Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mouloud Mammeri, Tizi-Ouzou

Faculté du Génie Electrique et Informatique

Département d'Automatique

MEMOIRE DE MAGISTER

Spécialité : Automatique

Option: Automatique des Systèmes Continus et Productique

Présenté par :

M. SAIDOUN Yacine

Thème :

Commande des réseaux TCP par l'approche des systèmes à retard.

Soutenu le 24/10/2012, devant le jury d'examen composé de :

M. MAIDI Ahmed	**M.C, Classe A à l'UMMTO**	**Président**
M. DJENNOUNE Saïd	**Professeur à l'UMMTO**	**Rapporteur**
M. ZIANI Areski	**M.C, Classe A à l'UMMTO**	**Examinateur**
M. LAHDIR Mourad	**M.C, Classe A à l'UMMTO**	**Examinateur**
M. SI-AMMOUR Amar	**M.C, Classe B à l'UMMTO**	**Examinateur**

Remerciements

Le travail présenté dans ce mémoire a été effectué au sein du Laboratoire de Conception et Conduite des Systèmes de Production (L2CSP) de l'Université Mouloud Mammeri de Tizi-Ouzou (UMMTO).

Je tiens tout d'abord à exprimer mes sincères remerciements à mon promoteur M. DJENNOUNE Saïd, Professeur à l'UMMTO pour la confiance qu'il m'a accordée, sa disponibilité, ses conseils et son aide assez précieuse, tout au long de ma thèse.

Je remercie M. MAIDI Ahmed, Maitre de conférences classe A à l'UMMTO, pour avoir accepté d'examiner ce travail et de présider le jury de soutenance de ce mémoire.

Je remercie très vivement M. ZIANI Areski, Maître de conférences Classe A à l'UMMTO, M. LAHDIR Mourad, Maître de conférences classe A à l'UMMTO, et M. SI-AMMOUR Amar, Maître de conférences classe B à l'UMMTO qui ont accepté de rapporter sur mon mémoire et m'ont fait l'honneur d'être membres du jury de soutenance.

Mes remerciements s'adressent également à tous les membres du Laboratoire de Conception et Conduite des Systèmes de Production (L2CSP) et tous les collègues pour l'excellente ambiance de travail qu'ils ont créé.

Enfin, je remercie toute ma famille, et tous mes amis, qui de prés ou de loin m'ont supporté, soutenu et encouragé tout au long de ces années.

Table des matières

Table des figures

Notations

$\mathbb{R}$	L'ensemble des nombres réels.
$\mathbb{R}_+$	L'ensemble des nombres réels positifs ou nuls.
$\mathbb{R}^n$	L'espace euclidien de dimension n.
$\mathbb{R}^{nxm}$	L'espace des matrices réelles de dimension n × m.
t	La variable temps.
τ	Le retard temporel constant.
$\tau(t)$	Le retard temporel variable.
$[a, b]$	un intervalle fermé de $\mathbb{R}$, d'extrémités a et b..
$C = C([-\tau, 0];\ \mathbb{R}^n)$	L'ensemble des fonctions continues de $[-\tau, 0]$ dans $\mathbb{R}^n$.
$x_t \in C$	La fonction segment à l'instant t, $x_t(\theta) = x(t+\theta), \forall\ t\ \in [-\tau,\ 0]$
$x \in \mathbb{R}^n$	Le vecteur d'état instantané.
$\dot{x}(t) = \frac{dx}{dt}$	La dérivée temporelle de l'état x.
$V(.)$	La fonction de Lyapunov.
$\dot{V}(.)$	La dérivée de la fonction de Lyapunov V (.) par rapport au temps.
x^T	Le vecteur transposé du vecteur x.
$\|x\|$	La norme Euclidienne de x.
A^T	La matrice transposée de la matrice A.
$A > 0$	La matrice A est définie positive.
$A < 0$	La matrice A est définie négative.

Acronymes

AIMD	Additive Increase Multiplicative Decrease
AQM	Active Queue Management
IP	Internet Protocol
LMI	Linear Matrix Inequality / Inégalité Matricielle Linéaire
NCS	Networked Control System
QoS	Quality of Service / Qualité de Service
RED	Random Early Detection

RTT	Round Trip Time / Temps d'un aller-retour
SCR	Systèmes Commandés en Réseau
TCP	Transport Control Protocol / Protocole de contrôle de transfert
UDP	User Datagram Protocol

Introduction générale

Les réseaux informatiques de communication (Internet, Ethernet, Intranet,…) constituent un passage obligé dans les traitements de l'information et la commande à distance des systèmes. Si les réseaux de communication constituent une avancée technologique importante dans la célérité de traitement d'information en temps réel, ils restent néanmoins source de nombreux problèmes dans beaucoup d'applications : problèmes de retard, problèmes de congestion et problèmes de pertes d'informations pour ne citer que les plus importants.

En automatique, de nombreux travaux ont été consacré récemment à apporter quelques solutions à ces problèmes. Cependant même si on retrouve des similitudes entre les problèmes posés, il est à souligner que deux orientations complètement différentes dans les travaux menés à ce jour, doivent être mentionnées :

- Dans la première orientation, on s'intéresse à la commande des systèmes via les réseaux informatiques (**SCR** : Systèmes Commandés en Réseaux ou bien **NCS** : Networked Controlled Systems). Là, il s'agit essentiellement d'essayer de trouver des commandes afin de palier aux problèmes de retard et de pertes d'informations induite par le réseau, causant inévitablement des instabilités.

- Dans la seconde orientation, le réseau de communication est considéré lui-même comme le système à contrôler. Il s'agit d'abord de modéliser le réseau (système dynamique de propagation) puis de le commander afin d'éviter les congestions et les retards.

C'est dans cette deuxième voie que s'inscrit ce sujet de magister qui est une thématique pluridisciplinaire explorant les liens existants entre la théorie de la commande et une classe de réseaux informatiques. Comme le suggère le titre de ce mémoire, l'enjeu de ce travail est double. Il consiste d'une part à faire une étude assez générale sur les réseaux informatiques, le protocole TCP (*Transmission Control Protocol*), le processus de communication et le problème de congestion au niveau des routeurs. D'autre part, il propose d'employer les outils théoriques ainsi développés, dans le cadre de l'automatique, pour le

contrôle de congestion d'un routeur lors des communications TCP (*Transmission Control Protocol*). Notre thématique se compose donc à la fois d'un aspect fondamental, constitué par la recherche de conditions générales de stabilité pour la classe de systèmes considérés et d'un aspect applicatif lié au monde des réseaux de communication.

La communication entre plusieurs ordinateurs nécessite au préalable la mise en place d'un "langage" commun, c'est-à-dire un protocole de communication. Le protocole TCP, inventé en 1981 [4], s'est imposé comme le standard pour la communication dans l'Internet. Il est chargé de gérer la quantité de données à émettre sur le réseau lors d'un transfert d'informations d'un ordinateur source vers son destinataire. C'est un protocole de communication dit de *bout en bout* puisque, du fait de son niveau d'abstraction, il établit une connexion directe entre l'émetteur et le récepteur, sans se préoccuper des dispositifs (logiciels ou matériels) mis en œuvre pour traverser le réseau. La communication effective est, quant à elle, établie par les protocoles de niveau inférieur (par exemple : IP, Ethernet) offrant ainsi une liaison directe entre les machines communicantes. Sa propriété principale est de garantir une communication fiable. En appliquant un mécanisme d'acquittement, il assure que l'intégralité des données soit transmise au destinataire. Ainsi, s'il y a une perte, la source renverra le paquet manquant. Cependant, étant donné la concurrence des différents utilisateurs pour l'accès au réseau, la question est : *à quel débit un émetteur peut-il envoyer ses données ?* Il s'agit de maximiser, équitablement, les débits des sources tout en évitant la saturation de la capacité du réseau. Cette problématique constitue le point de départ du *contrôle de congestion* et plus généralement de la *Qualité de Service* (QdS).

Le phénomène de congestion se manifeste lorsqu'un dispositif reçoit plus d'information qu'il ne peut en traiter, c'est-à-dire, dès lors que le débit de données entrant est supérieur au débit sortant. Il est clair que dans le cas d'un routeur agrégeant une certaine quantité de flux de données, l'ensemble des émetteurs sont en compétition pour l'accès aux ressources. Si l'intensité du trafic dépasse la capacité du routeur, ce dernier est en état de congestion et les données reçues sont mises en attente avant de pourvoir être traitées. Plus précisément, le buffer de réception du routeur se remplit, puis, une fois saturé tous les nouveaux paquets arrivant sont éjectés, donc perdus. La congestion entrainant la perte de paquets, TCP doit, pour maintenir un service fiable, retransmettre l'ensemble des données perdues. Mais si la surcharge du réseau n'a pas été résolue, les paquets réémis seront certainement de nouveau perdus. De ce fait, le protocole a été doté d'un algorithme

supplémentaire pour l'*évitement de congestion*. A partir de la "vision" bout en bout de TCP, ce mécanisme se base sur le postulat que la perte d'un paquet est synonyme de congestion à l'intérieur du réseau. Par conséquent, il ajuste son taux d'émission en fonction de la réception ou non-réception d'un acquittement. Cependant, de nombreuses études ont montré que cet algorithme empirique entraine de fortes oscillations du trafic et l'inégalité parmi les utilisateurs.

Au cours de ce mémoire, nous nous sommes intéressés au contrôle de congestion d'un routeur dans les réseaux IP. Plus précisément, notre étude s'est focalisée sur le partage d'un lien de communication entre plusieurs émetteurs situés sur des sites distants. Le lien en aval du routeur congestionné étant emprunté par N flux, chaque source applique le mécanisme d'évitement de congestion pour remédier au problème de surcharge du réseau. Des travaux de modélisation fluide proposant une représentation mathématique du comportement du protocole TCP ont été développés dans le but d'effectuer une analyse quantitative du problème de congestion.

Ainsi, sous certaines hypothèses, l'évolution temporelle du taux d'émission d'une source est régie par une équation différentielle *retardée,* dépendant du taux de perte au niveau du routeur.

Par conséquent, l'intensité des flux étant sensible à la perte de paquets, des dispositifs d'*Active Queue Management* (AQM), capable d'éjecter prématurément des paquets de la file d'attente ont été proposés afin d'anticiper la saturation du buffer et de réduire les fortes oscillations de la file. Dès lors, des chercheurs issus de la communauté de l'automatique se sont intéressés à ce sujet et au problème de commande sous-jacent. C'est dans ce contexte applicatif que nous avons développé des outils d'analyse propres aux systèmes à retards, offrant alors un cadre de travail approprié pour l'étude de stabilité du système de communication considéré. Toutefois, l'analyse de stabilité d'une telle classe de systèmes reste un problème de recherche ouvert. Nous nous sommes appliqués, dans ce mémoire, à utiliser l'approche de Lyapunov-Krasovskii et les inégalités matricielles linéaires (LMIs) pour la synthèse d'un régulateur par retour d'état pour la régulation de la file d'attente au niveau du routeur.

Le premier chapitre de ce mémoire présente une lecture générale sur les réseaux informatiques et internet en général afin de mieux comprendre le processus de communication et les divers concepts utiles pour la suite de ce mémoire.

Le deuxième chapitre est consacré à la présentation de notre application, à savoir le fonctionnement du protocole TCP et son mécanisme d'évitement de congestion lors des communications. Nous détaillerons le fonctionnement du protocole TCP et analyserons les conséquences de son mécanisme d'évitement de congestion lors des communications. A partir de là, nous serons en mesure de mettre en évidence la problématique du phénomène de saturation d'un routeur. Nous verrons alors comment il est possible de réguler le trafic TCP à l'aide du dispositif d'AQM, localisé au niveau du routeur, afin d'améliorer le contrôle de congestion. Nous présenterons aussi dans ce chapitre, la modélisation mathématique du comportement de TCP.

Dans le troisième chapitre, nous présenterons les systèmes à retard en général et les systèmes linéaires à retard en particulier ainsi que le critère de Lyapunov-Krasovskii pour la stabilisation d'une telle classe de systèmes que nous utiliserons par la suite pour le contrôle de congestion d'un routeur.

Le quatrième chapitre sera consacré à la partie pratique de ce mémoire, c'est-à-dire, à la synthèse d'un régulateur issu de l'approche des systèmes à retard pour la régulation de la file d'attente en niveau d'un routeur et ainsi éviter la congestion du réseau. Afin d'illustrer clairement les différents points abordés dans ce travail, des simulations accompagneront toute l'étude théorique.

Finalement, nous résumerons les différents points abordés tout au long de ce mémoire et tenterons de dégager quelques pistes ouvertes pour des travaux futurs.

Chapitre I

Lecture sur les réseaux

Nous proposons dans ce chapitre de présenter une introduction et quelques généralités sur l'architecture des réseaux de communication. Nous présenterons divers définitions et concepts propres au domaine des réseaux informatiques et internet, mais notons bien qu'il ne s'agit pas de détailler avec précision le monde des réseaux de communication mais de donner les notions et concepts nécessaires pour comprendre la *problématique* abordée dans ce mémoire. La documentation sur les réseaux et abondante, et le lecteur peut se référer aux ouvrages [1], [2], [3] pour plus de détails et de précisions.

I.1 Introduction

Parmi les éléments essentiels à l'existence humaine, le besoin de communiquer arrive juste après le besoin de survie. Le besoin de communiquer est aussi important pour nous que l'air, l'eau et la nourriture.

Les méthodes dont nous nous servons pour partager idées et informations changent et évoluent sans cesse. Si le réseau humain se limitait autrefois à des conversations en face à face, aujourd'hui les découvertes en matière de supports étendent sans cesse la portée de nos communications. De la presse écrite à la télévision, chaque innovation a développé et amélioré nos moyens de communication.

À l'image de tous les progrès dans le domaine des technologies de la communication, la création et l'interconnexion de réseaux de données solides ont un profond impact. Si les premiers réseaux de données se limitaient à échanger des informations reposant sur des caractères entre des systèmes informatiques connectés, les réseaux modernes ont évolué pour prendre en charge le transfert audio, des flux vidéo, du texte et des graphismes entre des périphériques de types très différents. Des moyens de communication autrefois séparés et bien distincts convergent maintenant sur une plateforme commune. Cette plateforme offre une

large gamme de méthodes de communication aussi nouvelles que différentes qui permettent aux individus d'interagir directement, et presque instantanément.

Avant de commencer à communiquer, nous établissons des règles, ou conventions, qui régissent la conversation. Ces règles ou protocoles doivent être respectés pour que le message soit correctement transmis et compris. Parmi les protocoles qui régissent nos communications pour qu'elles se déroulent correctement, citons :

-L'identification de l'expéditeur et du destinataire.

-Le recours à une méthode de communication convenue (face-à-face, téléphone, lettre, photographie).

-L'utilisation d'une langue et d'une syntaxe communes.

-La vitesse et le rythme d'élocution.

-La demande de confirmation ou de reçu.

Les règles régissant la communication peuvent varier en fonction du contexte. Si un message mentionne un fait ou un concept important, il est nécessaire de confirmer que le message a été reçu et compris. Les messages moins importants n'exigent pas toujours de reçu de la part du destinataire.

Les techniques utilisées dans le cadre des communications réseaux partagent ces mêmes exigences fondamentales avec les conversations directes entre personnes. Étant donné qu'un grand nombre des protocoles s'appliquant aux communications humaines sont implicites ou intégrés à notre culture, certaines règles n'ont pas besoin d'être précisées. Mais lorsque nous établissons des réseaux de données, nous devons être beaucoup plus explicites sur la façon dont la communication s'effectuera et sur ce qui en assurera le succès.

I.2 Généralités sur les réseaux

I.2.1 Eléments d'un réseau

Si au départ un réseau se résumait à un ensemble de terminaux connectés à un unique calculateur central, il désigne maintenant un ensemble d'ordinateurs autonomes capables d'échanger entre eux des informations. Un réseau informatique se définit comme l'ensemble

des moyens matériels et logiciels mis en œuvre pour assurer la communication entre plusieurs terminaux.

Le diagramme ci-dessous montre les éléments constituant le plus souvent un réseau, à savoir des périphériques, des supports et des services reliés par des règles et qui collaborent pour envoyer des messages. Le terme messages nous sert à désigner les pages Web, les courriels, les messages instantanés, les appels téléphoniques et les autres formes de communication prises en charge par le réseau. Pour qu'un réseau soit opérationnel, il faut que les périphériques le composant soient interconnectés. Les connexions réseau peuvent être câblées ou sans fil.

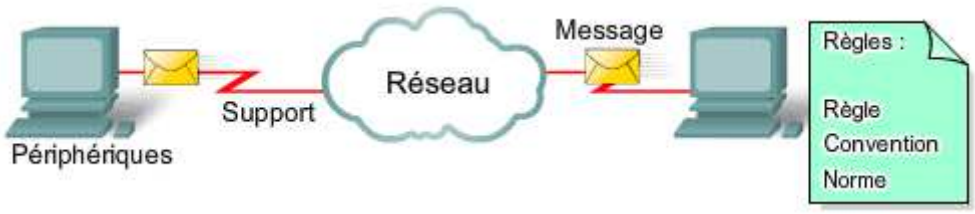

Fig. I.1 : Éléments d'un réseau.

Un périphérique est un composant matériel qui fait parti du réseau. Il peut être final ou terminal, c'est-à-dire un récepteur ou un émetteur comme c'est le cas d'un ordinateur, d'un smart phone, d'un PDA, d'une d'imprimante. Il peut être aussi intermédiaire comme le cas d'un routeur ou d'un Switch.

Un support est le moyen sur lequel les données sont transférées. Par exemple, une paire torsadée blindée représente un type de support de réseau.

Un message est l'information à transmettre à travers le réseau. Lors de la première étape de son voyage, le message est converti en un format pouvant être transmis sur le réseau. Tous les types de messages doivent être convertis en bits, c'est-à-dire en signaux numériques codés en binaire, avant d'être envoyés vers leurs destinations. Ceci est obligatoire quel que soit le format d'origine du message : texte, vidéo, audio ou données informatiques. Une fois le message converti en bits, il est prêt à être envoyé sur le réseau, jusqu'à son destinataire.

Les règles ou protocoles constituent un autre aspect important des réseaux. Ces règles sont les normes ou protocoles qui définissent la façon dont les messages sont envoyés, orientés sur le réseau puis interprétés par les périphériques de destination. Ainsi, dans le cas de la messagerie instantanée par exemple, les protocoles XMPP, TCP et IP sont tous des ensembles de règles jouant un rôle important dans l'établissement de la communication.

I.2.2 Réseaux convergents

Les réseaux classiques de transfert de données téléphoniques, de radio, de télévision ou informatiques intègrent tous leur propre version des quatre éléments de base constituant les réseaux. Autrefois, chacun de ces services nécessitait une technologie différente pour acheminer son signal de communication particulier. De plus, chaque service utilisait son propre ensemble de règles et de normes pour assurer le succès du transfert de son signal sur un support précis.

Les progrès technologiques nous permettent aujourd'hui de réunir ces réseaux disparates sur une même plateforme, une plateforme définie comme étant un réseau convergent. Le fait que les flux vocaux, vidéo et de données empruntent le même réseau rend inutile la création et la maintenance de réseaux séparés. Si de nombreux points de contact et périphériques spécialisés (par exemple des ordinateurs personnels, téléphones, télévisions, assistants personnels et lecteurs sur le point de vente) continuent à cohabiter sur un réseau convergent, l'infrastructure réseau, quant à elle, est unique et commune.

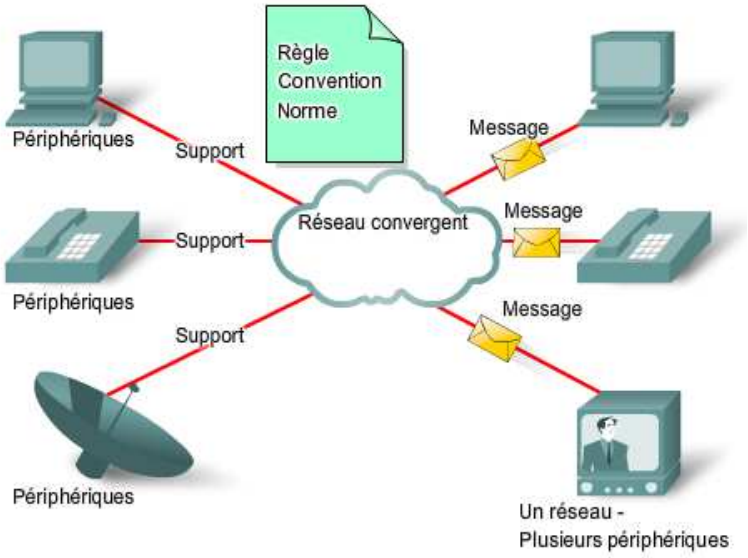

Fig. I.2 : Structure d'un réseau convergent.

I.3 Architecture réseau

Les réseaux doivent d'une part prendre en charge une large gamme d'applications et de services et d'autre part fonctionner sur de nombreux types d'infrastructures physiques. Dans le contexte actuel, l'expression « architecture réseau » désigne aussi bien les technologies prenant en charge l'infrastructure que les services programmés et les protocoles qui déplacent les messages dans l'infrastructure. Alors qu'Internet, et les réseaux en général, évoluent, nous découvrons que les architectures sous-jacentes doivent prendre en considération quatre caractéristiques de base si elles veulent répondre aux attentes des utilisateurs : tolérance aux pannes, évolutivité, qualité de service et sécurité.

