une certaine perte de paquets, cette perte est interprétée comme étant due à un état de congestion dans le réseau. Ainsi, la source réagit en diminuant le débit de transmission afin de ne pas aggraver la situation en fournissant plus de trafic au réseau congestionné.

2.2 Le contrôle de flux

Si le contrôle de congestion est le mécanisme mis en oeuvre par TCP pour éviter de congestionner le réseau, le contrôle de flux est le mécanisme mis en oeuvre pour éviter de faire déborder la mémoire en réception du récepteur. Ces deux mécanismes

impliquent qu'une entité TCP qui s'apprête à transmettre un segment de données doit examiner préalablement si deux conditions sont satisfaites :

1. Si le réseau possède les ressources pour router le segment.
2. Si l'entité TCP réceptrice est prête pour recevoir le segment (si elle a l'espace pour le stocker et le traiter).

Chacune de ces deux conditions est matérialisée par une variable appelée fenêtre. Cette dernière évolue avec la transmission de segments de données et la réception d'acquittements :
- CWND ou fenêtre de congestion : représente le nombre d'octets (ou paquet) pouvant être transmis sans risque de congestionner le réseau.
- SWND ou fenêtre d'émission : représente le nombre d'octets (ou paquets) pouvant être transmis sans provoquer de débordement dans la file d'attente du récepteur.

En conséquence, une entité TCP (1) qui tend à transmettre des octets doit s'assurer que le nombre des octets transmis est inférieur à la taille de sa fenêtre de congestion et à la taille de la fenêtre de réception de l'entité réceptrice (2). Cela se traduit par l'équation suivante :

$$Nombre\ des\ octets\ transmis < Min(cwnd1, swnd2) \tag{2.1}$$

En réalité, le contrôle de flux ne pose pas de problèmes pour le protocole TCP. En effet, les entités TCP s'échangent, en permanence, les tailles de leurs fenêtres de réception grâce à un champ réservé qui fait partie de l'entête TCP.
Ce qui semble être le moins évident et qui constitue un sujet de controverse, c'est le contrôle de congestion. A savoir que les solutions proposées n'affectent pas seulement le protocole TCP mais également plusieurs entités au sein du réseau.

2.3 Gestion de la fenêtre CWND de TCP

Avant de montrer comment le protocole TCP gère sa fenêtre de congestion, il est important de mentionner quelques caractéristiques concernant le contrôle de congestion dans TCP.
En principe, le contrôle de congestion de TCP ne se base pas sur une signalisation entre les entités TCP et les routeurs dans le réseau. Les entités TCP induisent l'état du réseau, et, par conséquent, la taille de la fenêtre cwnd, par l'unique observation du flux de segments émis et des acquittements reçus.
Une source TCP sera, notamment, attentive aux pertes des segments émis car elle assimilera ces pertes à des congestions du réseau et agira en conséquence. Le réseau n'impose aucun contrôle direct sur les sources, ces dernières appliquent donc une auto-discipline.
La technique qui vient proposer une nouvelle approche dans l'évaluation de l'état du réseau est la technique Explicit Congestion Notification (ECN). Cette technique recommande un marquage explicite des paquets (par les routeurs congestionnés), de

façon à permettre aux éléments du réseau de signaler aux sources la présence de la congestion. Pour mettre en oeuvre la technique ECN il est nécessaire de faire évoluer le protocole TCP de manière à lui permettre d'interpréter le contenu de quelques champs supplémentaires dans l'entête TCP qui étaient des champs réservés.
Supposons pour le moment que la détection de la congestion est liée à la détection des pertes. La figure 2.1, ci-dessous montre comment une entité TCP se comporte lors de la transmission des données.

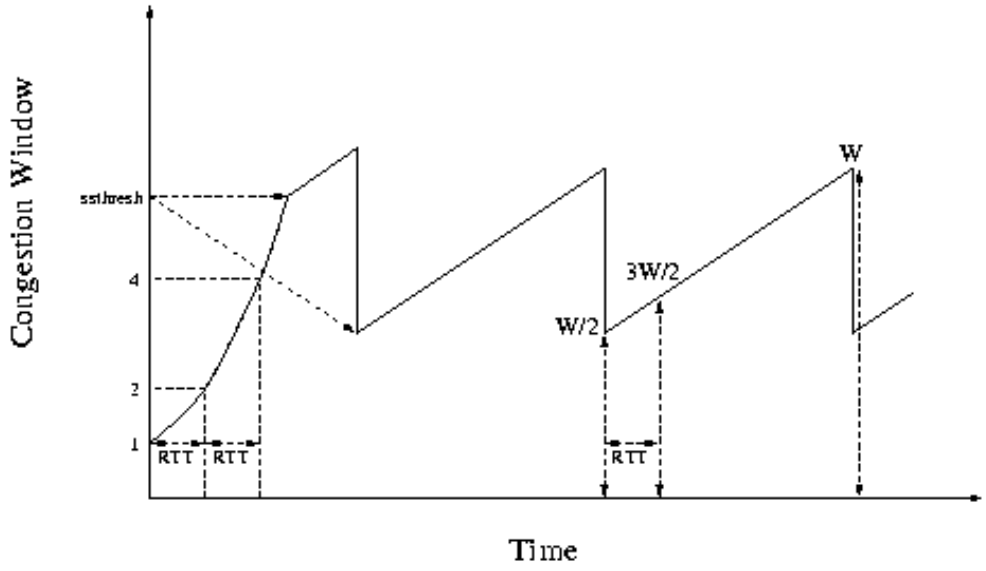

Figure 2.1 – Comportement de la fenêtre Cwnd TCP

Une entité source TCP passe par trois états différents de contrôle de congestion :
– **Slow Start** (démarrage lent)
– **Congestion Avoidance** (évitement de congestion)
– **Fast Recovery** (recouvrement rapide)

Chaque état possède ses modalités propres pour la gestion de la fenêtre *cwnd*. Les changements d'état sont opérés par une entité, lorsque celle-ci détecte une perte de segment ou lorsque sa fenêtre atteint un seuil prédéfini.

2.4 Les différentes améliorations proposées pour le protocole TCP

Les deux mécanismes "Slow-Start" et "Congestion Avoidance" ont été proposés dans la première version du protocole TCP nommée "TCP Tahoe" [RFC793]. Ils permettent de contrôler la fenêtre de congestion.

2.4.1 Slow Start (SS)

Au départ, la fenêtre cwnd est initialisée à 1 MSS (Maximum Segment Size). L'entité TCP source utilise ce droit pour émettre un segment. Ensuite, elle attend l'acquittement du routeur. Au retour de l'acquittement, la source recouvre son droit initial, et, de plus, reçoit le droit d'émettre un segment supplémentaire. Elle peut émettre donc une salve de deux segments. Le retour des deux acquittements (ou d'un acquittement cumulé), provoque, en appliquant le même raisonnement, la transmission de 4 paquets, et ainsi de suite. A noter, l'allure exponentielle de la courbe d'évolution de la fenêtre en fonction du temps.

2.4.2 Congestion Avoidance (CA)

En phase de slow start, la fenêtre de congestion démarre à 1 MSS. Puis, elle augmente rapidement (exponentiellement) ; d'où le risque d'une saturation rapide du réseau.
C'est pour cela que, TCP s'impose un infléchissement du rythme d'augmentation quand la fenêtre de congestion atteint un certain seuil "threshold". Dans ce cas, le rythme d'augmentation devient linéaire ; on entre, alors, dans la phase dite de congestion avoidance.
En phase de congestion avoidance, la norme exige que la fenêtre cwnd n'augmente pas plus de 1 MSS par RTT.
Les processus "slow start" et "congestion Avoidance" [RFC2581] sont essentiels à la stabilité actuelle de l'Internet. Ces mécanismes ont été conçus pour faciliter l'utilisation de réseaux ne fournissant pas d'informations explicites sur leur congestion. Des mécanismes expérimentaux comme la recommandation [RFC2481] tendent vers la notification explicite de la congestion. Le mécanisme ECN permet quant à lui d'obtenir des informations avant la perte des paquets et d'éviter ainsi leur retransmission.
Les connexions TCP présentant des taux d'erreurs élevés sur leurs chemins ne conviennent pas avec les phases "slow start" et "congestion avoidance". En effet, les taux d'erreurs élevés provoquent une interprétation ambiguë des pertes car ils empêchent l'émetteur de savoir si les pertes détectées sont dues à une congestion ou à une corruption de données. TCP préfère alors la sécurité et présume que les pertes sont dues à une congestion.
Tous les mécanismes TCP conformes aux normes instaureront les phases "Slow Start" et "congestion avoidance", comme stipulé par le mot-clé "DOIT" dans le document STD 3 [RFC1122].
Dans cette première version, les pertes sont détectées par l'expiration d'un temporisateur (Time Out), au bout duquel les entités présument qu'il y a une congestion et commencent à diminuer leur débit. Cette phase se traduit par un passage de nouveau à l'état "Slow Start" avec un CWND réinitialisé à 1. De plus, le seuil d'infléchissement "threshold" est réduit jusqu'à la moitié de la fenêtre courante (ou à 2*MSS, si cette moitié est plus petite que 2*MSS).
Dans la section suivante, nous présentons l'algorithme Fast Retransmit/Fast Reco-

very qui permet de détecter plus rapidement une perte grâce aux trois acquittements dupliqués. C'est l'amélioration qui a été proposée dans la version TCP Reno.

2.4.3 Processus "Fast Retransmit/Fast Recovery" (TCP Reno)

TCP permet une transmission fiable des données sous la forme d'un flux d'octets vers une application. Ainsi, lorsqu'un segment est perdu (qu'il s'agisse d'une congestion ou d'une perte de transmission), l'émetteur TCP doit attendre de recevoir les informations manquantes avant de transmettre les données à l'application réceptrice. Le récepteur TCP détecte, au fur et à mesure, les segments manquants qui arrivent dans le mauvais ordre.
Lorsqu'il reçoit des données dans le mauvais ordre [RFC2581], TCP doit alors immédiatement envoyer un accusé de réception stipulant le numéro du prochain segment attendu, afin que l'émetteur puisse retransmettre les données requises aussi rapidement que possible et ainsi continuer la transmission de données vers l'application réceptrice. Lorsqu'un accusé de réception porte le même numéro qu'un accusé de réception précédemment envoyé pour le dernier segment ordonné reçu, on dit que ces accusés de réception sont en double.

Les réseaux IP étant autorisés à réordonner les paquets, le récepteur peut alors envoyer des accusés de réception en double pour des segments qui arrivent en désordre. Ceci, en raison de modifications d'itinéraire, de retransmission de niveau couche, etc. Lorsqu'un émetteur TCP reçoit un accusé de réception en trois exemplaires, le mécanisme "fast retransmit" [RFC2581] lui permet d'inférer qu'un segment est perdu. L'émetteur retransmet ce qu'il considère être le segment perdu sans attendre la temporisation de la retransmission. Ce procédé permet ainsi de gagner du temps au lieu d'attendre l'expiration du temporisateur.

Après cette phase, un émetteur réduit de moitié sa fenêtre de congestion et invoque l'algorithme "fast recovery" [RFC2581], par lequel il appelle la phase "congestion avoidance" à partir d'une fenêtre de congestion réduite. Il n'a alors pas recours à la phase "slow start" depuis une fenêtre de congestion d'un seul segment comme il le ferait suite à une temporisation de retransmission. Etant donné que l'émetteur reçoit toujours des accusés de réception en double, il sait que le récepteur reçoit les paquets envoyés.
En général, TCP agrandira sa fenêtre de congestion au delà du produit délai bande passante. La stratégie de contrôle de congestion de TCP répond donc au concept "additive-increase, multiplicative-decrease", signifiant que si d'autres pertes sont détectées avant que la fenêtre de congestion retrouve sa taille initiale, après une réduction de moitié, la fenêtre de congestion donne alors l'effet d'une "spirale descendante" avec de nouvelles réductions de 50%. Même en utilisant les algorithmes "Fast Retransmit/Fast Recovery", l'émetteur réduira de moitié la fenêtre de congestion pour chaque fenêtre présentant un ou plusieurs segments perdus. Puis, il réouvrira la fenêtre par un segment supplémentaire pour chaque accusé de réception reçu par

fenêtre de congestion.
Lorsqu'une connexion traversant un lien, perd un ou plusieurs segments au cours de la phase de restauration, une nouvelle réduction de moitié se produit, mais, cette fois sur une fenêtre de congestion réduite. Cette spirale descendante continuera à maintenir la fenêtre de congestion en dessous de la capacité du lien jusqu'à ce que la connexion soit capable de se restaurer complètement par augmentation additive et sans perte. Lorsque le taux de pertes est constamment élevé, on ne parle naturellement pas de spirale descendante. La fenêtre de congestion reste réduite. Dans ce cas, la phase d'augmentation multiplicative "slow start" sera réduite et la fenêtre de congestion reste petite pendant la connexion TCP. Dans les liens présentant un taux de pertes élevé, la fenêtre TCP peut rester relativement petite pendant de longs moments.
Une petite fenêtre, notamment une fenêtre constituée de moins de quatre segments, empêche effectivement l'émetteur de tirer partie du processus "Fast Retransmit". En outre, une restauration efficace de différentes pertes au sein d'une seule et même fenêtre nécessite l'adoption de nouvelles méthodes (NewReno [RFC2582]).

2.4.4 Accusés de réception sélectifs (TCP Sack)

Il est nécessaire de noter que plusieurs modifications ont été proposées pour améliorer le comportement de TCP envers le contrôle de congestion. Il est important de mentionner l'existence des versions Tri-S et Vegas qui proposent un autre comportement de TCP dans la phase congestion avoidance.
Ces deux versions ne recommandent pas un rythme d'augmentation linéaire dans la phase congestion avoidance uniquement si le réseau est non congestionné. Pour les deux algorithmes, si le RTT indique une augmentation de délai, cela sera interprété comme étant dû à une congestion. Les sources **diminuent** ou bien **elles fixent les tailles de leurs fenêtres** au lieu de les augmenter. Bien que ces deux algorithmes possèdent le potentiel de diminuer le taux de pertes dans le réseau, l'intérêt de ces deux algorithmes dans le cas d'une congestion persistante n'est pas très évident. En effet, en modifiant le scénario d'augmentation linéaire puis de diminution rapide, ces modifications ne semblent pas assurer un partage juste entre les différentes connexions qui utilisent le lien.
Les accusés de réception sélectifs [RFC2018] permettent de pallier la perte de plusieurs segments par fenêtre sans avoir à effectuer un (ou plusieurs) aller-retour par perte.
La [RFC2883] propose une petite extension aux accusés de réception sélectifs (SACK) que TCP permet de recevoir afin de fournir plus d'informations concernant l'ordre de livraison des segments. Cette méthode permet ainsi un fonctionnement plus robuste dans un environnement de paquets réordonnés ou répliqués, de perte d'accusé de réception et/ou de temporisations de retransmission anticipées. Dans ce document, sauf indication contraire, l'accusé de réception sélectif (ou "SACK") réfère à la combinaison de [RFC2018] et de [RFC2883].
Les accusés de réception sélectifs sont particulièrement pratiques dans les réseaux dits éléphants (LFN - Long Fat Network) en raison de la longue durée d'un aller-

retour dans ces environnements, tel que cela est expliqué dans la section 1.1 de la [RFC1323]. En outre, ces accusés de réception sont également pratiques lorsque de grandes fenêtres sont requises, car, au moment de la congestion, plusieurs pertes au sein d'une même fenêtre sont susceptibles de se produire. Dans ce cas, l'accusé de réception sélectif offre alors l'avantage d'accélérer la restauration et d'empêcher toute réduction inutile de la taille de la fenêtre.
Dans le cas où les accusés de réception sélectifs ne peuvent être activés aux deux extrémités de la connexion, les émetteurs TCP peuvent utiliser la norme NewReno [RFC2582] pour mieux manipuler ces accusés de réception et faire face à plusieurs pertes au sein d'une même fenêtre.

2.4.5 TCP NewReno

Cette implémentation est une évolution de la version TCP Reno. L'algorithme "fast recovery" considère qu'il a terminé son travail dès qu'il reçoit un acquittement supérieur à celui qui avait été dupliqué.
Dans la version NewReno, décrite dans la [RFC2582] et datant de 1999, les auteurs proposent de rester dans le mode "fast recovery" tant que l'on n'a pas reçu d'acquittement pour le segment envoyé portant le plus grand numéro de séquence. Si on reçoit un acquittement supérieur à celui qui était dupliqué, mais qui n'acquitte pas tout ce qui a été envoyé, on reste en mode "fast recovery". On considère ainsi que le segment suivant a aussi été perdu. C'est une méthode permettant de récupérer rapidement lorsque plusieurs segments ont été perdus, sans avoir besoin de mettre en oeuvre les acquittements sélective (SACK). Même si on implémente SACK, il est toujours intéressant d'y ajouter NewReno, car, pour mettre en oeuvre les acquittements sélectifs, les deux extrémités doivent reconnaître ces extensions de TCP.

2.4.6 TCP Vegas

Le Slow Start permet un démarrage en douceur tout en atteignant rapidement une vitesse de croisière. Une fois on atteint la phase congestion avoidance, la taille de la fenêtre de congestion augmente systématiquement de 1 avec une période de temps RTT. On tente d'augmenter doucement jusqu'à obtenir des pertes, et on en déduit un point de fonctionnement.
En clair, pour éviter les congestions, on en provoque volontairement pour voir jusqu'ou on peut aller ! L'algorithme Vegas [BRA94] propose de trouver la vitesse acceptable sans provoquer de congestion.
On remplace le mode congestion avoidance par Vegas. Celui-ci peut augmenter la fenêtre d'une unité pendant le temps RTT, mais aussi rester constant, voire même diminuer. On décide quel cas adopter en fonction de la vitesse mesurée. En effet, si le réseau est proche de la congestion, les files des routeurs sont pleines et le RTT augmente. Une augmentation de RTT provoquera donc une réduction de la taille de la fenêtre de congestion. Une diminution de RTT indique un réseau dégagé, donc on peut augmenter la taille de la fenêtre. En cas de perte, au lieu de réduire la fenêtre

de moitié, on se contente de 3/4.
Bien que cet algorithme possède le potentiel de diminuer le taux de pertes dans le réseau, son intérêt dans le cas d'une congestion persistante n'est pas très évident. En effet, en modifiant le scénario d'augmentation linéaire puis de diminution rapide, ces modifications ne semblent pas assurer un partage juste entre les différentes connexions qui utilisent le lien.

2.5 Résumé des recommandations

Le mécanisme "Fast Retransmit/Fast Recovery" a permis aux entités de réparer plus rapidement les pertes sans compromettre la sécurité du contrôle de congestion. Pour que ce mécanisme fonctionne, la taille de la fenêtre de congestion doit être assez large (supérieure à trois paquets). Ceci, permet au récepteur, au moment de la détection d'une perte, d'envoyer trois accusés de réception dupliqués avant l'expiration du délai de temporisation de retransmission (TO), forçant ainsi la phase complète "slow-start". Nous pouvons ainsi dire que le mécanisme "Fast Retransmit/Fast Recovery" convient plus aux connexions longues qui ont beaucoup de données à envoyer. Ceci leur permet d'avoir des fenêtres de congestions suffisamment larges tout au long de leur transfert, ainsi, les pertes seront détectées, plus souvent, grâce aux trois acquittements dupliqués qu'à l'expiration du TO.
La version TCP SACK ne peut être utilisée que si elle est installée sur les deux hôtes qui font l'échange ce qui explique pourquoi elle est très peu utilisée. La difficulté présente dans la version TCP Vegas consiste à calculer le RTT à chaque fois que l'on reçoit un acquittement, pour savoir si on augmente ou diminue la fenêtre de congestion. Cette version n'a pas été largement déployée et on trouve ainsi que les versions TCP Reno et New Reno sont les plus utilisées. Dans notre travail, c'est la version TCP New Reno qui a été utilisée, cette dernière implémente le mécanisme "Fast Retransmit/Fast Recovery", d'où une détection et une récupération plus rapide des pertes. Elle permet aussi, quand plusieurs segments dans la même fenêtre de congestion sont perdus, de restaurer plus rapidement les pertes que dans le cas de TCP Reno.

Bien que ces mécanismes aient apporté beaucoup d'amélioration à la version TCP Tahoe d'origine ; ils restent incapables, d'une part, d'assurer un partage équitable entre les différents flux transportés, et d'autre part, de garantir un débit minimal par flux. De plus, quand le réseau est chargé, les ressources sont mal utilisées. Ceci est surtout dû aux grands nombres des paquets retransmis. Certains chercheurs continuent à proposer de nouveaux mécanismes de contrôle de congestion pour remplacer la version TCP Tahoe comme le SCTCP, T-TCP..., d'autres proposent de nouvelles approches de QoS qui avec TCP permettent d'assurer une bonne qualité de service ; une direction que nous avons adoptée dans nos travaux de recherche.

Dans le paragraphe suivant, nous montrons la façon dont la bande passante est partagée entre des flux avec même RTT et RTT différents

2.6 Partage statistique de la bande passante

Une des hypothèses fondamentales sur lesquelles se fondent les modèles de performance à l'échelle des flux repose sur l'existence d'un partage équitable de débit entre les flux TCP. Cependant, il existe un certain nombre de paramètres et de contraintes qui agissent en pratique sur la manière dont TCP partage les ressources. Par conséquent, le partage n'est plus équitable. Le débit qu'un flux peut atteindre durant sa durée de vie dépend : du temps aller-retour, de la fenêtre de congestion maximale et des pertes et la façon avec laquelle les flux réagissent à ces pertes. Face à ces paramètres, le modèle de partage équitable n'est plus valable, et ne peut pas représenter le comportement des flux TCP. Nous étudions, dans la suite, la façon dont la bande passante est partagée entre différents flux TCP avec même RTT et RTT différents. Nous considérons ainsi des flux assez longs pour qu'ils puissent atteindre la phase stationnaire. Nous étudions aussi la sensibilité des différents flux TCP face aux pertes, nous remarquons ainsi que les flux courts ont besoin d'attendre, dans la plupart du temps, un "Time Out" pour récupérer un paquet perdu, au moment où un flux long arrive à récupérer plusieurs pertes au bout d'un RTT. Ceci est dû au fait que les flux courts ont, en général, des fenêtres assez petites, alors, ils détectent souvent une perte grâce au "Time Out". Tandis que les flux longs ont des fenêtres de congestion assez larges, ce qui leur permet de détecter une perte grâce au trois acquittements dupliqués.

2.6.1 Le temps aller-retour

C'est le temps entre le moment d'envoi d'un paquet et le moment de la réception de l'acquittement de ce dernier. Il représente les temps de transmission, les temps de propagation et les temps d'attente dans les files.
Un flux augmente sa fenêtre de congestion au fur et à mesure qu'il reçoit des acquittements. Plus le temps aller-retour est important, plus les paquets mettent du temps pour arriver, ainsi que les acquittements. Cela se traduit par un débit plus faible.
Dans la figure 2.2, nous traçons le débit moyen d_{moy} obtenu par une source en fonction du temps aller-retour. La capacité du lien est fixée à 10 Mbits/s et nous faisons varier le RTT. Nous remarquons, que pour un RTT=5ms, le débit moyen est élevé et il est limité par la capacité du lien (d_{moy}=C=10Mbits/s) ; à partir de RTT=10ms, le débit moyen commence à diminuer au fur et à mesure que RTT augmente. Nous constatons ainsi que le débit d'une connexion est inversement proportionnel à son RTT.

A faible charge, les temps d'attente dans les files sont négligeables ; le temps aller-retour est dominé par les temps de propagation. Ainsi, un émetteur qui est proche de la station de service bénéficiera d'un débit élevé, tandis qu'un autre émetteur, plus loin, va avoir un débit plus faible. Cela résulte du temps additionnel que les paquets mettent pour traverser le réseau.
A forte charge, le temps aller-retour sera plus dominé par les temps d'attente dans

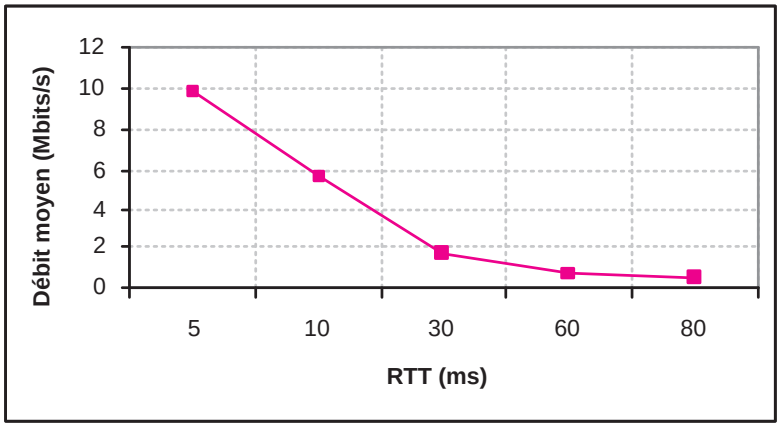

Figure 2.2 – Débit instantané en fonction du RTT

les files. Par conséquent, un émetteur proche de la station de service ne sera plus favorisé par rapport à un autre qui est un peu plus loin. Et la contrainte du temps aller-retour aura moins de sens.

2.6.1.1 Connexions avec le même temps aller-retour

Dans ce paragraphe, nous montrons la façon dont la bande passante est partagée entre deux flux de même RTT (Round Trip Time). Nous prenons des flux assez longs, pour atteindre la phase stationnaire. Les flux courts sont moins sensibles au RTT, car ils ont moins de données à envoyer et font un nombre moins important d'aller-retour.

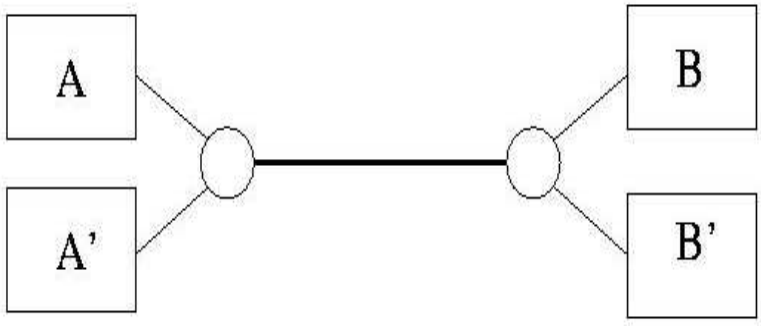

Figure 2.3 – Topologie avec un bus d'interconnexion

Dans la figure 2.4, nous traçons le débit instantané de deux flux TCP qui ont le même temps aller-retour. Les simulations sont faites avec le simulateur NS et représentent le réseau décrit dans la figure 2.3. A l'instant t=0, une connexion TCP est ouverte entre A et B pour transporter un flux FTP continu. A l'instant t= 1s,

une autre connexion TCP est ouverte entre A'et B' pour transporter un flux FTP qui s'arrêtera avant la fin de la simulation (la durée totale de la simulation est de 3 secondes).
La simulation dure 3 secondes. A partir des débits relevés sur les deux courbes nous pouvons remarquer que la première connexion TCP qui démarre à t=0 s augmente son débit au fur et à mesure que la taille de la fenêtre de congestion augmente jusqu'à atteindre, à peu près, le débit du lien (qui est 10 Mbits/s). A t=1 s, quand la seconde connexion TCP est ouverte, le débit de la première connexion chute jusqu'à atteindre la moitié de son débit initial. Le débit de la deuxième connexion se trouve aussi égal au débit de la première. Après t=2s, la deuxième connexion est fermée. Le débit de la première connexion augmente pour reprendre la partie libre de la bande passante et atteindre ainsi le débit du lien. Ainsi, nous pouvons constater que le débit d'une certaine connexion TCP est égal au débit en ligne divisé par le nombre de connexions TCP. Ce raisonnement se base sur le fait que toutes les connexions TCP ont les mêmes contraintes d'accès au bus principal, et, qu'elles sont suffisamment longues pour atteindre le régime stationnaire. Nous pouvons ainsi déduire que les connexions TCP, qui ont le même RTT, partagent équitablement la bande passante.

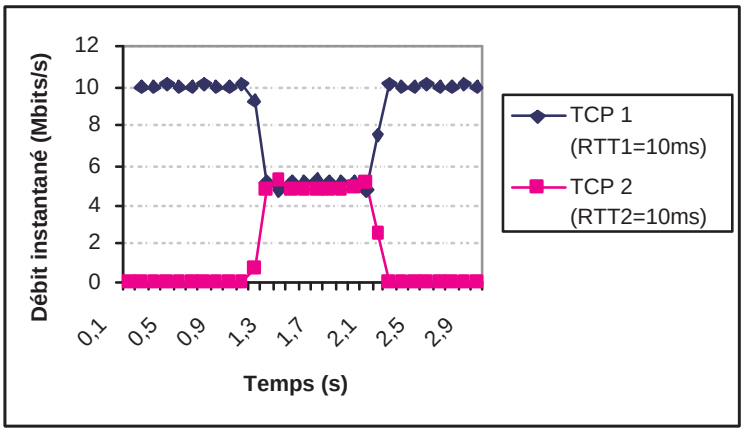

Figure 2.4 – Débit instantané pour deux flux TCP avec même RTT

Dans la section suivante, nous étudions l'équité entre différentes connexions TCP qui ont un RTT différent.

2.6.1.2 Connexions avec des RTT différents

Dans cette partie, nous étudions le comportement du mécanisme TCP face à des connexions avec des aller-retour différents partageant le même chemin. Nous reprenons la même topologie de la figure 2.3, à ceci près que le délai de propagation entre A' et B' est augmenté de 40ms.
La figure 2.5 trace les courbes du débit instantané de deux connexions TCP. TCP

1 (RTT=10ms) est plus rapide que TCP 2 (RTT=50ms). Nous constatons que le débit obtenu par la connexion 1 est plus important que celui de la connexion 2. Cette iniquité vient de l'algorithme d'augmentation de la fenêtre de congestion utilisé par TCP ; en effet à chaque fois que l'émetteur reçoit un acquittement, il a le droit d'augmenter sa fenêtre de congestion d'une unité ou de 1/Cwnd selon qu'on est dans la phase "Slow Start" ou "Congestion Avoidance" respectivement. Ce qui fait que les connexions avec un RTT plus petit augmentent plus rapidement leur fenêtre de congestion.

Le partage d'une même bande passante entre différentes connexions TCP nécessite

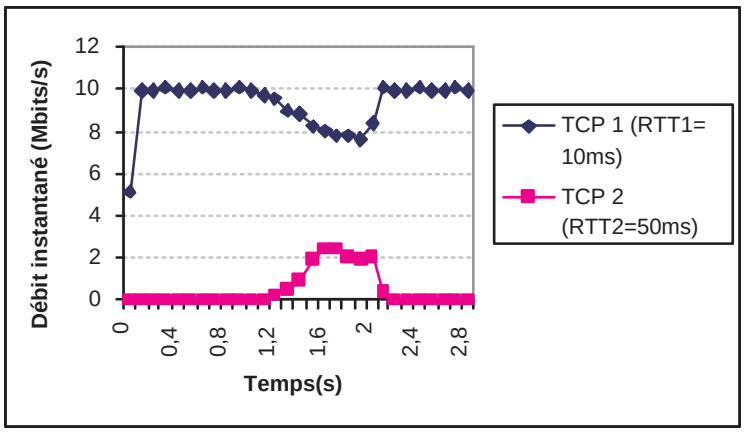

Figure 2.5 – Débit instantané pour deux flux TCP avec RTT différent

une certaine équité, une chose qui n'est pas assurée par le protocole TCP. Le but principal de TCP est de contrôler la congestion sur Internet, mais ce contrôle se révèle être inefficace s'il n'assure pas un partage équitable du medium. Cependant, on remarque que le problème avec TCP est que les connexions ayant un plus faible "Round Trip Time" bénéficient d'un plus grand débit par rapport aux connexions avec un RTT plus important.

Cette première étude des flux TCP multiples dans un même conduit nous montre donc un point très important : le partage de la bande est très défavorable aux flux TCP dont le RTT est important.

Dans le chapitre suivant, nous faisons une étude plus détaillée sur l'influence du RTT sur le débit. Cette étude est faite dans un contexte plus général, dont nous considérons une topologie avec plusieurs sources partageant le même chemin. Nous étudions l'impact de la taille de file d'attente et de la charge sur les performances du système et le partage du lien. Nous montrons aussi l'avantage de classifier les flux TCP longs selon le RTT.

2.6.2 Comportement des flux TCP face aux pertes

Les flux TCP ne réagissent pas de la même façon quand ils détectent une perte. D'ailleurs une perte peut être détectée selon deux méthodes :

– Par l'expiration d'un temporisateur (TO : Time Out) (Figure 2.6(a))
– Par la détection de trois acquittements dupliqués (Figure 2.6(b))

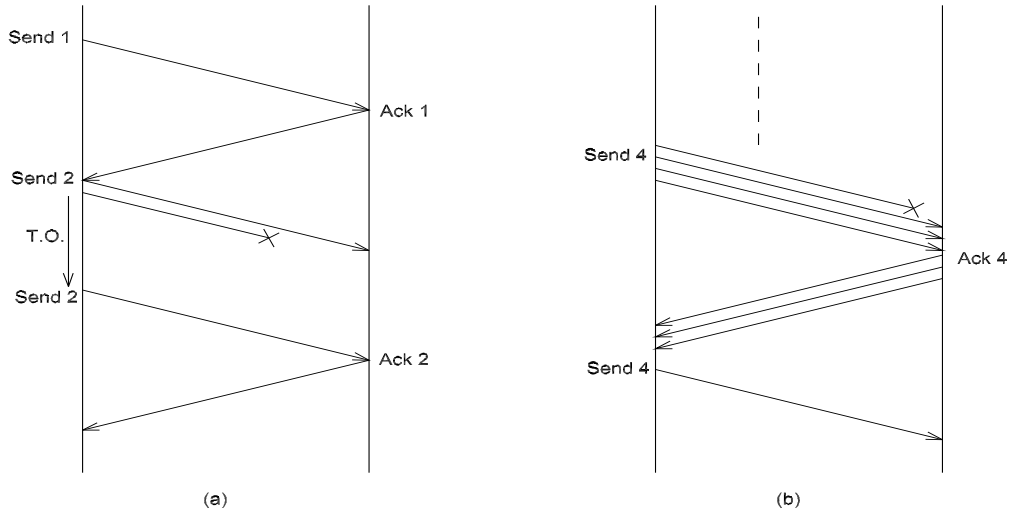

Figure 2.6 – Détection d'une perte par T.O (a) ou par trois acquittements dupliqués (b)

Ces deux méthodes entraînent deux comportements différents. A l'expiration d'un TO, une connexion TCP divise sa fenêtre de congestion par deux, afin de fixer le nouveau seuil d'infléchissement, et redémarre à 1 paquet (elle revient à la phase Slow Start). Tandis qu'à la réception de trois acquittements dupliqués, une connexion TCP divise sa fenêtre de congestion par deux, elle récupère les paquets perdus (c'est le mécanisme de Fast retransmit/Fast recovery) et elle continue à partir de la nouvelle fenêtre (elle reste dans la phase Congestion Avoidance).
Comme les flux courts ont toujours un petit nombre de paquets à envoyer, ils opèrent souvent dans la phase Slow Start. Par conséquent, leurs fenêtres de congestion sont souvent petites, ce qui les rend très sensibles aux pertes puisqu'ils ont souvent recours au TO pour détecter une perte. Cela entraîne des temps plus longs pour la détection et la récupération des paquets perdus et ne laisse pas la fenêtre de congestion atteindre des valeurs plus grandes. Cependant, le débit qu'un flux peut atteindre est lié à la fenêtre de congestion : plus elle augmente, plus le débit augmente.
En revanche, les flux longs ont un grand nombre de paquets à envoyer et arrivent à atteindre des fenêtres de congestion assez larges. Ainsi, une perte est souvent détectée grâce aux trois acquittements dupliqués, ce qui les rend moins sensibles aux pertes puisqu'ils arrivent à récupérer plusieurs pertes au bout d'un RTT.
Plusieurs études ont été faites pour trouver une relation entre le débit et le taux de pertes. Dans son article [PAD98a], Padhye et al. font une analyse détaillée du mécanisme de contrôle de congestion et de sa performance qui aboutit à une formulation relativement simple du débit réalisé, au cours d'une connexion TCP, en fonction du taux de perte de paquets et du temps aller-retour. Les résultats sont validés par

comparaison avec un modèle stochastique plus rigoureux [PAD99] mais d'utilisation moins pratique car ne fournissant pas d'expressions analytiques. Ce modèle conduit aux expressions analytiques suivantes qui donnent le débit d'émission DE d'une connexion TCP en régime stationnaire, en fonction du RTT, de la durée élémentaire de "Time Out" T_O, du taux de perte p et de la taille de la fenêtre de congestion maximale W_{max} imposée par le récepteur. Il faut noter que dans ce modèle seules les pertes détectées par l'expiration du temporisateur sont caractérisées, les auteurs ne tiennent pas compte des trois acquittements dupliqués.

$$DE = \begin{cases} \dfrac{\dfrac{1-p}{p} + W(p) + \dfrac{Q(p,W(p))}{1-p}}{RTT[W(p)+1] + \dfrac{Q(p,W(p))G(p)T_o}{1-p}} & W(p) < W_{max} \\ \dfrac{\dfrac{1-p}{p} + W_{max} + \dfrac{Q(p,W_{max})}{1-p}}{RTT\left[\dfrac{W_{max}}{4} + \dfrac{1-p}{pW_{max}} + 2\right] + \dfrac{Q(p,W_{max})G(p)T_o}{1-p}} & W(p) \geq W_{max} \end{cases} \tag{2.2}$$

où

$$W(p) = \frac{2}{3} + \sqrt{\frac{4(1-p)}{3p} + \frac{4}{9}}$$

$$Q(p,W_{max}) = Min(1, \frac{(1-(1-p)^3)(1+(1-p)^3(1-(1-p)^{W_{max}-3}))}{1} - (1-p)^{W_{max}})$$

$$G(p) = 1 + p + 2p^2 + 4p^3 + 8p^4 + 16p^5 + 32p^6$$

Des expressions similaires s'obtiennent pour le débit de réception DR :

$$DR = \begin{cases} \dfrac{\dfrac{1-p}{p} + \dfrac{W(p)}{2} + Q(p,W(p))}{RTT[W(p)+1] + \dfrac{Q(p,W(p))G(p)T_o}{1-p}} & W(p) < W_{max} \\ \dfrac{\dfrac{1-p}{p} + \dfrac{W_{max}}{2} + Q(p,W_{max})}{RTT\left[\dfrac{W_{max}}{4} + \dfrac{1-p}{pW_{max}} + 2\right] + \dfrac{Q(p,W_{max})G(p)T_o}{1-p}} & W(p) \geq W_{max} \end{cases} \tag{2.3}$$

Dans le cas de faibles taux de perte de paquets, une très bonne approximation du débit d'émission est fournie par l'expression plus simple suivante :

$$DE \approx Min(\frac{W_{max}}{RTT}, \frac{1}{RTT\sqrt{\frac{4p}{3}} + T_o Min(1, 3\sqrt{\frac{3p}{4}})p(1+32p^2)}) \tag{2.4}$$

Des résultats similaires peuvent être trouvés dans [ALT00]. Les auteurs proposent des modèles analytiques avec une loi générale pour les inter-arrivées des pertes. Dans le cas sans perte, les auteurs trouvent une relation entre le débit et le RTT qui est donnée par la formule suivante :

$$DE = \frac{1}{RTT^{\alpha}} \qquad \text{avec} \quad 1 < \alpha < 2 \tag{2.5}$$

Cette relation entre le débit et le temps aller-retour sera étudiée plus amplement dans le chapitre suivant. Nous considérons, ainsi, plusieurs connexions avec des RTT différents. Le but est de garantir un débit minimal par flux, ainsi les flux avec des grands RTT ne seront plus défavorisés.

Dans la section suivante, nous étudions la différence entre les flux courts et longs TCP. Nous trouvons que ces deux types de flux ont des caractéristiques et des contraintes de QoS différentes.

2.7 Partage de la bande passante entre flux courts et flux longs

Les flux TCP représentent la grande majorité du trafic Internet (voir section 1.4). Parmi ces flux, nous pouvons distinguer les flux courts et les flux longs. Les premiers sont générés par des applications comme HTTP, SMTP.... Ils sont appelés courts car ils ont, en général, très peu de paquets à envoyer. Les flux longs sont des flux qui durent assez longtemps et ils ont un très grand nombre de paquets à envoyer, ils sont en général générés par les applications comme FTP, peer to peer,...
Les performances des flux courts sont étroitement liées à la phase Slow-Start du protocole TCP, qui est la phase de démarrage. Par conséquent, leur fenêtre de congestion est souvent petite. Ainsi, même dans le cas où toute la capacité est disponible, le débit d'un flux court n'arrivera jamais à dépasser la valeur théorique imposée par l'algorithme TCP (qui est le double de la fenêtre de congestion chaque RTT). Les flux TCP avec des fenêtres de congestion petites sont très sensibles aux pertes car ils ont souvent recours au TO pour détecter une perte. Par conséquent, le temps de transfert augmente et les flux mettent plus de temps pour finir [AYE02]. Les performances des flux longs dépendent plus de la phase Congestion Avoidance que de la phase Slow Start, ce qui leur permet d'augmenter suffisamment leur fenêtre de congestion. Durant la phase Congestion Avoidance, les flux entrent dans un régime permanent et arrivent à mieux partager la bande passante. Les flux TCP avec des fenêtres de congestion larges sont plus tolérants aux pertes, car ils ont souvent recours aux trois acquittements dupliqués pour détecter une perte.
Dans [MAT00], les auteurs montrent que les flux courts sont très défavorisés en présence des flux longs. Ainsi, les flux longs peuvent complètement arrêter les flux courts. En conséquence, leur temps de transfert sera plus long, ce qui crée des problèmes de performance pour les flux courts qui sont en général très sensibles au délai.

Plusieurs études [AYE02, BEN01, ROB98, MAS99] ont supposé que le partage de la bande passante entre plusieurs flux est du type " Processor Sharing ". Cette hypothèse s'avère bonne, dans certain contexte, pour les flux TCP longs. Sa justification est plus difficile dans le contexte des flux courts. Par ailleurs, il est important de signaler que les objectifs de QoS ne sont pas les mêmes pour les flux longs et les flux courts : les premiers sont sensibles à la bande passante qui leur est allouée tandis que les deuxièmes sont, en général, plus sensibles au temps de traversée du réseau.

Dans les études citées plus haut, un modèle analytique (M/G/1 Processor Sharing) a été défini pour calculer les paramètres de performances qu'un utilisateur peut avoir. Les auteurs ont considéré des flux assez larges, ce qui leur permet d'atteindre la phase stationnaire. Par conséquent, le mode de partage obtenu à partir du mécanisme du contrôle de congestion de TCP peut être approximé par une discipline PS. Nous trouvons ainsi que le débit moyen obtenu par un flux est donné par la formule suivante :

$$d = C(1 - \rho) \tag{2.6}$$

Avec C la capacité du lien et ρ la charge du lien. Cette formule est très théorique et ne peut pas représenter le débit réel qu'un utilisateur peut recevoir durant sa durée de vie. Elle représente plutôt le débit restant. Celui-ci ne peut pas être toujours atteint dû à plusieurs limitations expliquées ci-dessus. Même si cette formule est réelle, nous remarquons qu'elle ne tient pas compte des flux courts qui représentent 90% des flux TCP même s'ils n'introduisent que 15 à 20% de la charge totale.

Pour pallier à ce manque, nous avons proposé un modèle de contrôle d'admission qui tient compte des flux courts et longs TCP et des contraintes de la QoS de chaque type de flux. Cette nouvelle approche ainsi que d'autres améliorations seront détaillées dans les chapitres 4 et 5. Dans le paragraphe suivant, nous expliquons l'intérêt d'utiliser le contrôle d'admission pour les flux courts et longs TCP.

2.8 Congestion à l'échelle des flux

Nous avons vu que le protocole TCP reste insuffisant pour assurer d'une part, un partage équitable entre les différents flux, et d'autre part, un débit minimal par flux. En fait, les performances des flux TCP sont acceptables tant que la charge est faible ; à forte charge nous avons plus des flux qui arrivent que des flux qui partent et les performances ne sont pas contrôlées.

La figure ci-dessus, trace le nombre de flux actifs et le débit moyen obtenu par flux en fonction du temps. Nous considérons un lien de capacité C=20 Mbits/s, qui subisse une surcharge de 30%. Les flux sont de taille moyenne et générés suivant la loi de Poisson.
Lorsque la charge augmente, le nombre de flux en cours augmente et par conséquent, leur débit diminue et peut tendre vers zéro, puisque le taux d'arrivée λ reste supérieur au taux de service des flux.

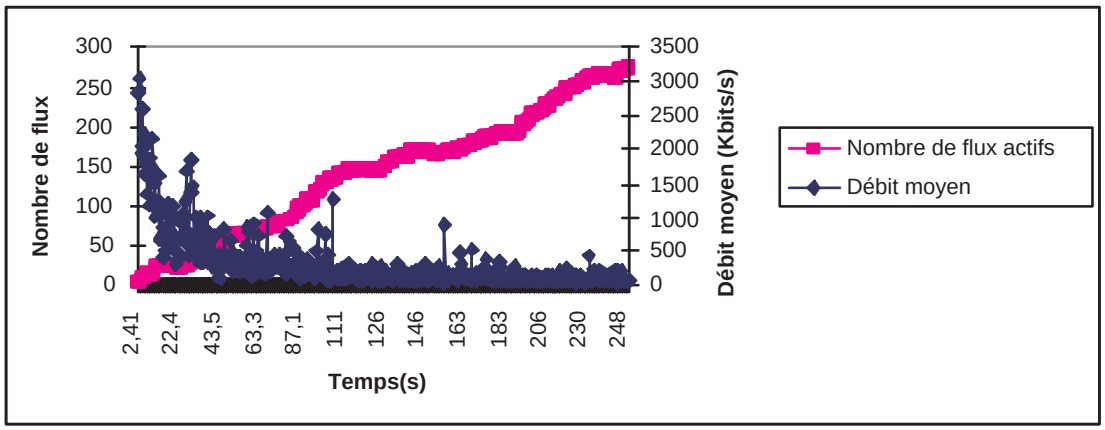

Figure 2.7 – Débit moyen et nombre de flux actifs en fonction du temps

Il est intéressant de noter que la vitesse d'augmentation du nombre de flux dépend de manière significative de la distribution de la taille des flux [JEA94]. La vitesse de croissance est plus petite lorsque la proportion des flux de petite taille est plus grande. Ces derniers parviennent à terminer, bien que le débit continu à régresser. Au moment où le temps de réponse des grands flux augmente rapidement. Il faut noter qu'il n'est pas facile de détecter le degré de congestion sur un lien du réseau en observant simplement le processus d'arrivée des paquets. La fréquence d'arrivée des paquets est contrôlée par TCP, idéalement vers une valeur proche du taux de service, bien que le nombre de flux présents puisse augmenter rapidement. La congestion du trafic élastique se manifeste essentiellement au niveau flux plutôt qu'au niveau paquet.
Dans la section suivante, nous expliquons l'intérêt d'utiliser un contrôle d'admission pour limiter le nombre total des flux actifs. Ce mécanisme ne sera efficace que s'il est appliqué sur les deux types de flux TCP : courts et longs.

2.8.1 Besoin du contrôle d'admission pour les flux TCP

Dans son article [SHE95], Shenker conclut que le contrôle d'admission n'est pas utile pour les flux élastiques. Cette conclusion a été contestée par Massoulié et Roberts dans [MAS99] qui soulignent que l'objectif du contrôle d'admission n'est pas tant d'assurer un débit minimal acceptable aux flux élastiques, que de préserver l'efficacité globale du réseau en temps de surcharge. En l'absence d'un tel mécanisme, le réseau devient théoriquement instable lorsque le volume de trafic offert excède sa capacité. Le nombre de flux en cours croît alors indéfiniment et le débit utile par flux tend vers zéro. Dans un réseau réel, ce phénomène ne se produit pas car en temps de surcharge le débit utile se dégrade au point que certains utilisateurs finissent par interrompre leur session. Cela peut être dû à l'impatience des usagers, un abandon au niveau TCP ou à celui des protocoles de couches supérieures. Les transferts par-

tiels gaspillent la bande passante et ne génèrent aucune utilité. Actuellement, nous manquons de statistiques sur le phénomène d'impatience des utilisateurs. Il est clairement très difficile d'observer ce phénomène dans la pratique, notamment, parce que toutes les interruptions des flux ne sont pas la réaction au temps de réponse excessif. La plupart des phénomènes d'impatience se manifeste par l'interruption d'une session et peut ne pas être discernable comme événement anormal. Dans sa thèse, G. Régnié [REG02] a modélisé le phénomène de l'impatience. Il a proposé une résolution théorique dans le cas où des hypothèses statistiques markoviennes sont posées. La proportion de flux écoulés non-impatients, le volume moyen des flux écoulés non-impatients et le débit utile écoulé ont ainsi été calculés.

Un modèle de contrôle d'admission limite les phénomènes d'impatience : un flux non accepté ne saurait être impatient, un flux accepté a moins de raison d'être impatient. Le débit du lien s'en trouve amélioré.
Le contrôle d'admission devient nécessaire pour maintenir un nombre limité de flux dans le réseau. Il détermine principalement si un nouveau flux doit être admis avec une décision, selon si le réseau peut assurer la QoS du nouveau flux et celle des flux existants. Mais, son application sur les flux TCP sans faire la différence entre flux courts et flux longs, peut pénaliser les flux courts sans nécessairement améliorer les performances. Puisque les flux courts représentent plus que 90% des flux TCP et ils induisent seulement entre 15 et 20% du trafic total.
Dans cette thèse, nous nous sommes intéressés à l'amélioration des performances des flux TCP. Nous avons proposé des nouvelles approches de contrôle d'admission qui tiennent compte des flux TCP courts et longs. Ces approches seront détaillées dans les chapitres suivants.

2.9 Conclusion

TCP est un protocole de bout en bout dont la vocation est de compenser les faiblesses du protocole IP et de fournir aux applications un transport fiable, c'est à dire, avec un taux d'erreur et de perte résiduel nul. TCP effectue le contrôle d'erreur, de perte et de séquencement par détection, signalisation et par la récupération des pertes par retransmission. TCP a donc été donc conçu pour s'adapter à un type de couche réseau (IP), pour des réseaux locaux et des réseaux longue distance à faible débit et pour des classes d'application bien spécifiques (transfert fiable de fichiers, messagerie textuelle et accès distant).
Avec l'évolution des applications et des infrastructures, TCP, malgré le grand nombre d'amélioration que nous avons évoqué, reste incapable d'assurer un débit minimal par flux et un partage équitable entre les différents flux TCP.
Nous suggérons que la clé pour espérer une qualité de service acceptable consiste dans un dimensionnement adéquat couplé aux stratégies de routage de trafic et de contrôle d'admission conçues pour éviter les surcharges. Plutôt que de compter sur l'impatience des utilisateurs pour stabiliser un lien surchargé menant à une très mauvaise qualité de service, il est préférable de mettre en place un contrôle

d'admission au niveau flux ou session, en maintenant un débit suffisant pour les flux admis et en évitant ainsi un gaspillage de bande passante dû aux transferts inachevés. L'étude et la définition d'un tel contrôle d'admission font l'objet des chapitres suivants.

Chapitre 3

Avantages de classifier les flux TCP longs selon le RTT

3.1 Introduction

Comme nous avons expliqué dans le chapitre 2, les différentes versions de TCP ont permis à ce protocole de s'améliorer au niveau de la détection et de la récupération des paquets perdus ce qui a rendu la transmission plus efficace.

Un des objectifs principaux de TCP est de contrôler la congestion dans l'Internet [JAC88]. Ce contrôle n'est pas efficace s'il n'assure pas un partage équitable des ressources du réseau. Un problème important dans TCP est sa discrimination contre les connexions ayant un large RTT [BAR00, FLO91, LAK97]. Ces connexions ne peuvent pas réaliser le même débit que les autres connexions se partageant le même chemin et ayant un RTT plus petit. Ceci est dû à l'algorithme d'augmentation de la fenêtre de congestion adopté par TCP. En effet, TCP emploie la stratégie de "additive-increase multiplicative-decrease" pour contrôler la congestion [JAC88, STE97]. Nous savons qu'un tel genre de stratégies mène à l'équité quand toutes les connexions augmentent leur fenêtre avec le même rythme [CHI89]. Nous parlons ici de l'équité dans le partage de la bande passante d'un goulot d'étranglement indépendamment du volume de ressources consommées par une connexion sur les autres liens du réseau. Ce genre d'équité s'appelle dans la littérature le partage max-min [FLO91]. Cependant, d'autres types d'équité existent. L'objectif est de partager, non seulement, les ressources au niveau du goulot d'étranglement, mais aussi, les ressources sur d'autres parties du réseau. Dans le cas de TCP et en présence des connexions avec RTT différents, un partage équitable ne peut pas être assuré puisque la croissance de la fenêtre de congestion est inversement proportionnelle au RTT (un paquet par RTT dans la phase congestion avoidance). Ceci mène à une augmentation du taux de transmission à un taux inversement proportionnel à un produit de son RTT. Notons que le taux de transmission peut être approximé, à tout moment, par la taille de la fenêtre divisée par le RTT. Les connexions avec un petit RTT augmentent plus rapidement leur fenêtre et utilisent la plus grande part de la bande passante disponible.

Les débits obtenus par les connexions qui sont générées par deux sources différentes, dont l'une est située à une distance plus grande du routeur d'entrée, peuvent être reliés par l'équation suivante [LAK97] :

$$\frac{d_1}{d_2} = \left(\frac{RTT_2}{RTT_1}\right)^{\alpha} \qquad 1 < \alpha < 2 \tag{3.1}$$

Les débits ont été calculés en supposant que les fenêtres des différentes connexions partageant le goulot d'étranglement varient d'une façon synchronisée [BRO00, LAK97]. Toutes les connexions sont censées réduire leurs fenêtres simultanément quand survient une période de congestion. Ce phénomène de synchronisation a été aussi observé dans le cas des connexions avec RTT petits [ZHA98]. Il est principalement provoqué par l'utilisation des files d'attente de type Drop Tail.

Dans ce chapitre, nous proposons de classifier les flux TCP longs selon le RTT. Nous considérons des flux TCP assez larges, ce qui leur permet d'atteindre le régime stationnaire (la phase "congestion avoidance"). Les flux courts sont moins sensibles au RTT puisqu'ils ont toujours très peu de paquets à envoyer. Par conséquent, ils font moins d'aller-retour. Nous considérons des files d'attente assez petites pour que les RTT des différentes connexions soient dominés par les temps de propagation. Le RTT (Round Trip Time) est le temps entre le moment d'envoi d'un paquet et le moment de la réception de l'acquittement de ce dernier. Il est la somme des temps de transmission, des temps de propagation et des temps d'attente dans les files.

Nous considérons ensuite des problèmes de QoS et de dimensionnement. D'abord, nous étudions un modèle avec deux types de sources qui ont des RTT différents. Nous montrons, en se basant sur une étude théorique et sur des simulations, l'avantage de séparer les flux TCP selon le RTT. Celui-ci se caractérise, d'une part, par un débit plus important pour les connexions générées par la source avec un RTT plus large, et d'autre part, par un système plus prédictible puisque les flux d'un agrégat arrivent à partager équitablement la bande passante qui leur est allouée. De plus, quand un contrôle d'admission est appliqué, pour garantir un débit minimal par flux, nous constatons que le modèle avec WFQ nous permet d'accepter plus de flux rapides (RTT petit) que dans le cas où aucune séparation n'a été faite.

Après avoir étudié le cas de deux types de source, nous avons généralisé le modèle pour tenir compte de plusieurs sources avec des RTT différents. Nous savons que la solution qui consiste à séparer en N classes, N étant égale au nombre de sources, devient coûteuse et difficile à implémenter qu'en N croît. Pour cela, nous avons proposé de séparer les connexions en deux classes seulement, ainsi les connexions avec $RTT_i < RTT_k$ (K varie entre 1 et N) seront placées dans la première classe et le reste des connexions sera placé dans la deuxième classe. Pour garantir un débit minimal par flux, nous cherchons à trouver la valeur de k qui minimise C (C est la capacité du lien). Nous montrons, en optimisant l'équation qui relie le débit minimal et les différents RTT avec C, que C est minimal pour K $<$ N (N est le nombre total des

sources). Ainsi, la séparation des flux en deux classes nous permet d'attribuer une capacité plus petite, pour garantir un débit minimal par flux, que dans le cas sans séparation.

Dans la section suivante, nous étudions plus amplement la relation qui relie les débits de deux connexions qui ont des RTT différents.

3.2 Relation entre le débit de deux types de connexion

Dans le cas de deux types de flux avec RTT différent, une formulation simple peut être trouvée [ALT00] pour relier les débits et les temps aller-retour des deux flux (formule 3.1).

Dans son article [BRO00], Brown étudie l'impact de la taille de la file d'attente sur les performances du système et le partage du lien. Il considère des files d'attente de taille supérieure à deux fois le produit délai bande passante. Pour des files d'attente de petite taille, il constate que le partage du lien est non-équitable et l'avantage est du côté des connexions ayant un petit RTT. Cependant, les performances de chaque connexion dépend de l'utilisation globale du lien. Cette tendance change dès que la charge du système atteint un et la file d'attente tend à être tout le temps pleine. Pour des files d'attente larges, Brown constate que le lien tend à être partagé équitablement tandis que la taille de la file augmente, même si la convergence est lente.

Ainsi, $\frac{RTT_2}{RTT_1}$ tend vers 1, par conséquent $\left(\frac{RTT_2}{RTT_1}\right)^{\alpha}$ tend vers 1 et d_1 sera égal à d_2. En effet, quand la file d'attente est large et le réseau est chargé, le temps d'attente dans les files augmente et domine ainsi le temps aller-retour. Par conséquent, les RTT des différentes connexions se rapprochent et les ressources seront partagées équitablement indépendamment de la distance qui sépare les sources du routeur d'entrée.

3.2.1 Variation de α

Nous avons simulé sur NS (Network Simulator) la topologie donnée dans la figure 3.1. La taille de la file d'attente placée à l'entrée du réseau est de 10 paquets, ainsi le RTT sera dominé par les temps de propagation et non pas par le temps d'attente dans les files. Pour cela, nous avons considéré que le RTT des différentes sources reste constant. Nous avons ensuite calculé le débit moyen obtenu par chaque source (d_1 et d_2) pour différentes valeurs de N. N représente le nombre total de source, sachant que nous avons toujours un même nombre de sources de chaque type. Plus N augmente, plus le réseau sera chargé. Les sources génèrent des flux qui durent tout au long de la simulation.

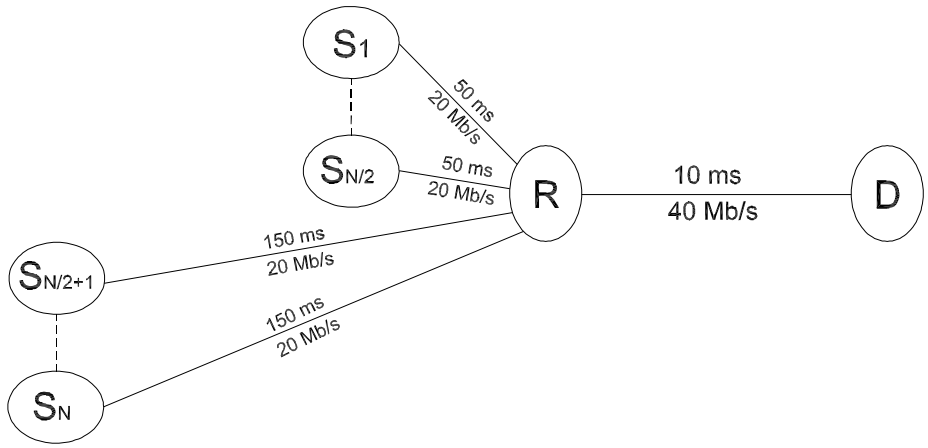

Figure 3.1 – La toplogie du modèle étudié

La figure 3.2 nous montre que α varie très peu avec la charge et prend toujours des valeurs entre 1 et 1.1. Ainsi, nous pouvons la fixer à 1.1. Une valeur qui sera utilisée tout au long de cette étude.

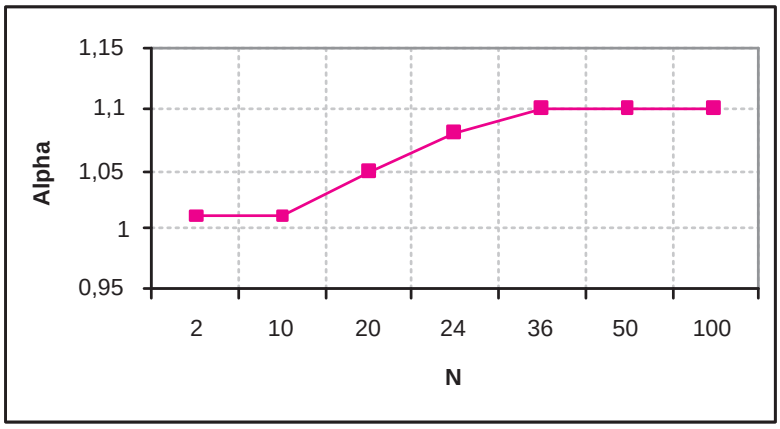

Figure 3.2 – Variation de α en fonction de la charge

3.3 Les modèles étudiés

Dans un premier temps, nous proposons une architecture assez simple qui consiste à séparer les flux selon le RTT, pour protéger ainsi les flux ayant un large RTT. Ensuite, nous proposons de limiter le nombre total de flux accepté dans chaque classe. Ceci nous permet de garantir un débit minimal par flux. Nous étudions une topologie avec deux types de sources et considérons que la deuxième source est située à une distance trois fois plus grande que la première (Figure 3.1). Par conséquent, les

paquets envoyés par cette source mettent plus du temps pour atteindre le routeur d'entrée. Cette architecture sera ensuite comparée à un modèle où aucune séparation n'a été faite. Dans le premier cas, les paquets des agrégats des flux RTT_1 et RTT_2 se partagent la bande passante disponible selon la politique WFQ, ainsi, deux files d'attentes sont utilisées. Lorsqu'une file est vide, l'autre utilise toute la capacité de transmission. Quand aucune file n'est vide, chacune dispose d'un pourcentage fixe de la bande passante disponible. Dans le deuxième cas, un partage en mode FIFO est introduit, ainsi les paquets des deux agrégats sont placés dans une même file d'attente.
Dans un deuxième temps, nous étudions une topologie avec plusieurs sources qui ont des RTT différents. La solution de séparer en N classes, N étant égale au nombre de sources, est coûteuse et difficile à implémenter. Pour cela, nous proposons de séparer les sources en deux classes uniquement, en utilisant la politique WFQ, et cherchons ensuite à dimensionner le lien pour trouver la capacité nécessaire qui nous permet de garantir un débit minimal par flux. Le RTT sera mesuré au moment de l'ouverture de la connexion comme nous l'expliquons dans la section suivante.