I.3.1 Tolérance aux pannes

Comme des millions d'utilisateurs attendent d'Internet qu'il soit constamment disponible, il faut une architecture réseau conçue et élaborée pour tolérer les pannes. Un réseau tolérant aux pannes est un réseau qui limite l'impact des pannes du matériel et des logiciels et qui peut être rétabli rapidement quand des pannes se produisent. De tels réseaux dépendent de liaisons, ou chemins, redondantes entre la source et la destination d'un message. En cas de défaillance d'une liaison (ou chemin), les processus s'assurent que les messages sont instantanément routés sur une autre liaison et ceci de manière totalement transparente

pour les utilisateurs aux deux extrémités. Aussi bien les infrastructures physiques que les processus logiques qui dirigent les messages sur le réseau sont conçus pour prendre en charge cette redondance. Il s'agit d'une caractéristique essentielle des réseaux actuels.

À ses débuts, Internet était le produit de recherches financées par le Département de la défense des États-Unis (DoD) dont l'objectif premier était de disposer d'un support de communication capable de résister à la destruction de nombreux sites et établissements de transmissions sans interruption de service. Ceci explique que les travaux de conception initiaux de cet interréseau se soient concentrés sur la tolérance aux pannes. Les premiers chercheurs ont examiné les réseaux de communication existant alors, lesquels étaient principalement destinés à la transmission du trafic sonore, afin de déterminer ce qui pourrait être fait pour améliorer leur degré de tolérance aux pannes.

Réseaux à commutation de circuits orientés connexions

Pour comprendre les défis auxquels les chercheurs du DoD ont été confrontés, il faut rappeler le mode de fonctionnement des premiers systèmes téléphoniques. Lorsqu'une personne utilise un téléphone conventionnel, l'appel commence par un processus de configuration au cours duquel tous les emplacements de commutation téléphonique existant entre la personne qui appelle et le poste téléphonique appelé sont identifiés. Un chemin, ou circuit, temporaire est créé à travers les divers commutateurs à utiliser pendant la durée de l'appel téléphonique. Si une liaison ou un périphérique quelconque du circuit concerné connaît un dysfonctionnement, l'appel est abandonné. Pour recréer une connexion, il faut passer un nouvel appel et créer un nouveau circuit entre le poste téléphonique source et celui de destination. Ce type de réseau orienté connexions est appelé réseau à commutation de circuit. Les premiers réseaux à commutation de circuit ne recréaient pas dynamiquement les circuits rompus. Pour assurer la reprise après une défaillance, il fallait initier de nouveaux appels et élaborer de nouveaux circuits de bout en bout.

De nombreux réseaux à commutation de circuit donnent la priorité au maintien des connexions sur les circuits existants aux dépens des requêtes de nouveaux circuits. Dans ce genre de réseaux orientés connexions, une fois qu'un circuit a été établi, il demeure connecté même si aucune communication n'a lieu entre les personnes à chaque extrémité de l'appel, et les ressources sont réservées jusqu'à ce que l'une des parties mette fin à l'appel. Étant donné

que la capacité à créer de nouveaux circuits n'est pas illimitée, il est parfois possible de recevoir un message indiquant que tous les circuits sont occupés et que l'appel ne peut être établi. Devant le coût de la création de nombreux chemins de remplacement disposant d'une capacité suffisante pour prendre en charge un grand nombre de circuits simultanés, et des technologies nécessaires pour recréer dynamiquement les circuits abandonnés en cas de défaillance, le DoD a été conduit à s'intéresser à d'autres types de circuits.

Réseaux à commutation de paquets sans connexion

Dans leur quête d'un réseau capable de supporter la perte d'un nombre important de points de transmission et de commutation, les premiers concepteurs d'Internet ont reconsidéré les recherches préalables sur les réseaux à commutation de paquets. L'idée de base pour ce type de réseaux est qu'un message peut être décomposé en de multiples blocs de message. Les blocs individuels contenant des informations d'adressage indiquent le point d'origine ainsi que la destination finale. Grâce à ces informations intégrées, les blocs de message, appelés paquets, peuvent être envoyés sur le réseau en empruntant des chemins variés avant d'être réassemblés pour recomposer le message d'origine une fois parvenus à destination.

Utilisation des paquets

Au sein du réseau même, les périphériques n'ont pas accès au contenu des paquets individuels. Seuls l'adresse de la destination finale et le prochain périphérique sur le chemin vers cette destination leur sont indiqués. Aucun circuit réservé n'est établi entre l'expéditeur et le destinataire. Chaque paquet est envoyé d'un emplacement de commutation à un autre de façon indépendante. À chaque emplacement, une décision de routage est prise pour déterminer le chemin qui sera emprunté pour transmettre le paquet vers sa destination finale. Si un chemin précédemment utilisé n'est plus disponible, la fonction de routage peut choisir dynamiquement le meilleur chemin suivant disponible. Comme les messages sont fragmentés au lieu d'être envoyés sous forme de message unique complet, il est possible de retransmettre sur un chemin différent les quelques paquets qui pourraient s'être perdus en cas de défaillance. Dans bien des cas, le périphérique de destination ignore les défaillances ou modifications de routages qui sont intervenues.

Les chercheurs du DoD ont compris qu'un réseau à commutation de paquets sans connexion disposait des capacités requises pour prendre en charge une architecture réseau résiliente et tolérante aux pannes. Dans ce type de réseau, il est inutile de réserver un circuit unique de bout en bout. Chaque morceau du message peut être envoyé sur le réseau par l'intermédiaire de n'importe quel chemin disponible. Des paquets contenant des morceaux de messages provenant de sources différentes peuvent emprunter simultanément le même réseau. Ceci résout le problème des circuits sous-utilisés ou actifs car toutes les ressources disponibles peuvent être utilisées simultanément pour livrer des paquets à leur destination finale. Parce qu'il permet d'utiliser dynamiquement les chemins redondants sans intervention de l'utilisateur, Internet est devenu un moyen de communication tolérant aux pannes et extensible.

Réseaux orientés connexions

Bien que les réseaux à commutation de paquets sans connexion répondent aux besoins du DoD et continuent à constituer l'infrastructure de base d'Internet aujourd'hui, un système orienté connexion comme le système téléphonique à commutation de circuit présente quelques avantages. Étant donné que les ressources des divers emplacements de commutation ont pour vocation de fournir un nombre précis de circuits, la qualité et la cohérence des messages transmis sur un réseau orienté connexion peuvent être garanties. En outre, le fournisseur du service peut facturer la période de temps pendant laquelle la connexion est active aux utilisateurs du réseau, ce qui est un autre avantage. Pouvoir facturer aux utilisateurs les connexions actives sur le réseau est un élément essentiel de l'industrie des services de télécommunication.

I.3.2 Évolutivité

Un réseau extensible est en mesure de s'étendre rapidement afin de prendre en charge de nouveaux utilisateurs et applications sans que ceci n'affecte les performances du service fourni aux utilisateurs existants. Chaque semaine, des milliers de nouveaux utilisateurs et fournisseurs de services se connectent à Internet. La capacité du réseau à prendre en charge ces nouvelles interconnexions dépend de l'existence d'un modèle hiérarchisé à plusieurs couches appliqué à l'infrastructure physique et à l'architecture logique. Il est possible

d'insérer des utilisateurs ou des fournisseurs de service au niveau de chaque couche sans perturber l'ensemble du réseau. Grâce aux progrès technologiques, les capacités de transport des messages et les performances des composants de l'infrastructure physique augmentent au niveau de chaque couche. Ces progrès, associés aux nouvelles méthodes d'identification et de localisation de chaque utilisateur au sein d'un interréseau, permettent à Internet de continuer à répondre aux attentes des utilisateurs.

I.3.3 Qualité de service (QdS)

Actuellement, Internet offre un niveau de tolérance aux pannes et d'évolutivité correct à ses utilisateurs. Cependant, le fait que de nouvelles applications soient mises à la disposition des utilisateurs sur les interréseaux crée des attentes supplémentaires en termes de qualité des services fournis. Les transmissions audio et vidéo en direct exigent un niveau de qualité constant et un service ininterrompu qui n'était pas indispensable aux applications informatiques traditionnelles. La qualité de ces services est évaluée par rapport à la qualité que l'on obtiendrait en assistant en personne à la même présentation audio ou vidéo. Les réseaux audio et vidéo traditionnels sont conçus pour ne prendre en charge qu'un seul type de transmission. Ils peuvent donc offrir un niveau de qualité acceptable. De nouvelles exigences en matière de prise en charge de cette qualité de service sur un réseau convergent modifient cependant la façon dont les architectures réseau sont conçues et implémentées.

I.3.4 Sécurité

Autrefois simple interréseau d'organisations éducatives et gouvernementales strictement contrôlées, Internet a évolué pour devenir un moyen de transmission de communications professionnelles et personnelles largement accessible. Les exigences du réseau en matière de sécurité ont donc évidemment changé. Les exigences de sécurité et de confidentialité résultant de l'utilisation d'interréseaux pour échanger des informations confidentielles et commerciales d'importance critique excèdent ce que l'architecture actuelle peut offrir. L'expansion rapide de secteurs des communications qui n'étaient précédemment pas desservis par des réseaux de données traditionnels renforce le besoin d'intégrer la sécurité à l'architecture du réseau. C'est pourquoi des efforts considérables sont consacrés à ce secteur de recherche et de développement. En attendant, de nombreux outils et procédures sont implémentés pour combattre les failles de sécurité inhérentes à l'architecture réseau.

I.4 Communications sur un réseau

Au niveau humain, certaines règles de communication sont formelles et d'autres sont simplement comprises ou implicites, en fonction de la coutume et de la pratique. Afin que des périphériques puissent communiquer correctement, une suite de protocoles réseau doit décrire des exigences et des interactions précises.

Les suites de protocoles réseau décrivent des processus tels que :

-le format ou la structure du message.

-la méthode selon laquelle des périphériques réseau partagent des informations sur des chemins avec d'autres réseaux.

-comment et à quel moment des messages d'erreur et système sont transférés entre des périphériques.

-la configuration et l'arrêt des sessions de transfert de données.

Des protocoles individuels dans une suite de protocoles peuvent être spécifiques au fournisseur et propriétaire. Propriétaire, dans ce contexte, signifie qu'une société ou qu'un fournisseur contrôle la définition du protocole et la manière dont il fonctionne. Certains protocoles propriétaires peuvent être utilisés par différentes organisations avec l'autorisation du propriétaire. D'autres peuvent uniquement être implémentés sur du matériel fabriqué par le fournisseur propriétaire.

Souvent, de nombreux protocoles qui comprennent une suite de protocoles font référence à d'autres protocoles largement utilisés ou normes de l'industrie. Une norme est un processus ou un protocole reconnu par l'industrie du réseau et ratifié par une organisation de normes, telle que l'Institute of Electrical and Electronics Engineers (IEEE) ou le groupe de travail Internet Engineering Task Force (IETF).

L'utilisation de normes dans le développement et l'implémentation de protocoles garantit que les produits provenant de différents fabricants peuvent fonctionner ensemble pour créer des communications efficaces. Si un fabricant spécifique n'adhère pas strictement à un protocole, son équipement ou ses logiciels risquent de ne pas communiquer correctement avec les produits d'autres fabricants.

I.4.1 Utilisation de modèles en couche

Pour visualiser l'interaction entre différents protocoles, un modèle en couches est généralement utilisé. Un modèle en couches décrit le fonctionnement des protocoles au sein de chacune des couches, ainsi que l'interaction avec les couches supérieures et inférieures. Mais il faut bien noter qu'un modèle de réseau est uniquement une représentation du fonctionnement du réseau et il n'est pas le réseau réel.

Il existe deux types de modèles de réseau de base : le modèle OSI et le modèle TCP/IP. La figure ci-dessous montre une représentation des deux modèles.

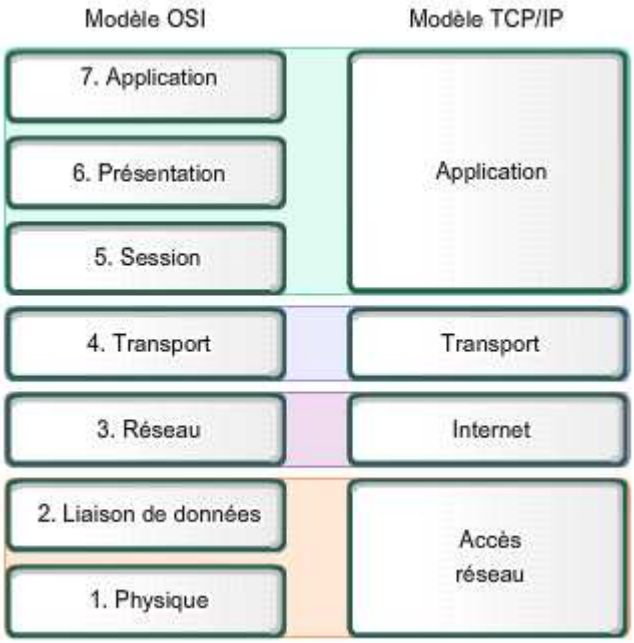

Fig. I.3 : Comparaison des modèles OSI et TCP/IP.

Les protocoles qui constituent la suite de protocoles TCP/IP peuvent être décrits selon les termes du modèle de référence OSI. Dans le modèle OSI, la couche d'accès réseau et la couche application du modèle TCP/IP sont encore divisées pour décrire des fonctions discrètes qui doivent intervenir au niveau de ces couches.

La couche application est la couche supérieure des modèles OSI et TCP/IP. Elle est la couche qui sert d'interface entre les applications que nous utilisons pour communiquer et le réseau sous-jacent via lequel nos messages sont transmis. Les protocoles de couche application sont utilisés pour échanger des données entre les programmes s'exécutant sur les hôtes source et de

destination. Il existe de nombreux protocoles de couche application et de nouveaux protocoles sont constamment en cours de développement. Les protocoles de la couche application les plus utilisés sont DNS (Domain Name Service) utilisé pour traduire les addresses internet en adresses ip, HTTP (Hypertext Transfer Protocol) utilisé dans le transfert des pages web, SMTP (Simple Mail Transfer Protocol) utilisé dans la messagerie électroinque, Telnet utilisé dans l'accés à distance aux serveurs et aux perifiques réseaux, FTP (File Transfer Protocole) utilisé dans le transfert de fichiers.

La couche transport segmente les données et se charge du contrôle nécessaire au réassemblage de ces blocs de données dans les divers flux de communication. Pour ce faire, il doit :
-effectuer un suivi des communications individuelles entre les applications résidant sur les hôtes source et de destination.
-segmenter les données et gérer chaque bloc individuel.
-réassembler les segments en flux de données d'application.
-identifier les différentes applications.

Les deux protocoles de la suite de protocoles TCP/IP les plus couramment employés sont le protocole TCP (Transmission Control Protocol) et le protocole UDP (User Datagram Protocol). Ces deux protocoles gèrent les communications de nombreuses applications. Ce sont les fonctions spécifiques implémentées par chaque protocole qui les différencient.

Le protocole UDP est un protocole simple, sans connexion, décrit par le document RFC 768 [4]. Il présente l'avantage d'imposer peu de surcharge pour l'acheminement des données. Les blocs de communications utilisés dans le protocole UDP sont appelés des datagrammes. Ces datagrammes sont envoyés « au mieux » par ce protocole de couche transport.
Le protocole UDP est notamment utilisé par des applications de :
-Système de noms de domaine (DNS)
-Lecture vidéo en continu
-Voix sur IP (VoIP)

Le protocole TCP est un protocole avec connexion décrit dans le document RFC 793 [4]. Le protocole TCP impose une surcharge pour accroître les fonctionnalités. Le protocole TCP spécifie d'autres fonctions, à savoir la livraison dans l'ordre, l'acheminement fiable et le

contrôle du flux. Chaque segment du protocole TCP utilise 20 octets de surcharge dans l'en-tête pour encapsuler les données de la couche application alors que chaque segment du protocole UDP n'ajoute sur 8 octets de surcharge.
Le protocole TCP est utilisé par des applications de :
-Navigateurs Web
-Courriel
-Transfert de fichiers

La couche Réseau ou Internet est responsable de la récupération des segments formatés à partir de la couche transport, de leur encapsulation en paquets, de l'affectation des adresses appropriées et de la sélection du meilleur chemin vers l'hôte de destination. Le protocole de la couche réseau utilisé est le protocole ip (internet protocol).

La couche accés réseau décrit deux fonctions principales : la gestion des liaisons de données et la transmission physique des données sur les supports. Les protocoles de gestion de liaison de données prennent les paquets depuis le protocole IP et les formatent pour les transmettre à travers les supports. Les normes et les protocoles des supports physiques régissent la manière dont les signaux sont envoyés à travers les supports, ainsi que leur interprétation par les clients destinataires. Des émetteurs-récepteurs sur les cartes réseau implémentent les normes appropriées pour les supports en cours d'utilisation. Le protocole Ethernet est un exemple de protocole accès réseau.

I.4.2 Processus de communication

Le modèle TCP/IP décrit la fonctionnalité des protocoles qui constituent la suite de protocoles TCP/IP. Ces protocoles, qui sont implémentés sur les hôtes émetteurs et récepteurs, interagissent pour fournir une livraison de bout en bout d'applications sur un réseau.

Un processus de communication complet comprend ces étapes :

1. Création de données sur la couche application du périphérique d'origine.
2. Segmentation et encapsulation des données lorsqu'elles descendent la pile de protocoles dans le périphérique source dans la couche transport. Chaque segment TCP reçoit une étiquette, appelée en-tête, qui contient des informations pour désigner le

processus s'exécutant sur l'ordinateur de destination qui doit recevoir le message. Il contient également les informations pour permettre au processus de destination de réassembler les données selon leur format d'origine.

3. Dans la couche réseau ou internet la totalité du segment TCP est encapsulée dans un paquet IP, qui ajoute une autre étiquette, appelée en-tête IP. L'en-tête IP contient des adresses IP d'hôtes source et de destination, ainsi que des informations nécessaires à la livraison du paquet à son processus de destination correspondant.

4. Dans la couche accès au réseau le paquet IP est encapsulé dans un en-tête de trame et une queue de bande (ou en-queue de trame). Chaque en-tête de trame contient une adresse physique source et de destination. L'adresse physique identifie de manière unique les périphériques sur le réseau local. L'en-queue contient des informations de vérification d'erreur.

5. Transport des données via l'inter réseau, qui est constitué de supports et de n'importe quels périphériques intermédiaires.

6. Réception des données au niveau de la couche d'accès au réseau du périphérique de destination.

7. Décapsulation et assemblage des données lorsqu'elles remontent la pile dans le périphérique de destination et transmission de ces données à l'application de destination, au niveau de la couche application du périphérique de destination.

La figure ci-dessous montre l'encapsulation des données et l'ajout d'en-tête à chaque couche réseau :

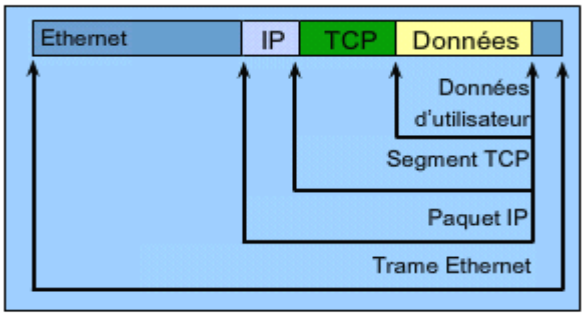

Fig. I.4 : Structure d'un segment, d'un paquet et d'une trame.

I.5 Conclusion

Nous avons consacré ce chapitre à la présentation des réseaux et Internet qui sera utile dans la suite de ce mémoire. De nombreux concepts et définitions ont été introduits et il convient alors de faire un court récapitulatif. Nous avons présenté, dans un premier temps, l'architecture des réseaux dans l'Internet et son principe de couche qui garantissent un certain niveau d'abstraction pour un protocole de communication donné. Nous avons vu comment fonctionne le processus de communication de l'émetteur au récepteur. D'abord le message est décomposé et encapsulé en segments au niveau de la couche Transport, en paquets au niveau de la couche Réseau (Internet) puis en trames au niveau de la couche accès au réseau qui seront ensuite envoyées en empruntant des chemins qui peuvent être différents et cela grâce à l'architecture réseau tolérante aux pannes qu'on a vu auparavant. Une fois arrivés au destinataire, ils seront réassemblés pour former le message original envoyé.

Chapitre II

Le Protocole TCP, modélisation et Contrôle de congestion

Nous proposons dans ce second chapitre de présenter le contexte de l'application. Après une introduction sur l'architecture des réseaux de communication, nous développons maintenant l'objet de notre étude, le protocole TCP (Transmission Control Protocol). En vue de l'utilisation des outils de l'automatique, nous introduirons une modélisation mathématique du comportement de TCP, extraite de la littérature. Enfin, nous verrons comment nous pouvons réguler le trafic de flux TCP au niveau d'un routeur grâce au dispositif d'*Active Queue Management* (AQM).

II.1 Généralités

II.1.1 Contexte

Le protocole TCP, inventé en 1981 [4], s'est imposé comme le standard pour la communication dans l'Internet. Situé au niveau de la couche transport, TCP est chargé de gérer la quantité de données à émettre sur le réseau ainsi que d'assurer un service de fiabilité aux applications de la couche supérieure. C'est un protocole de communication dit de *bout en bout* puisque, du fait de son niveau d'abstraction, il est en "discussion" directe avec le récepteur. La communication effective (à travers éventuellement plusieurs routeurs) est, quant à elle, établie par les couches inférieures offrant ainsi une liaison directe entre les échanges TCP de l'émetteur avec le récepteur. Dans l'Internet, les échanges sont basés sur la transmission à commutation de paquets par opposition aux réseaux à commutation de circuits. En d'autres termes, plutôt que de réserver un chemin pour transmettre l'ensemble des données, chaque paquet est indépendant et peut emprunter un parcours différent. Aussi TCP doit numéroter les paquets de manière à ce que le récepteur puisse les réordonner.