3.3.1 Mesure du RTT

Pour séparer les flux selon le RTT, nous avons besoin de mesurer ce dernier. Pour ceci, nous utilisons la même méthode proposée dans TCP vegas.
La version de TCP Vegas (voir L.S. Brakmo et al [BRA94]) change le procédé avec lequel elle fait varier les tailles des fenêtres de congestion par rapport aux autres versions de TCP. Son principe est d'évaluer la taille des files d'attente en entrée des routeurs et d'en observer leur évolution sur la base d'informations calculées à partir des mesures de RTT. Le RTT est calculé sur la base d'un segment envoyé dans le réseau sans qu'il y ait congestion. Il est égal à la durée entre le temps au bout duquel un paquet est émis et le temps au bout duquel son accusé de réception est reçu.

Dans notre modèle, le RTT est calculé une seule fois au moment de l'ouverture de la connexion. Cette dernière se déroule en trois étapes qui peuvent être résumées par la figure 3.3. Ainsi, nous mesurons le temps entre la réception du premier paquet, qui consiste en l'envoie par le client d'un paquet SYN avec un numéro initial de séquence, et la réception du deuxième paquet du client consistant en l'acquittement du paquet envoyé par le serveur. Le temps qui sépare ces deux paquets (SYN et ACK) sera le RTT.

Une fois calculé, le RTT sert d'identificateur de tous les paquets appartenant à la même connexion pour laquelle nous avons calculé le RTT. Ceci qui nous aide ensuite à les classer dans les files d'attentes correspondantes.

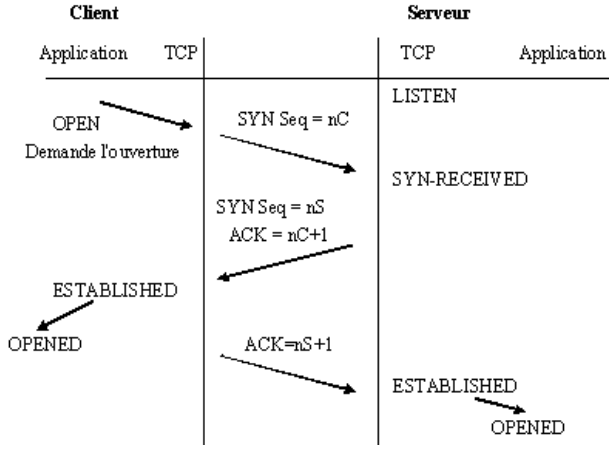

Figure 3.3 – Ouverture d'une connexion TCP

3.4 Cas de deux types de connexions TCP

Nous considérons plusieurs sources TCP de type 1 ainsi que plusieurs sources de type 2. Les sources de types 2 sont situées à une distance égale du routeur d'entrée. La distance quant à elle est trois fois plus grande que celle des sources du type 1 qui sont aussi situées à distance égale du routeur. Nous supposons que les sources ont toujours des paquets à envoyer. Ces derniers partagent le même chemin qui représente le goulot d'étranglement (figure 3.1). La file d'attente à l'entrée du goulot d'étranglement (R) est limitée à 10 paquets. Cette valeur a été choisie pour que les RTT restent dominer par les temps de propagation et non pas par le temps d'attente dans la file. Ainsi, nous pouvons considérer que les RTT sont constants et ne varient pas avec la charge.
Dénotons les RTT par T_1 et T_2, ainsi que le débit moyen obtenu par une connexion générée par les sources du type 1 et 2 par d_1 et d_2 respectivement.

D'après la formule 3.1, d_1, d_2, T_1 et T_2 peuvent être reliés par l'équation suivante :

$$\frac{d_1}{d_2} = \left(\frac{T_2}{T_1}\right)^{\alpha} \tag{3.2}$$

D'après la section 3.2, nous constatons que α peut être fixé à 1.1 et comme T_1 est plus petit que T_2, alors

$$b = \left(\frac{T_2}{T_1}\right)^{\alpha} > 1 \tag{3.3}$$

En partant de l'équation 3.2, nous montrons, en se basant sur deux études : qualitative et quantitative, l'avantage de séparer les deux types de sources selon le RTT.

3.4.1 Etude qualitative

Comme nous l'avons indiqué dans la section 3.1, le mécanisme TCP dans sa construction n'arrive pas à partager équitablement les ressources entre les connexions avec RTT différent. Il n'arrive pas non plus à garantir un débit minimal par flux puisque les ressources seront dominées par les connexions rapides. De ce fait, nous avons besoin, d'une part, de séparer les flux selon le RTT, et d'autre part, de limiter le nombre de flux dans chaque classe pour qu'un débit minimal par flux (connexion) soit garanti.

Dans le cas où aucune capacité minimale n'est garantie à chaque classe, les paquets des agrégats des flux avec T_1 et T_2 se partagent la bande passante disponible selon la politique FIFO. Ainsi, la bande passante sera partagée de la façon suivante :

$$N_1 * d_1 + N_2 * d_2 = C \tag{3.4}$$

Avec $d_1 = d_2 * \left(\frac{T_2}{T_1}\right)^{\alpha}$ (Voir équation 3.2). N_1 et N_2 sont le nombre de source de type 1 et 2 respectivement.

En remplaçant d_1 dans 3.4, nous obtenons :

$$N_1 * d_2 * b + N_2 * d_2 = C \tag{3.5}$$

Pour garantir un débit minimal d_{min}= d_2, nous constatons que nous pouvons accepter jusqu'à N_1 flux du type 1 et jusqu'à N_2 flux du type 2, tel que :

$$N_2 + N_1 * b = \frac{C}{d_{min}} \tag{3.6}$$

Tandis que, quand un partage en mode WFQ est introduit, deux files d'attente sont utilisées et les flux arrivent à mieux se partager la bande passante qui leur est allouée. Nous supposons ainsi que nous donnons C_1 aux connexions du type 1 et C_2 à celles de type 2 avec :

$$C_1 + C_2 = C \tag{3.7}$$

Ainsi pour garantir un débit minimal d_{min} pour les flux du type 1, nous pouvons accepter jusqu'à N_1' flux tel que :

$$\frac{C_1}{d_{min}} = N_1' \tag{3.8}$$

De même pour garantir un débit minimal d_{min} pour les flux du type 2, nous pouvons accepter jusqu'à N_2' flux tel que :

$$\frac{C_2}{d_{min}} = N_2' \tag{3.9}$$

En remplaçant 3.8 et 3.9 dans 3.7, nous obtenons l'équation suivante :

$$N_2' + N_1' = \frac{C}{d_{min}} \tag{3.10}$$

Ainsi, en comparant les deux équations 3.6 et 3.10 et en fixant le nombre maximal des flux du type 2 que nous pouvons accepter à N'_2=N_2 dans les deux cas FIFO et WFQ, nous constatons que nous pouvons accepter plus de flux du type 1 dans le cas WFQ que dans le cas FIFO.
Dans le cas FIFO, N_1 est donné par l'équation suivante (voir 3.6) :

$$N_1 = \frac{\frac{C}{d_{min}} - N_2}{b} \tag{3.11}$$

Tandis que dans le cas WFQ, où une séparation entre les deux agrégats est faite, N'_1 est donné par l'équation suivante (voir 3.10) :

$$N'_1 = \frac{C}{d_{min}} - N'_2 \tag{3.12}$$

Ainsi, nous constatons que dans le cas WFQ nous pouvons accepter plus de flux rapides que dans le cas FIFO (la différence est en fonction de b). La figure 3.4 trace les champs d'admission dans les cas FIFO et WFQ.

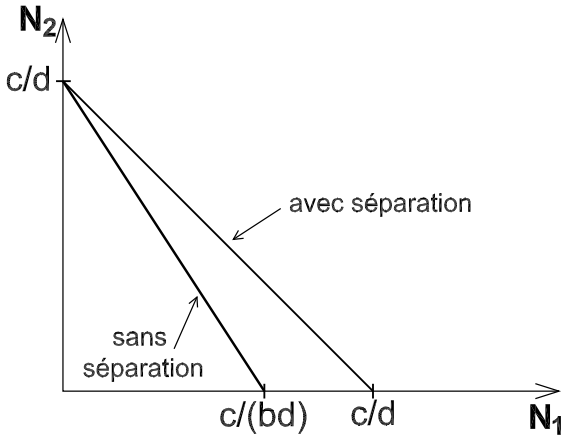

Figure 3.4 – Les champs d'admission dans les cas sans et avec séparation

Pour garantir un débit minimal par flux, le modèle avec WFQ nous permet d'accepter un nombre plus important de connexions rapides. Dans le cas FIFO, plus nous acceptons des connexions rapides, plus la part de la bande passante utilisée par les connexions lentes sera petite. Ces dernières n'arrivent pas à atteindre des fenêtres de congestion assez larges.

Dans la section suivante, nous simulons la topologie donnée dans la figure 3.1 en utilisant NS [NET]. Les deux cas FIFO et WFQ sont comparés pour trouver le gain réel que nous pouvons obtenir en séparant les deux classes.

3.4.2 Etude quantitative

Nous reprenons l'architecture donnée dans la figure 3.1 et nous fixons le débit minimal d_{min} à garantir à 400kb/s. α est fixé à 1.1 (voir figure 3.2).

En reprenant les équations 3.6 et 3.10 obtenues dans l'étude théorique, nous constatons qu'en fixant N_2=N'_2 à 25, nous pouvons accepter jusqu'à 9 flux du type 1 dans le cas FIFO et jusqu'à 25 flux dans le cas WFQ.

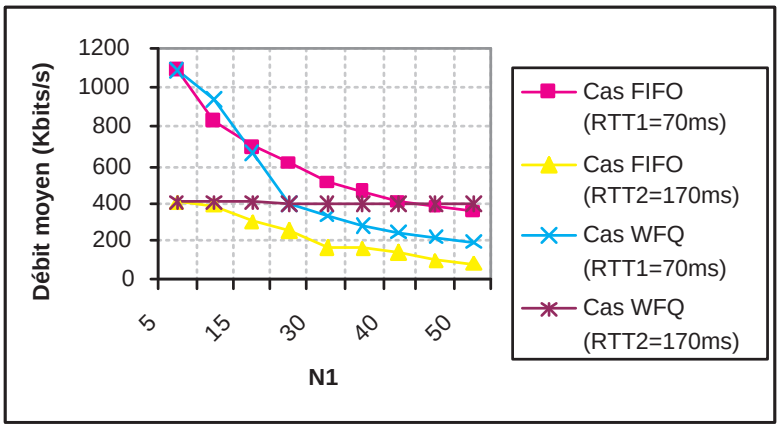

Figure 3.5 – Comparaison du débit moyen obtenu dans les deux cas : avec et sans séparation, N_2=N'_2=25

La figure 3.5 trace les débits moyens obtenus pour les deux types de flux dans les cas FIFO et WFQ. Nous avons fixé N_2 et (N'_2) à 25 et nous faisons varier N_1 (N'_1). D'après les courbes nous constatons que :

- Quand aucune séparation n'est faite, le débit minimal d_{min} fixé à 400kb/s reste garanti pour les flux lents tant que N_1 ne dépasse pas la valeur 9. Pour N_1 égale à 9 le débit moyen obtenu par les connexions du type 2 est égal à 395Kbit/s. Tandis que, quand une séparation est faite, un débit égal à 400kbits/s reste garanti pour les deux types de flux tant que (N'_1) reste égale ou inférieur à 25.

- La solution de séparer les flux TCP selon le RTT nous permet d'accepter un nombre plus grand de flux rapides quand un débit minimal par flux est garanti. De plus, quand une séparation est faite, le réseau devient plus prédictible et plus facile à dimensionner puisque les flux d'un même agrégat arrivent à partager équitablement la bande passante.

3.4.2.1 Pourcentage du gain en fonction de T_2/T_1

Nous avons aussi remarqué que plus l'écart entre T_1 et T_2 augmente, et, plus le nombre de flux rapides qui peut être accepté dans le cas WFQ sera intéressant par

rapport au cas FIFO.

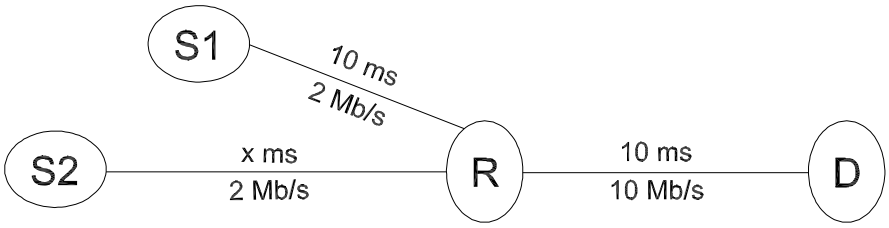

Figure 3.6 – La toplogie avec deux sources

La figure 3.7 trace le gain obtenu, quant au nombre de flux rapides que nous pouvons accepter, quand une séparation est faite par rapport au cas sans séparation en fonction de T_2/T_1. Ainsi, nous avons fixé T_1 à 10ms et avons fait varier T_2 entre 10 et 100 ms (voir figure 3.6). α a été fixé à 1.1.

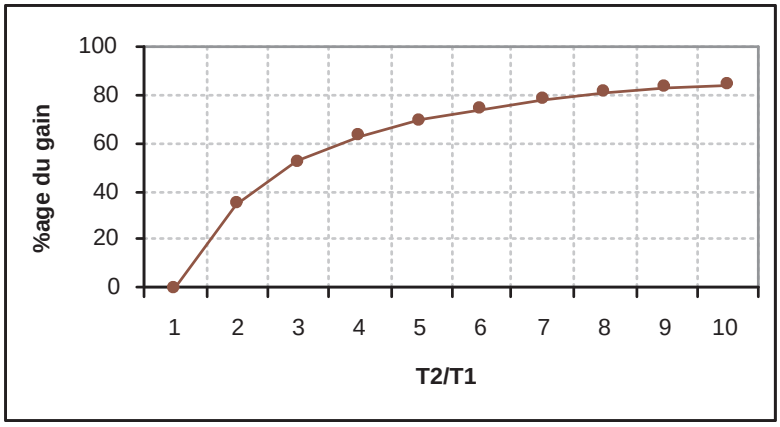

Figure 3.7 – Le pourcentage du gain obtenu en fonction de T_2/T_1, T_1=10ms

3.5 Dimensionnement d'un lien

Dans cette partie, nous considérons N sources avec des RTT différents $RTT_1<\ldots<RTT_N$ (figure 3.8). Nous voulons garantir un débit minimal d_{min} à tous et ne voulons faire qu'une séparation en deux classes en utilisant un WFQ.
Pour cela, nous cherchons à trouver la valeur de 1<k<N qui sépare les deux classes de sorte que si $RTT_i \leq RTT_k$ les paquets arrivant de la source i seront placés dans la classe 1, tandis que les sources avec $RTT_i > RTT_K$ seront placées dans la deuxième classe. Nous cherchons à trouver la capacité du lien C minimum (C est en fonction de k) nécessaire pour pouvoir offrir au moins d_{min} à chaque connexion.

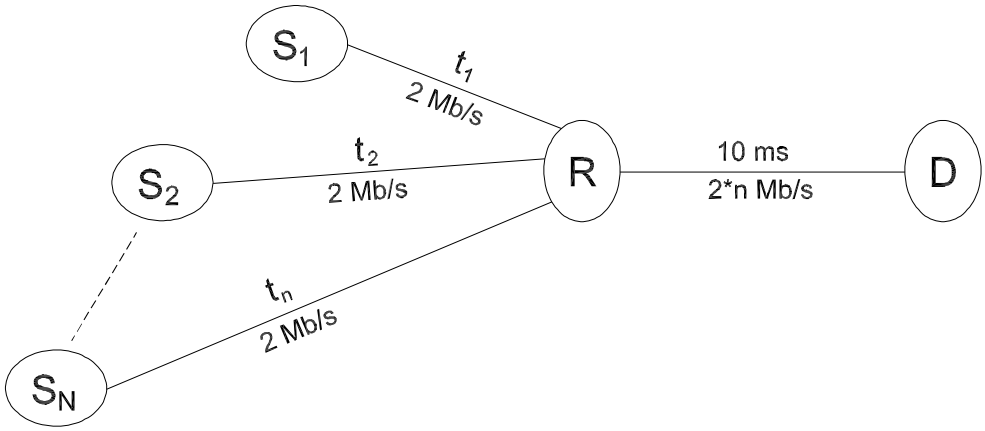

Figure 3.8 – Modèle avec plusieurs sources

Dans chaque classe, les débits moyens réalisés par chaque source sont reliés par les équations suivantes :

$$\sum_{i=1}^{k}(d_i) = C_1 \tag{3.13}$$

$$\sum_{j=k+1}^{N}(d_j) = C_2 \tag{3.14}$$

Avec

$$C_1 + C_2 = C \tag{3.15}$$

Dans chaque classe, nous calculons le débit obtenu par chaque source en fonction du débit le plus faible. Ce dernier correspond au débit de la source qui est située à la distance la plus éloignée du routeur d'entrée. Dans la première classe, c'est la source k qui est la plus éloignée. Par conséquent, le débit d_k que cette source peut atteindre sera forcément plus petit que ceux obtenus par les autres sources partageant la même classe (voir section 3.4). Le débit d_i obtenu par la source i ($1 \leq i < k$) peut être donné par la relation suivante :

$$d_i = d_k * \left(\frac{RTT_k}{RTT_i}\right)^{\alpha} \tag{3.16}$$

En remplaçant 3.16 dans 3.13 nous obtenons l'équation suivante :

$$\sum_{i=1}^{k} d_k * \left(\frac{RTT_k}{RTT_i}\right)^{\alpha} = C_1 \tag{3.17}$$

De la même façon, nous calculons les débits des différentes sources dans la deuxième classe en fonction de d_N (qui est le débit de la source N, cette dernière étant la plus

éloignée du routeur d'entrée). Ainsi, $d_j((k+1) \leq j < N)$ sera donné par l'équation suivante :

$$d_j = d_N * \left(\frac{RTT_N}{RTT_j}\right)^{\alpha} \tag{3.18}$$

En remplaçant cette dernière dans 3.14, nous obtenons l'équation suivante :

$$\sum_{i=k+1}^{N} d_N * \left(\frac{RTT_N}{RTT_j}\right)^{\alpha} = C_2 \tag{3.19}$$

En remplaçant 3.17 et 3.19 dans 3.15 nous obtenons l'équation suivante :

$$C(k) = C_1 + C_2 = (d_k * \sum_{i=1}^{k} \left(\frac{RTT_k}{RTT_i}\right)^{\alpha}) + (d_N * \sum_{j=k+1}^{N} \left(\frac{RTT_N}{RTT_j}\right)^{\alpha}) \tag{3.20}$$

En fixant les débits d_k et d_N à d_{min} qui est le débit minimal à garantir, nous pouvons écrire C(k) sous la forme suivante :

$$C(k) = C_1 + C_2 = d_{min} * (\sum_{i=1}^{k} \left(\frac{RTT_k}{RTT_i}\right)^{\alpha} + \sum_{j=k+1}^{N} \left(\frac{RTT_N}{RTT_j}\right)^{\alpha}) \tag{3.21}$$

L'optimisation de cette équation consiste à trouver la valeur de k qui minimise C(k). Deux cas sont étudiés :

- L'écart entre RTT_i et RTT_{i+1} est toujours constant et est égal à RTT_1. Ainsi, dans l'équation 3.21, RTT_k/RTT_i sera remplacé par k/i et RTT_N/RTT_j par N/j (puisque $RTT_i = i * RTT_1$).
- L'écart n'est pas constant et nous définissons ainsi quatre classes. Ces dernières sont choisies selon la distance géographique.

En fixant α à 1.1 et d_{min} à 400kbits/s, nous cherchons à calculer k dans les deux cas. Ceci, en faisant varier k entre 0 et N pour trouver ensuite la valeur de k qui correspond à C(k) minimal.

3.5.1 Cas avec RTT proportionnels

Dans cette partie, nous étudions le cas où les RTT sont choisis de sorte qu'ils soient proportionnels. Ensuite, nous calculons k pour différentes valeurs de N (voir figure 3.9).

La figure 3.9 représente C(k) pour différentes valeurs de N. Ainsi, nous pouvons constater que la valeur de k qui minimise la capacité est toujours comprise dans l'intervalle]0, N[$\forall$ la valeur de N. Ceci implique que, lorsqu'une séparation est faite, nous avons besoin de moins de bande passante pour assurer un débit minimal par flux. Nous constatons que pour k=0 ou N (aucune séparation est faite), C(k) est maximale.

Nous remarquons aussi que les courbes ont une forme convexe et que la valeur de

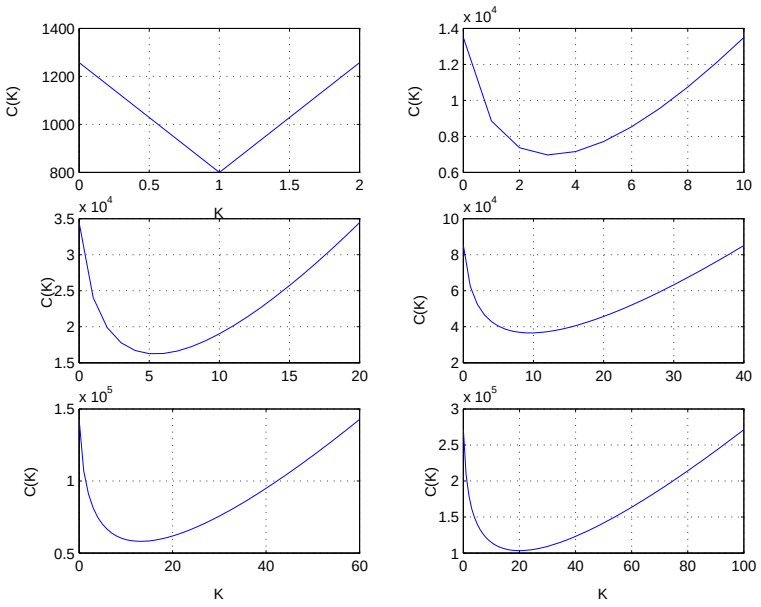

Figure 3.9 – La capacité (Kbits/s) en fonction de K, N=2, 10, 20, 40, 60 et 100

C(k) représente une certaine robustesse au voisinage du minimum. Ainsi, vu la difficulté de trouver une relation analytique entre k et N, cette robustesse nous permet de fixer, pour un intervalle donné de N, une valeur approximative de k. Par exemple, pour $40 \leq N \leq 100$, nous pouvons fixer k =15, qui donne des valeurs de C(k) proches du minimum, même si ce dernier est obtenu pour k égale à 10, 13 et 20 pour N= 40, 60 et 100 respectivement.

3.5.2 Cas avec RTT non proportionnels : quatre classes sont choisies

Plusieurs classes de RTT peuvent être distinguées. Dans notre travail, nous nous sommes limités à quatre. Chacune représente une distance géographique, allant du réseau local (LAN) jusqu'au satellitaire. Entre ces deux classes, nous distiguons celles qui représentent les réseaux MAN et WAN.

Le tableau 3.1 donne les temps aller-retour obtenus en faisant des pings sur des différentes adresses IP. Ces dernières représentent des réseaux situés à différents endroits. Les pings ont été effectués le soir, entre 21h et 22h, pour éviter que le réseau soit congestionné. Nous tenons à signaler, qu'en cas de congestion, le RTT augmente (dû aux temps d'attente dans les files) et cette classification ne sera plus valable.
La figure 3.10 trace la capacité en fonction de k en présence de quatre types de connexion (LAN, MAN, WAN et satellitaire). Elles sont représentées respectivement par un RTT égal à 5, 20, 60 et 400ms. Nous constatons que pour k=2 la capacité est

Classe	Adresse IP	Pays	RTT
Local	137.194.164.220 (ENST)	France	<10ms
Local	137.194.192.136 (ENST)	France	<10ms
MAN	134.157.81.129 (Jussieu)	France	13ms
MAN	132.227.73.20 (Lip6)	France	21ms
WAN	147.83.20.2	Espagne	57ms
WAN	193.136.128.9	Portugal	76ms

TAB. 3.1 – Mesure du RTT des différentes classes

minimale. Cette séparation nous permet d'attribuer une bande passante plus petite que dans le cas sans séparation.

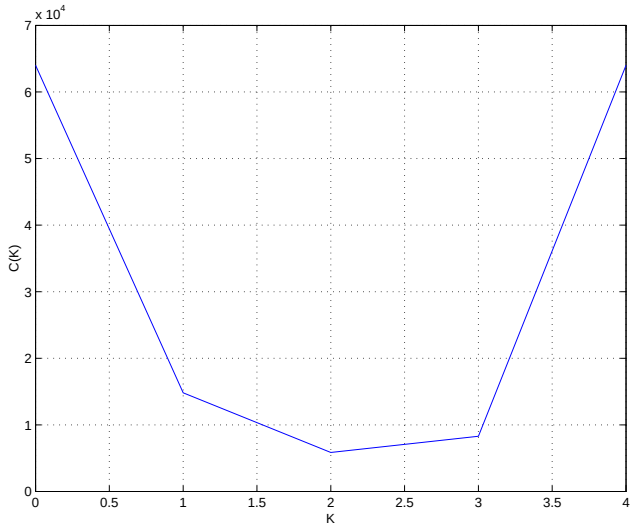

Figure 3.10 – La capacité (Kbits/s) en fonction de K, N=4

Cette étude nous montre l'avantage de séparer les flux en deux classes dans le cas où plusieurs sources, ayant des RTT différents, se partagent le même chemin. La séparation des flux en deux classes nous permet de diminuer l'iniquité entre eux, de mieux protéger les flux lents contre l'agressivité des flux rapides et de minimiser la bande passante nécessaire pour garantir un débit minimal par flux.

3.6 Discussion

Cette étude a été faite dans les conditions suivantes :

– Les flux générés ont toujours un grand nombre de paquets à envoyer. Ainsi, la grande partie du transfert est effectuée dans la phase "Congestion Avoidance" du mécanisme TCP.

- La charge introduite est inférieure ou égale à la capacité du lien. Par conséquent, nous avons toujours un faible taux de perte.
- Les files d'attente considérées sont de petites tailles. En effet, quand les files d'attente sont larges et le réseau est chargé, le temps d'attente dans les files augmente et domine ainsi le temps aller-retour. Par conséquent, les RTT des différentes connexions se rapprochent et les ressources seront partagées équitablement indépendamment de la distance qui sépare les sources du routeur d'entrée.

Ces conditions ne sont pas toujours assurées et surtout dans un réseau WAN comme le réseau Internet. Dans ce dernier, un paquet traverse plusieurs routeurs pour être acheminé jusqu'à la destination finale. Ceci nous a poussé à définir une nouvelle architecture qui tient compte des flux courts et longs TCP. Le but de cette architecture est d'assurer une bonne QoS aux trafics élastiques.

3.7 Conclusion

Dans ce chapitre, nous avons montré l'intérêt de classifier les flux TCP longs selon le RTT. Cette classification nous permet, d'une part, de protéger les connexions lentes qui mettent, en général, plus du temps à atteindre le routeur d'entrée, et d'autre part, d'avoir un réseau plus prédictible et plus facile à dimensionner.
Quand un contrôle d'admission est appliqué pour garantir un débit minimal par flux, nous montrons que nous pouvons accepter plus de flux rapides dans le cas avec séparation que dans le cas sans séparation. Dans cette dernière, plus nous acceptons des flux rapides et plus les performances des flux lents sont détériorées. En effet, aucune part de bande passante ne leur a été allouée.

Dans le cas de plusieurs sources, puisque la solution qui consiste à séparer les flux selon le RTT est difficile et très coûteuse à implémenter, nous avons proposé de séparer les flux en deux classes uniquement. Le RTT_k qui sépare les deux classes doit être choisi de sorte à minimiser la bande passante du lien, qui est nécessaire pour assurer un débit minimal par flux. En étudiant plusieurs valeurs de N et en faisant varier K entre 1 et N, nous constatons que la valeur de k qui minimise la capacité est toujours comprise dans l'intervalle $]0, N[$ $\forall$ la valeur de N. Ceci montre, qu'en présence de plusieurs sources, nous pouvons attribuer une bande passante plus petite, si nous voulons garantir un débit minimal par flux, en les séparant en deux classes.

Cette étude a été faite avec des files d'attente assez petites (10 paquets). Quand la charge du système atteint un et en présence des files d'attente assez larges, le RTT sera dominé par les temps d'attente dans les files. Ainsi, le temps de propagation aura moins d'influence. Par conséquent, les flux parviennent à partager équitablement la bande passante.
Il est important de signaler que, outre l'amélioration des performances dans les cas étudiés, l'approche proposée fourni un outil de dimensionnement du réseau permet-

tant d'atteindre, pour une structure de trafic donnée, les mesures de performances attendues, ce qui est bien plus difficile à réaliser dans un cas sans différentiation.

Dans le chapitre suivant, nous étudions un autre type de classification. Cette dernière est basée sur la taille du flux nous menant à les séparer en courts et longs. Après avoir classifié les flux, nous proposons une approche de contrôle d'admission afin d'améliorer les performances de ces deux types de flux.

Chapitre 4

Contrôle d'admission basé sur les flux courts et longs TCP

4.1 Introduction

Tout réseau se caractérise par le fait qu'il est constitué d'un ensemble de ressources déployées pour supporter un ensemble de services. Les modèles de service réseaux sont caractérisés par les approches utilisées pour le partage des ressources entre les différentes classes de services offerts voire entre les différents flux transportés. L'Internet de première génération se base sur une approche extrême qui consiste en l'acceptation de tous les flux qui se présentent au réseau et en l'effort de partager les ressources de manière équitable entre les différents flux transportés. Même dans le cas théorique où un partage équitable est réussi, le réseau ne peut garantir aucun niveau de Qualité de Service (QoS) ; en effet, les ressources disponibles pour chaque flux individuel décroissent avec le nombre de flux se partageant les ressources et le nombre de flux acceptés n'est pas contrôlé.
Le partage des ressources dans l'Internet découle des mécanismes de contrôle de congestion du protocole TCP. Ces mécanismes réagissent à la perte et de ce fait, plus le nombre de flux est important, plus le taux de perte est élevé et plus de bande passante est gaspillée à cause des retransmissions. De plus, un nombre de flux important implique une bande passante limitée pour chaque flux et en conséquence des temps de transfert longs, ce qui se traduit, dans certains cas comme expliqué dans la section 2.8 , par l'impatience des usagers qui arrêtent des transferts en cours. Ceci représente une deuxième cause de gaspillage des ressources disponibles (voir, par exemple, [YAN01]).
L'introduction d'un contrôle d'admission est un moyen de garantir une QoS minimale aux flux et de limiter le gaspillage de ressources dû aux phénomènes cités plus haut. Son utilisation pour les flux élastiques a été l'objet d'études récentes (voir par exemple, [KUM00, MOR00 et BEN01]).
Les travaux cités supposent que le partage de la bande passante entre plusieurs flux est du type "Processor Sharing". Cette hypothèse s'avère bonne dans le contexte des flux TCP longs ayant le même RTT. Par contre, sa justification est plus difficile dans le contexte des flux courts. Par ailleurs, il est important de signaler que les objectifs

de QoS ne sont pas les mêmes pour les flux longs et les flux courts : les premiers sont sensibles à la bande passante qui leur est allouée tandis que les deuxièmes sont en général plus sensibles au temps de traversée du réseau. Finalement, rappelons que le multiplexage de flux longs et courts dans les mêmes liens engendre des trafics avec des dépendances longues, pour lesquels il est difficile de dimensionner le réseau.
Dans ce chapitre, nous proposons une approche de contrôle d'admission pour les flux TCP (nous ne nous intéressons pas ici aux flux UDP qui restent encore minoritaires en nombre et en volume de ressources consommées) qui prend en compte la caractérisation en flux longs et flux courts ainsi que les contraintes de QoS propres à chaque type de flux. Nous montrons par une étude analytique et des simulations l'intérêt d'une telle approche en termes de performances. Dans un premier temps, nous nous intéressons au cas où les deux classes de flux sont multiplexées à l'aveugle. Dans un deuxième temps, nous étudions l'intérêt de garantir une bande passante minimale pour les agrégats des flux longs et courts. Afin d'éviter des gaspillages de ressources, dans ce dernier cas le partage de la bande passante entre agrégats se fait par un mécanisme du type WFQ de sorte que si un agrégat n'utilise pas la bande passante minimale qui lui est attribuée, la bande passante résiduelle puisse être utilisée par l'autre agrégat. Cette approche est utilisée aux routeurs de bordures, mais les résultats peuvent être étendus à une approche de bout en bout dans une architecture DiffServ ou dans un contexte MPLS. Dans ce dernier cas, nous proposons d'acheminer les flux courts et les flux longs sur des "Label Switched Paths" différents.

4.2 Caractérisation des flux

Les mesures reportées dans http : //ipmon.sprintlabs.com montrent que le trafic TCP représente plus de 90% du volume du trafic Internet "en octets" et plus de 80% des flux. Dans notre travail, nous nous sommes uniquement intéressés aux flux élastiques. Nous définissons un flux comme une connexion TCP suivie d'une période d'inactivité.
Les résultats de plusieurs campagnes de mesures montrent que la grande majorité des flux élastiques sont des flux de petit volume (appelés courts ou "souris") tandis qu'une faible minorité des flux sont de volume très important (appelés longs ou "éléphants"). La plus grande partie du volume du trafic est engendrée par cette minorité des flux.

4.2.1 Identification d'une connexion TCP

Les connexions TCP sont bidirectionnelles et nécessitent l'ouverture de la connexion dans les deux sens. Pour cela, le terminal local envoie un paquet dit SYN (pour synchronisation), puis le terminal distant lui répond par un message ACK (pour acquittement) et un message SYN afin d'ouvrir la connexion dans l'autre sens (les deux messages sont en général transportés dans un seul et même paquet). Par la suite, le terminal local acquitte le SYN du terminal distant, la connexion est alors établie.

L'arrivée d'un nouveau flux TCP peut donc être détectée par les messages SYN et SYN/ACK correspondants (voir, par exemple, [KUM00, MOR00]). Ainsi, quand un contrôle d'admission est appliqué, il suffit de rejeter ces premiers paquets (SYN et SYN/ACK) pour refuser une connexion. Cette approche n'est pas toujours suffisante puisque nous optons pour une stratégie de contrôle d'admission qui reconnaît les flux, il est indispensable que tous les paquets appartenant à un même flux subissent le même traitement.
Une autre approche intéressante est donc utilisée pour détecter l'arrivée d'un nouveau flux. Elle consiste à utiliser une description minimale de chaque flux. Un flux TCP sera ainsi identifié par les valeurs des champs suivants : adresse IP source, adresse IP destination, numéro de port source et numéro de port destination, présents dans l'en-tête TCP/IP (voir par exemple [BEN01]). Au niveau du routeur en charge du contrôle d'admission, une liste des flux en cours et de l'instant de transmission du dernier paquet de chaque flux est maintenue. Cette dernière information est utilisée pour détecter la fin d'un flux à expiration d'un timer associé à chaque flux qui représente la fin du temps d'inactivité.
La maintenance d'un état par flux au niveau de chaque noeud du réseau est actuellement difficilement envisageable. Cependant, dans le cadre d'une mise en oeuvre de MPLS (Multiprotocol Label Switching) [RFC3031], les flux traversant le domaine de contrôle d'admission pourraient être identifiés uniquement par les routeurs de bordure. Ainsi, à chaque flux serait associé un état qui est maintenu au niveau de l'interface d'entrée de routeur via lequel ce flux a pénétré le réseau.

4.2.2 Pourquoi un contrôle d'admission niveau flux ?

Comme expliqué dans la section 1.2, la notion de flux est particulièrement difficile à définir dans le contexte de l'Internet du fait qu'il s'agit d'un réseau en mode non connecté. Les définitions utilisées sont donc liées soit à l'espacement entre paquets ayant de caractéristiques communes (adresses source et destination, numéros de port UDP/TCP, etc.) soit à des notions applicatives.
Il s'avère que le contrôle d'admission est plus facile à appliquer au niveau flux qu'au niveau paquet. Ceci est vrai pour trois raisons.
D'abord parce que le trafic est plus facile à modéliser au niveau flux. En effet, il est possible de caractériser la taille d'un flux en bits, cette notion nous permet de définir la demande en trafic comme le produit du taux d'arrivée des flux par leur taille moyenne. La demande en trafic est une donnée importante qui permet de prédire les performances du réseau.
Deuxièmement, le processus d'arrivée des flux peut être approximé par une arrivée Poissonienne [PAX95], en dépit des corrélations qui peuvent exister entre les flux d'une même session Web ou FTP, il semble raisonnable de modéliser le processus d'arrivée des flux au niveau d'un lien dorsal, fédérant un grand nombre de sessions mutuellement indépendantes. Dans sa thèse Régnié [REG02] explique que désormais l'hypothèse de Poisson n'est pas valide pour modéliser l'arrivée des flux. Il explique ainsi que cette hypothèse peut être valide pour modéliser l'arrivée des sessions. Dans le cadre de notre travail, l'hypothèse de Poisson est d'autant plus justifiée, qu'elle est

utilisé pour des modèles descriptifs dont l'objectif est de comprendre l'impact des mécanismes de contrôle d'admission sur les performances des flux, par opposition aux modèles prédictifs dont l'objectif est d'obtenir des résultats quantitatifs précis. Cette hypothèse est inacceptable pour modéliser l'arrivée des paquets. Le processus décrivant l'arrivée des paquets possède la caractéristique d'auto-similarité [PAX94, PAX95, FEL00]. Cette propriété rend très difficile l'évaluation des performances du réseau au niveau paquet.
Troisièmement, en appliquant le contrôle d'admission au niveau flux, tous les paquets d'un même flux vont subir le même sort. Ainsi si le premier paquet d'un flux est accepté/rejeté tous les paquets appartenant à ce flux seront aussi acceptés/rejetés. Cette méthode est plus efficace que celle d'un contrôle d'admission appliqué au niveau paquet. Puisque le mécanisme TCP retransmet les paquets perdus.

4.2.3 Classification des flux élastiques et contraintes de QoS associées

Nous considérons que la classification flux stream-flux élastique peut s'avérer insuffisante pour un bon dimensionnement du réseau. En effet, parmi les flux élastiques ont compte ceux générés par des applications très diverses requérant des niveaux de QoS différents (voir section 1.5.1). Pour mieux comprendre cela, nous faisons la différence entre la QoS perçue par l'usager et les performances du réseau. Prenant l'exemple du transfert de fichiers. La qualité de service sera perçue à travers le temps de transfert. Ce temps est inversement proportionnel au débit de transfert. Le temps de transfert étant en général plus grand de plusieurs ordres de grandeur que le délai de traversée du réseau (pour un paquet), ce dernier paramètre de performance ne peut pas être considéré comme un facteur de la QoS perçue par l'usager. Par contre, dans le contexte du rapatriement d'une page Web de petite taille transportée sur un nombre réduit de paquets, le temps de récupération de la page sera fortement lié au temps de traversée du réseau (en particulier, parce que tout le transfert se fera avec une taille de fenêtre d'anticipation de taille réduite).
Par ailleurs, dans ce dernier cas, le partage des ressources n'est pas toujours bien modélisé par un ordonnancement du type " Processor Sharing ". En effet, tout le transfert se fera dans le début d'une période " Slow Start " du mécanisme de contrôle de congestion de TCP (voir [RFC2581]).
Nous constatons donc que selon la durée du flux les objectifs de QoS sont différents et les mécanismes de partage de ressources n'engendrent pas le même type de partage.
Une classification plus fine des flux est donc nécessaire afin d'obtenir une meilleure QoS pour chaque type de flux et un meilleur dimensionnement du réseau.
Nous dirons par la suite qu'un flux élastique est court, s'il est formé de quelques dizaines de paquets. Nous verrons plus loin l'impact de la quantification de cette frontière dans les performances des mécanismes de contrôle que nous proposons.

Avant de commencer la transmission d'un flux, le réseau doit savoir à l'avance la taille de ce dernier. Une telle information peut ne pas être aisément disponible.

Dans cette étude, nous nous basons sur le type d'application pour classifier les flux. Ainsi, un flux FTP est considéré comme long tandis qu'un flux HTTP ou SMTP est considéré comme court.

4.3 Mécanismes de gestion de trafic proposés

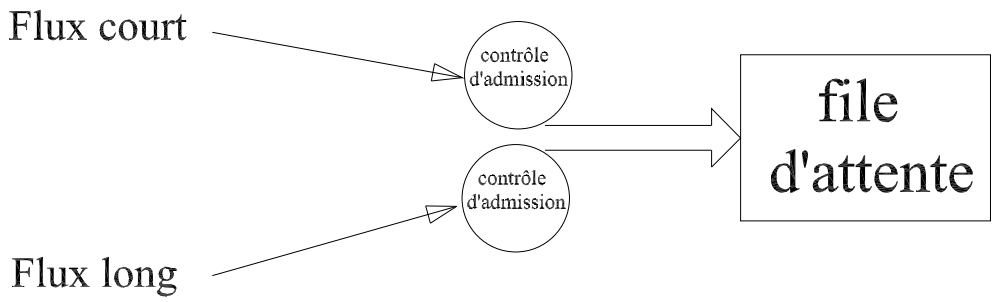

Figure 4.1 – Approche du C.A. proposée

Dans cette partie, nous décrivons le dispositif de contrôle d'admission que nous proposons. Nous nous sommes intéressés à deux cas de figure. Dans le premier, les paquets des agrégats des flux courts et longs se partagent la bande passante disponible selon une politique FIFO. Dans le deuxième, un partage en mode WFQ est introduit. Dans le premier cas, les paquets des deux agrégats sont placés dans une même file d'attente. Dans le deuxième cas, deux files d'attente sont utilisées. Lorsqu'une file est vide, l'autre utilise toute la capacité de transmission. Quand aucune file n'est vide, chacune dispose d'un pourcentage fixe de la bande passante disponible. La part de la bande passante attribuée aux flux courts est égale à la charge moyenne introduite par ces derniers. Le reste de la bande passante est attribué aux flux longs. En effet, les flux courts représentent plus de 90% des flux TCP, mais juste entre 15 à 20% de la charge moyenne introduite (voir section 1.4). Par conséquent, en les favorisant, nous arrivons à diminuer leur taux de blocage sans pour autant charger le réseau.
Nous insistons sur le fait que ce partage est mis en place entre les agrégats, autrement dit, l'ordonnanceur ne distingue pas les flux individuels mais uniquement les agrégats de flux courts et flux longs. On peut imaginer que les paquets d'un même agrégat constituent un "Behavior Agregate" dans l'architecture DiffServ afin qu'ils puissent être distingués grâce au champ TOS.
La séparation des flux obtenue grâce à la politique WFQ permet de dimensionner plus facilement les seuils d'acceptabilité pour chaque classe. En particulier, le fait de diminuer à l'intérieur d'un agrégat la variance de la taille des flux rend le trafic plus prédictible ce qui facilite également le dimensionnement. Ceci est particulièrement vrai pour les flux courts ce qui facilitera un dimensionnement permettant d'atteindre les objectifs attendus de QoS en terme de délai de traversée du réseau.

4.3.1 Mécanismes de contrôle d'admission proposés

La procédure de contrôle d'admission proposée est implicite, il consiste à limiter le nombre de flux courts et le nombre de flux longs admis simultanément. Nous définissons donc deux seuils, que nous notons N_1 et N_2 et qui représentent, respectivement, le nombre maximum de flux courts et de flux longs qui peut être admis. Le taux d'arrivée très élevé des flux au niveau d'un lien du réseau et la petite taille de la majorité des flux élastiques exigent une procédure implicite de contrôle d'admission. En effet, Il est impensable de mettre en oeuvre un processus de signalisation et de réservation des ressources pour chaque flux. En outre, le critère d'admission qui décide si un lien peut ou pas accepter un nouveau flux doit être le plus fiable possible. Nous proposons deux mécanismes de contrôle d'admission : avec et sans attente.
Le mécanisme de contrôle d'admission avec attente consiste en deux étapes. Un flux arrivant au réseau et ne pouvant pas être admis est, dans un premier temps, mis en attente. Dans un deuxième temps, à expiration d'un temps d'attente limite, le flux est définitivement rejeté. Les flux mis en attente sont servis selon une approche FIFO. Nous évaluons l'intérêt de garder des flux en attente. L'attente peut être implémentée en pratique par le retard de la transmission des paquets SYN. Le temps d'attente limite peut modéliser l'impatience des usagers ou une contrainte protocolaire. Dans notre travail nous modélisons le temps initial de l'expiration du temporisateur, qui au bout duquel, si on reçoit pas l'acquittement d'un paquet qui a été envoyé, le paquet sera retransmis. Ce paramètre est appelé "ITO (Initial Time Out)", et d'après le [RFC 1122], ce temps est fixé à 3 secondes. Nous aurons pu aussi modéliser le temps de la fin d'une connexion TCP qui est en général fixé à 3 minutes [RFC 1122]. Mais cette méthode va introduire un trafic supplémentaire dû aux requêtes envoyées pour solliciter la retransmission du premier paquet qui a été retardé dans la file d'attente .
Dans le cas du mécanisme sans attente un flux arrivant au réseau et ne pouvant pas être admis est définitivement rejeté.
Les seuils d'admissibilité ont bien sûr un impact significatif dans les performances du réseau (utilisation des liens par exemple) et dans la QoS offerte aux usagers. Il est donc important de les dimensionner de manière optimale. Ils sont fixés selon l'apprcohe choisie pour contrôler l'admission d'un nouveau flux. Il en existe trois : l'approche déterministe [KHA03], l'approche statistique [KHA04], et l'approche basée sur des mesures [OUE00, BEN02]. Les deux premières utilisent une estimation à priori tandis que la dernière est basée sur des mesures courantes de certains paramètres. L'approche déterministe effectue un calcul du pire des cas pour éviter toute violation de QoS. Cette technique, si elle est efficace pour les trafics fluides, l'est beaucoup moins pour les trafics en rafale et conduit à une sous-utilisation des ressources. Les deux autres approches autorisent une faible probabilité de violation de la QoS pour optimiser l'utilisation des ressources.
Dans notre travail, nous nous sommes basés sur une approche statistique pour déterminer les seuils d'admission. Ils ont été choisis afin de garantir un débit minimal par flux admis tout en maintenant un taux de blocage assez faible.
Dans la section suivante, nous introduisons un modèle analytique pour trouver les

seuils optimaux. Ce dernier modélise un lien avec une politique FIFO. Les hypothèses considérées ne sont pas toujours réelles, le but est d'avoir une idée approximative des seuils optimaux. Nous simulons ensuite l'architecture étudiée avec des hypothèses plus réelles.

4.4 Etude sur un lien

Nous définissons un modèle fluide au niveau flux pour calculer le seuil d'admission optimal pour chaque type de flux. Nous considérons un lien de capacité C=20Mbits/s simultanément traversé par les deux types de flux courts et longs. Nous supposons que les flux courts et longs sont générés selon un processus de Poisson d'intensité λ_1 et λ_2 respectivement. La taille des flux courts et longs suit une loi exponentielle de taille moyenne σ_1=17,5Ko et σ_2=1Mo. Dans le modèle, nous prenons l'hypothèse supplémentaire que les flux réalisent un partage parfaitement équitable. Ceci signifie qu'en présence de n flux simultanés, chaque flux reçoit une part de la bande passante égale à C/n. Nous supposons, en outre qu'un nouveau flux court est bloqué lorsque le nombre des flux courts actifs atteint N_1, ce qui garantit aux flux courts un débit minimum r_1 égal à ρ_1/N_1. Désignons par $\rho_1 = \lambda_1\sigma_1/C$ la charge moyenne introduite par les flux courts. De même, un flux long est bloqué lorsque le nombre des flux longs actifs atteint N_2, ce qui garantit aux flux longs un débit minimum r_2 égal à $(1-\rho_1)/N_2$. Avec ces hypothèses le système décrit ci-dessus peut être modélisé par une file d'attente $M/M/1$ avec la discipline dite Processor Sharing, à deux seuils N_1 et N_2. C'est une chaîne de Markov homogène et irréductible à deux variables d'états i et j, représentant le nombre de flux en cours appartenant aux deux classes courts et longs respectivement. Cette hypothèse nous permet de calculer les probabilités stationnaires $P(i,j)$, pour $0 \leq i \leq N_1$ et $0 \leq j \leq N_2$.
La figure 4.2 illustre le diagramme des transitions associé à la chaîne de Markov considérée.

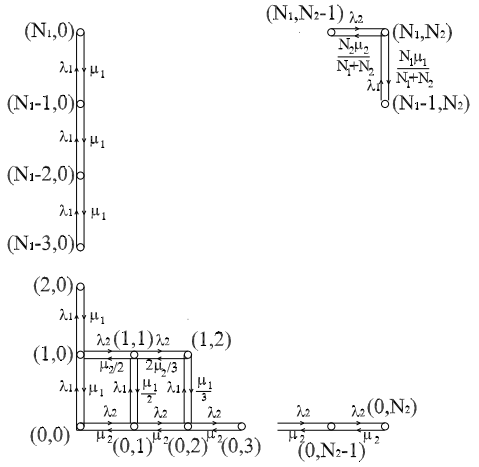

Figure 4.2 – Diagramme des transitions

L'écriture des équations de balance donne les relations de récursivité suivantes :

$$P(i-1,j)\lambda_1 + P(i,j-1)\lambda_2 = P(i,j)\left(\frac{i\mu_1 + j\mu_2}{i+j}\right), \quad i = N_1, j = N_2 \tag{4.1a}$$

$$P(i-1,j)\lambda_1 + P(i,j+1)\left(\frac{\mu_2}{i+1}\right) = P(i,j)(\mu_1 + \lambda_2), \quad i = N_1, j = 0 \tag{4.1b}$$

$$P(i,j-1)\lambda_2 + P(i+1,j)\left(\frac{\mu_1}{j+1}\right) = P(i,j)(\mu_2 + \lambda_1), \quad i = 0, j = N_2 \tag{4.1c}$$

$$P(i+1,j)\mu_1 + P(i,j+1)\mu_2 = P(i,j)(\lambda_1 + \lambda_2), \quad i = 0, j = 0 \tag{4.1d}$$

$$P(i-1,j)\lambda_1 + P(i+1,j)\mu_1 + P(i,j+1)\left(\frac{\mu_2}{i+1}\right) = P(i,j)(\lambda_1 + \lambda_2 + \mu_1), \quad 0 < i < N_1, j = 0 \tag{4.1e}$$

$$P(i,j-1)\lambda_2 + P(i,j+1)\mu_2 + P(i+1,j)\left(\frac{\mu_1}{j+1}\right) = P(i,j)(\lambda_1 + \lambda_2 + \mu_2), \quad i = 0, 0 < j < N_2 \tag{4.1f}$$

$$P(i,j-1)\lambda_2 + P(i-1,j)\lambda_1 + P(i,j+1)\left(\frac{(j+1)\mu_2}{i+j+1}\right) = P(i,j)\left(\frac{j\mu_2 + i\mu_1}{i+j} + \lambda_2\right), \quad i = N_1, 0 < j < N_2 \tag{4.1g}$$

$$P(i-1,j)\lambda_1 + P(i,j-1)\lambda_1 + P(i+1,j)\left(\frac{(i+1)\mu_1}{i+j+1}\right) = P(i,j)\left(\frac{j\mu_2 + i\mu_1}{i+j} + \lambda_1\right), \quad 0 < i < N_1, j = N_2 \tag{4.1h}$$

$$P(i,j-1)\lambda_2 + P(i-1,j)\lambda_1 + P(i+1,j)\left(\frac{(i+1)\mu_1}{i+j+1}\right) + P(i,j+1)\left(\frac{(j+1)\mu_2}{i+j+1}\right) = P(i,j)\left(\lambda_1 + \lambda_2 + \frac{j\mu_2 + i\mu_1}{i+j}\right), \quad 0 < i < N_1, 0 < j < N_2 \tag{4.1i}$$

En résolvant les équations de balance globale, nous pouvons calculer la probabilité stationnaire représentée par l'équation suivante :

$$P(i,j) = \frac{(i+j)!\rho_1^i\rho_2^j}{i!j!}P(0,0) \tag{4.2}$$

L'expression de $P(0,0)$ est donnée par :

$$P(0,0) = \frac{1}{\left(\sum_{i=0}^{N_1}\sum_{j=0}^{N_2}\frac{(i+j)!\rho_1^i\rho_2^j}{i!j!}\right)} \tag{4.3}$$

Soient P_1, D_1 et Q_1 la probabilité de blocage, le débit moyen et le nombre moyen des flux courts respectivement, ils sont donnés par les équations suivantes :

$$P_1 = \sum_{j=0}^{N_2} P(N_1, j) = \sum_{j=0}^{N_2} \frac{(N_1 + j)!}{N_1! j!} \rho_1^{N_1} \rho_2^j P(0,0) \tag{4.4}$$

$$D_1 = \sum_{i=1}^{N_1} \sum_{j=0}^{N_2} \frac{(i+j)! \rho_1^i \rho_2^j i C}{i! j! (i+j)} P(0,0) \tag{4.5}$$

$$Q_1 = \sum_{i=0}^{N_1} \sum_{j=0}^{N_2} i P(i,j) \tag{4.6}$$

En utilisant la formule de Little, nous pouvons calculer le temps moyen de séjour obtenu par un flux court individuel. Celui-ci est donné par la formule suivante :

$$s_1 = \frac{Q_1}{((1 - P_1)\lambda_1)} \tag{4.7}$$

Finalement le débit moyen obtenu par un flux court individuel est donné par la formule suivante :

$$d_1 = \sum_{i=1}^{N_1} \sum_{j=0}^{N_2} \frac{(i+j)! \rho_1^i \rho_2^j C}{i! j! (i+j)} P(0,0) \tag{4.8}$$

De la même façon, nous calculons les paramètres de performance des flux longs ils sont représentés par P_2, D_2, Q_2, s_1, d_1.

$$P_2 = \sum_{i=0}^{N_1} P(i, N_2) = \sum_{i=0}^{N_1} \frac{(i + N_2)!}{i! N_2} \rho_1^i \rho_2^{N_2} P(0,0) \tag{4.9}$$

$$D_2 = \sum_{i=0}^{N_1} \sum_{j=1}^{N_2} \frac{(i+j)! \rho_1^i \rho_2^j j C}{i! j! (i+j)} P(0,0) \tag{4.10}$$

$$Q_2 = \sum_{i=0}^{N_1} \sum_{j=0}^{N_2} j P(i,j) \tag{4.11}$$

$$s_2 = \frac{Q_2}{((1 - P_2)\lambda_2)} \tag{4.12}$$

$$d_2 = \sum_{i=0}^{N_1} \sum_{j=1}^{N_2} \frac{(i+j)! \rho_1^i \rho_2^j C}{i! j! (i+j)} P(0,0) \tag{4.13}$$

Après avoir trouvé les différents paramètres de performances pour chaque type de flux, nous cherchons à calculer les seuils d'admission optimaux. Ces seuils doivent être calculés afin d'assurer un débit minimal pour chaque flux court ainsi que pour chaque flux long. Ces derniers sont notés r_1 et r_2 respectivement. Pour cela, nous calculons, à partir des formules données ci-dessus, le débit moyen, le temps moyen

de séjour et la probabilité de blocage pour chaque type de flux pour différentes valeurs de r_1 et r_2. N_1 et N_2 sont calculés en divisant ρ_1 par r_1 et $(1-\rho_1)$ par r_2 respectivement.
Le calcul de ces équations sur Matlab ou Maple n'est pas tractable. En effet, les étapes intermédiaires pour résoudre ce type d'équations demandent beaucoup de capacité. Pour contourner ce problème, nous avons simplifié le calcul de ces équations en les transformant à des suites. Ainsi, nous pouvons les réécrire de la façon suivante :

$$(P(0,0))^{-1} = \sum_{i=0}^{N_1} a_{i0} + \sum_{i=0}^{N_1}\sum_{j=1}^{N_2} a_{ij} \tag{4.14}$$

a_{i0} et a_{ij} forment la suite suivante :

$$\begin{cases} a_{i0} = \rho_1^i \\ a_{ij} = \dfrac{i+j}{j}\rho_2 a_{i(j-1)} \end{cases}$$

$$d_1 = CP(0,0)(\sum_{i=1}^{N_1} b_{i0} + \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} b_{ij}) \tag{4.15}$$

b_{i0} et b_{ij} forment la suite suivante :

$$\begin{cases} b_{i0} = \dfrac{\rho_1^i}{i} \\ b_{ij} = \dfrac{i+j-1}{j}\rho_2 b_{i(j-1)} \end{cases}$$

$$d_2 = CP(0,0)(\sum_{j=1}^{N_2} c_{0j} + \sum_{i=1}^{N_1}\sum_{j=1}^{N_2} c_{ij}) \tag{4.16}$$

c_{i0} et c_{ij} forment la suite suivante :

$$\begin{cases} c_{0j} = \dfrac{\rho_2^j}{j} \\ c_{ij} = \dfrac{i+j-1}{i}\rho_1 c_{(i-1)j} \end{cases}$$

Nous procédons de la même façon pour calculer les autres paramètres de performance. En effet, cette méthode nous aide à éliminer les factorielles. Par conséquent, les équations sont plus faciles et plus rapides à calculer.
Les figures 3.3 jusqu'à 3.8 tracent les débits moyens, les temps moyens de séjour et les probabilités de blocage obtenus par les deux types de flux, en faible ($\rho = 0.9$) et forte ($\rho = 1.5$) charge. Ils sont tracés pour différentes valeurs de r_1 et r_2. Pour augmenter la charge, nous modifions la fréquence d'arrivée des flux, tous les autres paramètres restent constants.
Les seuils d'admission optimaux doivent être choisis afin d'assurer une bonne QoS pour tous les flux admis. Ceci, en maintenant assez faible la probabilité de blocage

des flux courts. En effet, ces derniers représentent 90% des flux TCP mais ils n'introduisent que 15 à 20% de la charge totale. Ainsi, en les bloquant, on n'allège pas vraiment le réseau, on ne fait qu'augmenter leur probabilité de blocage.

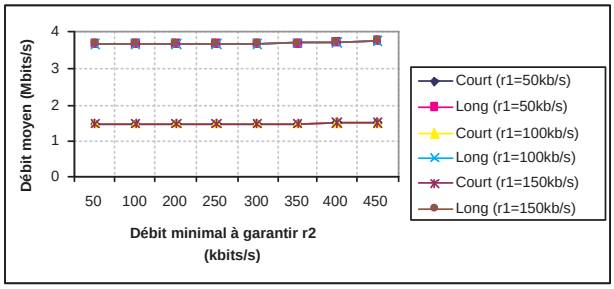

Figure 4.3 – Débits moyens obtenus par un flux court et par un flux long pour différentes valeurs de r_1, $\rho = 0,9$

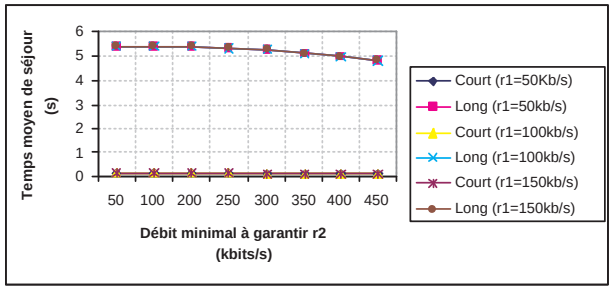

Figure 4.4 – Temps moyens de séjour obtenus par un flux court et par un flux long pour différentes valeurs de r_1, $\rho = 0,9$

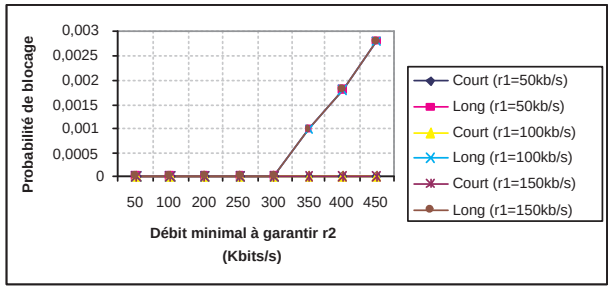

Figure 4.5 – Probabilités de blocage obtenues pour les flux courts et pour les flux longs pour différentes valeurs de r_1, $\rho = 0,9$

En faible charge (figures 4.3, 4.4 et 4.5), nous remarquons que pour les trois valeurs du débit minimal à garantir considérées (r_1=50, 100, 150Kbits/s), les résultats obtenus sont les mêmes. Ceci est dû au fait que le nombre instantané des flux courts

est toujours plus petit que N_1 (le nombre maximal des flux court admis) $\forall\ r_1$. Par conséquent, nous n'avons pas de blocage pour ce type de flux. Tandis que les paramètres de performances varient plus avec r_2 (le débit minimal à garantir pour un flux long), tout en restant très proches. Il faut noter qu'en réalité les débits obtenus par un flux court et par un flux long sont plus petits que ceux estimés par le modèle analytique. Ceci est vrai pour différentes raisons. Premièrement, parce que dans un réseau réel, le partage de la bande passante entre les différents flux TCP se fait selon le mécanisme de contrôle de congestion de TCP. Ce dernier ne peut pas toujours être approximé par une discipline Processor Sharing. Par conséquent, les flux courts n'arrivent pas à atteindre les débits estimés dans le modèle analytique suite aux limitations évoquées dans les chapitres précédents.
Deuxièmement, parce que le modèle analytique considéré ne tient pas compte des limitations dues aux RTT, à la fenêtre de congestion et aux retransmissions des paquets perdus qui avec lesquelles le système devient très difficile à modéliser.