La propriété principale de TCP est de garantir une communication fiable. En appliquant un mécanisme d'acquittement, il assure que l'intégralité des données soit transmise au destinataire. Plus précisément, ce dernier signale le prochain paquet attendu (via le système de numérotation) et s'il y a une perte, la source renverra le paquet manquant. Cependant, cette fiabilité coûte cher en temps à cause de l'attente de l'accusé de réception.

Supposons que la source attend un accusé de réception (ACK) pour chaque paquet, ceci lui évitera d'envoyer plus rapidement que la capacité du réseau: si le réseau est rapide l'accusé de réception (ACK) arrive rapidement et elle enverra les données rapidement ; si le réseau est lent les deux processus deviendront lents aussi. Cependant, ceci peut aussi lui causer une sous-exploitation de la capacité du réseau (la source envoie beaucoup plus lentement que la capacité du réseau). Supposons que le réseau a une capacité de 1000 paquets par seconde et un temps de propagation (Round-Trip Time) de 0.1 secondes. Dans ce cas la, la source peut envoyer que 10 paquets par seconde, or que la capacité du réseau est de 1000 paquets par seconde, soit 1% de la capacité du réseau.

Pour résoudre ce problème, TCP envoie un lot de paquets avant d'attendre un accusé de réception (ACK). Dans l'exemple cité au dessus, un lot de 100 paquets serait idéal. A la réception de l'accusé (ACK), TCP envoie un autre lot. Ce processus est appelé la fenêtre d'émission ou de congestion.

Le principe de la fenêtre d'émission autorise l'émetteur à envoyer plusieurs paquets à la suite. Puis, il attend le signal du récepteur avant de reprendre l'émission d'un nouveau lot de paquets. Etant donné la concurrence des différents utilisateurs pour l'accès au réseau, la question est *combien de paquets d'affilée un émetteur peut-il envoyer ?* Cette problématique constitue le point de départ du *contrôle de congestion*.

II.1.2 Définitions

Nous consacrons ce paragraphe à la définition de quelques termes relatifs au protocole TCP et au problème de congestion qui seront utiles pour la compréhension de la suite du sujet.

Un en-tête (ou ***header*** en anglais) de paquets est la partie du message contenant les informations de fonctionnement nécessaires à la transmission du paquet à travers le réseau : adresses de l'expéditeur et du destinataire, taille du message, code de détection d'erreurs... Son format est directement lié au protocole et au niveau de la couche utilisée. Comme nous avons pu voir dans le premier chapitre, chaque couche réseau ajoute son propre en-tête. Mais comme nous parlons ici du protocole TCP, l'en-tête de cette couche contient les numéros des paquets pour pouvoir les réassembler sur la machine du destinataire, le numéro de port pour identifier le processus pour lire le message.

Un acquittement, plus communément nommé accusé de réception, est un message (ACK) émis par le destinataire pour informer l'émetteur que le paquet transmis a effectivement été reçu (par référence au numéro de séquence spécifié dans l'en-tête du paquet). Plus précisément, le récepteur renvoie dans l'acquittement le prochain paquet attendu. Le protocole TCP utilise ce mécanisme afin de *fiabiliser* (en termes de Qualité de Service) la communication. Dans ce cas, la non-réception d'un acquittement, conséquente à la perte d'un paquet (où à un retard important), impliquera sa retransmission (Figure II.1).

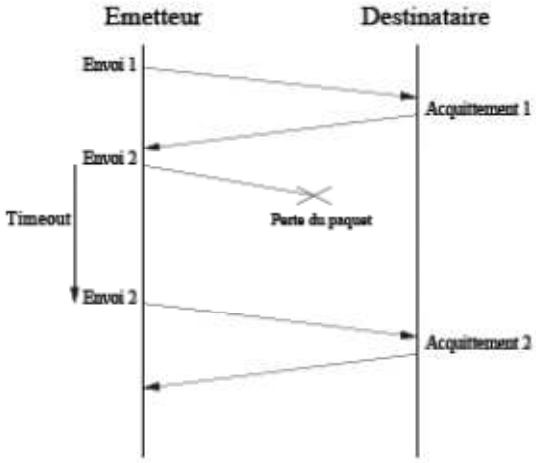

Fig. II.1 : Système d'accusé de réception et occurrence d'une Time Out

Le Time Out (TO) correspond au temps d'attente maximal d'un accusé de réception. Sur l'envoie d'un paquet, un timer se déclenche. Lorsque le temps fixé par le Time Out est écoulé, le paquet est supposé perdu, donc la source renvoie à nouveau ce paquet.

Le buffer ou mémoire tampon, est une zone de mémoire vive utilisée pour stocker temporairement des données. Ici on fait référence au buffer d'un routeur ou les paquets

envoyés sont stockés et mis en file d'attente temporairement avant d'être traités et routés par le routeur.

La congestion est un phénomène de saturation. Elle se produit lorsque le trafic est trop intense par rapport à ce que la capacité du réseau peut traiter. Ce phénomène peut survenir, par exemple, lorsqu'un nombre important d'utilisateurs se connecte à un même site, inondant alors le serveur de requêtes et provoquant le ralentissement des différentes connexions. Au niveau d'un routeur, ce phénomène arrive quand le buffer de celui-ci est plein et de ce fait tous les paquets arrivant seront ignorés.

Le Round Trip Time (RTT) d'une connexion est le temps nécessaire lors d'un échange aller-retour entre l'émetteur et le récepteur. Il se compose du temps de propagation sur le média ainsi que le temps de traitement dans les différents routeurs que le paquet et l'acquittement ont emprunté.

La fenêtre de réception désigne le nombre maximum de paquets que le destinataire est capable de recevoir (lié à la place disponible dans le buffer de réception). La taille de la fenêtre est donc fixée par le récepteur et constitue le principe de contrôle de flux évoqué au paragraphe précédent.

La fenêtre de congestion correspond au nombre de paquets que l'émetteur peut envoyer avant de devoir attendre la réception d'un accusé de réception. Une fois l'acquittement effectué, la transmission peut poursuivre. Bien évidemment, la fenêtre de congestion doit toujours rester inférieure à la fenêtre de réception.

Un acquittement dupliqué ou DupAck est un nouvel acquittement, identique au précédent, envoyé à l'émetteur pour signaler qu'un paquet n'est pas arrivé et donc toujours attendu par le récepteur.
La figure ci-dessous montre le principe d'une fenêtre de congestion et l'acquittement dupliqué.

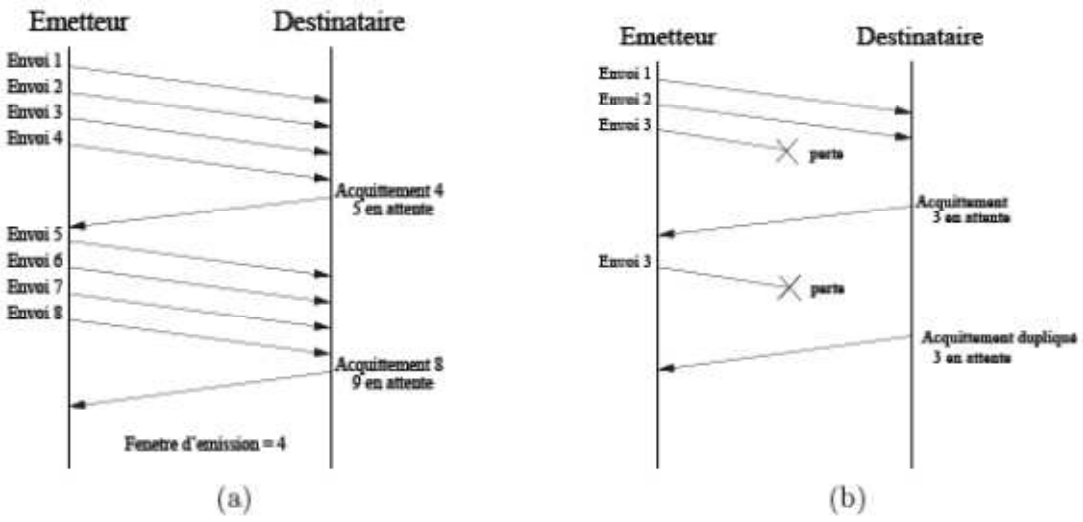

Fig. II.2 : (a) principe de la fenêtre de congestion, (b) acquittement dupliqué.

II.2 Le protocole TCP et le problème de congestion

II.2.1 Introduction

Le phénomène de congestion se manifeste lorsqu'un dispositif reçoit plus d'information que ce qu'il ne peut traiter, plus précisément, dès lors que le débit de données entrant est supérieur à celui sortant. En premier lieu et indépendamment du réseau, un émetteur est donc tenu d'envoyer une quantité de données au plus égale à la capacité du récepteur. Afin d'éviter une éventuelle surcharge, le récepteur informe l'émetteur de la capacité disponible en remplissant un champ dédié de l'en-tête du message d'acquittement. Ce mécanisme, appelé *contrôle de flux*, permet à la station réceptrice de régler la taille maximum du flux d'émission de la source.

Bien que le trafic généré par TCP soit adapté à la capacité du destinataire, il ne le sera pas forcément vis-à-vis du réseau. Contrairement au contrôle de flux, le problème est que ni l'émetteur ni le récepteur ne possèdent d'informations sur l'état de congestion du réseau et ceci d'autant plus que pour TCP, la communication à travers le réseau (réalisée par les couches inférieures) lui est transparente.

La congestion entrainant la perte de paquets, TCP doit, pour maintenir un service fiable, retransmettre l'ensemble des données perdues. Mais si la surcharge du réseau n'a pas été résolue, les paquets réémis seront certainement à nouveau perdus. Dans ces conditions, il est clair que ce mécanisme contribue à alimenter la congestion. Avec l'expansion d'Internet dans les années 80 et le succès du protocole TCP, le nombre grandissant d'utilisateurs a conduit à des séries d'effondrement du réseau, entrainant des baisses drastiques de performances. Des travaux de recherche ont ensuite été menés, notamment à l'Université de Berkeley (Californie, USA), afin de développer des algorithmes permettant à TCP d'empêcher ce phénomène, connu sous le nom de *congestion collapse*. Finalement, les travaux de *Jacobson* [5], [6] ont doté le protocole du *contrôle de congestion* et jeté les bases de la version de TCP la plus répandue actuellement.

II.2.2 Algorithmes d'évitement de congestion

Afin d'éviter la congestion d'un élément du réseau à partir de la vision bout en bout des échanges TCP, *Jacobson* [5] a proposé l'algorithme AIMD (*Additive-Increase Multiplicative-Decrease*). L'objectif de celui-ci est de transmettre un maximum d'informations tout en minimisant la perte de paquets

L'idée de base de cet algorithme est de considérer qu'une perte de paquets est synonyme de congestion. En effet, les pertes accidentelles, dues à un problème physique de la ligne de communication ou encore aux erreurs bits non corrigées par la couche liaison sont négligeables. En se rappelant qu'à chaque instant, les hôtes connectés sont en compétition pour l'accès aux ressources de transmission, le principe de l'algorithme AIMD est simple, chaque émetteur augmente progressivement son taux d'envoi (*Additive-Increase*). Cet accroissement s'effectue jusqu'à l'occurrence d'une perte, le cas échéant cela signifie qu'il y a une surcharge quelque part dans le réseau. Une perte peut être distinguée par deux indications différentes, les indications par acquittements dupliqués (3DupAck) et les indications par un Time Out. Bien évidemment il y a une différence entre ces deux indications ; une indication par acquittement dupliqué signifie que le récepteur réclame toujours le paquet suivant par contre une indication par un Time Out signifie que l'émetteur n'as plus de nouvelle du récepteur ce qui veut dire que la congestion du réseau est trop importante. A cet instant, les débits d'émission des sources concernées sont alors diminués et cette réduction doit être assez importante pour être sûr de sortir de l'état de saturation

(*Multiplicative-Decrease*). Notons par W le nombre de paquets que la source envoie. Nous pouvons donc présenter cet algorithme comme suit :

– Une source envoie W paquets (la fenêtre de congestion est donc égale à W).

– Le flux de paquets transite dans le réseau :

• Si le flux est transmis avec succès, sur réception de l'ACK (acquittement), l'émetteur augmente son taux d'envoi : $W \leftarrow W + 1$.

• S'il y a une perte, l'émetteur doit retransmettre le/les paquets perdus et réduire sa fenêtre de congestion W selon l'indication de perte :

• Si la source n'a pas de réponse du destinataire (indication par un Time Out) : $W \leftarrow 1$

• Si la source reçoit trois acquittements identiques (3DupAck) : $W \leftarrow W/2$

Le temps entre chaque échange correspond au trajet d'un aller-retour, c'est-à-dire un RTT.

La Figure ci-dessous nous montre le comportement caractéristique de l'algorithme AIMD.

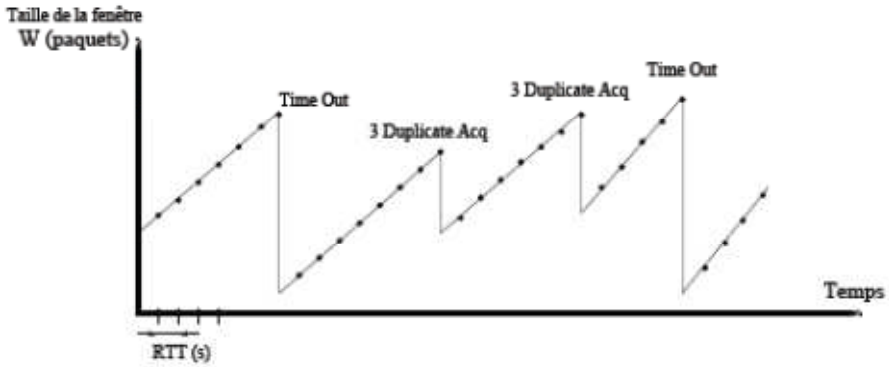

Fig. II.3 : Comportement caractéristique de l'algorithme AIMD

Au cours de son évolution, des algorithmes supplémentaires ont amélioré TCP pour le rendre plus efficace. Cette évolution a généré plusieurs versions de TCP [7] (Tahoe, Reno, SACK, New Reno, Vegas…), New Reno étant actuellement la plus utilisée. Nous décrivons ici quelques mécanismes caractéristiques.

L'algorithme du *slow start* est une phase d'initialisation et a pour but d'obtenir une estimation rapide du seuil de congestion. Pour cela, la taille de la fenêtre de congestion, initialement à 1, croît exponentiellement (de base 2) jusqu'à l'occurrence d'une perte. A cet instant, la moitié de la taille de la fenêtre courante est enregistrée dans la variable *ssthresh* (slow start threshold). Celle-ci définit le seuil en dessous duquel la fenêtre augmentera exponentiellement (phase de slow start) tandis qu'au dessus elle grandira linéairement (phase de congestion avoidance). La variable *ssthresh* est redéfinie à chaque fois qu'une perte se

produit. Ainsi la phase de progression linéaire débutera directement à partir d'une taille de fenêtre convenable, on considère qu'il n'est pas utile (perte de temps, donc de performance) d'être en phase d'évitement de congestion en dessous du seuil *ssthresh*. La figure ci-dessous nous montre le mécanisme du *slow start.*

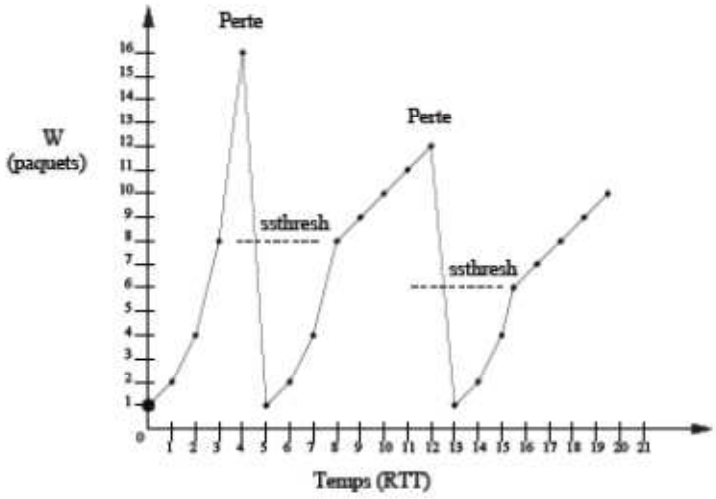

Fig. II.4 : Mécanismes du slow start et évitement de congestion.

L'algorithme du ***fast retransmit*** a été mis en place pour éviter que l'émetteur attende l'expiration du Time Out avant de retransmettre. Le destinataire signale, par ses accusés, où il en est dans la réception des données. Les paquets arrivant éventuellement dans le désordre, il est probable que la source reçoive des accusés identiques. Cependant, l'algorithme considère que sur réception de trois acquittements dupliqués (c'est-à-dire quatre acquittements portant le même numéro), un lien dans le réseau est encombré. Donc, plutôt que d'attendre la durée complète du Time Out, l'émetteur réduit la fenêtre de congestion et retransmet les informations attendues par le récepteur. Enfin, dans le cas d'une détection de perte par acquittements dupliqués, c'est l'algorithme de ***fast recovery***, souvent associé à celui du ***fast retransmit*** qui s'exécute. Dans ce cas, au lieu de se réinitialiser à un, la fenêtre se réduit à la moitié du seuil, puis passe donc directement à la phase d'évitement de congestion (phase linéaire). Les algorithmes de ***fast recovery*** et ***fast retransmit*** permettent à l'émetteur d'être plus réactif et d'anticiper un état de congestion avancé.

La figure ci-dessous nous montre maintenant l'interaction de tous ces algorithmes.

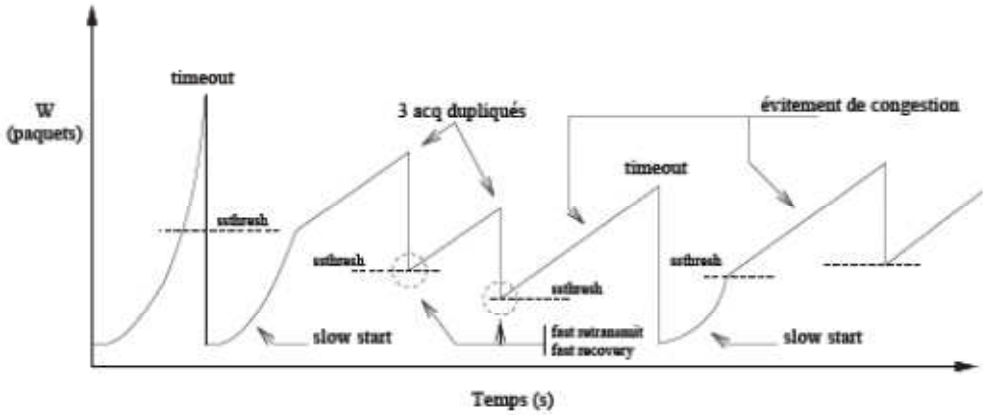

Fig. II.5 : Illustration des protocoles de slow start, fast retransmit et fast recovery (TCP Reno et NewReno).

II.2.3 Active Queue Management (AQM)

Le mécanisme d'évitement de congestion de bout en bout appliqué par TCP prévient l'effondrement du réseau mais il reste insuffisant pour assurer un contrôle efficace du trafic. Par conséquent, des études ont suggéré de faire participer le réseau au contrôle de congestion et la gestion des ressources partagées [8], [9].

Essentiellement, les routeurs appliquent un simple mécanisme de *store-and-forward* dans lequel les paquets entrants sont stockés dans un buffer avant d'être transférés (ou *routés*) vers le destinataire en utilisant une table de routage et selon la bande passante du lien. Généralement, les buffers sont gérés suivant une politique FIFO (*First-In First-Out*) et la saturation de leur capacité entraine la perte des paquets supplémentaires, et on appelle ça le principe du Drop Tail.

Les possibilités de gestion du trafic depuis les hôtes en bordure du réseau étant limitées, des mécanismes supplémentaires appelés Active Queue Management (AQM) doivent être ajoutés au niveau des routeurs et implémenter comme des programmes pour compléter les contrôles de bout en bout. Comme nous avons pu voir dans la section précédente, la perte de paquet, plus exactement la réception d'acquittements dupliqués, constituait une mesure de la congestion pour le protocole TCP. L'objectif du mécanisme d'AQM consiste alors à réguler, de façon *implicite*, le trafic TCP en agissant sur le taux de pertes. Pour cela, l'AQM a la possibilité de forcer la perte en *éjectant* un paquet du buffer afin d'anticiper la saturation totale du routeur. En effet, le paquet étant perdu (volontairement), il

n'atteindra pas le destinataire et celui-ci réclamera toujours le même paquet. L'émetteur, quant à lui, pensera que le paquet a été perdu par surcharge du réseau et diminuera en conséquence sa fenêtre d'émission. Ainsi, le mécanisme d'AQM s'appuie sur le principe de contrôle de congestion de bout en bout de TCP (c'est-à-dire l'algorithme AIMD) pour agir, à distance, sur le taux d'envoi des sources. Situé au niveau du routeur, il est le mieux placé pour analyser l'état de congestion du lien (taille de la file d'attente du buffer, intensité du trafic).
Puis, à partir de cette mesure et d'un algorithme décisionnel, il choisira d'éjecter ou non un paquet. La figure ci-dessous nous montre le principe d'un mécanisme AQM.