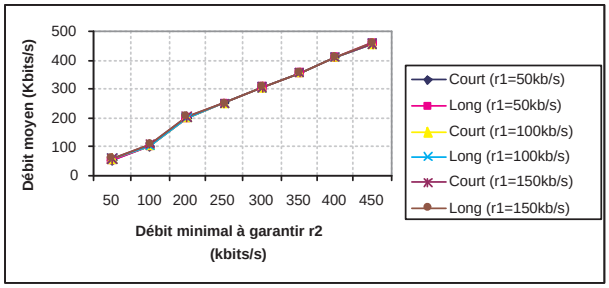

Figure 4.6 – Débits moyens obtenus par un flux court et par un flux long pour différentes valeurs de r_1, $\rho = 1,5$

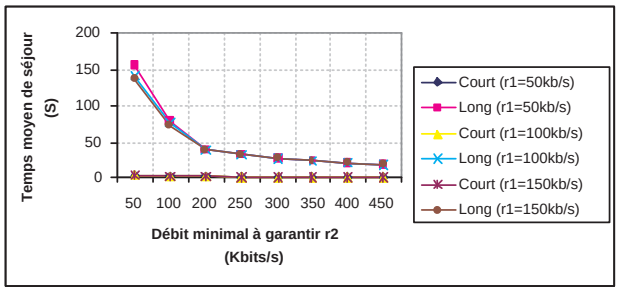

Figure 4.7 – Temps moyens de séjour obtenus par un flux court et par un flux long pour différentes valeurs de r_1, $\rho = 1,5$

Les figures 4.6, 4.7 et 4.8 tracent les différents paramètres de performances obtenus en forte charge (ρ=1.5) par les deux types de flux. D'après ces figures, nous remarquons que plus nous augmentons r_2 (le débit minimal à garantir par flux long) plus les performances des deux types de flux s'améliorent. Ceci est au détriment d'une

probabilité de blocage non négligeable pour les flux longs.
un seuil plus souple pour les flux longs, qui consiste à fixer r_2 à 250 ou 300 Kbits/s au lieu de 400 ou 450 Kbits/s (ce qui revient à accepter plus de flux longs), détériore les performances des deux types de flux sans pour autant diminuer la probabilité de blocage des flux longs. En effet, cette dernière est relativement stable et très proche de la limite fluide (Figure 4.8). Nous observons qu'un seuil inférieur à 250Kbits/s est extrêmement conservateur et engendre un blocage significatif pour les flux courts pour les trois valeurs de r_1 étudiés. Par conséquent, un seuil d'admission pour les flux longs qui correspond à r_2=350 ou 400kbits/s apparaît comme un choix raisonnable pour la configuration actuelle.

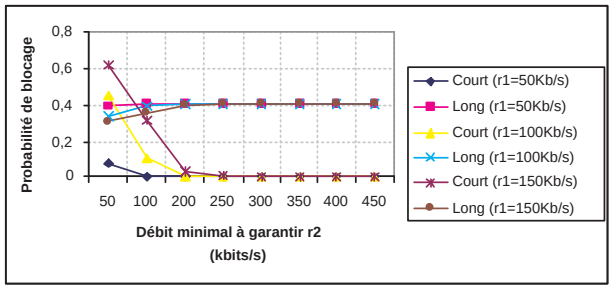

Figure 4.8 – Probabilités de blocage obtenues pour les flux courts et pour les flux longs pour différentes valeurs de r_1, $\rho = 1,5$

Nous cherchons maintenant un choix approprié pour r_1. Nous constatons que pour r_1=100 ou 150kbits/s, le taux de blocage des flux courts est assez élevé pour des valeurs de r_2 plus petites que 250kbits/s (Figure 4.8). Par contre, pour r_1=50kbits/s nous arrivons à garder un taux de blocage négligeable pour les flux courts. Mais, ceci est accompagné par une légère détérioration des performances des flux longs comparée aux deux autres seuils.
Nous tenons à noter qu'avec des valeurs de r_2 supérieures à 250Kbits/s, les trois valeurs de r_1 donnent les mêmes résultats. Ainsi, le seuil d'admission des flux courts n'aura pas beaucoup d'influence.
D'après cette étude analytique, nous constatons que les valeurs appropriées pour r_1 et r_2 sont respectivement 100 et 400 Kbits/s. Ceci nous permet de fixer les seuils d'admission N_1 et N_2 à (25,44) et (42,39) en faible et forte charge respectivement. Ils sont calculés en divisant ρ_1 par r_1 et $(1 - \rho_1)$ par r_2 (voir section 4.3.1). Nous tenons à noter que les seuils choisis ne sont que des valeurs approximatives. Dans la section 4.5, nous simulons le mécanisme TCP avec des hypothèses plus réelles pour choisir des seuils plus significatifs.

Dans le paragraphe suivant, nous comparons les paramètres de performance obtenus à partir du modèle analytique avec ceux obtenus à travers des simulations où le partage de la bande passante se fait selon le mécanisme de contrôle de congestion de TCP. Ceci avec les valeurs de (N_1, N_2) choisies ci-dessus.

4.4.1 Partage équitable Vs. Partage obtenu par TCP

Les tableaux 4.1, 4.2 et 4.3 comparent le débit moyen, le temps moyen de séjour et la probabilité de blocage obtenus, en faible et forte charge, à partir du modèle analytique et des simulations. Le premier représente le cas où un partage équitable est considéré. Le deuxième représente le mode de partage obtenu par le mécanisme de contrôle de congestion de TCP. Dans les simulations, nous reprenons toutes les autres hypothèses considérées dans le modèle analytique.
Les flux courts opèrent souvent dans la phase Slow-Start pour finir leurs transmissions. Cette dernière représente la phase de démarrage du mécanisme TCP. Elle peut servir jusqu'à 30 paquets[1] avant de passer à la phase Congestion Avoidance, si aucune perte n'a eu lieu. Par conséquent, leurs fenêtres de congestion sont souvent petites. Ce qui ne leur permet pas d'atteindre des débits très élevés comme ceux obtenus par les flux longs. Pour cette raison, nous remarquons que le débit moyen obtenu par un flux court dans le modèle analytique (noté analy.) est beaucoup plus élevé que celui obtenu par simulations (noté Sim.), surtout en faible charge.
En forte charge, le modèle analytique accorde le même débit moyen aux deux types de flux. Par contre, les simulations indiquent que les flux courts n'arrivent pas à atteindre ce débit moyen tandis que les flux longs parviennent à dépasser ce débit. En effet, la bande passante estimée et non utilisée par les flux courts sera exploitée par les flux longs dans les simulations. Ainsi, ces derniers atteingnent des débits plus élevés que ceux obtenus dans le modèle analytique.
Cette différence des résultats entre les simulations et le modèle analytique est due au fait que dans ce dernier nous considérons que les deux types de flux partagent équitablement la bande passante. Une hypothèse qui n'est pas toujours vraie.

	ρ=0.9		ρ=1.5	
	Analy.	Sim.	Analy.	Sim.
Court	1493	485	410	248
Long	3715	1992	411	696

TAB. 4.1 – Comparaison du débit moyen (kbits/s) entre le modèle analytique et les simulations

Dans cette partie, un modèle analytique, qui s'approche le mieux de l'architecture proposée, a été étudié. Ce modèle nous a aidé pour d'une part, trouver approximativement les seuils optimaux, et d'autre part, montrer que la discipline Processor Sharing n'est pas toujours une bonne hypothèse pour représenter le mode de partage obtenu par le mécanisme de contrôle de congestion de TCP.
Dans la partie suivante, nous simulons l'approche de contrôle d'admission proposée pour d'un côté, montrer son efficacité en le comparant avec un modèle sans contrôle d'admission et d'un autre côté, pour choisir les seuils optimaux.

[1] En général, le seuil threshold, qui représente le passage de la phase Slow-Start à la phase Congestion Avoidance, est fixé à 20.

	ρ=0.9		ρ=1.5	
	Analy.	Sim.	Analy.	Sim.
Court	0.0877	0.25	0.3857	1.9
Long	4.98	4.26	19.5	13.75

TAB. 4.2 – Comparaison du temps moyen de séjour (s) entre le modèle analytique et les simulations

	ρ=0.9		ρ=1.5	
	Analy.	Sim.	Analy.	Sim.
Courts	0	0.007	0	0.32
Longs	0.0018	0.0019	0.4074	0.25

TAB. 4.3 – Comparaison de la probabilité de blocage entre le modèle analytique et les simulations

4.5 Présentation du modèle de simulation

Afin d'évaluer les différents mécanismes de contrôle d'admission proposés et de mieux comprendre l'impact de leurs divers paramètres, nous les avons simulés dans le contexte d'un modèle de réseau simple en utilisant le simulateur NS "Network Simulator" [NET].

4.5.1 Modèle de simulation

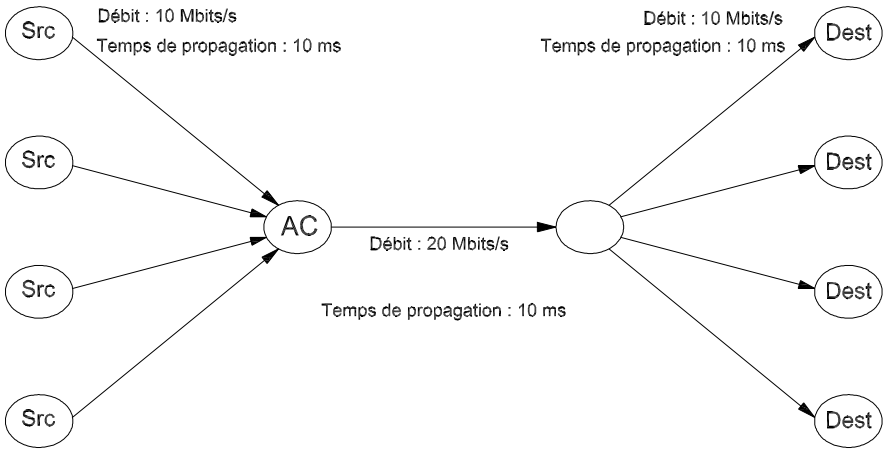

Figure 4.9 – Topologie du réseau simulé

Modèle du réseau : Nous considérons le modèle de réseau de la Figure 4.9. Il est constitué de quatre sources et quatre destinations, reliées respectivement aux rou-

teurs d'entrée et de sortie par des liens de 10 Mbits/s de débit et de 10 ms de temps de propagation.
Le contrôle d'admission est appliqué au niveau du routeur d'accès (AC) pour lequel le lien de sortie représente un goulot d'étranglement. Ce lien a un débit de 20Mbits/s et son temps de propagation est de 10 ms. Les paquets des flux courts et longs partagent les mêmes liens physiques.
Selon l'approche de gestion de trafic étudiée, le routeur implémente une file FIFO ou deux files avec un ordonnancement WFQ sur le lien de sortie. Dans le deuxième cas, la bande passante attribuée à l'agrégat des flux courts est égale à la charge moyenne introduite par ces derniers quand aucune des deux files n'est vide. Ce choix découle du fait que les flux courts représentent 90% des flux totaux mais juste 15 à 20% de la charge introduite (voir section 4.3). Ainsi, en garantissant leur charge moyenne, nous arrivons à diminuer leur taux de blocage sans pour autant charger le réseau. Dans les deux cas, la politique de gestion des files d'attente utilisée est DropTail.
Dans le cas du mécanisme de contrôle d'admission avec attente (voir la section 4.3.1), deux files d'attente sont implémentées au niveau flux, l'une pour les flux courts et l'autre pour les flux longs. Il s'agit de files FIFO à buffer infini. Les flux en tête de file quittent le buffer soit lorsqu'un flux actif a terminé et quitte le système soit à l'expiration d'un timer. Ces files, au niveau flux, sont indépendantes des files de transmission des paquets des flux actifs.

Modèle de trafic : Pour la simulation, les processus d'arrivée des flux courts et longs sont modélisés par des processus de Poisson indépendants. Des travaux sont en cours pour améliorer cette modélisation en se basant notamment sur les résultats de (Régnié, 2002).
Chaque flux produit un certain nombre de paquets de taille 1 Koctets. La taille des documents varie de la manière suivante : 90% des documents sont des fichiers de petite taille avec une distribution de Pareto de taille moyenne 17.5 Koctets ; le reste, sont des fichiers de grande taille avec une distribution de Pareto de taille moyenne 1 Moctets.
Nous focalisons nos efforts sur l'évaluation des paramètres de qualité de service suivants : le débit moyen obtenu par chaque flux (volume divisé par le temps de transmission de tout le flux), le temps moyen de séjour (attente avant le début de la transmission plus le temps de transmission de tout le flux), le temps moyen d'attente avant le début de la transmission et la probabilité de blocage. Nous appelons ces mesures débit moyen, temps moyen de séjour, temps d'attente et probabilité de blocage. Ces mesures caractérisent la QoS vue au niveau flux et ne doivent pas être confondues avec les mesures au niveau paquet (délai de traversée du réseau pour un paquet, par exemple, ou bien le temps d'attente des paquets dans les files).
Dans la suite, nous présentons les résultats des simulations de trois études. Les deux premières concernent l'évaluation de l'approche de contrôle d'admission proposée dans le cas FIFO avec et sans attente. La troisième concerne la comparaison des performances entre les cas FIFO et WFQ.
Pour les différentes études, les performances sont évaluées pour différentes valeurs de r_1 et r_2. Ils sont les débits minimaux à garantir pour un flux court et long

respectivement et qui sont à la base des calculs des seuils d'admission. Les paramètres de qualité de service ont été étudiés en sous charge (ρ=0.9) et en surcharge (ρ=1.5). Nous jouons sur l'intensité des processus d'arrivée des flux pour définir la charge, en effet, tous les autres paramètres du modèle ont été fixés.

4.5.2 Contrôle d'admission Vs. sans contrôle d'admission

Dans cette partie, nous évaluons l'approche de contrôle d'admission avec attente dans le cas d'un ordonnancement FIFO. Nous comparons ainsi les paramètres de performances obtenus dans ce dernier noté (Avec C.A) avec ceux obtenus dans le cas où le contrôle d'admission n'est pas activé (noté sans C.A).
Nous procédons de la même façon que dans le modèle analytique pour choisir les seuils optimaux. Ainsi, nous calculons les paramètres de performance obtenus pour différentes valeurs de r_1 et r_2. Les résultats sont ensuite comparés avec ceux obtenus dans le cas où le contrôle d'admission n'est pas appliqué. Ce dernier est représenté par une ligne horizontale sur les figures.
Les seuils sont choisis de sorte à maintenir assez faible le taux de blocage perçu par les flux courts tout en améliorant les autres paramètres de performances. Pour les flux longs, la qualité de service perçue lors de la transmission de ces derniers est déterminée par le débit, alors que celle perçue par les flux courts est déterminée par le délai. Nous travaillons ainsi sur ces deux paramètres pour montrer l'intérêt de notre approche.

4.5.2.1 Résultats obtenus en faible charge

Les figures 4.10 et 4.11 représentent le débit moyen et le temps moyen de séjour ainsi que le temps d'attente obtenus, en faible charge, par un flux court, en fonction des débits minimaux à garantir r_1 et r_2. De même, les figures 4.12 et 4.13 représentent ceux obtenus par un flux long.

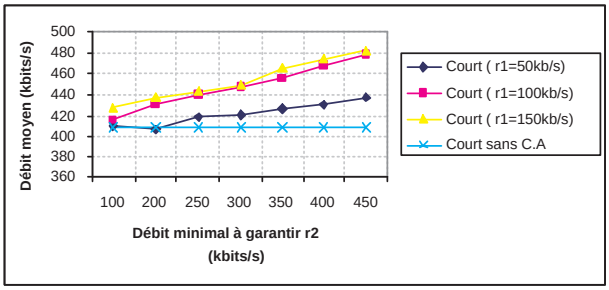

Figure 4.10 – Comparaison entre les modèles sans C.A et avec C.A du débit moyen (kbits/s) obtenu par un flux court, $\rho = 0.9$

La probabilité de blocage des flux courts est nulle pour toutes les valeurs considérées de r_1 et r_2 et est négligeable pour les flux longs (0.005 pour des valeurs de r_2 supérieures à 350kb/s avec r_1=50kb/s). Ceci est grâce à l'utilisation d'une phase

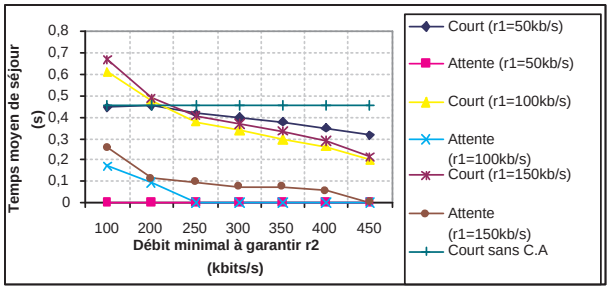

Figure 4.11 – Comparaison entre les modèles sans C.A et avec C.A du temps moyen de séjour (s) et du temps d'attente (s) obtenus par un flux court, $\rho = 0.9$

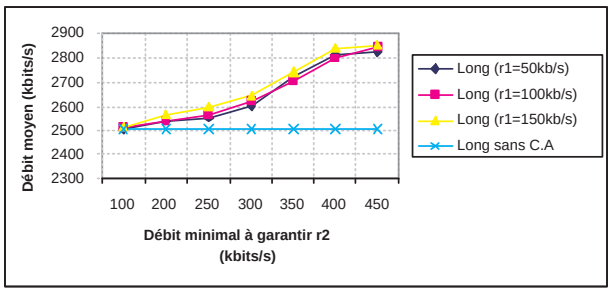

Figure 4.12 – Comparaison entre les modèles sans C.A et avec C.A du débit moyen (kbits/s) obtenu par un flux long, $\rho = 0.9$

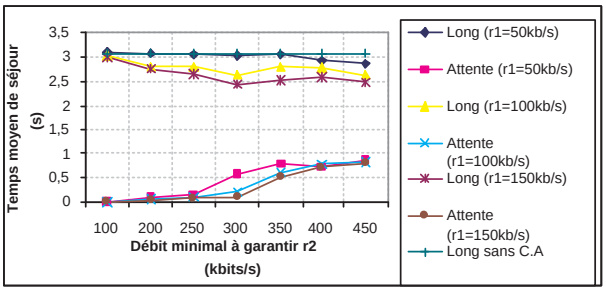

Figure 4.13 – Comparaison entre les modèles sans C.A et avec C.A du temps moyen de séjour (s) et du temps d'attente (s) obtenus par un flux long, $\rho = 0.9$

d'attente. Ainsi, un flux arrivant au réseau et ne pouvant pas être admis est mis en attente ; à l'expiration d'un temps d'attente limite[2] le flux est définitivement rejeté. Cette méthode nous aide d'un côté, à diminuer la probabilité de blocage et d'un autre côté, à mieux utiliser les ressources du réseau (voir section suivante).

[2]Le temps d'attente limite est représenté par l'expiration du ITO (Initial Time Out) qui vaut 3 secondes

Analyse des résultats

Les paramètres de performance obtenus par les flux courts varient avec les seuils r_1 et r_2, même si le dernier a beaucoup plus d'influence. En effet, en augmentant r_2 nous attribuons une part de la bande passante plus importante à chaque flux long. Par conséquent, le nombre des flux longs admis est plus petit. Les flux longs représentent 10% des flux totaux mais ils introduisent plus que 80% de la charge totale, vu leur grande taille.
D'après les figures, nous constatons qu'une valeur de r_1 égale à 100 ou 150 kbits/s aide à améliorer le débit moyen ainsi que le temps moyen de séjour obtenus par les flux courts. Tandis qu'avec r_1=50Kbits/s, les performances sont proches de celles obtenues quand le contrôle d'admission n'est pas appliqué. Celui-ci est représenté par une ligne horizontale sur les figures. Nous tenons à noter que le débit moyen est calculé en divisant le volume du fichier par le temps de transmission de tout le flux sans compter la durée de la phase d'attente. Durant laquelle le flux est en temps d'arrêt. Par contre, le temps de séjour représente toute la durée de vie du flux. Elle comprend le temps d'attente avant le début de la transmission et le temps de service de tout le flux.
D'après la figure 4.11, nous remarquons que la valeur de r_1=150kbits/s augmente le temps de séjour des flux courts comparé à celui obtenu avec r_1=100kbits/s. Ceci est à cause d'un temps d'attente élevé. Par conséquent, le temps de séjour est détérioré même si au contraire le temps de service est amélioré avec r_1=150kbits/s. Ce raisonnement est évident vu que le débit moyen est plus important avec r_1=150kb/s sans que ça soit le cas pour le temps moyen de séjour (dû au temps d'attente important).
De même, l'approche de contrôle d'admission proposée améliore les paramètres de performance des flux longs. Les meilleures valeurs du débit moyen et du temps moyen de séjour sont obtenues pour r_1=150kbits/s. En effet, cette dernière nous permet d'accepter un nombre de flux courts plus petit que celui obtenu avec les deux autres valeurs de r_1. Par conséquent, les flux longs sont moins perturbés par les flux courts et ils arrivent à finir plus rapidement leurs transferts.
Ces améliorations obtenues par les deux types de flux sont plus remarquables quand r_2 est grande. Etant que le nombre de flux longs acceptés diminue. Ainsi, nous avons moins de flux qui se concurrencent pour les mêmes ressources.
Pour $r_2 < 250$kbits/s, les paramètres de performance sont proches de ceux obtenus avec le modèle sans contrôle d'admission. En effet, pour des valeurs de r_1 et r_2 petites, nous n'avons quasiment pas des flux retardés ou bloqués. Tandis que, pour $r_2 > 400$kbits/s, les paramètres de performance des flux longs se stabilisent et aucune amélioration n'a été observée. Ainsi, augmenter le débit minimal à garantir ne fait qu'augmenter le taux de blocage des flux longs.
Dans le paragraphe suivant, nous traçons les paramètres de performances obtenus par les deux types de flux à forte charge ($\rho = 1.5$). Le but est de montrer d'un côté, l'efficacité de l'approche proposée et d'un autre côté, qu'avec la phase d'attente, nous arrivons à diminuer le taux de blocage éprouvé par les deux types de flux.

4.5.2.2 Résultats obtenus en forte charge

Les figures 4.14 jusqu'à 4.18 représentent les débits moyens et les temps moyens de séjour ainsi que les probabilités de blocage obtenus, en forte charge, par les deux types de flux. Les paramètres de performances sont tracés pour différentes valeurs de r_1 et r_2.

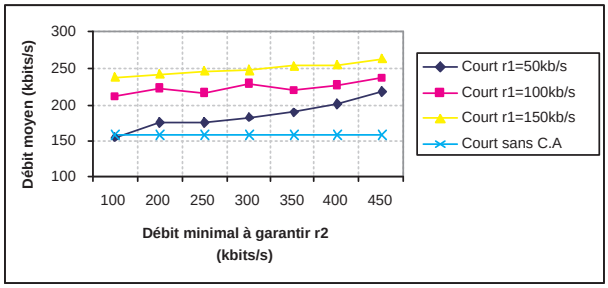

Figure 4.14 – Comparaison entre les modèles sans C.A et avec C.A du débit moyen (kbits/s) obtenu par un flux court, $\rho = 1.5$

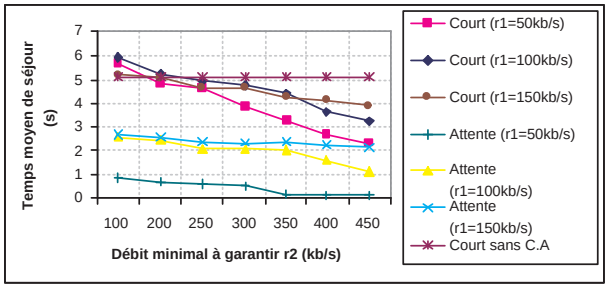

Figure 4.15 – Comparaison entre les modèles sans C.A et avec C.A du temps moyen de séjour (s) et du temps d'attente (s) obtenus par un flux court, $\rho = 1.5$

Analyse des résultats

A forte charge, les variations des paramètres de performance avec les seuils étudiés sont significatives. Nous étudions d'abord, l'influence de r_1 sur la QoS perçue par les deux types de flux. D'après les figures 4.16, 4.17 et 4.18, nous constatons que le débit moyen et le temps moyen de séjour ainsi que la probabilité de blocage des flux longs se sont améliorés quand nous augmentons r_1. En effet, en augmentant r_1 le nombre des flux courts admis simultanément diminue. Cette amélioration est au détriment d'une détérioration du temps de séjour des flux courts et d'un taux de blocage très élevé.

Nous aurons pu imaginer qu'avec des valeurs de r_1 grandes, les paramètres de performances des flux courts vont s'améliorer. Etant donné que nous acceptons moins de flux. Ceci a été uniquement vrai pour le débit moyen (figure 4.14). Tandis que, le temps moyen de séjour s'est détérioré à cause d'un temps d'attente très élevé (figure

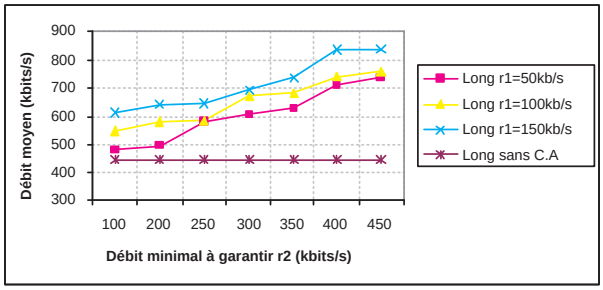

Figure 4.16 – Comparaison entre les modèles sans C.A et avec C.A du débit moyen (kbits/s) obtenu par un flux long, $\rho = 1.5$

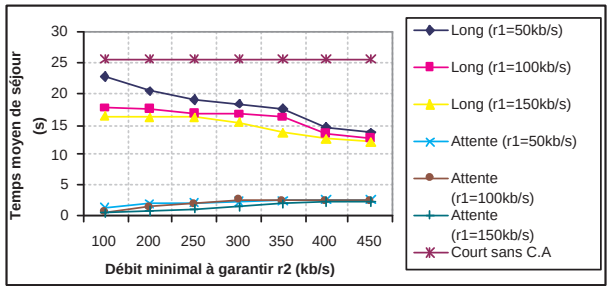

Figure 4.17 – Comparaison entre les modèles sans C.A et avec C.A du temps moyen de séjour (s) et du temps d'attente (s) obtenus par un flux long, $\rho = 1.5$

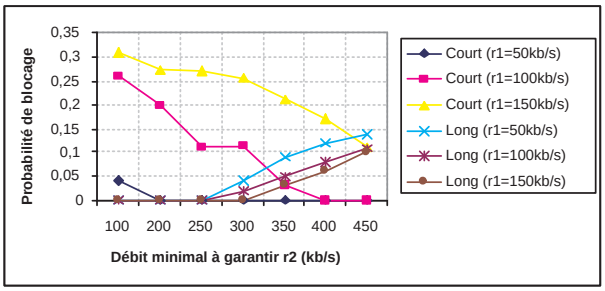

Figure 4.18 – Comparaison entre les modèles sans C.A et avec C.A des probabilités de blocage obtenus par les flux courts et longs, $\rho = 1.5$

4.15). D'après cette figure, nous remarquons que le temps d'attente le plus élevé est obtenu pour r_1=150kbits/s. Ce dernier est largement supérieur à celui obtenu avec les deux autres valeurs (r_1=50 et 100 kbits/s).
D'après cette étude, nous constatons que la valeur 50kbits/s est un choix approprié pour r_1. Cette valeur assure d'une part, une bonne QoS pour les flux courts, et d'autre part, un taux de blocage très faible. Par conséquent, le contrôle d'admission sera transparent vis à vis des utilisateurs. En effet, ce seuil maintient un taux de

blocage très faible pour les flux courts, même pour des petites valeurs de r_2 (r_2 < 200kb/s, ρ=1.5)(figure 4.18). Ce choix tient compte du fait que les flux courts appartiennent, en général, à la classe d'interactivité élevée ou moyenne (voir section 1.5.1), ce qui les rend plus sensibles au délai.
En revanche, les flux longs sont plus sensibles au débit et sont classés dans la catégorie d'interactivité faible. Ainsi, nous cherchons à trouver la valeur de r_2 qui garantit un débit moyen assez élevé par flux long. Par suite, le client n'interrompt pas sa session. S. Yang et G. de Veciana [YAN01] indiquent que la plupart des sessions interrompues sont générées par des flux longs. Le choix approprié pour r_2 est la valeur 400kbits/s. Cette dernière aide à améliorer les performances des deux types de flux tout en gardant un taux de blocage nul pour les flux courts et acceptable pour les flux longs, grâce à la phase d'attente.
En conclusion, nous constatons que les valeurs appropriées pour r_1 et r_2 sont respectivement 50 et 400 Kbits/s. Ceci nous permet de fixer les seuils d'admission N_1 et N_2 à (50,44) et (84,39) en faible et forte charge respectivement. Ils sont calculés en divisant ρ_1 par r_1 et $(1-\rho_1)$ par r_2.

4.5.2.3 Valeurs quantitatives

Dans cette partie, nous comparons les paramètres de performance obtenus dans le modèle avec contrôle d'admission (noté Avec C.A) avec ceux obtenus quand aucun contrôle d'admission n'est appliqué (noté Sans C.A). Dans le premier, les valeurs sont calculées pour les seuils N_1 et N_2 fixés dans la section précédente.

	ρ=0.9		ρ=1.5	
	Sans C.A	Avec C.A	Sans C.A	Avec C.A
Court	408	430	158	201
Long	2510	2813	442	711

TAB. 4.4 – Comparaison entre les modèles avec et sans C.A des débits moyens (kbits/s) obtenus par les deux types de flux

D'après le tableau 4.4, nous constatons qu'en appliquant l'approche de contrôle d'admission proposée, nous améliorons les débits moyens atteints par les deux types de flux. Ceci est vrai en faible et forte charge. Cette amélioration est de l'ordre de 27% pour les flux courts et de 60% pour les flux longs, en forte charge. Nous constatons ainsi que les flux longs, qui sont plus sensibles au débit, approuvent une forte amélioration. De même, les flux courts, qui sont plus sensibles au délai, approuvent une forte amélioration quant au temps de séjour. Le tableau 4.5 indiquent que cette dernière est de l'ordre de 23 et 47% en faible et forte charge respectivement.
L'amélioration des paramètres de performance est due au fait que le nombre de flux actifs est toujours contrôlé. En revanche, dans le cas où le contrôle d'admission n'est pas appliqué, nous constatons que ce nombre continue à augmenter même quand les ressources sont faibles. En effet, plus ce nombre augmente plus le débit par flux

diminue et les performances sont détériorées.

	ρ=0.9		ρ=1.5	
	Sans C.A	Avec C.A	Sans C.A	Avec C.A
Court	0.46	0.35	5.1	2.7
Long	3.08	2.92	25.65	14.66

TAB. 4.5 – Comparaison entre les modèles avec et sans C.A des temps moyens de séjour (s) obtenus par les deux types de flux

L'utilisation du contrôle d'admission constitue un dispositif pertinent. Il nous aide à maintenir une bonne QoS pour les deux types de flux même quand le réseau est chargé. De plus, en utilisant une phase d'attente, nous arrivons ainsi à préserver un taux de blocage nul pour les flux courts (Tableau 4.6). Ces derniers représentent 90% des flux totaux mais ils n'introduisent que 20% de la charge totale.

	ρ=0.9	ρ=1.5
	Avec C.A	Avec C.A
Court	0	0
Long	0	0.11

TAB. 4.6 – Les probabilités de blocage obtenues par les deux types de flux en faible et forte charge

4.5.2.4 Avantages du contrôle d'admission

Il est intéressant de constater les apports du contrôle d'admission par rapport au cas où toutes les connexions sont admises. La figure 4.19 met en relief l'évolution du nombre de connexions en cours en fonction du temps. Les connexions TCP courtes et longues démarrent sur un lien vide et subissent ensuite la concurrence avec d'autres connexions qui arrivent, produisant ainsi une charge de 1.5.
Sans contrôle d'admission, le nombre de flux courts et longs actifs croît indéfiniment, comme le montre la figure 4.19. Cette croissance est plus lente pour les flux courts. Ceci indique que la détérioration du débit moyen touche principalement les flux longs.
Avec le contrôle d'admission, le nombre de connexions actives se stabilisent autour des seuils 84 et 39 pour les flux courts et longs respectivement. Elles correspondent aux seuils d'admission fixés précédemment.
Dans la pratique, en cas de surcharge et en l'absence de contrôle d'admission, l'impatience des utilisateurs abandonnant prématurément leurs transferts limite la croissance du nombre de flux simultanés. Cependant, l'impatience mène clairement à l'inefficacité. Elle est discriminatoire envers les grands transferts.

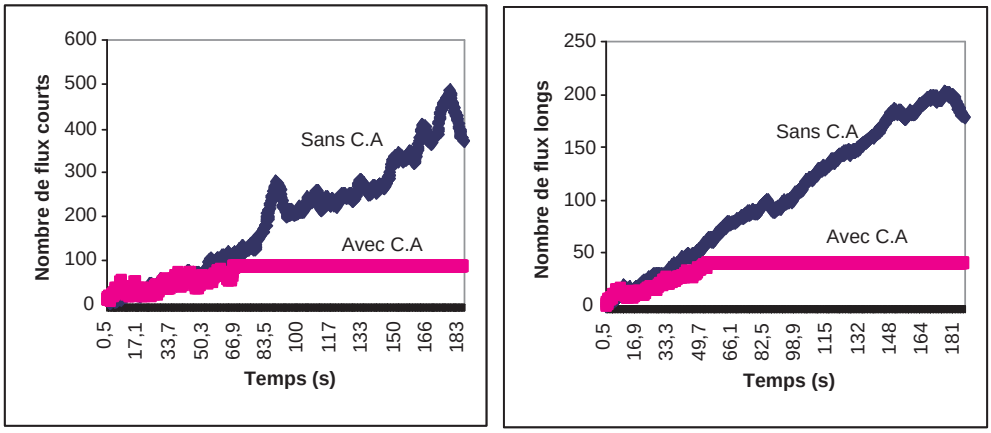

Figure 4.19 – Comparaison entre les modèles avec C.A et sans C.A du nombre de flux courts (à gauche) et longs (à droite) actifs, ρ=1.5

4.5.3 FIFO avec attente Vs. FIFO sans attente

Dans cette partie, nous évaluons l'intérêt d'utiliser une phase d'attente. Ainsi, un flux arrivant au réseau et ne pouvant pas être admis est, dans un premier temps, mis en attente. Dans un deuxième temps, à expiration d'un temps d'attente limite, le flux est définitivement rejeté. Les flux mis en attente sont servis selon une approche FIFO.

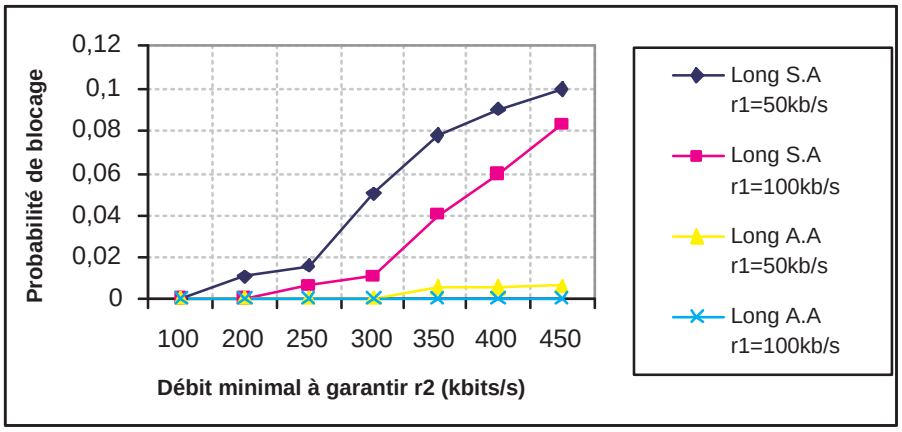

Figure 4.20 – Comparaison entre le modèle avec attente et le modèle sans attente de la probabilité de blocage obtenue pour les flux longs en fonction de r_1 et r_2 en faible charge, ρ=0.9

La figure 4.20 trace les probabilités de blocage obtenues par les flux longs en fonction de r_1 et r_2, en faible charge (ρ=0.9). Elle compare les résultats des deux modèles avec attente (noté A.A) et sans attente (noté S.A). Quand la phase d'attente est

utilisée, nous remarquons que la probabilité de blocage est toujours nulle sauf pour des valeurs de r_2 supérieures à 300kb/s (r_1=50kb/s). Durant la phase d'attente, un flux peut être retardé jusqu'à 3 secondes, qui est le temps limite au bout duquel le flux sera rejeté. Ce temps représente l'expiration du temporisateur ITO (Initial Time Out). Cette méthode nous permet de diminuer le taux de blocage sans pour autant augmenter le temps total de séjour. Ce dernier est toujours contrôlé puisque le nombre de flux accepté est toujours limité, ce qui permet aux flux actifs de finir plus rapidement leurs transferts. En revanche, quand la phase d'attente n'est pas implémentée, le taux de blocage tend à augmenter. Il croît avec r_2 et est plus fort pour r_1 égal 50kbits/s.

En faible charge, dans le modèle sans attente, la probabilité de blocage des flux courts est nulle pour r_1=50kbits/s et $\forall\, r_2$, ainsi que pour r_1=100kbits/s avec r_2>200kbits/s. Pour des valeurs de r_2<200kbits/s, la probabilité de blocage est négligeable et est de l'ordre de 0.003.

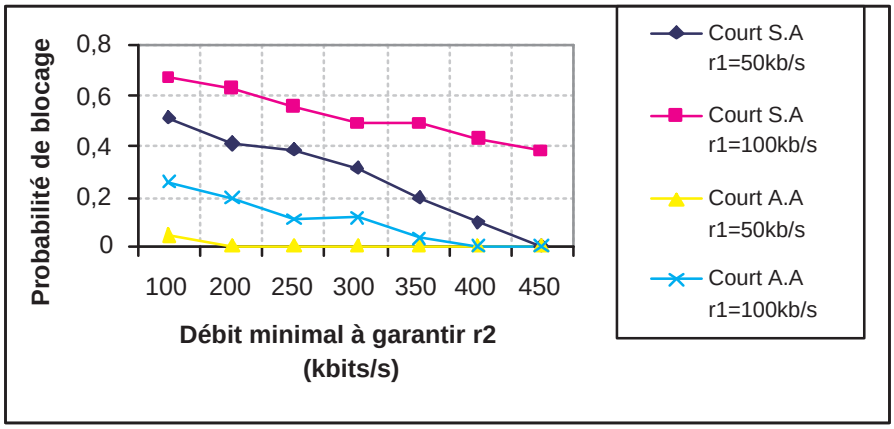

Figure 4.21 – Comparaison entre le modèle avec attente et le modèle sans attente de la probabilité de blocage obtenue pour les flux courts en fonction de r_1 et r_2 en forte charge, ρ=1.5

En forte charge (figure 4.21 et 4.22), quand la phase d'attente n'est pas utilisée, la probabilité de blocage augmente remarquablement pour les deux types de flux. Cette augmentation est plus importante pour les flux courts que pour les flux longs. En effet, les flux courts sont de petites tailles et leur fréquence d'arrivée est plus forte que celle des flux longs (7.5 flux/s contre 0.833 flux/s, en forte charge). Ainsi, quand la phase d'attente est utilisée, plusieurs flux courts déjà mis en attente parviennent à regagner le lien de transmission. Ils remplacent les flux actifs qui finissent leurs transferts. Ceci nous aide à diminuer leur taux de blocage.
Le tableau 4.7 donne les valeurs du taux de blocage obtenu pour les deux types de flux en faible et forte charge. Ils sont calculés pour r_1 égal à 50kbits/s et r_2 égal à 400kbits/s, qui sont les débits minimaux à garantir pour les flux courts et longs respectivement.

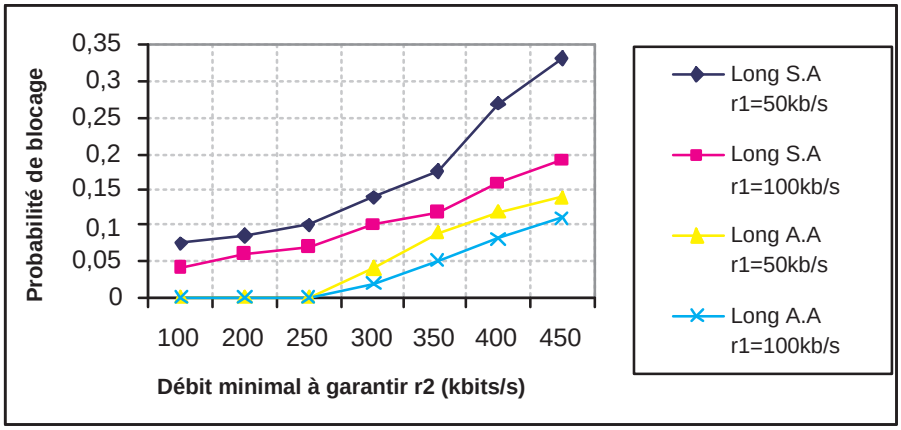

Figure 4.22 – Comparaison entre le modèle avec attente et le modèle sans attente de la probabilité de blocage obtenue pour les flux longs en fonction de r_1 et r_2 en forte charge, ρ=1.5

	ρ=0.9		ρ=1.5	
	Avec attente	Sans attente	Avec attente	Sans attente
Court	0	0	0	0.1
Long	0	0.09	0.11	0.27

TAB. 4.7 – les probabilités de blocage calculées pour r_1=50kbits/s et r_2=400kbits/s pour les deux types de flux

L'augmentation de la probabilité de blocage, quand la phase d'attente n'est pas utilisée, est sûrement accompagnée par une amélioration des autres paramètres de performance. Puisque la bande passante est moins utilisée. Mais ceci est au détriment d'un taux de blocage très élevé.

4.5.3.1 Débit instantané

Dans cette partie, nous comparons les débits instantanés obtenus par les deux modèles : avec et sans attente. Nous montrons ainsi l'avantage de la phase d'attente non seulement au niveau de l'amélioration du taux de blocage mais aussi au niveau de l'utilisation des ressources.
La figure 4.23 compare les débits instantanés en faible charge. Les débits minimaux à garantir sont fixés à 50 et 400kbits/s pour les flux courts et longs respectivement. Nous constatons que la bande passante est moins utilisée dans le cas sans attente. Ceci est dû au nombre plus important des flux rejetés dans ce dernier cas.
L'avantage de faire attendre les flux est d'un côté de mieux utiliser les ressources du réseau, et d'un autre côté, de diminuer le taux de rejet des flux.
Dans la section suivante, nous étudions l'intérêt de garantir une bande passante

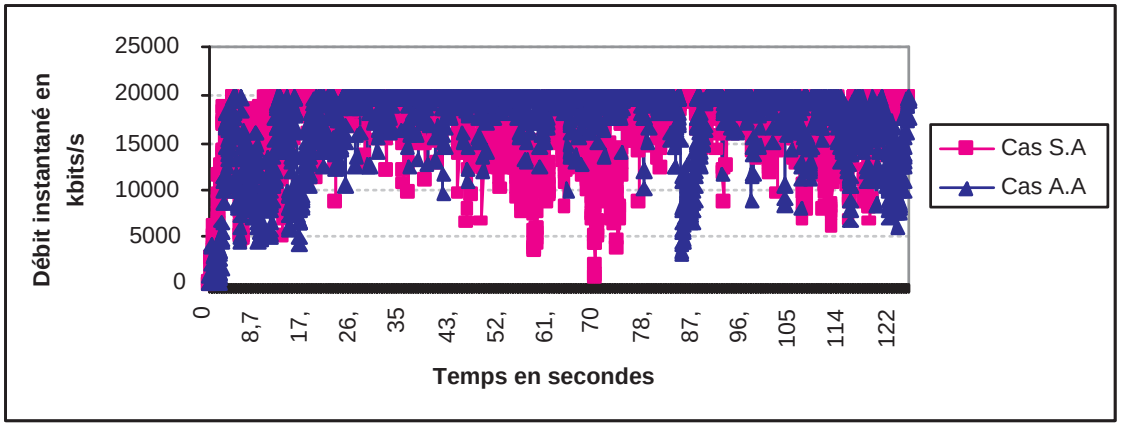

Figure 4.23 – Comparaison du débit instantané entre modèles avec et sans attente, $\rho = 0.9$

minimale pour les agrégats des flux longs et courts. Afin d'éviter des gaspillages de ressources, le partage de la bande passante entre agrégats se fait par un mécanisme du type WFQ.

4.5.4 FIFO avec attente Vs. WFQ avec attente

Nous nous intéressons maintenant au cas où l'ordonnancement est du type WFQ. L'intérêt d'introduire un tel mécanisme est évalué en le comparant avec le modèle FIFO. Le modèle étudié prend compte du contrôle d'admission ainsi que de l'attente au niveau des flux. Les valeurs de N_1 et N_2 ont été choisies comme indiqué dans la section 4.3 qui consiste à diviser ρ_1 par r_1 et (1-ρ_1) par r_2. r_1 et r_2 sont les débits minimaux à garantir pour un flux court et long respectivement.
La séparation des flux obtenue grâce à la politique WFQ permet de dimensionner plus facilement les seuils d'acceptabilité pour chaque classe. En particulier, le fait de diminuer à l'intérieur d'un agrégat la variance de la taille des flux rend le trafic plus prédictible et plus facile à dimensionner. Ceci est particulièrement vrai pour les flux longs qui arrivent à se partager équitablement la bande passante.

Les tableaux 4.8, 4.9 et 4.10 comparent les mesures de qualité de service des deux modèles.
Dans le modèle FIFO, les flux courts et longs partagent la même bande passante, il n'y a pas une capacité minimale garantie pour l'un ou l'autre. Ce phénomène pénalise les flux courts qui sont plus sensibles aux pertes. Les connexions TCP avec une large fenêtre de congestion sont plus tolérantes aux pertes. En effet, Les connexions longues peuvent recouvrir de multiples pertes au bout d'un RTT [FAL96] (la méthode de fast retransmission fast recovery étant utilisée, elle consiste à détecter les pertes grâce à la réception de trois acquittements dupliqués). Tandis que, les connexions courtes ont souvent besoin d'attendre l'expiration du TO (Time

	ρ=0.9		ρ=1.5	
	FIFO	WFQ	FIFO	WFQ
	$N_1 = 50, N_2 = 44$		$N_1 = 84, N_2 = 39$	
Court	430	510	201	317
Long	2813	2779	711	697

TAB. 4.8 – Comparaison du débit moyen en Kbits/s entre les modèles FIFO et WFQ avec C.A

Out) pour recouvrir un seul paquet perdu (vu que la taille de leurs fenêtres de congestion est en général petite, d'où la difficulté de recevoir trois acquittements dupliqués). Le tableau 4.8 montre qu'avec l'utilisation du WFQ, les flux courts bénéficient d'un débit plus important. Le prix à payer est une diminution du débit des flux longs. Néanmoins, il reste supérieur au débit minimal que nous voulons garantir (r_2=400Kbits/s).

	ρ=0.9		ρ=1.5	
	FIFO	WFQ	FIFO	WFQ
	$N_1 = 50, N_2 = 44$		$N_1 = 84, N_2 = 39$	
Court	0.35s	0.31s	2.7s	2.1s
Long	2.92s	3.1s	14.66s	16.56s

TAB. 4.9 – Comparaison du temps moyen de séjour entre les modèles FIFO et WFQ avec C.A

Le tableau 4.9 montre que le temps moyen de séjour des flux courts est clairement amélioré. En contrepartie, nous perdons un peu au niveau du débit et du temps moyen de séjour obtenus par les flux longs. Ainsi que le taux de blocage de ces derniers est faiblement augmenté (Tab. 4.10).

	ρ=0.9		ρ=1.5	
	FIFO	WFQ	FIFO	WFQ
	$N_1 = 50, N_2 = 44$		$N_1 = 84, N_2 = 39$	
Court	0	0	0	0
Long	0	0.001	0.11	0.13

TAB. 4.10 – Comparaison de la probabilité de blocage entre les modèles FIFO et WFQ avec C.A

Ces résultats montrent tout l'intérêt du contrôle d'admission proposé combiné à un ordonnancement WFQ. Non seulement les performances sont globalement améliorées

mais de plus, nous disposons dans ce contexte d'un bon contrôle du réseau permettant de le dimensionner correctement. Le dimensionnement est bien plus difficile à réaliser dans le cas où aucune séparation n'a été introduite.

4.6 Conclusion

Dans ce travail, nous avons proposé une approche de gestion de trafic pour l'Internet basée sur un contrôle d'admission prenant en compte la classification des flux élastiques en flux courts et longs. Un modèle analytique ainsi que des simulations ont été réalisés afin d'évaluer les performances des mécanismes proposés et d'analyser l'impact des seuils d'admission. Nous avons également analysé l'intérêt de faire attendre les flux qui ne peuvent pas être acceptés immédiatement.

Dans un premier temps, l'avantage d'appliquer le contrôle d'admission a été évalué en comparant ce modèle avec le modèle basic (sans contrôle d'admission). Les mesures de qualité de service ont été améliorées pour les deux types de flux. Cette amélioration se caractérise par une augmentation du débit pour les flux longs et une baisse du temps moyen de séjour pour les flux courts.
Dans un deuxième temps, nous avons mis en évidence l'importance d'introduire le concept d'attente au niveau des flux. Les avantages d'utiliser la phase d'attente est d'un côté de mieux utiliser les ressources du réseau, et d'un autre côté, de diminuer le taux de rejet des flux.
Dans un troisième temps, nous avons montré l'intérêt d'introduire un mécanisme de partage de ressources comme WFQ. Ce dernier nous aide à améliorer les performances des flux courts qui représentent 90% des flux TCP dans le réseau Internet. En contrepartie, une faible détérioration des performances des flux longs a été observée.

Il est important de signaler que, outre l'amélioration des performances dans les cas étudiés, l'approche proposée fourni un outil de dimensionnement du réseau permettant d'atteindre, pour une structure de trafic donnée, les mesures de performances attendues, ce qui est bien plus difficile à réaliser dans un cas sans différentiation entre flux longs et courts.

Dans l'architecture proposée, nous nous sommes basés sur le type d'application pour reconnaître la nature du flux et ensuite le classifier. Cette méthode de classification peut, quelquefois, introduire des erreurs quant au type de flux. Ceci nous a poussé à proposer une nouvelle architecture basée sur le traitement préférentiel. Cette architecture sera détaillée dans le chapitre suivant.

Chapitre 5

Traitement préférentiel appliqué sur les premiers paquets d'un flux

5.1 Introduction

Dans les chapitres 3 et 4, nous avons montré l'avantage de classifier les flux TCP que ce soit entre les flux longs avec RTT différent ou entre les flux courts et longs. Ces méthodes de classification, combinées avec des approches de contrôle d'admission, permettent d'améliorer les performances des différents types de flux. Elle assure un débit minimal par flux et diminue la variance de la taille des flux et du RTT au sein d'une même classe. Par conséquent, les flux arrivent à mieux se partager les ressources. Pour surmonter quelques difficultés constatées dans les modèles proposés, dans ce chapitre, une nouvelle approche basée sur le traitement préférentiel est proposée. Cette dernière est complémentaire aux études déjà abordées.

Comme nous l'avons expliqué dans le chapitre 2, le protocole de transport TCP couple le contrôle d'erreur avec le contrôle de congestion. La perte d'un paquet est toujours attribuée à une congestion dans le réseau. Elle est détectée par l'expiration d'un temporisateur de retransmission ou bien par la détection de trois acquittements dupliqués. Une connexion courte exige souvent un TO pour détecter une perte. Puisque, la plupart du temps, sa fenêtre de congestion a une valeur relativement petite, ainsi, elle n'a pas assez de paquets pour activer le mécanisme de trois acquittement dupliqués. Habituellement, une retransmission, après un TO (Time Out), peut sévèrement dégrader le taux de transmission pour ce type de connexion. Du fait que le TO est souvent très large. D'une manière primordiale, TCP se base sur ses propres paquets pour estimer la valeur appropriée du RTO. Pour les premiers paquets (SYN, SYN-ACK) et les premiers paquets de données, puisque aucune donnée n'est disponible, TCP doit employer une valeur initiale estimée du ITO comme RTO[1]. Perdre ces paquets peut avoir un effet désastreux sur les connexions courtes due à la grande période du TO. Pour ces raisons, les flux courts sont généralement

[1]La valeur recommandée d'ITO est de 3 secondes et est mise en application dans la plupart des logiciels d'exploitation modernes [SED01]. Quelques vieux systèmes emploient une valeur plus conservatrice de 6 secondes.

plus conservateurs que les flux longs et tendent, ainsi, à obtenir moins de bande passante quand ils se partagent un goulot d'étranglement avec les flux longs.

En tenant compte de ces différentes contraintes, nous proposons, dans ce chapitre, une nouvelle architecture assurant un traitement préférentiel aux premiers paquets de chaque connexion, favorisant ainsi les connexions courtes. Dans le chapitre précédent, nous avons étudié l'interaction entre les flux courts et longs et proposé alors de les séparer pour améliorer le temps de réponse et l'équité des flux courts. Ceci, en utilisant le mécanisme d'ordonnancement WFQ. En partant de cette architecture, à quelques différences près, nous proposons non pas de séparer les flux courts des flux longs, mais, de séparer les premiers paquets d'une connexion du reste des paquets. Ensuite, nous appliquons un traitement préférentiel à ces premiers paquets qui sont, en général, plus sensibles aux pertes. Cette architecture a la particularité de ne pas nécessiter le besoin de savoir la taille du flux à l'avance. Un flux arrivant au réseau sera classé comme court. Une fois il dépasse un certain seuil, le flux sera basculé dans une deuxième classe. Nous comptons également sur l'architecture DiffServ proposée dans [BER99] pour classifier les flux aux bordures d'un réseau. Plus spécifiquement, nous maintenons la longueur (en paquets) de chaque flux actif aux routeurs de bordures et l'employons pour classifier les paquets entrants. Cette architecture a la particularité de ne pas nécessiter le maintien en mémoire d'un état par flux au coeur du réseau. Dans ce dernier, nous utilisons la politique de gestion de file d'attente RED [FLO93] avec des seuils différents pour les deux types de classes. Ceci nous permet de réduire le taux de pertes éprouvé par les paquets des flux courts.

Nous montrons, à travers des analyses et des simulations, que le délai de transfert résultant pour les connexions de courte durée est réduit sans pour autant pénaliser les performances des autres connexions. D'ailleurs, notre méthode de classification traite les premiers paquets d'un flux long comme ceux d'un flux court. Pour cela, les premiers paquets de chaque flux éprouvent moins de pertes (y compris les paquets d'ouverture de connexion qui, autrement, causeraient des retards inutiles pour les sessions).
Nous confirmons également que notre modèle peut réaliser une meilleure équité et temps de réponse pour les flux courts que les modèles sans traitement préférentiel même lorsque la valeur de l'ITO est réduite à la configuration réelle, comme suggéré dans [SED01].

Pour améliorer les performances des flux TCP courts, des propositions comme [RFC2414], [PAD98b], [ZHA00] et [HEI97b] suggèrent l'utilisation d'une fenêtre de congestion initiale large, le partage d'information de mesure de réseau des précédents enregistrements ou la communication avec des voisins. Notons que ces propositions exigent la modification du protocole TCP et peuvent mener à l'augmentation du trafic. Notre architecture diffère de ces propositions puisque nous plaçons le contrôle à l'intérieur du réseau, et est ainsi transparent pour les clients. [GUO02a], [GUO02b] et [AVR04] explorent l'idée du PQ (Priority Queueing) traitant ainsi les flux courts avec une priorité absolue par rapport aux flux longs. Cette architecture reste effi-

cace tant que la charge des flux courts ne dépasse pas un certain seuil. En effet, un temps d'attente large entraîne les flux longs à avoir recours au TO pour détecter un paquet perdu. Ceci les force à diminuer leurs fenêtres de congestion à 1 paquet pour redémarrer en Slow-Start, ce qui dégrade leurs performances.

Notre modèle assure un traitement préférentiel aux premiers paquets d'un flux en leur attribuant une part de la bande passante assez large et en diminuant leur taux de perte. Notre but est de fournir un nouveau service meilleur que le service best effort qui tente à améliorer les performances des flux TCP courts. Ceci crée alternativement un environnement plus équitable et utilise mieux les ressources du réseau, particulièrement pour le trafic Web qui reste dominant dans le réseau Internet actuel.

5.2 Influence du taux de perte sur le temps de séjour

Dans cette section, nous étudions la sensibilité des flux courts et longs face aux pertes. Nous affirmons le fait que : pour réaliser des taux de transmission comparables à ceux obtenus par les flux longs, les flux courts devraient éprouver beaucoup moins de pertes, exigeant ainsi un traitement préférentiel au niveau des files d'attente du goulot d'étranglement.

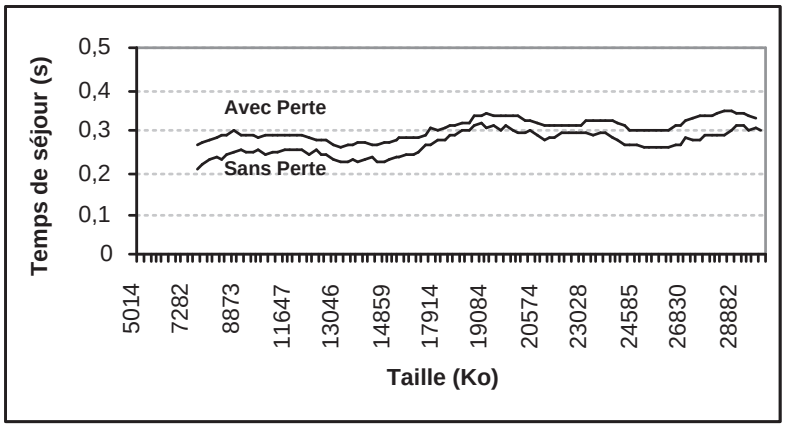

Figure 5.1 – Temps de séjour (s) obtenus par des fichiers de petites tailles, dans les deux cas : avec et sans perte

Nous simulons le scénario suivant : des sources génèrent des fichiers de petites tailles avec une distribution de Pareto de taille moyenne 17.5 Koctets sur un lien. Ce lien provoque des pertes avec une probabilité p=0.1. Le même scénario est répété avec des sources générant des fichiers de grandes tailles avec une distribution de Pareto de taille moyenne 1 Moctets. Les figures 5.1 et 5.2 tracent les temps moyens de séjour

obtenus par des fichiers de petites et grandes tailles respectivement. Dans chaque figure, nous représentons les deux cas : avec perte (p=0.1) et sans perte (p=0).

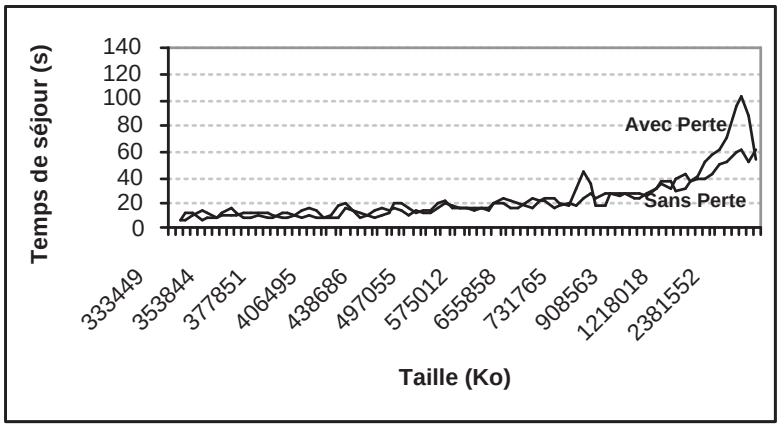

Figure 5.2 – Temps de séjour (s) obtenus par des fichiers de grandes tailles, dans les deux cas : avec et sans perte

Nous pouvons observer une tendance intéressante : pour les flux TCP de petite taille (Figure 5.1), l'augmentation du taux de perte mène à des temps de séjour élevés. Par contre, pour les flux longs, un taux de perte élevé a moins des effets négatifs sur leurs performances (Figure 5.2). Comme nous l'avons expliqué dans la section 5.1, pour les premiers paquets d'une connexion, il n'est pas possible d'activer le mécanisme de trois acquittements dupliqués quand une perte est détectée. Ainsi, un TO sera toujours exigé, ce qui cause l'augmentation du temps de séjour.

	Sans Perte	Avec Perte
Court	0.21 s	0.3 s
Long	20.36 s	22.01 s

TAB. 5.1 – Temps moyens de séjour obtenus par les flux courts et longs dans les deux cas : avec et sans perte

D'après le tableau 5.1, nous constatons que l'augmentation du temps moyen de séjour est beaucoup plus forte pour les flux courts et atteint 43%, tandis que cette augmentation est de l'ordre de 8.1% pour les flux longs.
Nous concluons ainsi, que la réduction du taux de perte pour les flux de petite taille est primordiale pour diminuer leurs temps de séjour. Pour cela, nous avons proposé un modèle avec un traitement préférentiel appliqué aux premiers paquets de chaque flux qui sont les plus sensibles aux pertes.

5.3 Traitement préférentiel

Pour classifier un flux à l'avance, le réseau doit connaître sa taille. Une telle information peut ne pas être aisément disponible. Dans le chapitre 4, nous nous sommes basés sur le type d'application pour classifier les flux. Ainsi, un flux FTP est considéré comme long tandis qu'un flux HTTP ou SMTP est considéré comme court. Ceci peut ne pas être toujours valable, comme par exemple, dans le cas d'un courriel avec un fichier attaché. Ce qui introduit un certain taux d'erreur.
Pour combler la difficulté de reconnaître la taille du flux à l'avance, nous proposons une nouvelle architecture qui assure un traitement préférentiel aux premiers paquets d'un flux. Ainsi, un flux arrivant au réseau passe par un ou deux étapes selon sa taille. Si cette dernière est plus petite qu'un certain seuil S_{seuil}, le flux finit son transfert en restant dans la première phase (classe). Sinon, le flux sera basculé dans la deuxième. Les flux qui ont un nombre de paquets supérieur à S_{seuil} finissent toujours leurs transferts dans la deuxième classe. Dans la suite, nous appelons un flux comme court ou long selon s'il finit son transfert dans la première ou deuxième classe respectivement. Les paquets de la première classe subissent un traitement préférentiel, d'un côté, en leur allouant une part de la bande passante égale à leur charge moyenne (sans qu'elle dépasse 50%), et d'un autre côté, en diminuant leur taux de perte.
Dans le réseau Internet actuel, la charge du trafic introduit par les flux courts ne dépasse pas 15 à 20% de la charge totale. Nous suggérons, dans le cas où ces proportions ne seront plus valables, d'appliquer un contrôle d'admission à l'entrée du réseau. Ce dernier sera appliqué au moment où la charge introduite par les flux courts dépasse un certain seuil.
Un traitement préférentiel pour les flux courts peut significativement améliorer leurs temps de séjour sans pour autant dégrader les performances des flux longs. Un flux long peut s'adapter à l'état du réseau et acquérir la quantité de ressources nécessaires pour transmettre un certain fichier. Ainsi, donnant un traitement préférentiel aux premiers paquets ne sacrifie pas les performances des flux longs. En effet, ce traitement préférentiel pourrait même améliorer leur QoS. D'une part, les flux longs fonctionnent dans un environnement plus stable (moins troublé par les flux courts) pendant des longues périodes, et d'autre part, leurs premiers paquets, qui sont aussi sensibles aux pertes, sont protégés.