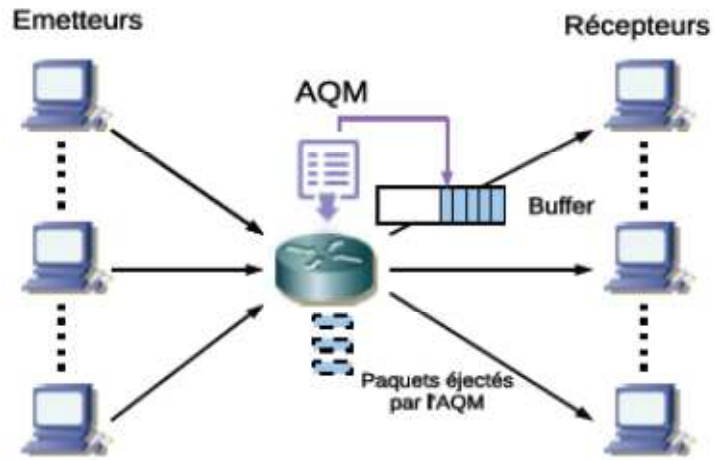

Fig. II.6 : Principe d'un mécanisme AQM.

Beaucoup de mécanismes AQM ont été proposés dans la littérature comme Random Early Detection (RED) [10], Random Early Marking (REM) [11], Blue [12], Adaptive Virtual Queue (AVQ) [13] ainsi que beaucoup d'autres pour ne citer que ceux la.
Nous allons faire ici, une petite description du mécanisme Random Early Detection (RED).

Principe

Le principe de l'algorithme du *Random Early Detection* (RED) [10] est d'anticiper la saturation du buffer. Ainsi, plutôt que d'attendre le remplissage complet du buffer et une saturation du réseau, l'idée suggérée est d'éjecter les paquets entrant au routeur avec une certaine probabilité pour indiquer à la source correspondante de réduire son taux d'envoi. Ce taux de perte est une fonction croissante de la taille moyenne de la file d'attente du buffer (Figure II.7) et s'exprime par :

$$p(\bar{b}) = \begin{cases} 0, & 0 \le \bar{b} < b_{min} \\ \frac{\bar{b} - b_{min}}{b_{max} - b_{min}} p_{max} & b_{min} \le \bar{b} \le b_{max} \\ 1, & b_{max} < \bar{b} \end{cases} \qquad (II.1)$$

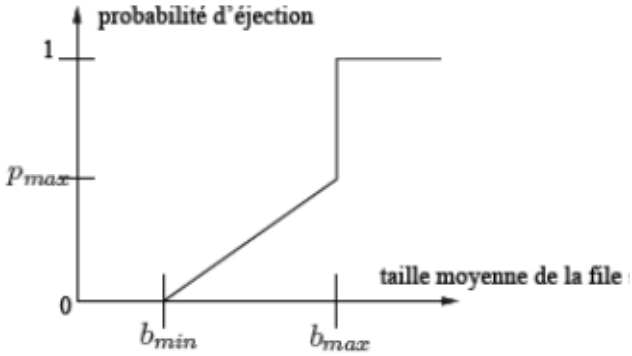

Fig. II.7 : Le mécanisme de Random Early Detection (RED).

Où $\bar{b}$ est la longueur moyenne de la file d'attente, b_{min}, b_{max} et p_{max} sont des paramètres à configurer. Donc si la file est petite ($\bar{b} < b_{min}$), chaque paquet entrant est accepté et mis en mémoire tampon. Tandis qu'au delà d'un certain seuil, une fraction des flux entrants, proportionnelle à la mémoire utilisée (en moyenne) $\bar{b}$, est rejetée. Enfin, au delà de la limite b_{max} tous les paquets supplémentaires sont refusés. Du point de vue commande, le RED peut être assimilé à un filtre passe-bas (calcul de la taille de file moyenne) avec une action proportionnelle.

L'étude de l'algorithme *Random Early Detection* (RED) a suscité beaucoup d'intérêts par la communauté scientifique [14], [15]. Des modifications du comportement du RED motivées par une difficulté de réglage ont été proposées [16], [17], [18], [19]. *Firoiu et al* [20] ont proposé quelques règles et recommandations pour le choix des paramètres du RED.

II.3 Modèle dynamique de TCP et utilisation de la théorie de commande pour le contrôle de congestion

Beaucoup de travaux de recherche se sont intéressé par l'utilisation de la théorie de commande pour le développement d'algorithmes AQM plus efficaces. C'est à partir des travaux de *Hollot et al* [21], [22] que la question a véritablement été reformulée en termes de problème de régulation au sens où l'entendent les automaticiens. Après le développement du modèle dynamique du comportement de TCP par *Misra et al* [23], [24] ou des correcteurs Proportionnel P et Proportionnel Intégral PI pour calculer la probabilité de perte $p(t)$ ont été proposé, de nombreux chercheurs se sont intéressés à cette nouvelle problématique, exploitant les différents outils de l'automatique. La plupart de ces travaux se sont focalisés sur la régulation de la file d'attente par diverses méthodes allant des correcteurs fréquentiels [25], à l'analyse non linéaire [26], en passant par la commande robuste [27] ou encore la commande prédictive [28]. Néanmoins, la plus part de ces travaux ne prennent pas en compte le retard dans le système. Si pour la communauté des réseaux de communication, le contrôle de congestion est un phénomène critique au regard de la Qualité de Service, la problématique a suscité un grand intérêt en termes de stabilité du trafic dans la communauté de l'automatique.

Nous présentons maintenant le modèle dynamique développé par *Misra et al* [24] :

Nous considérons ici une topologie d'un seul routeur recevant *N* flux TCP homogènes où toutes les sources avaient le même temps de communication, c'est-à-dire le même RTT et le même taux d'envoie comme celle de la Fig (II.6). Dans ce qui suit, nous ignorons le mécanisme du *slow start* et le *time out,* donc l'analyse sera faite dans la phase d'évitement de congestion uniquement. Le modèle dynamique fluide du comportement du TCP a été développé en utilisant des équations stochastiques et décrit par les équations différentielles non linéaires couplées suivantes :

$$\dot{w}(t) = \frac{1}{R(t)} - \frac{w(t)}{2}\frac{w\big(t-R(t)\big)}{R\big(t-R(t)\big)}p\big(t-R(t)\big) \qquad (II.2.a)$$

$$(II.2)$$

$$\dot{q} = \begin{cases} -C + \frac{N(t)}{R(t)}w(t), & q > 0 \\ max\left\{0, -C + \frac{N(t)}{R(t)}w(t)\right\}, & q = 0 \end{cases} \qquad (II.2.b)$$

Ou $\dot{x}$ représente la dérivée par rapport au temps et :

- $W \triangleq$ Taille moyenne de la fenêtre de congestion TCP (paquets)
- $q \triangleq$ Taille moyenne de la file d'attente au niveau du routeur (paquets)
- $R(t) \triangleq$ Round Trip Time (RTT) $= \frac{q(t)}{C} + T_p$ (secondes)
- $C \triangleq$ Capacité de la connexion au niveau du routeur (parquets/secondes)
- $N \triangleq$ Nombre de connexions TCP
- $P \triangleq$ Probabilité d'éjecter un paquet, mesure de congestion
- $T_p \triangleq$ Délai de propagation (secondes)

L'équation différentielle $(II.2.a)$ représente le comportement dynamique de la fenêtre de congestion TCP. Le terme $\frac{1}{R}$ modélise *l'additive-increase* tandis que le terme $\frac{W}{2}$ modélise le *multiplicative-decrease*. L'équation différentielle $(II.2.b)$ quant à elle, représente la taille de la file d'attente au niveau du routeur simplement comme étant la différence accumulée entre le flux de paquets entrant $\frac{NW}{R}$ et la capacité de connexion C. La taille de la file d'attente q et la fenêtre de congestion TCP W sont des quantités positives bornées, $q \in [0,\ q_{max}]$ et $W \in [0,\ W_{max}]$ ou q_{max} et W_{max} représentent la taille du buffer du routeur et la taille maximale de la fenêtre de congestion respectivement. Aussi, la probabilité de rejet de paquets prend des valeurs uniquement sur$[0,\ 1]$.

II.4 Conclusion

Avant de conclure ce chapitre, nous revenons sur les objectifs de notre travail et délimitons le cadre d'étude de notre application. Bien que nous nous focaliserons sur le

contrôle de congestion *d'un seul* routeur, nous montrons dans cette section que, au delà du problème théorique qu'il constitue, ce cas de figure peut être replacé dans un cadre concret.

Nous avons présenté dans ce chapitre le contexte de l'application à savoir le protocole TCP que nous avons expliqué d'une manière assez générale. Nous avons vu que TCP est non seulement en charge d'établir des communications fiables entre deux utilisateurs mais il doit également mettre en œuvre un mécanisme de contrôle de congestion. En effet, à chaque signal (acquittement) du récepteur, le protocole de niveau transport doit calculer le nombre de paquet à émettre (fenêtre de congestion). D'une part, il s'agit de maximiser le taux d'envoi pour avoir un débit de transfert intéressant et, d'autre part, il est essentiel que l'intensité cumulée des différents flux de paquets injectés dans le réseau ne dépasse pas la capacité de traitement des routeurs. L'algorithme d'évitement de congestion consiste à augmenter doucement le débit d'émission lorsque tout se passe bien puis à le diminuer drastiquement si une perte de paquet se produit. La perte de paquets, détectée par la réception de multiples acquittements dupliqués, constitue un signal d'information de la congestion.

Puis, nous avons vu qu'il existait des dispositifs, appelés AQM, destinés à contrôler le phénomène de congestion. Localisés au niveau des routeurs, le principe de fonctionnement de tels algorithmes consiste à forcer la perte de paquets de sorte à réduire le trafic de la source correspondante pour anticiper la saturation de réseau. Nous avons ensuite évoqué l'émergence de nombreux travaux dédiés à l'utilisation du principe de rétroaction propre à l'automatique pour la synthèse d'AQM efficaces assurant une certaine Qualité de Service.

Enfin, nous avons présenté le modèle dynamique du comportement de TCP développé par *Misra et al* [24] et utilisé par la communauté des automaticiens pour la synthèse d'AQM issus de la théorie de commande. C'est sur ce dernier point que nous intéresserons dans la suite de ce mémoire afin de proposer un AQM en utilisant la théorie des systèmes à retard. Mais avant cela, il est nécessaire de présenter *comment* nous étudierons la stabilité d'un tel système, affecté par le phénomène de retard. Ceci constitue l'objet du prochain chapitre.

Chapitre III

Systèmes à retard

Dans ce troisième chapitre, nous exposerons d'une manière assez générale les systèmes à retards avant de nous intéresser plus particulièrement à l'étude de leur stabilité. L'objectif n'est pas de présenter de façon exhaustive les problématiques associées à une telle classe de systèmes mais plutôt d'introduire les concepts qui nous seront nécessaires pour la synthèse d'un AQM pour le contrôle de congestion d'un routeur par la suite. Ainsi, nous nous intéresserons plus particulièrement à la stabilité des systèmes à retards au sens de *Lyapunov.*

III.1 Introduction

Lors de l'étude d'un système, l'étape de modélisation est essentielle car elle conditionne les méthodes qui seront ensuite utilisées pour analyser ses propriétés. Dans ce processus de modélisation, les phénomènes de retard apparaissent souvent dans de nombreux processus physiques. La biologie, l'écologie, l'économie, les sciences de l'ingénieur, les télécommunications sont des domaines ou interviennent des équations différentielles dont l'évolution dépond non seulement de la valeur de leurs variables à l'instant présent *t*, mais aussi d'une partie de leur histoire, c'est-à-dire des valeurs passées d'où leurs autre appellation de systèmes héréditaires. Dans ce cas, il faut mémoriser une partie de l'histoire du système pour connaitre son évolution.

En réalité tout processus physique comporte des retards et cela s'explique par le fait que chaque transmission physique s'accompagne inévitablement d'un délai de propagation ou latence. Sauf que dans les systèmes dits ordinaires, ces retards sont négligeables par rapport aux constantes de temps de ces systèmes, donc ils ne sont pas pris en compte. En revanche, si ces retards sont du même ordre de grandeur que certaines de ces constantes de temps, ils doivent être impérativement pris en compte dans les analyses comportementales et/ou la conception de dispositif contrôle/commande de tels processus. Ainsi, l'étude des systèmes à retard a fait l'objet de nombreux travaux en automatique durant ces dernières décennies pour

développer des méthodes d'analyse de leurs propriétés (stabilité et autres) [29], [30], [31] et établir des techniques de contrôle et de commande [32], [33].

III.2 Généralités sur les systèmes à retard

Cette section vise à présenter certaines notions fondamentales relatives aux systèmes à retard, avec un regard porté notamment sur les principaux modèles mathématiques de tels systèmes, les types de retards usuellement considérés.

III.2.1 Définition

Un système à retard est un système régi par un système d'équations différentielles fonctionnelles de la forme :

$$\begin{cases} \dot{x}(t) = f\big(t, x_t(\theta)\big), \quad t \geq t_0 \\ x_{t_0}(\theta) = \emptyset(\theta), \quad \theta \in [-\tau, 0] \end{cases} \tag{III.1}$$

Ou $x(t) \in \mathbb{R}^n$ est l'état du système (III.1) à l'instant t, et f une fonction supposée continue, localement Lipschitz par rapport à la seconde variable et telle que $f(t, 0) = 0$.

L'équation (III.1) indique que la dérivée de l'état $x(t)$, à l'instant t, dépond du temps présent t et de l'état $x_t(\theta) \in C$ définis par $x_t(\theta) = x(t + \theta), \forall \theta \in [-\tau, 0]$, et pour lesquels le domaine de définition C est un sous espace $C([-\tau, 0], \mathbb{R}^n)$ des fonctions continues définies dans l'intervalle $[-\tau, 0]$ et à valeurs dans $\mathbb{R}^n$. La condition initiale x_{t_0} peut ainsi être représentée par une fonction continue $\emptyset$ de la forme suivante :

$$x(t_0 + \theta) = \emptyset(\theta), \forall \theta \in [-\tau, 0] \tag{III.2}$$

L'état du système (III.1) à chaque instant t est constitué de l'ensemble des états instantanés $\{x(t + \theta) : \theta \in [-\tau, 0]\}$. Ainsi, l'espace d'état de ce système est de dimension infinie.

III.2.2 Modèles de systèmes à retard

Le modèle (III.1) constitue une représentation générale qui peut être déclinée sous plusieurs formes, en fonction des classes de systèmes concernées. Les sections suivantes présentent alors deux de ces classes les plus usuellement considérées.

III.2.2.1 Systèmes de type retardé

Les systèmes de types retardés sont des systèmes dynamiques régis par des équations différentielles fonctionnelles portant à la fois sur des valeurs présentes et passées du temps. Ces systèmes retardés sont alors décrits par un modèle de la forme :

$$\begin{cases} \dot{x}(t) = f(t, x_t, u_t), \ \ t \geq t_0 \\ y(t) = g(t, x_t, u_t) \\ x_{t_0} = \emptyset(\theta), \ \ \theta \in [-\tau, 0] \\ u_{t_0} = \emptyset(\theta), \ \ \theta \in [-\tau, 0] \end{cases} \tag{III.3}$$

Ou $x(t) \in \mathbb{R}^n$ est le vecteur d'état, $y(t) \in \mathbb{R}^p$ est le vecteur des sorties, $u(t) \in \mathbb{R}^m$ représente l'entrée de commande, f et g sont des fonctions vectorielles avec des propriétés analytiques : continuité, dérivabilité, Lipschitzienne, et les fonctions x_t et u_t sont définies, en respect de la notation de Shimanov [34], par :

$$x_t \, , u_t \begin{cases} [-\tau, 0] \ \rightarrow \ \mathbb{R}^n, \ \ \theta \ \in \ [-\tau, 0] \\ x_t(\theta) = x(t + \theta) \\ u_t(\theta) = u(t + \theta) \end{cases} \tag{III.4}$$

Où τ est un réel strictement positif qui représente le plus grand retard du système, et x_t , u_t caractérisant la présence de l'état $x(t)$ et la commande $u(t)$ sous formes retardées.
Pour connaître l'évolution du système à partir de l'instant initial t_0, deux informations sont nécessaires, dont, notamment, la connaissance de la condition initiale de l'état x sur l'intervalle$[t_0 - \tau, \ t_0]$. Ce système est donc bien un système de dimension infinie car nous avons besoin de la valeur de l'état en une infinité de points et non pas une seule valeur x_0 comme dans le cas des systèmes régis par des équations différentielles ordinaires.

Nous présentons maintenant les différents modèles de systèmes à retards rencontrés dans la littérature.

III.2.2.1.a Systèmes linéaires retardés avec un retard discret sur l'état

Cette classe de systèmes est la plus simple et, de fait, la plus fréquemment rencontrée. L'équation d'état de cette classe est donnée, dans sa forme basique, par :

$$\begin{cases} \dot{x}(t) = Ax(t) + A_d x(t - \tau) + Bu(t) \\ y(t) = Cx(t) \end{cases} \quad \text{(III.5)}$$

Ou $x(t) \in \mathbb{R}^n$ est le vecteur d'état, $y(t) \in \mathbb{R}^p$ est le vecteur des sorties, $u(t) \in \mathbb{R}^m$ est l'entrée de commande, $\tau > 0$ est le retard (constant ou variable), A, A_d, B et C sont des matrices constantes de dimensions appropriées, et la condition initiale est définie par :

$$x(t_0 + \theta) = \emptyset(\theta), \forall\theta \in [-\tau, 0] \quad \text{(III.2)}$$

Naturellement, le modèle (III.5) peut être facilement étendu à la classe des systèmes à état retardé et entrée retardée, en considérant alors la forme suivante :

$$\begin{cases} \dot{x}(t) = Ax(t) + A_d x(t - \tau) + Bu(t) + B_d u(t - \tau) \\ y(t) = Cx(t) \end{cases} \quad \text{(III.6)}$$

A signaler que cette description mathématique peut être exploitée pour prendre en compte l'existence de certaines formes de retards inhérents à des transferts de matière où d'informations, et a donc servi de base à plusieurs travaux relatifs au contrôle de congestion des réseaux TCP. C'est ce modèle qui nous intéressera dans la suite de ce mémoire.

III.2.2.1.b Systèmes linéaires retardés avec des retards discrets et distribués

Cette classe de systèmes contient une forme particulière de retards qualifiés usuellement de retards distribués, et qui se rencontrent dans certains processus réels tels que celui de la combustion du propane dans les missiles. Une telle classe est alors associée à une description de la forme suivante :

$$\begin{cases} \dot{x}(t) = Ax(t) + A_d x(t - \tau_1) + \int_0^{\tau_2} A_{d_2}(\theta) x(t - \theta) d\theta \; Bu(t) \\ x(t_0 + \theta) = \emptyset(\theta), \forall\theta \in [-\tau, 0] \end{cases} \quad \text{(III.7)}$$

III.2.2.2 Systèmes de type neutre

La classe des systèmes de type neutre est une classe plus générale que celles présentées dans la section précédente, dans le sens où les modèles de ces systèmes considèrent comme dérivée de l'état au temps présent, une fonction qui dépend non seulement des valeurs de l'état passé, mais aussi de la dérivée de l'état passé dans un intervalle. Ces systèmes sont ainsi régis par des équations de la forme générale suivante :

$$\begin{cases} \dot{x}(t) = f(t, x_t, \dot{x}_t, u_t), \ \ t \geq t_0 \\ x_{t_0} = \emptyset(\theta), \ \ \theta \in [-\tau, 0] \\ u_{t_0} = \emptyset(\theta), \ \ \theta \in [-\tau, 0] \end{cases} \quad \text{(III.8)}$$

Qui, en considérant la représentation d'état du système, revêt généralement la forme suivante :

$$\dot{x}(t) - F\dot{x}(t - \tau) = Ax(t) + A_d x(t - \tau) + Bu(t) + B_d u(t - \tau) \quad \text{(III.9)}$$

Avec

$$x(t_0 + \theta) = \emptyset(\theta), \forall\theta \in [-\tau, 0] \quad \text{(III.2)}$$

Où $x(t) \in R^n, u(t) \in \mathbb{R}^m$ sont respectivement le vecteur d'état du système et la commande. $A, A_d, F, B, B_d \in \mathbb{R}^{nxm}$ sont des matrices constantes connues de dimensions appropriées, et $\emptyset(\theta)$ est le vecteur des conditions initiales définies sur l'intervalle$[-\tau, 0]$.

Dans la pratique, le comportement dynamique de nombreux procédés physiques peut être décrit par un modèle de système neutre. En effet, la présence de l'argument $\dot{x}(t)$ (i.e. de la dérivée de l'état passé) dans le modèle, introduit une difficulté importante pour leur analyse. Néanmoins, l'étude de leur stabilité et leur stabilisation a été un sujet largement abordé par de nombreux chercheurs (voir par exemple [35], [36])

III.2.3 Types de retard

Comme nous l'avons déjà mentionné, les délais temporels inhérents aux phénomènes de transmission, de transport et de propagation interviennent assez naturellement dans tout les processus physiques. La caractérisation de ces délais peut alors être opérée suivant deux catégories élémentaires, qui constituent également deux hypothèses fondamentales

susceptibles d'orienter le choix ou le développement des techniques d'analyse comportementale des systèmes à retard. Ces catégories sont les suivantes :

(a) Retards inconnus : Dans ce premier cas, aucune hypothèse sur le retard n'est considérée. Qu'il soit constant ou variant dans le temps, il peut prendre toutes les valeurs dans $\mathbb{R}_+$ [37].

(b) Retards majorés : Cette seconde classe suppose la connaissance d'une valeur maximale sur le retard

$$0 \leq \tau(t) \leq \tau_{max}.$$

Si $\tau(t) = \tau$ est constant, il reste toujours incertain et la contrainte ci-dessus peut être utilisée pour la robustesse. Ce cas de figure a été largement considéré dans la littérature [38], [39].