5.3.1 Choix du seuil

Le seuil qui sépare les deux phases doit être soigneusement choisi. Il doit être suffisamment large pour laisser le temps à la fenêtre de congestion d'avoir suffisamment de paquets pour activer les trois acquittements dupliqués en cas d'une perte. Dans notre modèle, nous proposons de fixer le seuil S_{seuil} à 30 paquets. Nous supposons que les paquets sont de taille constante égale à 1000 Octets, ce qui correspond à 30 Koctets de données. Cette valeur a été choisie pour les raisons suivantes :

1. Un nouveau flux TCP a besoin d'une phase d'adaptation, appelée Slow-Start, pour ajuster le taux auquel il peut transférer ses paquets sans causer la conges-

tion dans le réseau. Cette phase se termine quand la fenêtre de congestion atteint 20 (cette valeur a été adoptée dans plusieurs versions du protocole TCP [RFC793], [RFC2581], [RFC2582]...), ce qui correspond à l'envoie de 30 paquets si aucune perte n'a eu lieu. Une fois le flux dépasse la phase Slow-Start, il continue son transfert dans la phase Congestion Avoidance. Durant cette phase, les flux arrivent à mieux se partager les ressources.

2. Dans la phase Slow Start, la plupart du temps, une perte est détectée par l'expiration du TO et non pas par la réception de trois acquittements dupliqués. Ceci dégrade la qualité du transfert, puisque, d'une part, le flux a besoin d'attendre 3 secondes[2] pour détecter une perte, et d'autre part, la fenêtre de congestion redémarre à 1 paquet. Un traitement préférentiel pour les premiers paquets d'un flux évite des temps d'attente très larges dues aux pertes.

3. Les flux courts sont défavorisés en présence des flux longs (voir chapitre 4), ainsi, un seuil égal à 30 paquets laisse le temps aux flux courts de finir leurs transferts dans la première phase.

Dans la section suivante, nous décrivons le modèle proposé ainsi que l'architecture et les mécanismes nécessaires pour son implémentation.

5.4 Modèle proposé : Architecture et Mécanismes

Dans cette section, nous décrivons le modèle proposé, y compris l'architecture de réseau que nous considérons et les mécanismes exigés pour différencier entre les flux TCP courts et longs.

5.4.1 Architecture

Nous supposons une architecture qui ressemble à un domaine DiffServ où les routeurs de bordures d'un domaine administratif maintient et contrôle toutes les informations concernant un flux. Les routeurs au coeur du réseau doivent seulement contrôler les classes des flux. La figure 5.3 illustre les opérations qu'un paquet a besoin pour traverser une telle architecture. Le routeur de bordure marque les différents paquets comme appartenant à la classe des flux courts ou longs. Une telle information est alors utilisée par les routeurs au coeur du réseau dans lesquels un mécanisme de gestion des files d'attente est utilisé.

[2]Dans leur article [SED01], Seddigh et al. proposent de fixer l'ITO à 1 seconde. Cette solution permet aux flux de détecter plus rapidement une perte, si la fenêtre de congestion n'a pas suffisamment de paquets pour activer les trois acquittements dupliqués. Cependant, une telle valeur agressive peut mener à des fortes congestions dans quelques parties du réseau où les liens sont lents et ont un large temps aller-retour. Employer des RTO agressifs sur des tels liens peut causer des retransmissions inutiles, un phénomène indésirable que les concepteurs de TCP ont toujours voulu éviter [JAC88].

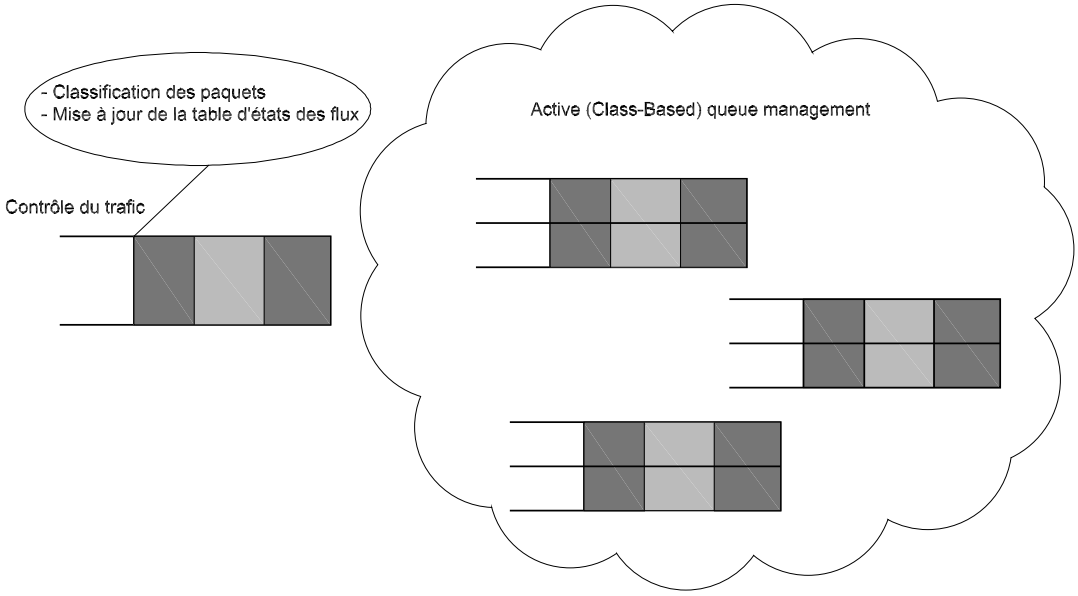

Figure 5.3 – Architecture proposée

5.4.2 Mécanismes

5.4.2.1 Routeur de bordure : Classification des paquets et maintien de l'état des flux

Vu la difficulté de déterminer si un paquet arrivant au réseau appartient à un flux court ou long, nous recourons à une méthode simple basée sur un seuil. Pour chaque flux actif, nous maintenons simplement un compteur qui dépiste combien de paquets ont été envoyés jusqu'ici. Une fois le compteur excède un certain seuil (S_{seuil}), nous considérons le flux comme étant long et classifions les paquets suivants en tant que long. Notons que sous une telle méthode de classification, les premiers paquets de tous les flux sont classifiés en tant que paquets appartenant à un flux court.
Les informations d'états par flux sont maintenues pour détecter sa fin. Plus spécifiquement, la table d'informations par flux est mise à jour périodiquement. Un flux est considéré terminé et sera éliminé du tableau, si aucun paquet de ce flux n'a été détecté au bout d'un temps T_{fin}[3].

5.4.2.2 Les routeurs au coeur du réseau : Application d'un traitement préférentiel pour les premiers paquets d'un flux

Il y a beaucoup de mécanismes de gestion et d'ordonnancement qui peuvent être utilisés pour une telle architecture. Nous choisissons la politique d'ordonnancement WFQ avec des files d'attente de type RED. Des taux de perte différents sont utilisés pour les flux courts et longs qui sont p et q respectivement. La figure 5.4 illustre

[3]Le temps, au bout duquel une connexion sera fermée si aucun paquet n'a été reçu, est fixé à 60 secondes [BRA95].

la méthode de marquage des paquets utilisée dans RED. Le marquage des paquets des flux longs est basé sur le calcul de la taille moyenne de la file d'attente totale (Q_{total}). Tandis que le marquage des flux courts est basé uniquement sur le calcul de la taille moyenne de la file d'attente des flux courts. Les valeurs recommandées dans l'article [CHR00] ont été utilisées.

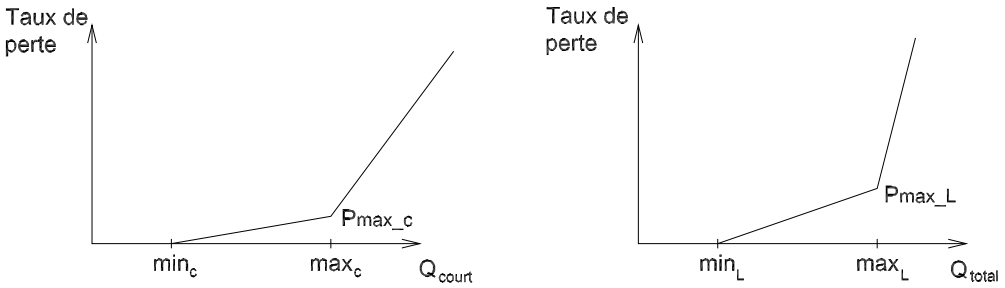

Figure 5.4 – Marquage des paquets courts (à gauche) et longs (à droite)

Dans chaque routeur deux files d'attente sont utilisées et sont gérées par le mécanisme d'ordonnancement WFQ. Un flux arrivant au réseau sera classé dans la première file. A partir d'un certain seuil S_{seuil} (voir section 5.3), le flux sera basculé dans la deuxième file. Les deux files disposent d'un pourcentage fixe de la bande passante disponible. Lorsqu'une file est vide, l'autre utilise toute la capacité de transmission. Quand aucune file n'est vide, chacune utilise la part de la bande passante qui lui a été réservée. Le pourcentage fixé pour la classe des flux courts est égal à 40% (à peu près deux fois la charge moyenne introduite par les flux courts). Ainsi, les paquets de cette classe subissent très rarement des pertes et sont protégés en période de congestion.

Dans la section suivante, nous faisons des simulations pour montrer l'avantage d'une telle architecture. Cette dernière sera comparée à deux autres architectures. La première consiste à un modèle classique où aucun traitement préférentiel n'a été envisagé pour la classe des flux courts. La deuxième consiste à appliquer une priorité absolue aux paquets des flux courts [GUO02a], [GUO02b] et [AVR04].

5.5 Simulations

Nous utilisons NS [NET] pour comparer les performances obtenues avec notre approche et celles obtenues avec : un modèle classique et un modèle où une priorité absolue est donnée aux flux courts.

5.5.1 Modèle de simulation

Modèle du réseau : Nous considérons le modèle du réseau de la figure 5.5. Le traitement préférentiel est appliqué au niveau du routeur d'accès (AC) pour lequel le

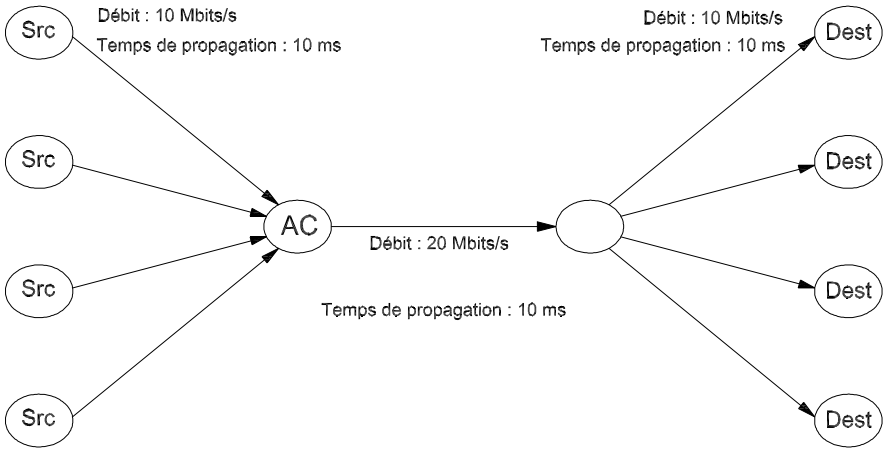

Figure 5.5 – Topologie du réseau simulé

lien de sortie représente un goulot d'étranglement. Ce lien a un débit de 20 Mbits/s et son temps de propagation est de 10 ms. Les paquets des flux courts et longs partagent les mêmes liens physiques.
Le routeur d'accès implémente deux files d'attente de la taille de 100 paquets chacune avec un ordonnancement WFQ. La bande passante attribuée à l'agrégat des flux courts est égale à 40% de la capacité totale. La politique de gestion des files d'attente utilisée pour les deux files est du type RED.
Les valeurs des seuils minimaux et maximaux de RED sont fixées à (80,100) et (20,80) pour les files des flux courts et longs respectivement [CHR00]. Comme nous l'avons indiqué dans la section précédente, le marquage des paquets des flux longs est basé sur le calcul de la taille moyenne de la file d'attente totale (Q_{total}). Tandis que le marquage des flux courts est basé sur le calcul de la taille moyenne de la file d'attente des flux courts uniquement. Les flux courts sont marqués avec un taux p qui est quatre fois plus petit que celui des flux longs q.

Modèle de trafic : Pour la simulation, les processus d'arrivée des flux courts et longs sont modélisés par des processus de Poisson indépendants. Chaque flux produit un certain nombre de paquets de taille fixe égale 1 Koctets. La taille des documents varie de la manière suivante : 90% des documents sont des fichiers de petite taille avec une distribution de Pareto de taille moyenne 17.5 Koctets. Les documents restant sont des fichiers de grande taille avec une distribution de Pareto de taille moyenne 1 Moctets. La version TCP New Reno est utilisée tout au long des simulations.
Nous focalisons nos efforts sur l'évaluation des paramètres de QoS suivants : le débit moyen obtenu par chaque flux (volume divisé par le temps de transmission de tout le flux) et le temps moyen de séjour (temps de transmission de tout le flux). Ces mesures caractérisent la QoS vue au niveau flux et ne doivent pas être confondues avec les mesures au niveau paquet (délai de traversée du réseau pour un paquet, par exemple, ou bien le temps d'attente des paquets dans les files).

Dans la suite, nous présentons les résultats des simulations de trois modèles. Le premier est le modèle classique dont aucune séparation ni traitement préférentiel n'ont été appliqués. Il sera désigné par Drop Tail. Le deuxième est l'approche que nous proposons qui sera désigné par TP. Le troisième consiste à donner une priorité absolue aux flux courts et il sera désigné par PQ (la priorité absolue est obtenue en faisant tendre le poids θ_c, fixé pour les flux courts, vers l'infini). Dans ces trois études, le même modèle de trafic est utilisé tandis que le modèle de réseau change selon le cas étudié.
Les paramètres de qualité de service ont été étudiés en sous charge (ρ=0.9) et en surcharge (ρ=1.8). Nous jouons sur l'intensité des processus d'arrivée des flux pour définir la charge. En effet, tous les autres paramètres du modèle ont été fixés.

5.5.2 TP Vs. Drop Tail et PQ

Dans cette section, nous analysons les résultats des simulations des trois études. Nous montrons ainsi l'avantage d'appliquer un traitement préférentiel aux flux courts.

5.5.2.1 Comparaison du temps moyen de séjour

Analyse des résultas en faible charge
La figure 5.6 trace les temps moyens de séjour, pour des fichiers de petites tailles, obtenus dans les cas TP, Drop Tail et PQ en sous charge (ρ=0.9). Les résultats indiquent que le traitement préférentiel, que ce soit avec TP ou PQ, diminue le temps moyen de séjour pour les flux courts de manière significative (50-60%).

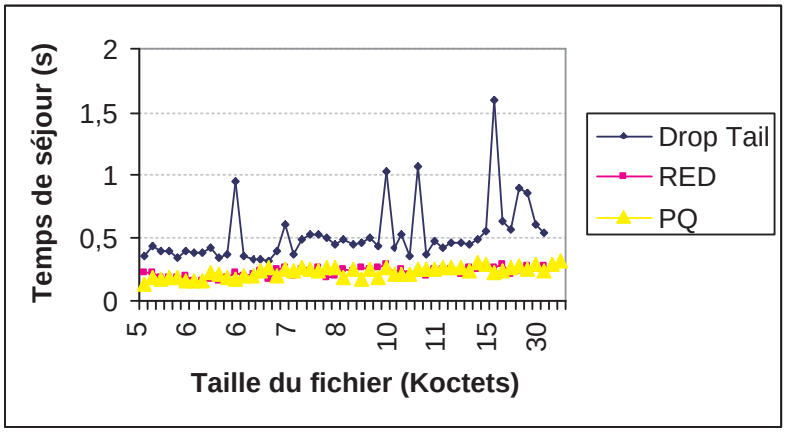

Figure 5.6 – Comparaison des temps moyens de séjour obtenus pour des flux de petites tailles, (ρ=0.9)

Quand un traitement préférentiel est appliqué pour les flux courts, ces derniers ne subissent pas des pertes (Tab. 5.2). La perte d'un paquet est le facteur principal de la détérioration des performances de ce type de flux. Le mécanisme TCP réagit aux

pertes en diminuant la fenêtre de congestion, par conséquent, les flux auront besoin de plus de temps pour finir leurs transferts.

	Drop Tail	TP	PQ
Courts	1642	0	0
Longs	5641	9463	9779

TAB. 5.2 – Pertes obtenues par les agrégats des flux courts et longs dans les cas : Drop Tail, TP et PQ, ρ=0.9

Nous constatons que l'amélioration des performances des flux courts n'est pas au détriment des performances des flux longs. En effet, le traitement préférentiel favorise aussi les premiers paquets des flux longs qui sont sensibles aux pertes. La figure 5.7 trace les temps moyens de séjour, pour des fichiers de grandes tailles, obtenus dans les trois modèles, en sous charge (ρ=0.9).

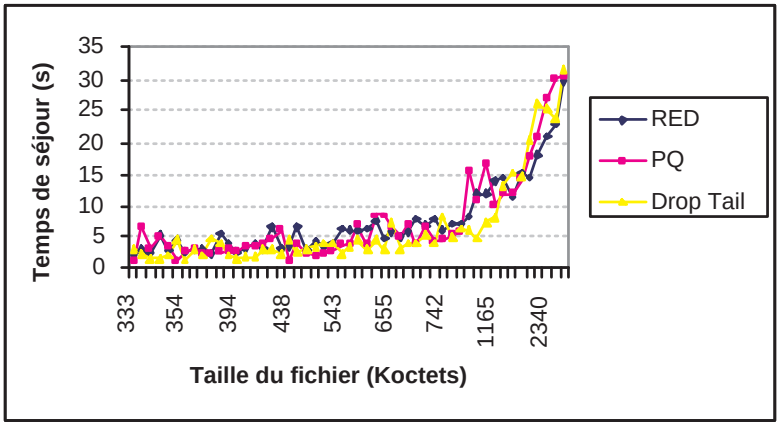

Figure 5.7 – Comparaison des temps moyens de séjour obtenus pour des flux de grandes tailles, (ρ=0.9)

Nous constatons, malgré le fait que les flux longs subissent plus de pertes dans les cas avec traitement préférentiel (Tab. 5.2), une légère détérioration du temps moyen de séjour a été observée (Figure 5.7). Cette dernière est de l'ordre de 17.5% et 23% dans les cas TP et PQ respectivement.
En effet, les pertes enregistrées, dans ces deux derniers cas, sont obtenues durant la phase Congestion Avoidance du mécanisme TCP. Ainsi, les conséquences qui en résultent sont moins désastreuses que si cela se produisait dans la phase Slow Start (voir section 5.3). Tandis que les pertes enregistrées pour les flux longs, dans le cas Drop Tail, sont obtenues durant la phase Slow Start ou Congestion Avoidance du mécanisme TCP.

Analyse des résultas en forte charge
En forte charge (ρ=1.8), le gain obtenu grâce au traitement préférentiel est très important. Ceci est valable pour les deux types de flux. En effet, les premiers paquets de chaque flux sont considérés comme courts et profitent aussi de ce traitement préférentiel. Le flux arrive ainsi à atteindre la phase Congestion Avoidance sans subir de perte.
Une fois les premiers paquets d'une connexion sont livrés avec succès, TCP gagne assez de connaissance du réseau pour estimer le temps de retransmission (RTO) et pour s'adapter à la capacité de transmission. Quand une perte est détectée dans la phase "Congestion Avoidance", le flux divise sa fenêtre de congestion par deux et récupère le paquet perdu. Ainsi, le flux n'a pas besoin de redémarrer sa fenêtre de congestion à 1.

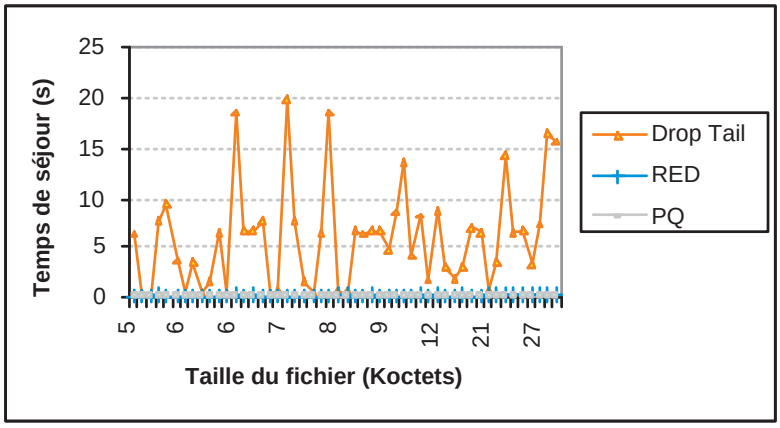

Figure 5.8 – Comparaison des temps moyens de séjour obtenus pour des flux de petites tailles, (ρ=1.8)

D'après les figures 5.8 et 5.9, nous constatons que le traitement préférentiel diminue le temps moyen de séjour pour les flux courts et longs de manière significative (97% et 50% respectivement).
Dans le cas de Drop Tail, les flux courts sont très défavorisés. Ils subissent un nombre de pertes très élevé (Tab. 5.3) ce qui diminue leurs taux de transfert et détériore leurs performances.
Pour les flux longs, malgré que le tableau 5.3 indique un nombre de pertes plus petit dans le cas Drop Tail, ces pertes sont observées durant les deux phases Slow Start et Congestion Avoidance. Tandis que, les pertes indiquées dans les deux autres cas sont observées uniquement dans la phase Congestion Avoidance ce qui diminue leurs effets négatifs.

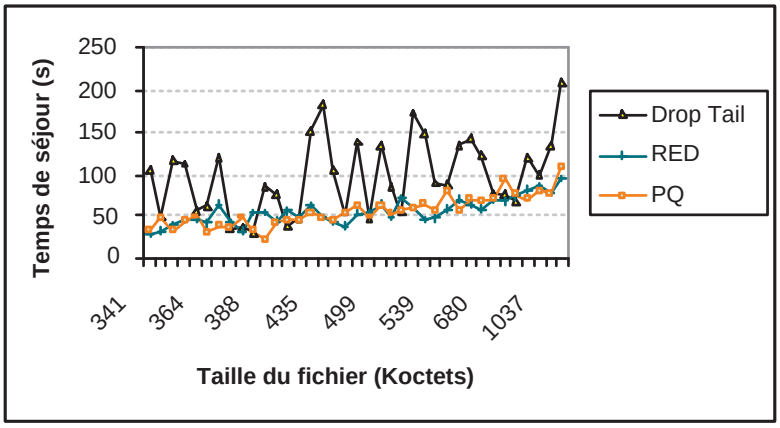

Figure 5.9 – Comparaison des temps moyens de séjour obtenus pour des flux de grandes tailles, (ρ=1.8)

	Drop Tail	TP	PQ
Courts	43147	0	0
Longs	76670	104233	106901

TAB. 5.3 – Pertes obtenues par les agrégats des flux courts et longs dans les cas : Drop Tail, TP et PQ, ρ=1.8

Valeurs moyennes

Les Tableaux 5.4 et 5.5 donnent les temps moyens de séjour ainsi que les débits moyens obtenus par les flux court et long en sous et sur charge respectivement.

	Drop Tail		TP		PQ	
	Débit	Temps de séjour	Débit	Temps de séjour	Débit	Temps de séjour
Court	259	0.57	404	0.22	407	0.22
Long	1478	5.73	1031	6.76	1023	7

TAB. 5.4 – Temps moyens de séjour (s) et débits moyens (kbits/s) obtenus dans les cas : Drop Tail, TP et PQ, ρ=0.9

Nous constatons qu'avec un traitement préférentiel pour les flux courts, ces derniers arrivent à atteindre des débits assez élevés et qui sont proches des débits obtenus par les flux longs. Ceci leur permet d'avoir un temps de séjour constant qui varie très peu avec la charge. En effet, favoriser les premiers paquets d'un flux aide à réduire leur taux de perte. Par conséquent, la plupart des flux courts parviennent à finir plus rapidement leurs transferts.

	Drop Tail		TP		PQ	
	Débit	Temps de séjour	Débit	Temps de séjour	Débit	Temps de séjour
Court	65.41	9.98	379.39	0.23	397.08	0.22
Long	149.63	63.19	463.68	32.6	406.36	37.01

TAB. 5.5 – Temps moyens de séjour (s) et débits moyens (kbits/s) obtenus dans les cas : Drop Tail, TP et PQ, ρ=1.8

5.5.2.2 Nombre de connexions actives

Dans les figures 5.10 et 5.12, nous traçons le nombre instantané des connexions courtes actives dans les trois architectures considérées, en sous et sur charge respectivement. De même, les figures 5.11 et 5.13 tracent celui des connexions longues. Elles représentent le changement de la stabilité du réseau.

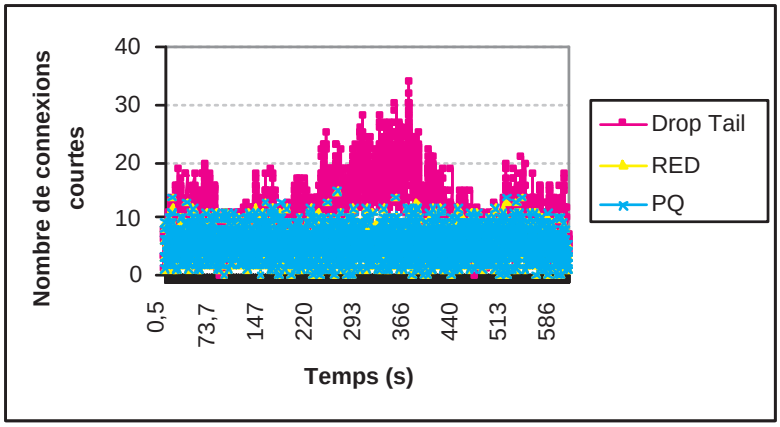

Figure 5.10 – Comparaison du nombre des connexions courtes instantanées entre les cas : Drop Tail, TP et PQ, (ρ=0.9)

Les architectures TP et PQ sont capables de réduire considérablement le nombre instantané de connexions courtes dans le réseau aussi bien que la variabilité (en sous et sur charge, figures 5.10 et 5.12 respectivement). Les figures montrent que TP est aussi efficace que PQ pour réduire le nombre global de flux actifs. La réduction du nombre de connexions courtes actives est accompagnée par une légère augmentation du nombre de connexions longues actives.
Les tableaux 5.6 et 5.7 donnent les nombres moyens des flux courts et longs, en sous et sur charge respectivement, obtenus dans les trois architectures considérées. Ainsi, nous constatons qu'un traitement préférentiel pour les flux courts aide à garder le nombre de connexions courtes assez petit sans pour autant augmenter le nombre de connexions longues. En effet, quand le nombre de flux augmente, le débit par flux diminue. Puisque la part de la bande passante obtenue par un flux dépend du

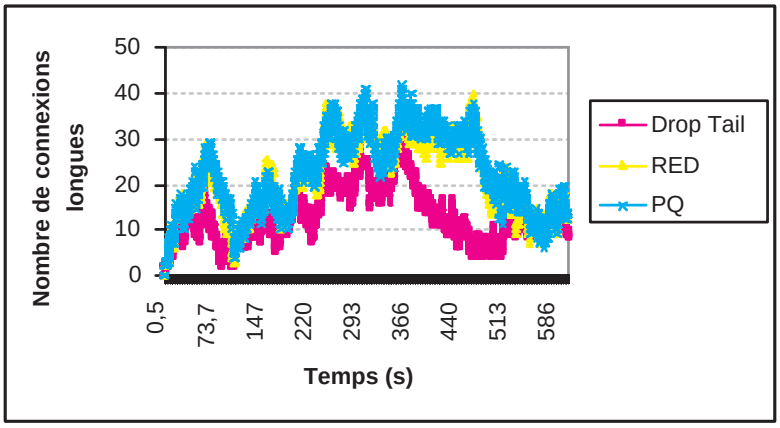

Figure 5.11 – Comparaison du nombre des connexions longues instantanées entre les cas : Drop Tail, TP et PQ, (ρ=0.9)

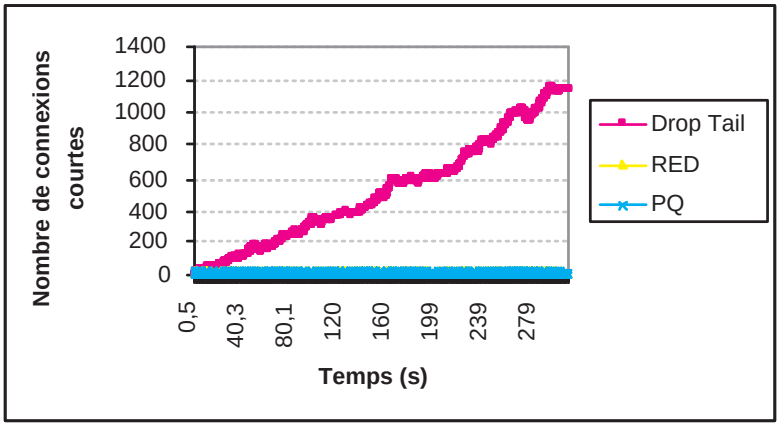

Figure 5.12 – Comparaison du nombre des connexions courtes instantanées entre les cas : Drop Tail, TP et PQ, (ρ=1.8)

nombre total de flux (dans le cas idéal, elle est égale à C/n où C est la capacité du lien et n est le nombre total de flux).

	Drop Tail	TP	PQ
Courtes	11	5.3	5.2
Longues	13.2	19.9	21.9

TAB. 5.6 – Nombre moyen de connexions courtes, ρ=0.9

D'après cette étude, nous montrons l'intérêt de l'architecture proposée. Il consiste principalement à l'amélioration significative des performances des flux courts ainsi

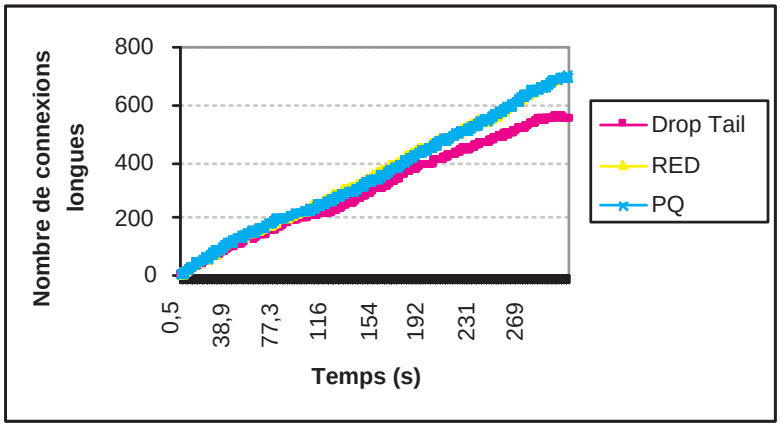

Figure 5.13 – Comparaison du nombre des connexions longues instantanées entre les cas : Drop Tail, TP et PQ, (ρ=1.8)

	Drop Tail	TP	PQ
Courtes	522	11	10.6
Longues	304	359	366

TAB. 5.7 – Nombre moyen de connexions longues, ρ=1.8

que celles des flux longs. Le trafic TCP constitue la majorité des octets circulant sur l'Internet d'aujourd'hui. Ce trafic est généré en grande partie par les flux appelés longs. Les flux courts représentent 90% des flux TCP mais ils n'introduisent que 15 à 20% de la charge totale. Le modèle présenté tient compte de ces proportions et des contraintes de QoS propres à chaque type de flux.