(c) Retards bornés : Moins abordée que le cas précédent, cette dernière catégorie suppose que le retard vérifie la contrainte

$$\tau_{min} \leq \tau(t) \leq \tau_{max}.$$

Le phénomène de retard souvent induit par le transport d'information, impose dans ce cas un délai (dû au temps de propagation) minimum incompressible. La littérature concernant ce type de retard est moins vaste

Dans le cas des retards variant dans le temps, une contrainte supplémentaire relative à sa dérivée peut être ajoutée

$$|\dot{\tau}(t)| \leq d, d \in \mathbb{R}_+$$

Indiquant alors une limitation sur la vitesse de variation du retard $\tau(t)$. En pratique la contrainte $d \leq 1$ est souvent utilisé afin d'assurer que le retard ne varie pas plus rapidement que le temps et que les informations retardées arrivent dans l'ordre chronologique.

Par ailleurs, indépendamment du procédé physique et du type du retard associé, certains auteurs [40], se sont intéressés à la robustesse de la stabilité d'un système vis-à-vis du retard. Il s'agit dans ce cas d'estimer les intervalles (ou encore *clusters* en anglais) sur le

retard tel que le système reste stable. Un cas particulier considère un système à retards stable lorsque celui-ci est nul et le but est de trouver la valeur maximale du retard telle que la stabilité du système soit préservée.

III.2.4 Modélisation incertaine d'un système linéaire avec retard

De manière générale, les modèles mathématiques ne peuvent pas décrire parfaitement le comportement dynamique d'un système. Tout d'abord, les équations régissant un procédé donné sont généralement complexes (non linéaires, à coefficients variant dans le temps) et des simplifications sont donc nécessaires pour pouvoir les traiter avec les outils de l'automatique.

Deuxièmement, les paramètres caractérisant un système ne peuvent être identifiés avec exactitude, ceci étant dû, par exemple, aux imprécisions de mesure. Ainsi, linéarisation, dynamiques négligées, incertitudes d'identification sont autant d'imprécisions affectant la pertinence du modèle considéré. De plus, ces paramètres peuvent être variant dans le temps.

La question est donc : *comment traiter ces erreurs de modélisation ?*

Afin de tenir compte des incertitudes de modélisation, il convient d'adapter notre modèle linéaire à retard pour obtenir un modèle linéaire *incertain* à retard

$$\begin{cases} \dot{x}(t) = A(\Delta)x(t) + A_d(\Delta)x(t-\tau) \\ y(t) = C_0(\Delta)x(t) + C_1(\Delta)x(t-\tau) \end{cases} \qquad (III.10)$$

Où Δ représente le caractère incertain du modèle. $A, A_d, C_0\ et\ C_1$ sont des matrices de dimensions appropriées qui ne sont pas complètement connues mais on suppose qu'elles appartiennent à un ensemble donné Ω appelé *ensemble incertain* $\forall \Delta \in \Delta$. Δ est l'ensemble des incertitudes admissibles. Le principe de la modélisation incertaine est d'étudier non plus un modèle unique mais un ensemble de modèles qui englobe le comportement du système réel.

Incertitudes polytopiques

Un polytope est un ensemble constitué de n_p sommets et est défini comme l'enveloppe convexe de ces derniers. Cette modélisation est applicable seulement dans le cas où les matrices du système (III.10) sont affines en les paramètres incertains. Si l'on suppose que chacune des incertitudes évoluent entre deux bornes alors l'ensemble incertain caractérisant

$$\sum(\Delta) = \begin{bmatrix} A(\Delta) & A_d(\Delta) \\ C_0(\Delta) & C_1(\Delta) \end{bmatrix} \in \boldsymbol{\Omega}, \text{ pour tout } \Delta \in \triangle,$$

est un polytope et peut s'exprimer comme l'enveloppe convexe des sommets de cette matrice

$$\Omega = \text{co}\{\sum^{[1]}, \sum^{[2]}, \dots, \sum^{[n_p]}\}$$

III.3 Stabilité des systèmes à retards

La notion de stabilité constitue une problématique centrale de l'automatique. Souvent liée à la façon d'appréhender un système, la stabilité possède un large éventail de définitions.

III.3.1 Notions sur la stabilité des systèmes à retard

Dans cette section, nous présentons quelques notions de base concernant la stabilité des systèmes à retards. Rappelons tout d'abord que la stabilité d'un point d'équilibre d'un système avec ou sans retard, consiste toujours à observer que son évolution reste proche du point d'équilibre lorsqu'on s'en écarte d'un certain voisinage. La stabilité asymptotique, en plus de garantir la condition précédente, indique que le système reviendra exactement au point d'équilibre, au bout d'un temps éventuellement infini (si on s'en écarte 'légèrement'). La stabilité exponentielle garantit, quant à elle, non seulement le caractère asymptotique mais aussi la rapidité de la convergence.

Ces définitions peuvent également s'exprimer de manière mathématique. Afin d'introduire de telles formulations, considérons à nouveau le cas de la classe des systèmes à retard décrits par un modèle général de la forme :

$$\begin{cases} \dot{x}(t) = f\big(t, x_t(\theta)\big), & t \geq t_0 \\ x_{t_0}(\theta) = \emptyset(\theta), & \theta \in [-\tau, 0] \end{cases} \qquad \text{(III.1)}$$

Ou $f(t,0) = 0$ et $x_t(.)$, pour $t \geq t_0$ donné, représente la restriction de $x(.)$ sur l'intervalle $[t-\tau,\ t]$ translaté sur $[-\tau, 0]$, i.e :

$$x_t(\theta) = x(t\ + \theta), \forall \theta\ \in\ [-\tau, 0]$$

Hypothèses :

a) L'application $f(t,\emptyset)\colon \mathbb{R}^+ xC \to \mathbb{R}^n$ (avec $f(t,0) = 0$) est continues et lipschitzienne en la deuxième variable (en l'occurrence : $\emptyset$).

b) $x\big(t, t_0, \emptyset(.)\big) = x_{(t_0,\emptyset)}(.)$ représente la solution de l'équation différentielle fonctionnelle avec la condition initiale $(t_0, \emptyset) \in \mathbb{R}^+ xC$.

Avec $\beta(0,b)$ avec $b > 0$ un voisinage de l'origine.

Nous pouvons, à présent, introduire l'ensemble des définitions suivantes.

Définition [41]

La solution triviale $x(t) = 0$ du système (III.1) est dite stable si, pour n'importe quel $k > 0$ et pour n'importe quel t_0 , il existe un $\delta(t_0, k)$ tel que pour toutes les valeurs initiales $\emptyset \in \beta(0,\delta)$, la solution $x(t_0,\emptyset)$ satisfait $x_t(t_0,\emptyset) \in \beta(0,k)$ pour tout $t \geq t_0$.

Définition [41]

La solution triviale $x(t) = 0$ du système (III.1) est dite asymptotiquement stable si elle est stable et il existe un $b_0(t_0) > 0$ tel que, pour toutes les valeurs initiales $\emptyset \in \beta(0, b_0)$, la solution $x(t_0,\emptyset)$ satisfait $x_t(t_0,\emptyset) \to 0$ quand $t \to \infty$.

Définition [41]

La solution triviale $x(t) = 0$ du système (III.1) est dite uniformément stable si, pour n'importe quel $k > 0$ et pour n'importe quel t_0, il existe un $\delta(k)$ indépendant de t_0 tel que, pour toutes les valeurs initiales $\emptyset \in \beta(0,\delta)$, la solution $x(t_0,\emptyset)$ satisfait $x_t(t_0,\emptyset) \in \beta(0,k)$ pour tout $t \geq t_0$.

Définition [41]

La solution triviale $x(t) = 0$ du système (III.1) est dite exponentiellement stable avec un taux de décroissance $\propto$, s'il existe un $M > 0$ et un $\propto > 0$ tels que, pour n'importe quelle condition initiale $\emptyset$ avec $\|\emptyset\|_c \leq v_0 \leq v$, la solution $x(t_0, \emptyset)$ satisfait l'inégalité :

$$\|x_t(t_0, \emptyset)(\theta)\| \leq Me^{-\propto(t-t_0)}\|\emptyset\|_c, \qquad \forall\theta \in [\tau, 0]$$

Interprétations

- Le première définition signifie que la solution de (III.1) est stable si, pour n'importe quel voisinage de l'origine $\beta(0, k)$ dans C et pour toute valeur initiale t_0, nous pouvons toujours trouver un autre voisinage $\delta = \delta(t_0, k)$ de l'origine $(0, \delta)$, tel que, pour n'importe quelle condition initiale dans $(0, \delta)$, nous garantissons que la solution $x_t(.)$ se trouve dans $\beta(0, k)$ pour tout $t \geq t_0$.

- La seconde définition dit, quant à elle, que s'il existe un b_0 dépendant du point t_0 tel que l'origine soit un attracteur du système pour n'importe quelle condition initiale $\emptyset \in \beta(0, b_0)$, alors la stabilité asymptotique est garantie.

- La troisième définition précise que si le δ est indépendant de t_0 alors, il y a stabilité uniforme par rapport à la condition initiale (en remarquant que si, de plus, l'origine est un attracteur pour ce système, il y a alors stabilité uniforme asymptotique).

- La notion de stabilité exponentielle implique l'existence d'un taux de décroissance $\propto$ tel que toutes les solutions sont bornées par une exponentielle $\exp(-\propto(t - t_0))$ "modulo" la condition initiale $\emptyset$.

Dans la littérature, il existe principalement trois méthodes pour l'étude de la stabilité et la commande des systèmes à retard : l'approche fréquentielle à partir de l'équation caractéristique en utilisant l'extension du critère de Routh-Hurwitz et la méthode du lieu des racines, l'approche robuste et l'approche par la théorie de Lyapunov. Nous allons considérer ici la dernière méthode que nous allons utilisé pour la synthèse d'AQM pour le contrôle de

congestion d'un routeur du fait que c'est une méthode pratique et effective, et nous fournis des critères LMI (Inégalité Matricielle Linéaire) facile à résoudre.

III.3.2 Etude de la stabilité par la seconde méthode de Lyapunov

Dans le cadre des équations différentielles ordinaires, la seconde méthode de Lyapunov, appelée également méthode directe car elle ne nécessite pas la résolution des équations (i.e. la connaissance explicite des solutions), repose sur l'existence d'une fonction $V(t)$ définie positive telle que sa dérivée soit définie négative. Une telle méthode peut être appliquée sans modification majeure aux systèmes à retards. Cependant celle-ci présente, dans le cas général, un inconvénient majeur qui est d'imposer des conditions sévères sur le système pour montrer que la dérivée de la fonction de Lyapunov calculée le long des trajectoires est négative (en effet, cette dérivée n'est plus une fonction ordinaire, mais une fonctionnelle : elle dépend aussi de certaines valeurs passées de l'argument t). Cette méthode est donc difficilement exploitable pour de nombreux cas de systèmes à retards. Deux extensions de la seconde méthode de Lyapunov ont alors été développées d'un coté par Krasovskii et, de l'autre par Razumikhin, dans le cadre des équations différentielles à retards. Une brève présentation de ces deux extensions constitue l'objet des deux paragraphes suivants.

III.3.2.1 Approche de Lyapunov-Krasovskii

Cette extension de la méthode de Lyapunov est due à Krasovskii [42] et permet d'analyser la stabilité en termes de propriétés de certaines fonctionnelles associées aux systèmes considérés. Ceci se traduit par l'énoncé du théorème suivant.

Théorème III.1 : *[42] Soit le système à retard défini par l'équation différentielle (III.1). Soient* $u, v, w : \mathbb{R}^+ \to \mathbb{R}^+$ *sont des fonctions continues non décroissantes, avec en plus* $u(\theta)$ *et* $v(\theta)$ *positives pour* $\theta > 0$, $u(0) = v(0) = 0$. *S'il existe une fonctionnelle continue et différentiable* $V(t, \emptyset)$: $\mathbb{R} \times C \to \mathbb{R}$ *telle que :*

a. $u(\|\emptyset(0)\|) \leq V(t, \emptyset) \leq v(\|\emptyset\|_c)$

b. $\dot{V}(t, x_t) \leq -w(\|x(t)\|)$, *pour tout* $t \in \mathbb{R}$.

Ou $\dot{V}(t, x_t)$ *est la dérivée dans le sens de Dini*

$$\dot{V}(t, x_t) = lim \sup_{\varepsilon \to 0^+} \frac{V(t + \varepsilon, x_{t+\varepsilon}) - V(t, x_t)}{\varepsilon}$$

Alors la solution triviale $x = 0$ de (III.1) est uniformément stable. Si de plus $w(\theta) > 0$ quand $\theta > 0$, alors la solution triviale est uniformément asymptotiquement stable.

Interprétation : Le problème de la stabilité asymptotique ou uniforme est réduit à trouver une fonctionnelle appelée fonctionnelle de Lyapunov-Krasovskii qui satisfait aux conditions (a) et (b). La condition (a) signifie que la fonctionnelle est définie positive (existence de la fonction $u(.)$) et possède une borne supérieure infinitésimale (existence de la fonction $v(.)$), i.e. un comportement borné de la fonctionnelle V pour toutes les valeurs de $t \in \mathbb{R}$ et $\emptyset \in C$. La condition (b), quant à elle, stipule simplement que la dérivée de la fonctionnelle V calculée le long des trajectoires de (III.1) doit être définie négative.

De manière générale, la méthode de Lyapunov se compose de deux points essentiels. Premièrement, le pessimisme de l'analyse sera conditionné par le choix de la fonctionnelle et de la nature des matrices de Lyapunov (invariantes, dépendantes du retard, dépendantes de paramètres incertains). Des transformations de modèle sont couramment appliquées au système pour :

a) introduire de nouvelles variables de décision et offrir des degrés de liberté supplémentaires aux critères à tester. Nous pouvons citer, par exemple, la formule de Newton-Leibniz [43]

$$x(t-\tau) = x(t) - \int_{t-\tau}^{t} \dot{x}(\theta)d\theta = x(t) - \int_{t-\tau}^{t} [Ax(\theta) + A_d x(\theta-\tau)]d\theta$$

Le lecteur intéressé pourra consulter, par exemple, les références [44], [29] pour un rapide tour d'horizon sur les modèles de comparaison.

b) pour obtenir une meilleure description de la dynamique du système, par exemple, par augmentation du modèle (fractionnement du retard, décalage temporel).

La seconde difficulté réside dans les techniques utilisées pour vérifier que la condition de décroissance sur la dérivée de V soit satisfaite. Généralement, il s'agit de majorer cette dernière quantité en introduisant des inégalités plus ou moins conservatives. Parmi les plus utilisées, nous pouvons citer les inégalités de Park [45], Moon [46] ou encore Jensen [47]. De plus, la recherche de conditions de stabilité exploitables fait appel à diverses manipulations

mathématiques telles que des changements de variables linéarisants, le complément de Schur ou encore le lemme de Finsler.

III.3.2.2 Approche de Lyapunov-Razumikhin

Une autre approche de la stabilité des systèmes à retard a été introduite par Razumikhin [48], et s'appuie, pour sa part, sur des fonctions de Lyapunov $V(t, x(t))$ plutôt que des fonctionnelles $V(t, x_t)$. Le résultat de Razumikhin peut s'énoncer au travers le théorème suivant :

Théorème III.2 : *[37] supposons que la fonction* $f: \mathbb{R} \times C \to \mathbb{R}^n$ *tel que l'image par* f *de* $\mathbb{R} \times (un\ ensemble\ borné\ de\ C)$ *est un ensemble borné de* $\mathbb{R}^n$*, et que* $u, v, w: \mathbb{R}^+ \to \mathbb{R}^+$ *sont des fonctions continues non décroissantes, avec en plus* $u(\theta)$ *et* $v(\theta)$ *positives pour* $\theta > 0$*, et* $u(0) = v(0) = 0, v$ *est strictement croissante. S'il existe une fonction continue différentiable* $V\big(t, x(t)\big): \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}$ *telle que :*

a) $u(\|x\|) \leq V(t, x) \leq v(\|x\|),\ t \in \mathbb{R}, x \in \mathbb{R}^n.$

b) $\dot{V}(t, x_t) \leq -w(\|x\|)$ *si* $V\big(t + \theta, x(t + \theta)\big) < V\big(t, x(t)\big),\ \forall \theta \in [-\tau, 0].$

Alors la solution triviale $x = 0$ *du système (III.1) est uniformément stable.*

Si en plus $w(\theta) > 0$ *pour* $\theta > 0$*, et s'il existe une fonction continue non décroissante* $p(s): \mathbb{R}^+ \to \mathbb{R}^+$, $p(\theta) > \theta$ *pour* $\theta > 0$ *telle que :*

a') $u(\|x\|) \leq V(t, x) \leq v(\|x\|),\ t \in \mathbb{R}, x \in \mathbb{R}^n.$

b') $\dot{V}\big(t, x(t)\big) \leq -w(\|x\|)$ *si* $\big(t + \theta, x(t + \theta)\big) < pV\big(t, x(t)\big),\ \forall \theta \in [-\tau, 0].$

Alors la solution triviale est uniformément asymptotiquement stable.

Interprétation : Le théorème de Razumikhin donne une condition suffisante pour l'analyse de la stabilité d'un système (III.1), en regard de l'existence d'une fonction de Lyapunov particulière nommée : fonction de Lyapunov-Razumikhin. Les conditions (a) et (a') représentent les conditions classiques imposées aux fonctions de Lyapunov candidates, et les conditions (b) et (b') constituent des conditions sur la dérivée de la fonction $V(t, x(t))$. Donc, dans le cas du théorème de Razumikhin, l'étude de la stabilité passe par la recherche d'une fonction de Lyapunov et non pas celle d'une fonctionnelle bornée, comme c'est le cas pour le théorème de stabilité de Lyapunov-Krasovskii.

III.4 Etude de la stabilité d'un système linéaire avec retard discret sur l'état

Nous allons considérer ici la classe de systèmes à retard que nous allons rencontrer dans la suite de ce mémoire, en l'occurrence les systèmes linéaires retardés avec un retard discret sur l'état.

Considérons un système linéaire à états retardés décrit par l'équation :

$$\dot{x}(t) = Ax(t) + A_d x(t-\tau) \qquad \text{(III.11)}$$
$$x(t_0 + \theta) = \emptyset(\theta), \theta \in [-\tau, 0]$$

Où τ est le retard du système, supposé constant et incertain (autrement dit, le retard est supposé connu dans intervalle borné).

Considérons également l'hypothèse suivante, à savoir :

Le système régi par (III.11) sans retard ($\tau = 0$) :

$$\dot{x}(t) = (A + A_d)x(t)$$
$$x(t_0) = x_0 \in \mathbb{R}^n$$

est asymptotiquement stable (i.e la matrice $(A + A_d)$ est stable au sens de Hurwitz) négatives).

Cette hypothèse, assez naturelle, consiste simplement à supposer que le système est asymptotiquement stable en l'absence de retard (i.e. pour $\tau = 0$).Pour $\tau \neq 0$, il existe alors deux possibilités envisageables, à savoir :

1. La stabilité asymptotique est assurée pour toutes les valeurs positives du retard τ, et, dans ce cas, la stabilité est qualifiée d'indépendante de la taille du retard.

2. Il existe une valeur non nulle τ^*, telle que le système est asymptotiquement stable pour n'importe quel retard positif τ tel que $0 \leq \tau < \tau^*$, et le système devient instable pour $\tau = \tau^*$. Dans ce cas, la stabilité est dite dépendante de la taille du retard.

Nous allons présenter uniquement la notion de stabilité indépendante du retard que nous utiliserons ultérieurement pour la synthèse d'AQM pour la régulation du trafic TCP au niveau d'un routeur.

Afin d'appréhender la notion de stabilité indépendante du retard, rappelons que le système (III.11) est dit stable indépendamment de la taille du retard si, quelle que soit la valeur du retard $\tau > 0$ (constant), il reste stable. Dans ce cas, l'analyse de la stabilité peut ainsi directement porter sur le système d'origine. Pour mieux comprendre cette particularité, considérons le système (III.11), pour lequel le choix classique de la fonctionnelle de Lyapunov-Krasovskii est le suivant [41] :

$$V(t) = x(t)^T P x(t) + \int_{-\tau}^{0} x(t+\theta)^T S x(t+\theta) d\theta$$

où $0 < P = P^T$ et $0 < S = S^T$ sont des matrices symétriques définies positives (garantissant ainsi que la fonctionnelle de V (t) est définie positive). La stabilité asymptotique du système est alors garantie par la proposition suivante (voir [37], [49]) :

Proposition

Le système (III.11) est asymptotiquement stable $\forall\, \tau \geq 0$, s'il existe deux matrices symétriques et définies positives P et S telles que :

$$\begin{bmatrix} A^T P + PA + S & PA_d \\ A_d^T P & -S \end{bmatrix} < 0 \qquad (III.12)$$

Soit satisfaite.

Remarques

En respect de la définition de la stabilité indépendante du retard, la matrice A doit être stable au sens de Hurwitz.

Le retard τ n'apparaît pas explicitement dans la condition LMI (III.12), ce qui justifie l'appellation de critère indépendant de la taille du retard.

III.5 Conclusion

Dans ce chapitre, nous avons tout d'abord présenté le contexte général des systèmes à retards, mentionné leur apparition fréquente dans les processus physiques, et décrit certains modèles de systèmes à retards ainsi que ceux des retards en eux-mêmes, parmi les modèles les plus fréquemment rencontrés dans la littérature. Puis, nous nous sommes attachés à la présentation de quelques techniques permettant d'analyser la stabilité de tels systèmes.

Nous nous sommes intéressés plus particulièrement à la classe de systèmes linéaires avec un retard discret sur l'état qui est la classe la plus fréquemment rencontrée et celle qui nous intéresse dans la suite de notre travail. Nous avons ainsi pu introduire les outils qui nous seront nécessaires pour mener à bien les objectifs de notre application : le contrôle des flux TCP et le contrôle de congestion d'un routeur.

Chapitre IV

Régulation du protocole TCP pour le contrôle de congestion

Nous revenons maintenant sur la problématique introduite au Chapitre II concernant le phénomène de congestion d'un routeur dans un réseau de communication. Nous proposons, dans cette quatrième partie, d'utiliser la méthodologie développée au cours du dernier chapitre afin et réguler le protocole TCP. Une telle régulation de trafic sera mise en œuvre par un mécanisme d'*Active Queue Management* (AQM), implanté au niveau du routeur. Son rôle sera non seulement de réguler la congestion mais aussi de préserver un certain niveau de *Qualité de Service* (QoS).

IV.1 Introduction

Comme cela a été mentionné auparavant, nous nous sommes intéressés dans ce mémoire au problème de partage d'un lien de communication traversé par différentes sources distantes.

Comme nous l'avons déjà mentionné au chapitre II, beaucoup de travaux de recherche se sont intéressé par l'utilisation de la théorie de commande pour le développement d'algorithmes AQM plus efficaces. Cela a été rendu possible après le développement du modèle dynamique du comportement de TCP par *Misra et al* [23], [24]. Grace à ce modèle, la problématique de régulation de la file d'attente au niveau d'un routeur a été reformulée en un problème purement automatique. De nombreux chercheurs se sont intéressés par la suite à cette nouvelle problématique, exploitant les différents outils de l'automatique. La plupart de ces travaux se sont focalisés sur la régulation de la file d'attente par diverses méthodes : les régulateurs Proportionnel P et Proportionnel Intégral PI pour calculer la probabilité de perte de paquets $p(t)$ [21], [22], [23], [24], les correcteurs fréquentiels [25], l'analyse non linéaire [26], la commande robuste [27] ou encore la commande prédictive [28]. Nous allons utiliser

dans ce qui suit l'approche des systèmes à retard et plus précisément l'approche de *Lyapunov-Krasovskii* et résoudre des critères de stabilisation en inégalités matricielles linéaires (LMIs) pour trouver des lois de commande par retour d'état pour la régulation de la file d'attente au niveau d'un routeur et ainsi éviter la saturation de son buffer et améliorer la qualité de service pour les communications dans le réseau. Cette problématique est représentée, de façon schématique, sur la Figure IV.1.

Nous considérons ici une topologie d'un seul routeur recevant *N* flux TCP homogènes, c'est-à-dire toutes les sources ont le même temps de communication : le même RTT et le même taux d'envoi W:

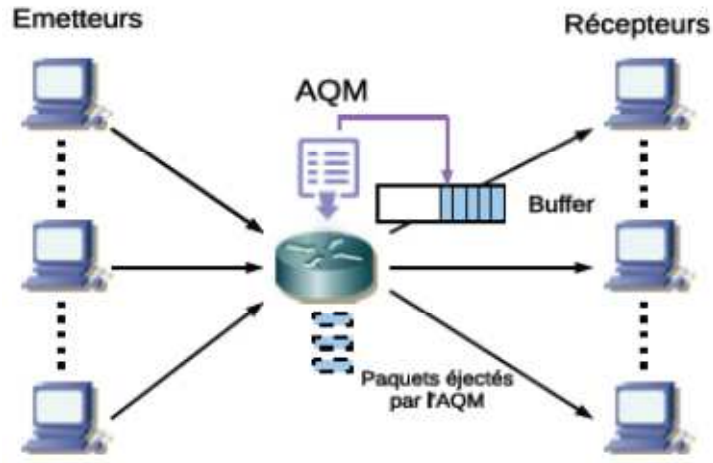

Fig. IV.1 : Topologie étudiée.

IV.2 Contrôle de congestion d'un routeur

IV.2.1 Modélisation

Nous rappelons ici le modèle dynamique de l'évolution de la fenêtre de congestion d'une source TCP, introduit plus tôt au Chapitre II :

$$\dot{w}(t) = \frac{1}{R(t)} - \frac{w(t)}{2}\frac{w\big(t-R(t)\big)}{R\big(t-R(t)\big)}p\big(t-R(t)\big) \qquad (IV.1.a)$$

$$(IV.1)$$

$$\dot{q} = \begin{cases} -C + \frac{N(t)}{R(t)}w(t), & q > 0 \\ max\left\{0, -C + \frac{N(t)}{R(t)}w(t)\right\}, & q = 0 \end{cases} \qquad (IV.1.b)$$

Ou $\dot{x}$ représente la dérivée par rapport au temps et :

- $w \triangleq$ Taille moyenne de la fenêtre de congestion TCP (paquets)
- $q \triangleq$ Taille moyenne de la file d'attente au niveau du routeur (paquets)
- $R(t) \triangleq$ Round Trip Time (RTT) $= \frac{q(t)}{C} + T_p$ (secondes)
- $C \triangleq$ Capacité de la connexion au niveau du routeur (parquets/secondes)
- $N \triangleq$ Nombre de connexions TCP
- $P \triangleq$ Probabilité d'éjecter un paquet, mesure de congestion
- $T_p \triangleq$ Délai de propagation (secondes)

Il s'agit d'un modèle dynamique fluide du comportement du TCP qui a été développé en utilisant des équations stochastiques et décrit par ces équations différentielles non linéaires couplées.

Notre objectif est de réguler la taille de file d'attente $q(t)$ ainsi que le débit des flux $w(t)$ autour d'un point d'équilibre. Il convient pour cela de définir ce point d'équilibre du modèle non linéaire (IV.1).

Connaissant les paramètres du réseau : le nombre de connexions TCP (N) , la capacité de connexion au niveau du routeur (C), et le délai de propagation T_p , nous définissons le point d'équilibre suivant (w_0, q_0, p_0) :

$$\begin{cases} \dot{w} = 0 \ \Rightarrow w_0^2 p_0 = 2 \\ \dot{q} = 0 \Rightarrow w_0 = \frac{R_0 C}{N}, R_0 = \frac{q_0}{C} + T_p \end{cases} \qquad (IV.2)$$

Le modèle dynamique du comportement de TCP peut être approximer autour de ce point d'équilibre par le système linéaire à retard suivant : [24]

$$\begin{cases} \delta\dot{w}(t) = -\frac{N}{R_0^2 C}\left(\delta w(t) - \delta w(t - R_0)\right) \\ -\frac{1}{R_0^2 C}\left(\delta q(t) - \delta q(t - R_0)\right) - \frac{R_0 C^2}{2N^2}\delta p(t - R_0) \\ \delta\dot{q}(t) = \frac{N}{R_0}\delta w(t) - \frac{1}{R_0}\delta q(t) \end{cases} \qquad (IV.3)$$

Avec $\delta w(t) = w - w_0$, $\delta q = q - q_0$, $\delta p = p - p_0$ représentent les perturbations de $w(t), q(t), et\ p(t)$ autour du point de fonctionnement.

Ce modèle peut être réécrit comme suit :

$$\begin{cases} \dot{x}(t) = Ax(t) + A_d x(t - R_0) + Bu(t - R_0) \\ y(t) = Cx(t) \\ x_0(\theta) = \emptyset(\theta), \ \ avec \ \ \theta \in [-R_0, 0] \end{cases} \quad \text{(IV.4)}$$

Avec

$$A = \begin{bmatrix} -\frac{N}{R_0^2 C} & -\frac{1}{R_0^2 C} \\ \frac{N}{R_0} & -\frac{1}{R_0} \end{bmatrix}, \quad A_d = \begin{bmatrix} -\frac{N}{R_0^2 C} & \frac{1}{R_0^2 C} \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -\frac{R_0 C^2}{2N^2} \\ 0 \end{bmatrix}, \quad C = [0 \ \ 1]$$

$x(t) = [\delta w(t) \quad \delta q(t)]^T$ est le vecteur état et $u(t) = \delta p(t)$ est l'entrée de commande.

Donc finalement, l'étude de la stabilité et la régulation de la longueur de la file d'attente $q(t)$ au niveau d'un routeur revient à étudier un système linéaire avec un retard discret sur l'état et sur l'entrée. Nous allons considérer le cas ou le retard est connu et constant.

Pour cela, nous allons utiliser la méthode de *Lyapunov-Krasovskii* déjà introduite dans le chapitre précédent. Cette dernière est une méthode effective et pratique parce qu'elle nous fournit des critères de stabilisation en inégalités matricielles linéaires (LMI) faciles à résoudre.

IV.2.2 Stabilisation par l'approche des systèmes à retard

Nous voulons stabiliser l'état $x(t) = [\delta w(t) \quad \delta q(t)]^T$ du système (IV.4) autour du point de fonctionnement désiré afin d'assurer une certaine qualité de service pour les communications réseaux. En d'autres termes, éviter la saturation du buffer du routeur et la congestion qui s'en suit.

Pour cela, un régulateur doit être synthétisé. Nous allons utiliser une synthèse par retour d'état :

$$u(t) = Kx(t) \tag{IV.5}$$

pour la commande du système (IV.4) (K est une matrice gain constante). Ce régulateur représente notre AQM.

En remplaçant $u(t)$ par $Kx(t)$ dans (IV.4) nous obtiendrons le système en boucle fermée suivant :

$$\begin{cases} \dot{x}(t) = Ax(t) + \tilde{A}_d x(t - R_0) \\ y(t) = Cx(t) \\ x_0(\theta) = \emptyset(\theta), \ \ avec \ \ \theta \in [-R_0, 0] \end{cases} \tag{IV.6}$$

Avec $\tilde{A}_d = (A_d + BK)$.

Proposition [37]

S'il existe des matrices symétriques définies positives R, S et une matrice $Z \in \mathbb{R}^{mxn}$ tel que :

$$\begin{bmatrix} RA^T + AR + S & A_d R + BZ \\ RA_d^T + Z^T B^T & -S \end{bmatrix} < 0 \tag{IV.7}$$

Alors le système (IV.4) est stabilisé par la commande (IV.5) avec le gain

$$K = ZR^{-1} \tag{IV.8}$$

IV.2.3 Exemple numérique

Pour exposer les résultats établis dans le paragraphe précédent, nous présentons maintenant un exemple numérique. Nous considérons un réseau de communication simple, constitué de N sources homogènes de trafics émettant des données vers leurs destinataires respectifs à travers un routeur de capacité C comme illustré dans la figure (IV.1).

Prenons comme exemple le cas ou $N = 100$ sources homogènes, $C = 1250$ $(paquets/seconde)$ équivalent à une connexion de 5 Mbps avec une taille moyenne des paquets de 500 octets, un délai de propagation $T_p = 0.092$ $secondes$. La taille

du buffer du routeur est $q_{max} = 300\ paquets$. Pour éviter la congestion il faut maintenir la taille de la file d'attente au niveau du routeur à une valeur désirée tel que le buffer du routeur ne soit ni vide ni saturé, l'idéal est de garder le buffer à moitié plein c'est-à-dire $q_d = q_0 = 150\ paquets$.

De l'équation (IV.2) nous aurons $R_0 = 0.212\ secondes$, $W_0 = 2.65\ paquets$ et $p_0 = 0.2848$.

Nous auront finalement

$$A = \begin{bmatrix} -1.7800 & -0.0178 \\ 471.6981 & -4.7170 \end{bmatrix}, \quad A_d = \begin{bmatrix} -1.7800 & 0.0178 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -16.5625 \\ 0 \end{bmatrix}$$

En utilisant Matlab pour la résolution de l'inégalité matricielle linéaire (LMI) (IV.7) nous obtiendrons les résultats suivants :

$$R = 10^8 * \begin{bmatrix} 0.0005 & 0.0285 \\ 0.0285 & 2.8549 \end{bmatrix}, \quad S = 10^5 * \begin{bmatrix} 1.3997 & 0 \\ 0 & 01.3997 \end{bmatrix},$$

$$Z = 10^3 * [-2.3155 \quad 0.0319]$$

Les matrices R et S sont symétriques et définies positives donc notre système est stabilisé par le gain de retour d'état suivant

$$K = ZR^{-1} = [-0.1075 \quad 0.0011]$$

Les figures ci-dessous montrent la simulation sous Matlab Simulink de l'évolution de la taille de la file d'attente au niveau du routeur et la fenêtre de congestion au niveau des sources avec le gain $K = [-0.1075 \quad 0.0011]$

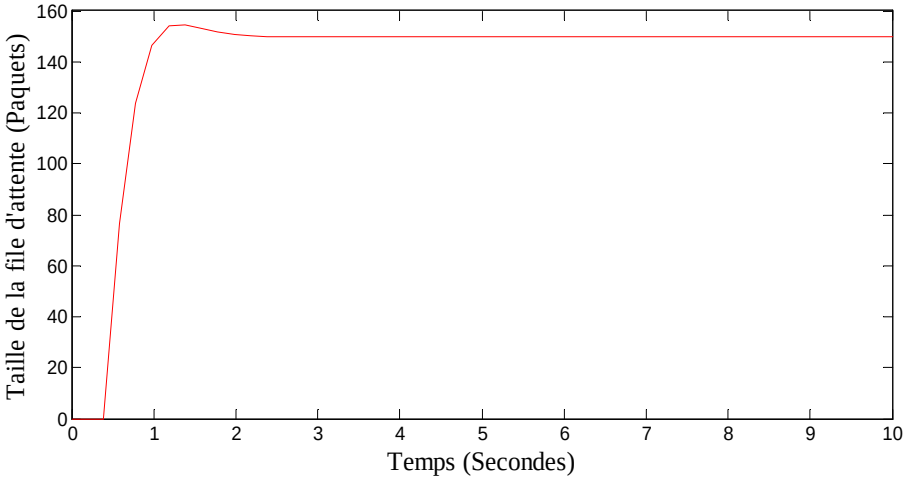

Fig. IV.2-a : Evolution de la taille de la file du d'attente au niveau du routeur

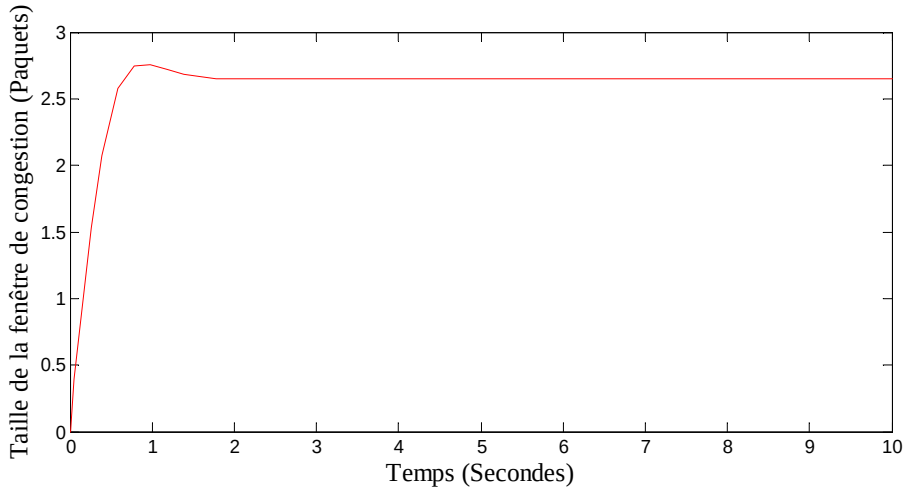

Fig. IV.2-b : Evolution de la taille de la fenêtre de congestion au niveau des sources

Nous pouvons remarquer de ces figures que notre régulateur par retour d'état assure la stabilité et la convergence vers l'état désiré, c'est-à-dire, une taille de la file d'attente $q_d = q_0 = 150\, paquets$ et une fenêtre de congestion de $w_0 = 2.65\, paquets$. Le régime permanant quant à lui est atteint au bout d'environs 2 secondes. Notons aussi que la taille de la fenêtre de congestion commence a progressé dés $t = 0\, seondes$, par contre la taille de la file d'attente nécessite un certain délai, chose très logique puisque les paquets envoyés des sources nécessitent un temps de propagation pour atteindre le buffer du routeur.

Maintenant que nous avons montré par un exemple numérique l'efficacité de notre régulateur issu de la théorie des systèmes à retard pour la régulation du trafic TCP au niveau d'un routeur, il faut comparer ces résultats avec ceux d'autres mécanismes AQM afin d'en tirer des conclusions. Pour cela, nous allons comparer notre régulateur avec l'AQM bien connu du RED (*Random Early Detection*) [10] que nous avons déjà présenté brièvement au chapitre 2. Notre choix pour le RED est du a sa popularité et aux nombreux travaux de recherche déjà réalisés autour de lui. Comme il est parmi les premiers mécanismes AQM inventés, il est souvent utilisé comme un cas d'étude comparative. Une étude assez intéressante a été réalisée dans [50] sur ce mécanisme d'AQM, son réglage (tuning), et de nombreuses simulations intéressantes ont été faites. Nous avons tiré les paramètres du RED de cette étude.

IV.2.4 Etude comparative avec l'algorithme RED (Random Early Detection)

Le principe de l'algorithme du *Random Early Detection* (RED) est d'anticiper la saturation du buffer. Ainsi, plutôt que d'attendre le remplissage complet du buffer et une saturation du réseau, l'idée suggérée est d'éjecter les paquets entrant au routeur avec une certaine probabilité pour indiquer à la source correspondante de réduire son taux d'envoi. Ce taux de perte est une fonction croissante de la taille moyenne de la file d'attente du buffer (Figure IV.3) et s'exprime par :

$$p(\bar{b}) = \begin{cases} 0, & 0 \le \bar{b} < b_{min} \\ \dfrac{\bar{b} - b_{min}}{b_{max} - b_{min}} p_{max} & b_{min} \le \bar{b} \le b_{max} \\ 1, & b_{max} < \bar{b} \end{cases} \qquad (IV.9)$$

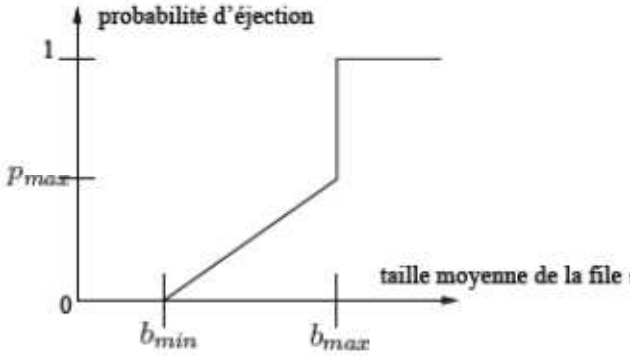

Fig. IV.3 : Le mécanisme de Random Early Detection (RED).

Où $\bar{b}$ est la longueur moyenne de la file d'attente, b_{min}, b_{max} et p_{max} sont des paramètres à configurer. Donc si la file est petite ($\bar{b} < b_{min}$), chaque paquet entrant est accepté et mis en mémoire tampon. Tandis qu'au delà d'un certain seuil, une fraction des flux entrants, proportionnelle à la mémoire utilisée (en moyenne) $\bar{b}$, est rejetée. Enfin, au delà de la limite b_{max} tous les paquets supplémentaires sont refusés. Du point de vue commande, le RED peut être assimilé à un filtre passe-bas (calcul de la taille de file moyenne) avec une action proportionnelle.

Nous allons utiliser les paramètres de RED suivants : [50]

$$b_{min} = 150; \quad b_{max} = 200; \quad p_{max} = 0.1$$

Nous allons faire des simulations pour 4 différentes valeurs de N pour mieux comparer le comportement de ces deux régulateurs.

Pour $N = 40$:

Nous prenons toujours $C = 1250\ (paquets/seconde)$, un délai de propagation $T_p = 0.092 secondes$. La taille du buffer du routeur est $q_{max} = 300\ paquets.$ $q_d = q_0 = 150\ paquets.$

De l'équation (IV.2) nous aurons $R_0 = 0.212\ secondes$, $w_0 = 6.62\ paquets$ et $p_0 = 0.0456.$

Et nous aurons finalement

$$A = \begin{bmatrix} -0.7120 & -0.0178 \\ 188.6792 & -4.7170 \end{bmatrix}, \quad A_d = \begin{bmatrix} -0.7120 & 0.0178 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -103.5156 \\ 0 \end{bmatrix}$$

La résolution par Matlab de l'inégalité matricielle linéaire (LMI) (IV.7) nous donne les résultats suivants :

$$R = 10^8 * \begin{bmatrix} 0.0032 & 0.0977 \\ 0.0977 & 3.9071 \end{bmatrix}, \quad S = 10^5 * \begin{bmatrix} 4.0014 & 0 \\ 0 & 4.0014 \end{bmatrix},$$

$$Z = [-506.9931 \quad 14.5866]$$

Les matrices R et S étant symétriques et définies positives donc notre système est stabilisé par le gain de retour d'état

$$K = ZR^{-1} = [-0.0069 \quad 0.0002]$$

Les figures ci-dessous montrent les résultats obtenus.