5.6 Discussion

Les simulations ont été effectuées dans un contexte où le trafic est unidirectionnel et où tous les flux TCP ont le même temps aller-retour. De ce fait, un tel modèle peut ne pas représenter la topologie vue par les utilisateurs dans le réseau Internet. Cependant, une telle hypothèse est seulement faite pour faciliter la comparaison des performances obtenues dans les différents contextes. Quand un trafic inverse est présent, notre modèle aura même une performance supérieure que celle du modèle classique Drop Tail. C'est parce que l'ouverture d'une session Web (et de la plupart des autres sessions de TCP) exige que le client échange une courte séquence de paquets de contrôle avec le serveur. Un tel échange sera mieux protégé dans notre modèle, particulièrement en présence d'un trafic dense dans la même direction.
Lorsque des flux TCP avec des RTT différents se partagent les mêmes liens, il suffit d'appliquer l'architecture proposée dans le chapitre 3 où une troisième classe sera

ajoutée dans laquelle nous classifions les flux qui sont moins rapides.
Notre modèle donne des résultats proches du modèle PQ avec une légère amélioration des performances des flux longs. Mais, l'avantage que le premier représente est qu'un contrôle sur la charge introduite par les flux courts est toujours possible, tandis qu'avec le modèle PQ, plus la charge des flux courts augmente, plus les performances des flux longs seront détériorées. Ceci est dû au fait que les flux courts sont toujours traités en priorité.

5.7 Conclusion

Dans ce chapitre, nous avons proposé une nouvelle architecture qui offre un nouveau service pour les flux élastiques. Elle ressemble à celle de DiffServ. Les routeurs de bordures implémentent une méthode de classification basée sur la taille du flux. Nous isolons ainsi les flux qui fonctionnent la plupart du temps dans différents régimes : les flux courts qui opèrent souvent dans la phase Slow Start et les flux longs qui opèrent dans la phase Congestion Avoidance. Les routeurs au coeur du réseau utilisent des politiques de gestion comme WFQ et RED pour assurer un traitement préférentiel aux premiers paquets d'un flux (favorisant ainsi les flux courts).
Cette architecture dispose de plusieurs avantages :

1. Les performances de la majorité des flux TCP (les transferts courts qui constituent 90% des flux TCP) sont significativement améliorées en termes de temps moyen de séjour et du débit moyen. De plus, le partage de la bande passante est plus équitable.
2. Les flux TCP courts ainsi que les premiers paquets des flux longs sont servis plus rapidement. Par conséquent, les performances de ces derniers (les grands transferts) sont également améliorées ou au pire légèrement affectées.
3. L'architecture proposée est extrêmement flexible parce que la fonctionnalité qui définit le nouveau service TCP peut être uniquement implémentée au niveau des routeurs de bordures.
4. L'architecture proposée ne nécessite plus de reconnaître la taille d'un flux à l'avance.

Notre but est de fournir un nouveau service meilleur que le service best effort qui tente à améliorer les performances des flux TCP courts et longs. Un tel service crée alternativement un environnement plus équitable et utilise mieux les ressources du réseau, particulièrement pour le trafic Web qui reste dominant dans le réseau Internet actuel.

Conclusions et Perspectives

Dans ce chapitre, nous présentons les conclusions tirées des études décrites dans ce rapport. D'abord, nous résumons les principales contributions. Ensuite, nous précisons les perspectives de prolongation qui se dégagent.

Contributions

Cette thèse met en avant l'importance d'assurer une qualité de service au trafic élastique. Nous nous plaçons dans le cadre d'une architecture de réseau orientée flux qui accorde autant d'importance à assurer une bonne qualité de service aux flux courts (un temps de séjour au dessus d'un seuil minimal) qu'aux flux longs TCP (un débit moyen au dessus d'un seuil minimal). Le réseau devrait être dimensionné de sorte à écouler l'ensemble du trafic avec une bonne qualité de service.
Pour garantir la qualité de service de ces deux classes de trafic, deux modèles de service ont été proposés, selon que la taille du flux est connue ou pas à l'avance. Le premier utilise une méthode de classification combinée avec une approche de contrôle d'admission appliqué aux deux types de flux. Le deuxième propose d'utiliser un traitement préférentiel pour les premiers paquets d'un flux, indépendamment de sa taille.

Partage de la bande passante : le partage de la bande passante entre les flux élastiques a toujours été supposé équitable et fidèle au modèle théorique de file d'attente avec une discipline de service "Processor Sharing". Or ce modèle analytique ne tient pas compte de la spécificité du protocole TCP, de ses paramètres et de ses limites. Pour cela, nous avons commencé nos études par analyser le partage statistique réalisé par TCP dans plusieurs configurations. Nous avons montré l'impact de certains paramètres comme le RTT et la taille maximale de la fenêtre de congestion sur le partage de la bande passante. Les flux TCP avec des RTT différents ne parviennent pas à partager équitablement la bande passante ainsi que les flux TCP avec des tailles différentes. Pour résoudre le problème de l'iniquité obtenu par le mécanisme de contrôle de congestion de TCP, nous avons proposé de classifier (séparer) les flux en plusieurs classes.

Classification des flux : la classification des flux a été étudiée dans deux cas de figures : premièrement, entre les flux TCP longs qui ont des RTT différents et deuxièmement entre les flux TCP courts et longs.

Dans un premier temps, nous montrons, en se basant sur une étude théorique et sur des simulations, l'avantage de séparer les flux TCP longs selon le RTT. Dans le cas de deux types de sources, l'avantage se caractérise, d'une part, par un débit plus important pour les connexions générées par la source avec un RTT plus large, et d'autre part, par un système plus prédictible puisque les flux d'un agrégat parviennent à partager équitablement la bande passante qui leur est allouée. Dans le cas de plusieurs sources, la séparation des flux en deux classes nous permet d'attribuer une bande passante plus petite, pour garantir un débit minimal par flux.
Dans un deuxième temps, nous montrons en se basant sur une étude analytique et sur des simulations l'avantage d'allouer une capacité minimale pour les agrégats des flux courts et longs. La séparation des flux obtenue grâce à la politique WFQ permet de protéger les flux courts et d'améliorer leurs performances. En particulier, le fait de diminuer à l'intérieur d'un agrégat la variance de la taille des flux rend le trafic plus prédictible et plus facile à dimensionner.

Contrôle d'admission : le contrôle d'admission a été largement utilisé dans cette thèse. Nous l'avons étudié pour les flux TCP (nous ne nous intéressons pas ici aux flux UDP qui restent encore minoritaires en nombre et en volume de ressources consommées). Nous avons ainsi proposé une approche de contrôle d'admission qui prend en compte la caractérisation en flux longs et flux courts ainsi que les contraintes de QoS propres à chaque type de flux.
Dans un premier temps, nous nous intéressons au cas où les deux classes de flux sont multiplexées à l'aveugle. Dans un deuxième temps, nous étudions l'intérêt de garantir une bande passante minimale pour les agrégats des flux courts et longs. Afin d'éviter des gaspillages de ressources, dans ce dernier cas, le partage de la bande passante entre agrégats se fait par un mécanisme du type WFQ : si un agrégat n'utilise pas la bande passante minimale qui lui est attribuée, la bande passante résiduelle peut être utilisée par l'autre agrégat.
Un modèle analytique (M/M/1/(N_1 + N_2) PS), qui s'approche le mieux de l'architecture proposée, a été étudié. Ce modèle nous a aidé à trouver les valeurs approximatives des seuils optimaux. Ensuite, des simulations ont été réalisées avec des hypothèses plus réelles afin d'évaluer les performances des mécanismes proposés et d'analyser l'impact des seuils d'admission. L'analyse de l'impact du nombre maximum de flux courts et longs concurrents sur la probabilité de blocage et le temps moyen de réponse, nous a conduit à recommander les couples de valeurs (N_1, N_2) suivants : (50, 44) et (84, 39), pour les seuils d'admission. Ces couples de valeurs représentent les nombres maximaux des flux courts et longs qui peuvent être acceptés simultanément et ceci en faible et forte charge respectivement. Nous avons également analysé l'intérêt de faire attendre les flux qui ne peuvent pas être acceptés immédiatement. Les avantages d'utiliser la phase d'attente est d'un côté, de mieux utiliser les ressources du réseau, et d'un autre côté, de diminuer le taux de rejet des flux.
Il est important de signaler que, outre l'amélioration des performances dans les cas étudiés, l'approche proposée fourni un outil de dimensionnement du réseau permettant d'atteindre, pour une structure de trafic donnée, les mesures de performances

attendues.
Dans l'architecture proposée, nous nous sommes basés sur le type d'application pour reconnaître la nature du flux et ensuite le classifier. Cette méthode de classification peut, quelquefois, introduire des erreurs quant au type de flux. Ceci nous a poussé à proposer une nouvelle architecture basée sur le traitement préférentiel.

Traitement préférentiel : le traitement préférentiel est appliqué aux premiers paquets de chaque connexion, qui sont plus sensibles aux pertes, favorisant ainsi les connexions courtes. Un flux arrivant au réseau sera classé comme court. Une fois il dépasse un certain seuil, le flux sera basculé dans une deuxième classe. Nous comptons également sur l'architecture DiffServ proposée dans [BER99] pour classifier les flux aux bordures d'un réseau. Plus spécifiquement, nous maintenons la longueur (en paquets) de chaque flux actif aux routeurs de bordures et l'employons pour classifier les paquets entrants. Cette architecture a la particularité de ne pas nécessiter le maintien en mémoire d'un état par flux au coeur du réseau. Dans ce dernier, nous utilisons la politique de gestion de file d'attente RED [FLO93] avec des seuils différents pour les deux types de classes. Ceci nous permet de réduire le taux de pertes éprouvé par les paquets des flux courts.
Dans un premier temps, nous comparons l'architecture proposée à un modèle classique où aucun traitement préférentiel n'a été envisagé pour la classe des flux courts. Nous montrons, à travers des analyses et des simulations, que le délai de transfert résultant pour les connexions de courte durée est réduit sans pour autant pénaliser les performances des autres connexions. D'ailleurs, notre méthode de classification traite les premiers paquets d'un flux long comme ceux d'un flux court. Pour cela, les premiers paquets de chaque flux éprouvent moins de pertes (y compris les paquets d'ouverture de connexion qui, autrement, causeraient des retards inutiles pour les sessions). Nous confirmons également que notre modèle peut réaliser une meilleure équité et temps de réponse pour les flux courts que les modèles sans traitement préférentiel. Dans un deuxième temps, nous comparons l'architecture proposée à un modèle qui consiste à appliquer une priorité absolue aux paquets des flux courts [GUO02a], [GUO02b] et [AVR04]. Notre modèle donne des résultats proches du modèle PQ avec une légère amélioration des performances des flux longs. Mais, l'avantage que le premier représente est qu'un contrôle sur la charge introduite par les flux courts est toujours possible, tandis qu'avec le modèle PQ, plus la charge des flux courts augmente, plus les performances des flux longs seront détériorées. Ceci est dû au fait que les flux courts sont toujours traités en priorité.

Notre but est de fournir un nouveau service meilleur que le service best effort qui tente à améliorer les performances des flux TCP courts et longs. Un tel service crée alternativement un environnement plus équitable et utilise mieux les ressources du réseau, particulièrement pour le trafic Web qui reste dominant dans le réseau Internet actuel.

Perspectives

Nos études menées sur la classification, le contrôle d'admission et le traitement préférentiel appliqués sur les flux élastiques ouvrent un certain nombre de perspectives.

Implémentation des modèles proposés : L'implémentation des approches étudiées sur une plate-forme est importante à faire. L'objectif de l'expérimentation sera de valider les solutions développées et d'identifier leurs limitations.

Penser à d'autres méthodes pour le calcul des seuils d'admission : Les seuils d'admission ont été calculés en fixant les débits minimaux à garantir appropriés pour les flux courts et longs. Cette solution est certes efficace et simple mais elle présente l'inconvénient de ne pas être dynamique. Il est donc intéressant de penser à d'autres solutions qui s'adaptent rapidement avec le changement des besoins.

Modèle analytique : Il serait intéressant d'étendre les modèles analytiques étudiés pour tenir compte du partage obtenu par le mécanisme de contrôle de congestion de TCP.

Impact de trafic streaming : Il serait aussi intéressant d'étudier l'impact du trafic streaming et de proposer ainsi l'intégration des deux types de trafic.

Ètude des approches proposées dans d'autres types de réseau : Nous pensons à étudier l'avantage des modèles proposés dans un réseau mobile tel que l'UMTS.

Glossaire

AAL	Adaptation Application Layer
AC	Admission Control
AF	Assured Forwarding
ATM	Asynchronous Transfer Mode
BA	Behaviour Aggregate
CA	Congestion Avoidance
CBR	Constant Bit Rate
CBQ	Class Based Queuing
CLS	Controlled Load Service
Cwnd	Congestion Windows
DT	Drop Tail
ECN	Explicit Congestion Notification
EDF	Earliest Deadline First
FECN	Forward Explicit Congestion Notification
EF	Expedited Forwarding
FIFO	First In First Out
FTP	File Transfer Protocol
GPS	Global Positioning System
GPS	Generalized Processor Sharing
GS	Guaranteed Service
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
ICMP	Internet Control Message Protocol
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ITO	Initial Time Out
LAN	Local Area Network
LSP	Label Switched Path
MAN	Metropolitan Area Network
MPLS	MultiProtocol Label Switching
MSS	Maximum Segment Size
NS	Network Simulator
PHB	Per Hop Behaviour
PQ	Priority Queuing
PS	Processor Sharing

QoS	Quality of Service
RED	Random Early Detection
RIO	RED In Out
RSVP	Resource ReSerVation Protocol
RTO	Retransmission Time Out
RTT	Round Trip Time
SMTP	Simple Mail Transfer Protocol
SS	Slow Start
TCP	Transmission Control Protocol
TO	Time Out
UDP	User Datagram Protocol
VBR	Variable Bit Rate
WAN	Wide Area Network
WFQ	Weighted Fair Queuing
WRED	Weighted RED
WRR	Weighted Round Robin
WWW	World Wide Web

Bibliographie

[ALT00] Altman E., Avrachenkov K., Barakat C., *A stochastic model of TCP/IP with stationary random losses.* In proceedings of ACM SIGCOMM, Stockholm, August 2000.

[AUR97] Aurrecoechea C., Campbell A., Hauw L., *A survey of qos architecture.* 1997.

[AVR04] Avrachenkov A., Urtzi A., Brown P., Nyberg E., *Differentiation Between Short and Long TCP Flows : Predictability of the Response Time.* In Proceedings of Infocom 2004, Hong Kong March 7-11 2004..

[AYE02] Ayesta U., Avrachenkov K., *The Effect of the Initial Window Size and Limited Transmit Algorithm on the Transient Behavior of TCP Transfers.* In 15th ITC Specialist Seminar on Internet Traffic Engineering and Traffic Management, Wurzburg, Germany, July 2002.

[BAR94] Bardford P., Crovella M., *Generating representative Web workloads for network and server performance evaluation.* In proceedings of ACM sigmetrics, 1998.

[BAR00] Barakat C., Altman E., Dabbous W., *On TCP Performance in a Heterogenous Network : A Survey.* IEEE Communications Magazine, January 2000.

[BEN01] Ben Fredj S., Oueslati-Boulahia S., Roberts J.W., *Measurement-based Admission Control for Elastic Traffic.* Teletraffic Engineering in the Internet Era, ITC 17, Elsevier, December 2001.

[BEN02] Ben Fredj S., *Un contrôle d'admission pour les flux IP dans un réseau multiservice.* PhD thesis, ENST, Juillet 2002.

[BER99] Bernet Y., *A Framework for Differentiated Services.* Internet Draft hdraft-ietf-diffserv-framework-02.txti, February 1999.

[BHA00] Bhatti N., Bouch A., Kuchinsky A. J., *Integrating User-Perceived Quality into Web Server Design.* In Proceedings of WWW, Amsterdam, May 2000.

[BOU00] Bouch A., Sasse M., DeMeer H. G., *Of Packets and People : A User-Centered Approach to Quality of Service.* In Proceedings Of IWQoS, June 2000.

[BRA94] Brakmo L., O'Malley S., *TCP Vegas : New Techniques for Congestion Detection and Avoidance.* In SIGCOMM '94 Conference on Communications Architectures and Protocols, (London, United Kingdom), pp. 24-35, Octobre 1994. (2)

[BRA95] Braun H. W., Claffy K. C., Polyzos G. C., *A Parameterizable Methodology for Internet Traffic Flow Profiling.* IEEE Journal on Selected Area in Communications, 13(8) : 1481-1494, October 1995.

[BRO00] Brown P., *Resource Sharing of TCP Connections with Different Round Trip Times.* In IEEE INFOCOM'2000, Tel-Aviv, Israel, March 2000.

[BUS95] Busse I., Deffner B., Schulzrinne H., *Dynamic QoS control of multimedia application based on RTP.* In proceedings First International Workshop on High Speed Networks and Open Distributed Platforms, Saint-Petersburg, Russia, June 1995.

[CAC91] Caceres R., Danzig P., Jamin S., Mitzel D., *Characteristics of Wide-Area TCP Conversations.* In Proceedings of ACM Sigcomm, September 1991.

[CHA01] Charzinski J., *Problems of Elastic Traffic Admission Control in an HTTP Scenario.* International Workshop on Quality of Service (IWQoS), Karlsruhe, Germany Springer-Verlag 2001.

[CHI89] Chiu D., Jain R., *Analysis of the Increase/ Decrease Algorithms for Congestion Avoidance in Computer Networks.* Journal of Computer Networks and ISDN, Jun 1989.

[CHR00] Christiansen M., Jeffay K., Ott D., Smith F. D., *Tuning RED for Web Traffic.* In Proceedings ACM SIGCOMM 2000, Stockholm, Sweden, Aug.-Sep. 2000.

[CLA88] Clark D., *The design philosophy of the DARPA internet protocols.* In ACM SIGCOMM, pages 106-114, 1988.

[CLA98] Clark D., Fang W., *Explicit allocation of best-effort packet delivery service.* In IEEE/ACM Transactions on Networking, 6(4) :362-373, 1998.

[CRO97] Crovella M., Bestavros A., *Self-Similarity in World Wide Web Traffic : Evidence and Possible Causes.* In IEEE/ACM Transactions on Networking, December 1997.

[DRA00] Drashinchi A., Fdida S., *Congestion avoidance for unicast and multicast traffic.* In ECUMN 2000, IEEE.

[FAL96] Fall K., Floyd S., *Simulation-based Comparisons of Tahoe, Reno, and SACK TCP.* Computer Communications Review, V. 26 N. 3, pp. 5-21, July 1996.

[FED00] Fedelmann A., *Characteristics of TCP connection arrivals. In self-similar network traffic and performance evaluation.* Edited by K. Park, W. Willinger, John Wiley and sons, 2000.

[FEL00] Feldmann A., *Characteristics of TCP connection arrivals.* Self-similar network traffic and performance evaluation edited by K. Park and W. Willinger, J. Wiley and Sons, 2000.

[FER98] Ferguson P., Huston G., *Quality of Service, Delivering QoS on the Internet and in Corporate Networks.* Wiley Computer Publishing, New-York, January 1998.

[FER00] Ferrari T., Chimento P. F., *A measurement-based analysis of expedited forwarding PHB mechanisms*. In Proceedings of IWQoS'00, 2000.

[FLO91] Floyd S., *Connections with Multiple Congested Gateways in Packet-Switched Networks Part 1 : One-way Traffic*. Computer Communication Review, Oct 1991.

[FLO93] Floyd S., Jacobson V., *Random early detection gateways for congestion avoidance*. In IEEE/ACM Transactions on Networking, 1(4) :397-413, August 1993.

[FLO01] Floyd S., *A report on recent developments in TCP congestion control*. 39 :84-90, Avril 2001.

[FRA01] Fraleigh C., Moon S., Diot C., Lyles B., Tobagi F., *Packet-Level Traffic Measurements from a Tier-1 IP Backbone*. In Proceedings of PAM, April 2001.

[GUE99] Guerin R., Peris V., *Quality of service in packet networks : Basic mechanisms and directions*. In Computer Networks, volume 31, pages 169-189, February 1999.

[GUO01] Guo L., Matta I., *The war between mice and elephants*. In Proceedings of the 9th IEEE International Conference on Network Protocols ICNP'01, 2001.

[GUO02a] Guo L., Matta I., *Differentiated control of web traffic : A numerical analysis*. In Proceedings of SPIE ITCOM'2002 : Scalability and Traffic Control in IP Networks, Boston, May 2002.

[GUO02b] Guo L., Matta I., *Scheduling flows with unknown sizes : Approximate analysis*. Tech. Rep. BU-CS-2002-009, Boston University, March 2002.

[HAH86] Hahne E., *Round Robin scheduling for fair flow control*. PhD thesis, Dept. Elect. Eng. And Comput. Sci., M.I.T, Decembre 1986.

[HEI97a] Heidemann J., Obraczka K., Touch J., *Modelling the Performance of HTTP Over Several Transport Protocols*. In IEEE/ACM Transactions on Networking, October 1997.

[HEI97b] Heidemann J., *Performance Interactions Between P-HTTP and TCP Implementations*. ACM Computer Communication Review, vol. 27(2), pp. 65-73, April 1997.

[JAC88] Jacobson V., *Congestion avoidance and control*. In Proceedings of ACM SIGCOMM, Stanford, CA, August 1988.

[JEA94] Jean-Marie A., Robert P., *On the transient behavior of the processor sharing queue*. Queueing Systems Theory and Applications, 17 :129-136,1994.

[KHA03] Khanafer R., Kofman D., *Contrôle d'admission dans l'Internet basé sur la classification des flots*. CFIP, Paris, Octobre 2003.

[KHA04] Khanafer R., Kofman D., *Internet admission control based on short and long flows classification*. HET-NETS, Ilkley- Angleterre, Juillet 2004

[KHA04a] Khanafer R., *Avantages de classifier les flux TCP longs selon le RTT*. DNAC, Paris, Novembre 2004

[KUM00] Kumar A., Hedge M., Anand S.V.R., *NETMASTER : Experiences in Using Nonintrusive TCP Connection Admission Control for Bandwidth Management of an Internet Access Link.* IEEE Communications Magazine, May 2000.

[LAK97] Lakshman T.V., Madhow U., *The performance of TCP/IP for networks with high bandwidth-delay products and random loss.* IEEE/ACM Transactions on Networking, June 1997.

[MAH97] Mah B., *An Empirical Model of HTTP Network Traffic.* In Proceedings of INFOCOM, April 1997.

[MAS99] Massoulié L., Roberts J., *Arguments in Favor of Admission Control for TCP Flows.* In Proceedings of ITC 16. Edinburgh, UK 1999.

[MAT00] Matta I., Guo L., *Differentiated Predictive Fair Service for TCP Flows.* In Proceedings of ICNP'2000 : The 8th IEEE International Conference on Network Protocols, Osaka, Japan October 2000.

[MAY99] May M., Bolot J. C., Diot C., Lyles B., *Reasons not to deploy RED.* In Proceedings of IWQoS'99, 1999.

[MOG95] Mogul J., *The Case for Persistent-Connection HTTP.* In Proceedings of SIGCOMM, August 1995.

[MOR00] Mortier R., Pratt I., Clark C., Crosby S., *Implicit Admission Control.* IEEE Journal on Selected Areas in Communications, Décembre 2000.

[NET] *Network simulator, ver.2, (ns-2).* Available at http ://www-mash.cs.berkeley.edu/ns.

[NUZ00] Nuzman C. J., Saniee I., Sweldens W., Weiss A., *A compound model of TCP arrivals.* In Proceedings of ITC seminar on IP Traffic Modeling, eds, Elsevier, volume 4, 2000.

[OUE00] Oueslati-Boulahia S., *Qualité de service et routage des flots élastiques dans un réseau multiservice.* PhD thesis, ENST, Décembre 2000.

[PAD98a] Padhye J., Firoiu, Towsley D., Kurose J., *Modeling TCP Trhoughput : A simple model and its empirical validation.* In SIGCOMM, 1998.

[PAD98b] Padmanabhan V., Katz R., *TCP Fast Start : a Technique for Speeding Up Web Transfers.* In Proceedings IEEE Globecom'98 Internet Mini-Conference, November 1998.

[PAD99] Padhye J., Firoiu, Towsley D., *A stochastic model of TCP Reno congestion avoidance and control.* CMPSCI Technical Report, 1999.

[PAX94] Paxson V., *Empirically derived analytical models of wide-area TCP connections.* IEEE/ACM Transactions on Networking, 2(4), August 1994.

[PAX95] Paxson V., Floyd S., *Wide-Area Traffic, The Failure of Poisson Modelling.* IEEE/ACM Transactions on Networking, vol., 3, 1995.

[PIT98] Pitkow J., *Summary of WWW Characterizations.* In Computer Networks and ISDN Systems Journal, volume 30, April 1998.

[REG02] Régnié G., *Ecoulement du trafic et intégration de services dans l'Internet.* PhD thesis, Université Paris 6, LIAFA, Septembre 2002.

[RFC1144] Jacobson V., *Compressing TCP/IP Headers for Low-Speed Serial Links.* RFC 1144, Internet Engineering Task Force, 1990.

[RFC1633] Braden R., Clark D., Shenker S., Herzog S., Jamin S., *Integrated services in the Internet architecture : an overview.* RFC 1633, Internet Engineering Task Force, 1994.

[RFC793] Shenker S., Wroclawski J., *Transmission Control Protocol.* RFC 793, 1981.

[RFC2414] Allman M., Floyd S., Partridge C., *Increasing TCP's Initial Window.* Internet RFC 2414, September 1998.

[RFC2481] Ramakrishnan K., Floyd S., *A Proposal to add Explicit Congestion Notification (ECN) to IP.* RFC 2481, Janvier 1999.

[RFC2581] Allman M., Paxson V., Stevens W., *TCP Congestion Control.* RFC 2581, Avril 1999.

[RFC2582] Floyd S., Henderson T., *The NewReno Modification to TCP's Fast Recovery Algorithm.* RFC 2582, Avril 1999.

[RFC2018] Mathis M., Mahdavi J., Floyd S., Romanow A., *TCP Selective Acknowledgment Options.* RFC 2018, Octobre 1996.

[RFC2883] Floyd S., Mahdavi J., Mathis M., Podlosky M., *An Extension to the Selective Acknowledgement (SACK) Option for TCP.* RFC 2883, Août 1999.

[RFC1323] Jacobson V., Braden R., Borman D., Podlosky M., *TCP Extensions for High Performance.* RFC 1323, Mai 1992.

[RFC1122] Braden R., *Requirements for Internet Hosts – Communication Layers.* RFC 1122, Octobre 1989.

[RFC2748] Boyle J., Cohen R., Herzog S., Rajan R., *The COPS (Common Open Policy Service.* RFC 2748, IETF Janvier 2000.

[RFC3031] Rosen E., Viswanathan A., Callon R., *Multiprotocol Label Switching Architecture.* RFC 3031, IETF Janvier 2001.

[RFC1945] Berners-Lee T., Fielding R., Frystyk K., *Hypertext Transfer Protocol - HTTP/1.0.* RFC 1945, Internet Engineering Task Force, 1997.

[RFC2205] Braden R., Zhang L., Berson S., Herzog S., Jamin S., *Resource ReSerVation Protocol, Version 1 Functional Specification.* RFC 2205, Internet Engineering Task Force, 1997.

[RFC2212] Shenker S., Partridge C., Guerin R., *Specification of guaranteed quality of service.* RFC 2212, Internet Engineering Task Force, September 1997.

[RFC2210] Wroclawski J., *The use of RSVP with IETF integrated services.* RFC 2210, Internet Engineering Task Force, September 1997.

[RFC2215] Shenker S., Wroclawski J., *General Characterization Parameters for Integrated Service Network Elements.* RFC 2215, 1997.

[RFC2475] Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W., *An architecture for Differentiated Services*. RFC 2475, Internet Engineering Task Force, December 1998.

[RFC2330] Paxson V., Almes G., Mahdavi J., Mathis M., *Framework for IP performance metrics*. RFC 2330, Internet Engineering Task Force, May 1998.

[RFC2481] Ramakrishnan K., Floyd S., *A proposal to add explicit congestion notification (ECN) to IP*. RFC 2481, Internet Engineering Task Force, June 1999.

[RFC2598] Jacobson V., Nichols K., Poduri K., *An expedited forwarding PHB*. RFC 2598, Internet Engineering Task Force, June 1999.

[RFC2597] Heinanen J., Baker F., Weiss W., Wroclawski J., *Assured forwarding PHB group*. RFC 2597, Internet Engineering Task Force, June 1999.

[RFC1305] Mills L. D., *Network Time Propocol*. RFC 1305, Internet Engineering Task Force, Mars 1998.

[ROB98] Roberts J., Massoulié L., *Bandwidth Sharing and Admission Control for Elastic Traffic*. In Proceedings ITC Specialist Seminar. Yokohama, Japan 1998.

[ROB00] Roberts J. W., *Self-Similar Network Traffic and Performance Evaluation*. Chapter Engineering for Quality of Service, pages 401-420. Wiley-Interscience, 2000.

[ROB01] Roberts P., *Réseaux et files d'attente : méthodes probabilistes*. Springer Verlag, 2001.

[RIN00] Rincon D., Ferrer G., Hernandez X., *Self Similar Traffic in a Commercial Video on-Demand System*. Eunice summer school, Spain, Septembre 2000.

[SED01] Seddigh N., Devetsikiotis M., *Studies of TCP's Retransmission Timeout Mechanism*. In Proceedings ICC 2001, Helsinki, Finland, June 2001.

[SHE95] Shenker S., *Fundamental design issues for the future on Internet*. IEEE JSAC, 13(7), Septembre 1995.

[STE95] Steinmetz R., Nahrstedt K., *Multimedia : computing, communications and applications*. Prentice Hall, 1995.

[STE97] Stevens W., *TCP Slow-Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms*. RFC 2001, Jan 1997.

[STO98] Stoica I., Shenker S., Zhang H., *Corestateless fair queuing : Achieving approximately fair bandwidthallocations in high speed networks*. In ACM Computer Communication Review, volume 28, pages 118-130. ACM press, Septembre 1998.

[TAS02] Tasker R., Primet P., Bonnassieux F., Mealor P., *Network monitoring architecture*. Technical report, EU DATAGRID report Deliverable D7.2 - Approved by the EC Janvier 2002.

[TEI01] Teitelbaum B., *Future priorities for internet2 QoS*. Technical report, Internet2/Qbone, 2001.

[THO97] Thompson K., Miller G. J., Wilder R., *Wide-Area Internet Traffic Patterns and Characteristics*. In IEEE Network, Décembre 1997.

[WU98] Wu C., Irwin D., *Emerging Multimedia Computer Communication Technologies*. Prentice Hall, 1998.

[YAN01] Yang S., De Veciana G., *Bandwidth Sharing : The Role of User Impatience*. Proceedings of IEEE Globecom, 2001.

[YEO95] Yeong W., Howes T., Kille S., *Lightweight directory access protocol*. Technical Report 1777, March 1995.

[ZHA91] Zhang L., Shenker S., Clark D.D., *Observations on the Dynamics of a Congestion Control Algorithm : The Effects of Two-Way Traffic*. ACM SIGCOMM, Sep 1991.

[ZHA95] Zhang H., *Service disciplines for guaranteed performance service in packet-switching networks*. Proceedings of the IEEE, 83(10) :1374-1396, October 1995.

[ZHA00] Zhang Y., Qiu L., Keshav S., *Speeding Up Short Data Transfers : Theory, Architecture Support and Simulation Results*. In Proceedings NOSSDAV 2000, Chapel Hill, NC, June 2000.

Publications

Khanafer R., Kofman D., *Contrôle d'admission dans l'Internet basé sur la classification des flots.* CFIP, Paris, Octobre 2003.

Khanafer R., Kofman D., *Internet admission control based on short and long flows classification.* HET-NETS, Ilkley- Angleterre, Juillet 2004.

Khanafer R., *Avantages de classifier les flux TCP longs selon le RTT.* DNAC, Paris, Novembre 2004.