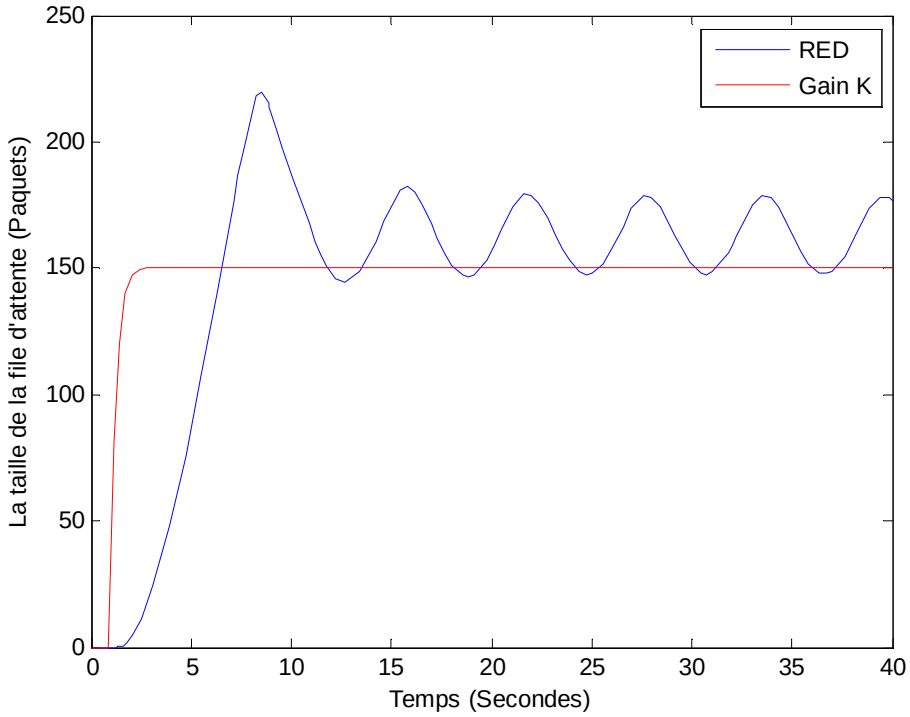

Fig. IV.3-a : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=40.

-Taille de la file d'attente-

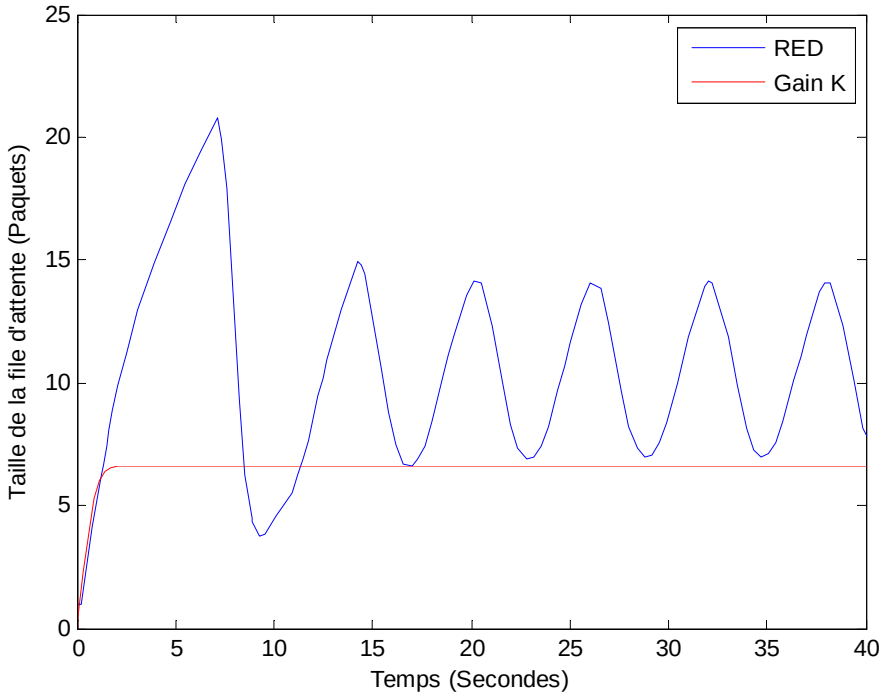

Fig. IV.3-b : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=40.
-Taille de la fenêtre de congestion-

Pour $N = 60$:

$C = 1250\ (paquets/seconde)$, le délai de propagation $T_p = 0.092\ secondes$. La taille du buffer du routeur est $q_{max} = 300\ paquets$. $q_d = q_0 = 150\ paquets$.

De l'équation (IV.2) nous aurons $R_0 = 0.212\ secondes$, $w_0 = 4.42\ paquets$ et $p_0 = 0.1024$.

et nous aurons

$$A = \begin{bmatrix} -1.0680 & -0.0178 \\ 283.0189 & -4.7170 \end{bmatrix}, \quad A_d = \begin{bmatrix} -1.0680 & 0.0178 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -46.0069 \\ 0 \end{bmatrix}$$

En utilisant Matlab pour la résolution de l'inégalité matricielle linéaire (LMI) (IV.7) nous obtiendrons les résultats suivants :

$$R = 10^8 * \begin{bmatrix} 0.0013 & 0.0539 \\ 0.0539 & 3.2355 \end{bmatrix}, \quad S = 10^5 * \begin{bmatrix} 2.3541 & 0 \\ 0 & 2.3541 \end{bmatrix},$$

$$Z = [-944.6976 \quad 19.3086]$$

Les matrices R et S sont symétriques et définies positives donc notre système est stabilisé par le gain de retour d'état

$$K = ZR^{-1} = [-0.0232 \quad 0.0004]$$

Les figures ci-dessous montrent les résultats obtenus.

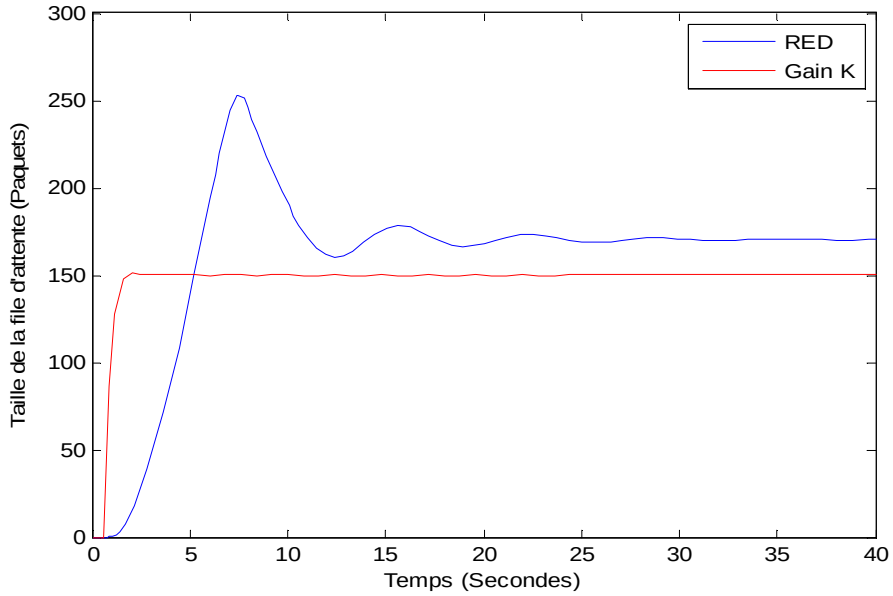

Fig. IV.4-a : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=60.
-Taille de la file d'attente-

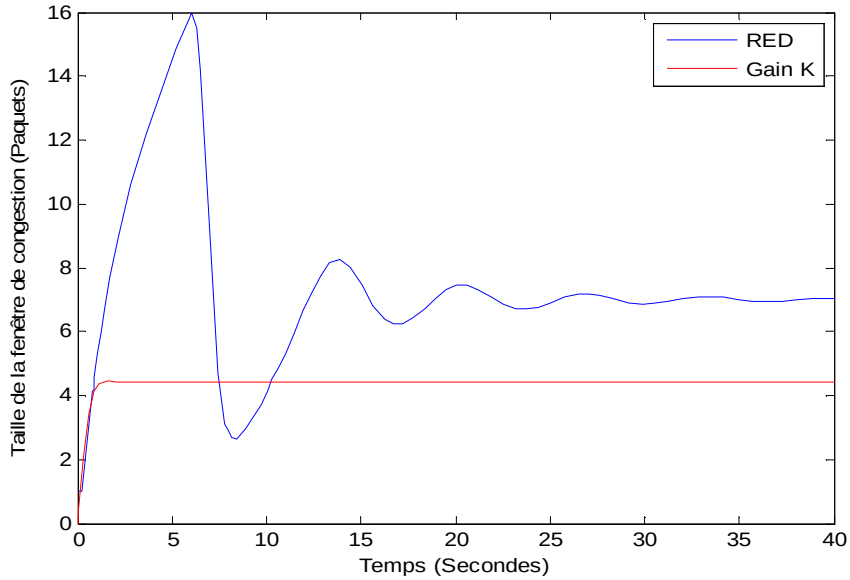

Fig. IV.4-b : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=60.
-Taille de la fenêtre de congestion-

Pour $N = 80$:

$C = 1250\ (paquets/seconde)$, le délai de propagation $T_p = 0.092\ secondes$. La taille du buffer du routeur est $q_{max} = 300\ paquets$. $q_d = q_0 = 150\ paquets$.

De l'équation (IV.2) nous aurons $R_0 = 0.212\ secondes$, $w_0 = 3.31\ paquets$ et $p_0 = 0.1825$.

et nous aurons

$$A = \begin{bmatrix} -1.4240 & -0.0178 \\ 377.3585 & -4.7170 \end{bmatrix}, \quad A_d = \begin{bmatrix} -1.4240 & 0.0178 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -25.8789 \\ 0 \end{bmatrix}$$

En utilisant Matlab pour la résolution de l'inégalité matricielle linéaire (LMI) (IV.7) on obtient les résultats suivants :

$$R = 10^8 * \begin{bmatrix} 0.0025 & 0.1237 \\ 0.1237 & 9.8997 \end{bmatrix}, \quad S = 10^5 * \begin{bmatrix} 5.7347 & 0 \\ 0 & 5.7347 \end{bmatrix},$$

$$Z = 10^3 * [-5.1387 \quad 0.0836]$$

Les matrices R et S sont symétriques et définies positives donc notre système est stabilisé par le gain de retour d'état

$$K = ZR^{-1} = [-0.0550 \quad 0.0007]$$

Les figures ci-dessous montrent les résultats obtenus.

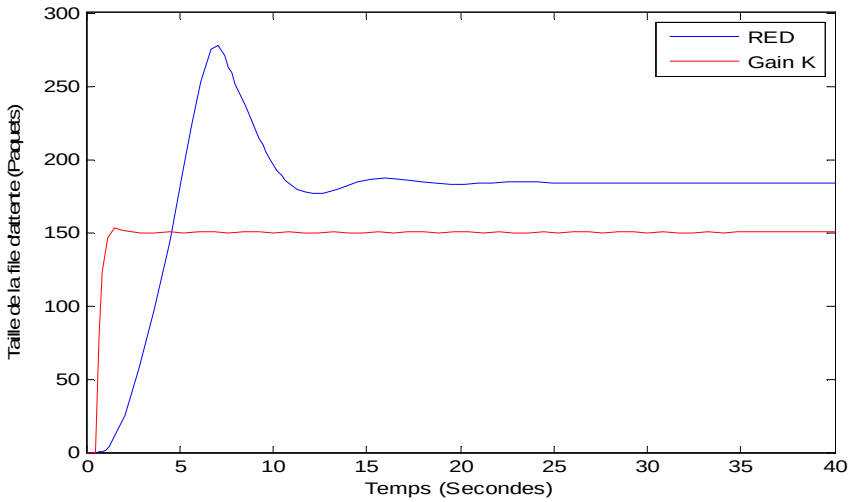

Fig. IV.5-a : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=80.

-Taille de la file d'attente-

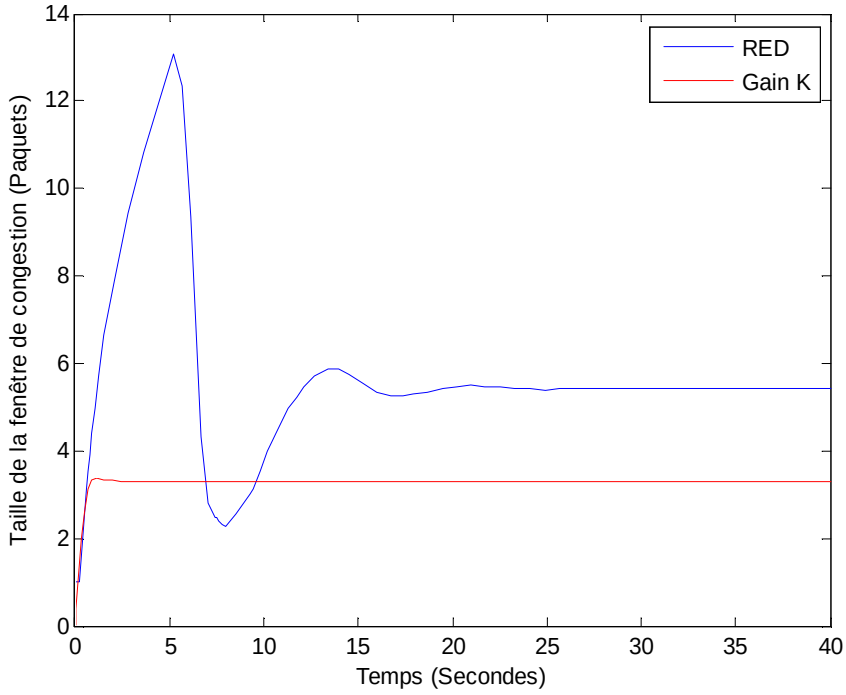

Fig. IV.5-b : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=80.
-Taille de la fenêtre de congestion-

Pour $N = 100$:

$C = 1250\ (paquets/seconde)$, le délai de propagation $T_p = 0.092\ secondes$. La taille du buffer du routeur est $q_{max} = 300\ paquets$. $q_d = q_0 = 150\ paquets$.

De l'équation (IV.2) nous aurons $R_0 = 0.212\ secondes$, $w_0 = 2.65\ paquets$ et $p_0 = 0.2848$.

Et nous aurons

$$A = \begin{bmatrix} -1.7800 & -0.0178 \\ 471.6981 & -4.7170 \end{bmatrix}, \quad A_d = \begin{bmatrix} -1.7800 & 0.0178 \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -16.5625 \\ 0 \end{bmatrix}$$

En utilisant Matlab pour la résolution de l'inégalité matricielle linéaire (LMI) (IV.7) nous obtiendrons les résultats suivants :

$$R = 10^8 * \begin{bmatrix} 0.0005 & 0.0285 \\ 0.0285 & 2.8549 \end{bmatrix}, \quad S = 10^5 * \begin{bmatrix} 1.3997 & 0 \\ 0 & 01.3997 \end{bmatrix},$$

$$Z = 10^3 * [-2.3155 \quad 0.0319]$$

Les matrices R et S sont symétriques et définies positives donc notre système est stabilisé par le gain de retour d'état

$$K = ZR^{-1} = [-0.1075 \quad 0.0011]$$

Les figures ci-dessous montrent les résultats obtenus.

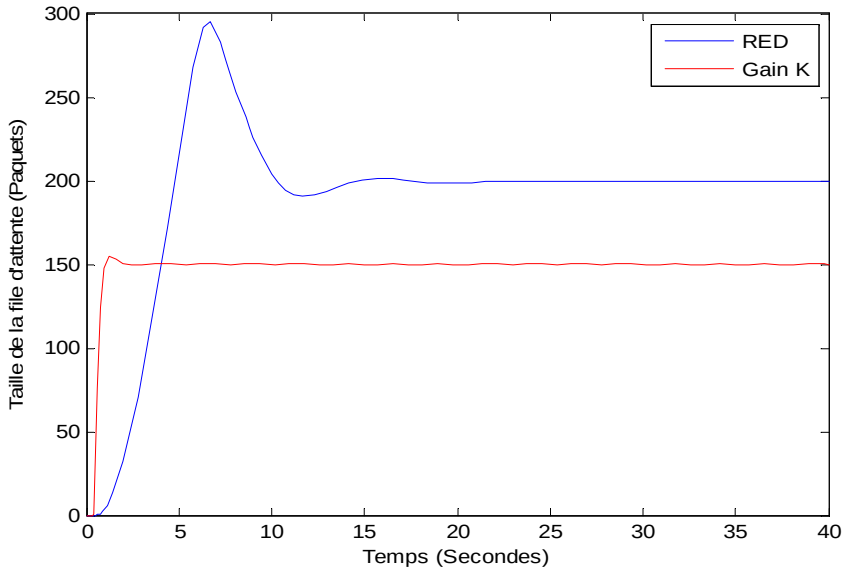

Fig. IV.6-a : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=100.

-Taille de la file d'attente-

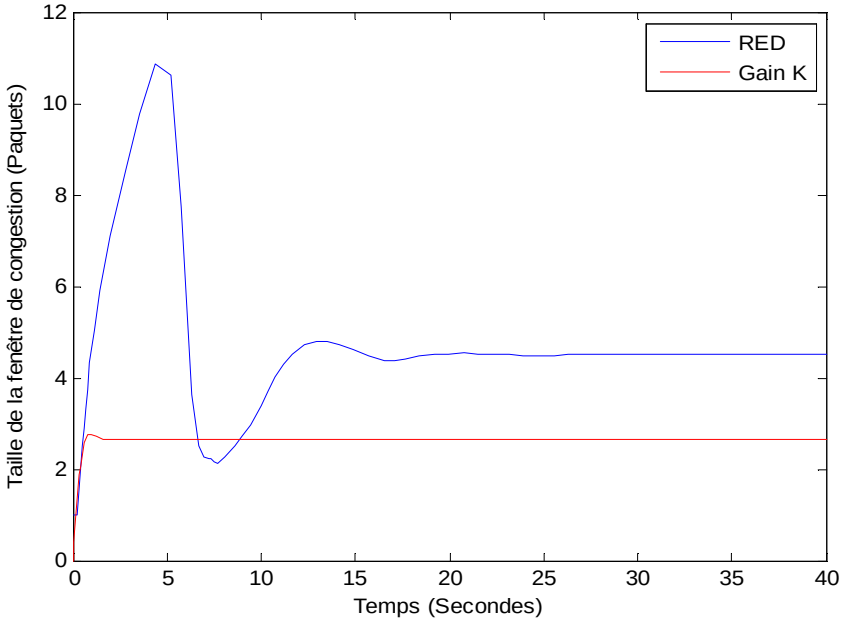

Fig. IV.6-b : Comparaison entre le régulateur par retour d'état et l'algorithme RED pour N=100.

-Taille de la fenêtre de congestion-

Remarques

Nous pouvons remarquer de ces différentes figures que notre régulateur par retour d'état et l'algorithme RED arrivent à stabiliser le système autour du point de fonctionnement mais il est clair que notre régulateur a un comportement bien meilleur pour les raisons suivantes :

-Le régulateur par retour d'état arrive au régime permanent beaucoup plus vite que le régulateur RED.
-Le régulateur par retour d'état a un petit dépassement (overshoot) par rapport au RED.
-Le régulateur RED représente des oscillations autour du point d'équilibre pour les petites valeurs de N, ces oscillations ont tendance à disparaitre quand N devient plus grand.
-Le régulateur RED ne stabilise pas le système à $q_0 = 150\, paquets$ contrairement au régulateur par retour d'état. En effet, le régulateur RED a tendance à stabiliser le système à une valeur qui dépond du nombre de sessions ouvertes$N$, par exemple pour $N = 40$ cette valeur est de 160 paquets pour devenir 200 paquets pour $N = 100$. Cela s'explique par le fait que pour le régulateur par retour d'état qui est un régulateur issu de l'automatique nous pouvons spécifier la valeur de la consigne, c'est-à-dire la valeur de la file d'attente désirée, ce qui n'est pas le cas pour le régulateur RED.

Finalement, nous pouvons conclure que notre régulateur issu de la théorie des systèmes à retard arrive à bien satisfaire les objectifs tracés, à savoir la régulation de la file d'attente au niveau d'un retour à une valeur désirée pour garantir ainsi une certaine qualité de service pour les comminations réseaux.

Néanmoins, notre régulateur n'est valable que pour les paramètres $N, C\ et\ R_0$ utilisés pour le calcul de la loi de commande du moment que les matrices $A, A_d\ et\ B$ du système dépendent de ces paramètres. Donc si un paramètre change, il va falloir calculer un autre gain K comme nous l'avons fait dans les premières simulations, mais ceci n'est pas pratique. Cela nous a poussé à penser à l'utilisation d'un modèle polytopique afin d'assurer la robustesse de la loi de commande.

IV.2.5 Approche robuste

Comme nous l'avons mentionné auparavant, les matrices $A, A_d \; et \; B$ dépendent des paramètres du système qui peuvent être incertains à cause des simplifications durant la modélisation, la linéarisation et des fois à cause des erreurs de mesure d'une part, et à cause de leurs variations au cours du temps d'autre part. Dans ce qui suit, on va considérer que la capacité C du routeur étant connue et fixe dans le temps ainsi que le délai de propagation T_p et par conséquent R_0 aussi. Notre attention se portera sur le nombre de sources N que nous supposerons variant.

En effet, le nombre d'utilisateurs sur un réseau varie sans cesse, donc il n'est plus commode de calculer une loi de commande pour un nombre fixe de N. L'idée est de supposer que le nombre de sources N peut varier entre deux valeurs données: $N_{min} \leq N \leq N_{max}$.

Nous allons réécrire le système (IV.4) comme suit :

$$\begin{cases} \dot{x}(t) = A(N)x(t) + A_d(N)x(t - R_0) + B(N)u(t - R_0) \\ \quad y(t) = Cx(t) \\ \quad x_0(\theta) = \emptyset(\theta), \;\; avec \;\; \theta \in [-R_0, 0] \end{cases} \tag{IV.10}$$

Avec

$$A = \begin{bmatrix} -\frac{N}{R_0^2 C} & -\frac{1}{R_0^2 C} \\ \frac{N}{R_0} & -\frac{1}{R_0} \end{bmatrix}, \quad A_d = \begin{bmatrix} -\frac{N}{R_0^2 C} & \frac{1}{R_0^2 C} \\ 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} -\frac{R_0 C^2}{2N^2} \\ 0 \end{bmatrix}, \quad C = [0 \;\; 1]$$

Avec l'approche polytopique, l'idée est d'assurer la stabilité pour un ensemble de systèmes. Comme nous avons supposé que $N \in [N_{min}\,,\; N_{max}]$, les matrices $A, A_d \; et \; B$ vont appartenir à un certain ensemble $\Omega = \{[A, A_d, B] \; avec \; N \in [N_{min}\,,\; N_{max}]\}$. Notre objectif est de trouver un AQM exprimé par un gain de retour K qui va assurer la stabilité pour le système (IV.10) pour toutes les matrices $A, A_d \; et \; B$ appartenant à Ω.

Nous remarquons que le terme N n'apparait pas linéairement dans la matrice B, donc l'ensemble Ω ainsi définit est non convexe. Une idée générale dans la commande robuste est de chercher un ensemble polytopique P qui inclut l'ensemble Ω. Si la stabilité de P est garantie alors la stabilité de Ω l'est aussi.

Pour créer le polytope P, on pose $\mathcal{N} = \frac{1}{N^2}$. On aura donc 2 paramètres variant, $N\ et\ \mathcal{N}$ et les matrices $A, A_d\ et\ B$ peuvent être exprimées comme suit :

$$A = \begin{bmatrix} -\frac{N}{R_0^2 C} & -\frac{1}{R_0^2 C} \\ \frac{N}{R_0} & -\frac{1}{R_0} \end{bmatrix}, \quad A_d = \begin{bmatrix} -\frac{N}{R_0^2 C} & \frac{1}{R_0^2 C} \\ 0 & 0 \end{bmatrix}, \quad B = \mathcal{N} \begin{bmatrix} -\frac{R_0 C^2}{2} \\ 0 \end{bmatrix}$$

Nous ne disposons désormais plus d'une seule représentation d'état du système, mais de 2^n (avec n le nombre de paramètres variant). Dans notre cas il y aura donc 4 représentations d'état ($n = 2$) et 4 inégalités matricielles linéaires à résoudre.

Proposition

Pour garantir la stabilité quadratique dans l'ensemble du polytope il faut trouver des matrices R et S symétriques et définies positives uniques et une matrice Z unique qui vérifie l'ensemble des 4 inégalités matricielles linéaires suivantes :

$$\begin{bmatrix} RA^{i^T} + A^i R + S & A_d^i R + B^i Z \\ RA_d^{i\ T} + Z^T B^{i^T} & -S \end{bmatrix} < 0 \qquad i = \{1, \dots, 4\} \qquad (IV.11)$$

Et le gain du retour d'état sera donné par l'expression suivante :

$$K = ZR^{-1}$$

Application numérique

Prenons toujours les données du cas étudié à savoir un routeur de capacité $C = 1250\ (paquets/seconde)$ équivalent à une connexion de 5 Mbps avec une taille moyenne des paquets de 500 octets, un délai de propagation $T_p = 0.092\ secondes$. La taille du buffer du routeur est $q_{max} = 300\ paquets$. La taille désirée est $q_d = q_0 = 150\ paquets$.

Quant au nombre de sources homogènes, nous prenons $N_{min} = 30\ et\ N_{max} = 80$.

$N \in [30,\ 80],\ donc\ \mathcal{N} \in [1.5625 * 10^{-4},\quad 0.0011]$.

Les 4 représentations d'état de notre système seront donc définîtes comme suit :

$$A^1 = \begin{bmatrix} -0.5340 & -0.0178 \\ 141.5094 & -4.7170 \end{bmatrix}, A_d^1 = \begin{bmatrix} -0.5340 & 0.0178 \\ 0 & 0 \end{bmatrix}, B^1 = \begin{bmatrix} -25.8789 \\ 0 \end{bmatrix}$$

$$A^2 = \begin{bmatrix} -0.5340 & -0.0178 \\ 141.5094 & -4.7170 \end{bmatrix}, A_d^2 = \begin{bmatrix} -0.5340 & 0.0178 \\ 0 & 0 \end{bmatrix}, B^2 = \begin{bmatrix} -182.1875 \\ 0 \end{bmatrix}$$

$$A^3 = \begin{bmatrix} -1.4240 & -0.0178 \\ 377.3585 & -4.7170 \end{bmatrix}, A_d^3 = \begin{bmatrix} -1.4240 & 0.0178 \\ 0 & 0 \end{bmatrix}, B^3 = \begin{bmatrix} -25.8789 \\ 0 \end{bmatrix}$$

$$A^4 = \begin{bmatrix} -1.4240 & -0.0178 \\ 377.3585 & -4.7170 \end{bmatrix}, A_d^4 = \begin{bmatrix} -1.4240 & 0.0178 \\ 0 & 0 \end{bmatrix}, B^4 = \begin{bmatrix} -182.1875 \\ 0 \end{bmatrix}$$

En utilisant Matlab pour la résolution de ces 4 inégalités matricielles linéaires (LMIs) nous obtiendrons les résultats suivants :

$$R = \begin{bmatrix} 0.0004 & 0.0120 \\ 0.0120 & 2.1289 \end{bmatrix},\quad S = \begin{bmatrix} 0.0005 & -0.0050 \\ -0.0050 & 6.3536 \end{bmatrix},$$

$$Z = 10^{-3} * [\,-0.0011 \quad 0.1607]$$

Les matrices R et S sont symétriques et définies positives donc notre système est stabilisé par le gain de retour d'état suivant

$$K = ZR^{-1} = [-0.0060 \quad 0.0001]$$

Maintenant nous allons faire des simulations avec ce gain K trouvé pour différentes valeurs de N afin de vérifier si notre régulateur ainsi conçu assure la robustesse par rapport aux variations du nombre de sessions. Les figures ci-dessous montrent les résultats de ces simulations :

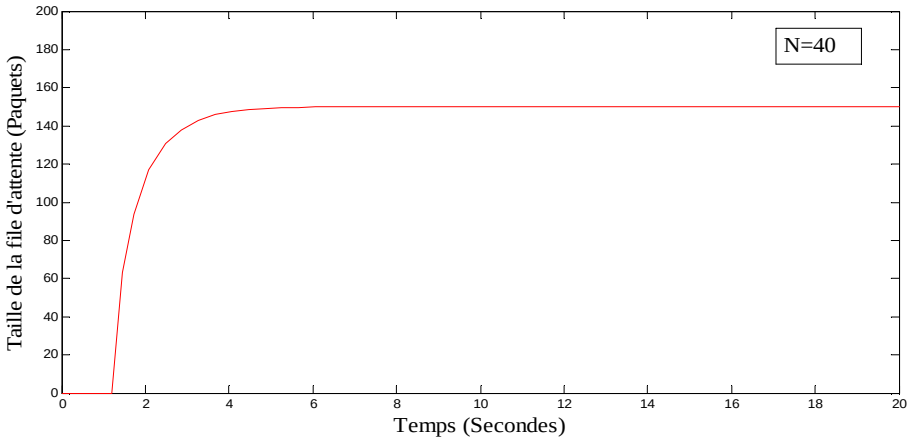

Fig. IV.7-a : Evolution de la taille de la file d'attente au niveau du routeur N=40

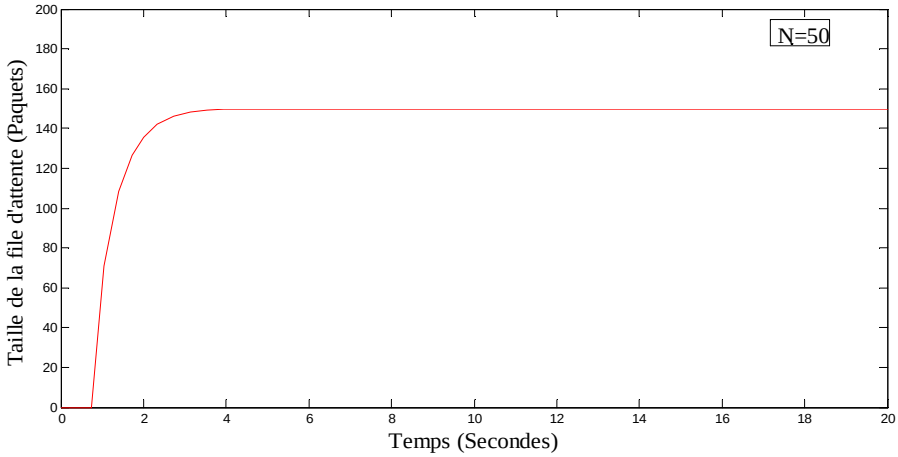

Fig. IV.7-b : Evolution de la taille de la file d'attente au niveau du routeur pour N=50

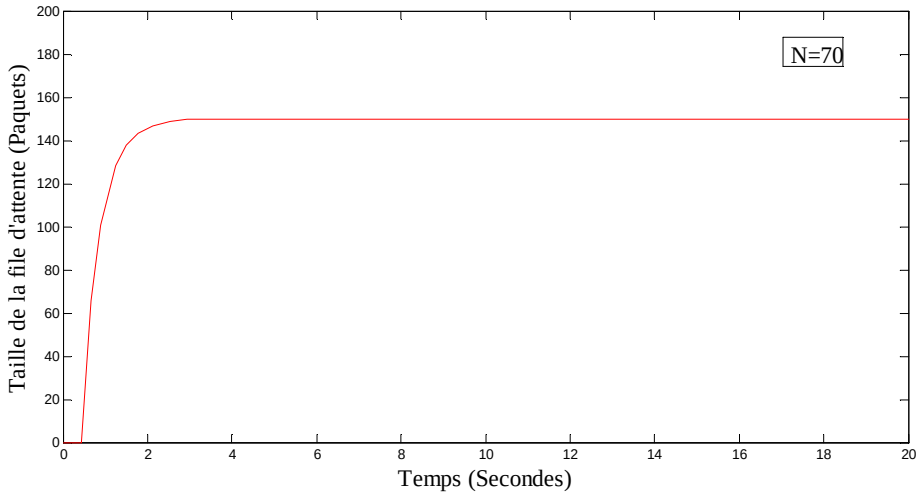

Fig. IV.7-c : Evolution de la taille de la file d'attente au niveau du routeur pour N=70

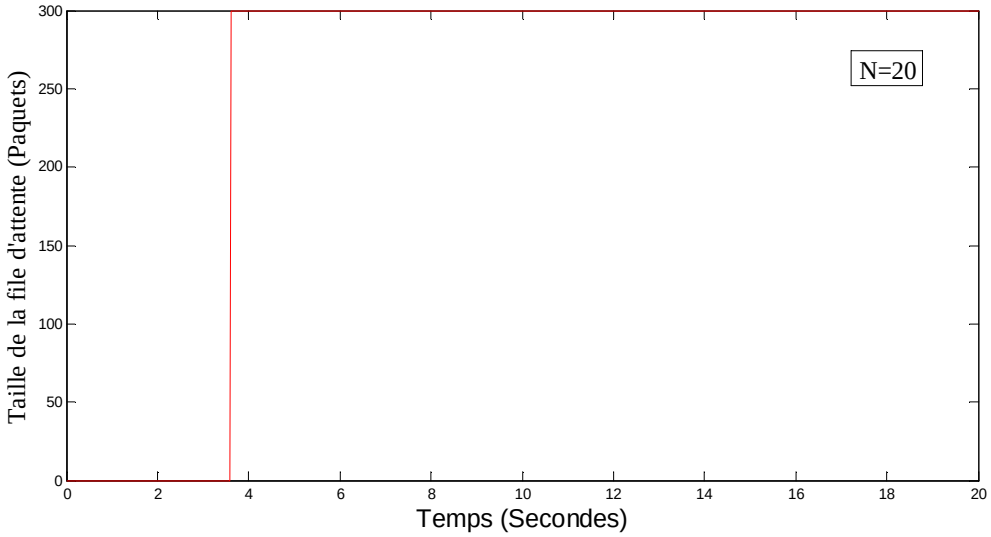

Fig. IV.7-d : Evolution de la taille de la file d'attente au niveau du routeur pou N=20

Remarques

Nous remarquons bien que notre régulateur par retour d'état assure la stabilité du système pour les valeurs de $N \in [30, 80]$, et ne satisfait pas cette stabilité pour $N = 20$ qui n'appartient pas aux valeurs admissibles de N, et nous remarquons dans ce cas la divergence du système qui s'exprime par la saturation du buffer au niveau du routeur à 300 paquets comme nous pouvons bien le voir sur la figure correspondante.

Maintenant nous allons voir le comportement de notre système pour une variation subite du nombre de sources N. Nous allons présenter 2 scénarios :

Scénario 1 :

A $t = 0\ s$ 60 sources commencent à envoyer leurs données au routeur et à $t = 40\ s$, 20 sources cessent subitement leurs émissions.

Scénario 2 :

A $t = 0\ s$ 60 sources commencent à envoyer leurs données au routeur et à $t = 40\ s$, 20 sources cessent subitement leurs envois. Et à $t = 70\ s$, 20 autres sources cessent aussi subitement leurs émissions.

Les résultats de la simulation sont donnés par les graphes suivants :

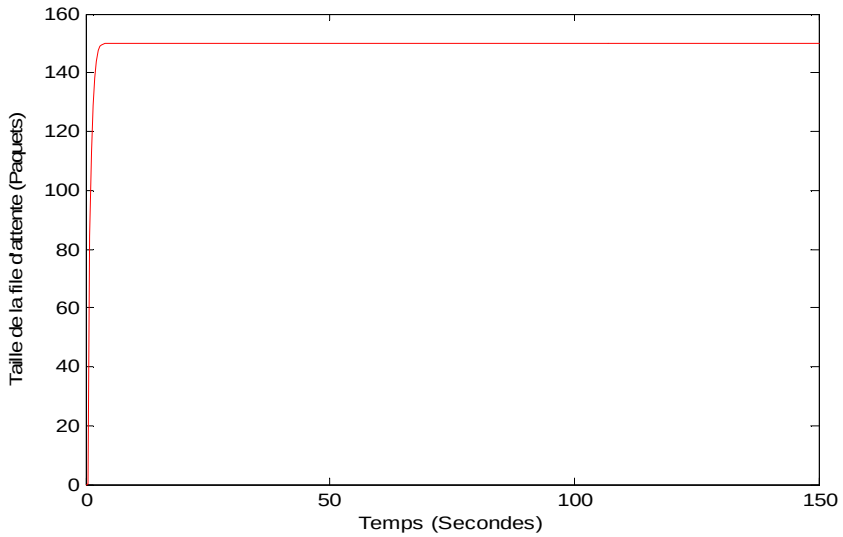

Fig. IV.8-a : Evolution de la taille de la file d'attente au niveau du routeur pour N passant de 60 à 40 à t=40 s

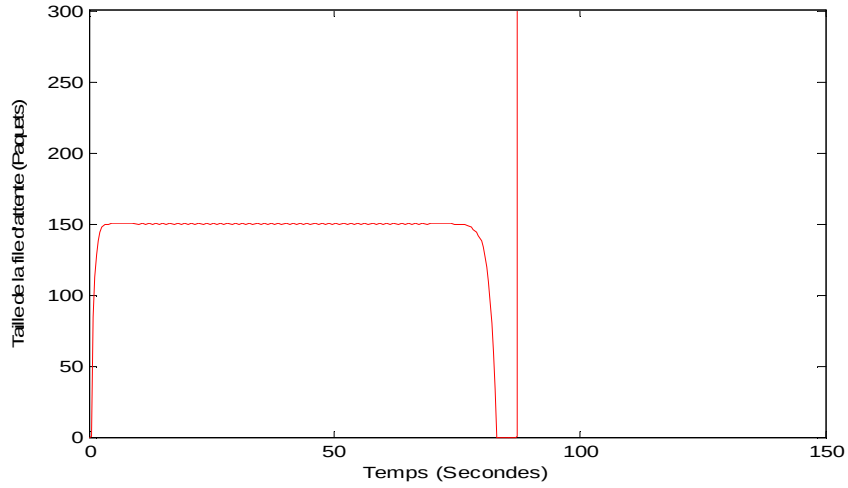

Fig. IV.8-b: Evolution de la taille de la file d'attente au niveau du routeur pour N passant de 60 à 40 à t=40 s et passant de 40 à 20 à t=70s

Remarques

Dans le premier graphe, nous pouvons remarquer que notre régulateur par retour d'état maintient la taille de la file d'attente au niveau du routeur à la valeur désirée c'est-à-dire 150 paquets malgré que 20 sources cessent d'envoyer leurs données à partir de $t = 40\ s$. Nous pouvons dire alors que notre régulateur n'est pas sensible à cette variation de N.

Dans le deuxième graphe, nous pouvons constater exactement la même chose à $t = 40\ s$, c'est-à-dire le maintien de la taille de la file d'attente à la valeur désirée quand les premières 20 sources cessent leurs envois, mais à $t = 70\ s$ le système devient instable et cela ce traduit par la saturation du buffer du routeur après que 20 autres sources cessent leurs envois (Ce qui rend $N = 20$ qui n'est pas inclus dans l'ensemble des valeurs admissibles de N).

Maintenant nous allons comparer le comportement de notre système avec le régulateur à retour d'état lors des variations subites du nombre de sources N avec celui de régulateur RED.Pour cela nous proposons le scénario suivant :

Scénario :
A $t = 0\ s$ 60 sources commencent à envoyer leurs données au routeur et à $t = 70\ s$, 20 sources cessent subitement leurs émissions pour le reprendre à partir de $t = 120\ s$.

Les résultats de la simulation sont donnés par les graphes suivants :

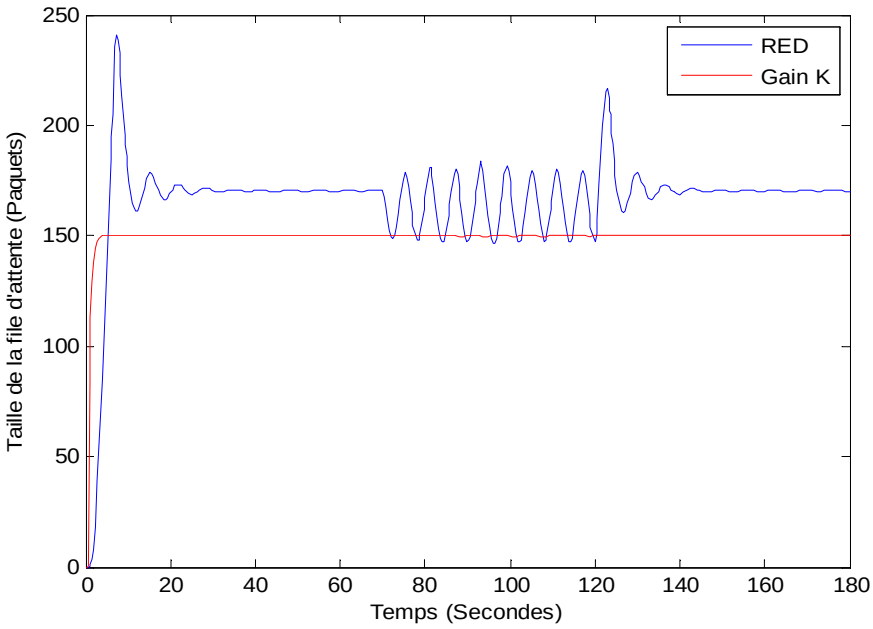

Fig. IV.9: Evolution de la taille de la file d'attente au niveau du routeur pour N passant de 60 à 40 à t=70 s et revenant de 40 à 60 à t=120s

Remarques

Nous pouvons voir clairement que le régulateur par retour d'état maintient la file d'attente à la valeur désirée, c'est-à-dire, 150 paquets bien que la valeur de N ait changé deux fois, en passant de 60 à 40 à t=70s et revenant à 60 à t=120s. Nous pouvons conclure alors que notre régulateur avec le gain calculé par l'approche robuste reste insensible aux variations du nombre de sources N pourvu que ce dernier soit compris dans l'intervalle des valeurs admissibles.

Par contre, le régulateur RED quant à lui, est sensible à ces variations de N. Nous remarquons bien qu'à l'instant t=70s la sortie du RED a changé de comportement et elle a commencé à avoir des oscillations (c'est le comportement du RED pour $N = 40$ sources comme nous l'avons déjà vu dans les simulations précédentes) ensuite à t=120s celle-ci à tendance à revenir à sa forme initiale (comportement du RED pour $N = 60$ sources).

IV.3 Conclusion

Nous avons montré dans ce dernier chapitre que la technique de commande par l'approche des systèmes à retard peut être appliquée avec succès aux problèmes de régulation du trafic et de la congestion d'un routeur par la synthèse d'un régulateur par retour d'état en utilisant la théorie de *Lyapunov-Krasovskii* et en résolvant des critères de stabilisation en inégalités matricielles linéaires (LMIs).

Afin de valider les résultats théoriques, des simulations ont été réalisées et des comparaisons ont été faites avec le régulateur du *Random Early Detection* (RED). Notre régulateur semble avoir un comportement bien meilleur que celui du RED, à savoir une meilleure stabilisation, pas d'oscillations, une régulation bien précise à la valeur désirée et finalement une insensibilité aux variations du nombre de sources en utilisant l'approche robuste.

Conclusion générale et perspectives

L'objectif général de ce mémoire est d'utiliser les techniques issues de l'automatique en général et de la théorie des systèmes à retard en particulier pour la synthèse de régulateurs pour le contrôle de congestion au niveau d'un routeur afin d'éviter les congestions et saturations et de garantir par conséquent une certaine qualité de service (QoS) pour les communications réseaux.

Ainsi, à partir des outils d'analyses développés au premier point, nous avons cherché à traiter les problèmes posés dans la partie applicative. Dans cette conclusion, nous revenons, chapitre par chapitre, sur les différents points abordés pour finalement dégager les futures pistes de travaux envisagées.

Dans un premier temps, nous avons présenté d'une manière générale au premier chapitre les réseaux informatiques et internet. Nous avons présenté les définitions et concepts nécessaires pour la suite de notre mémoire tels que le processus de communication de l'émetteur jusqu'au récepteur.

Dans le deuxième chapitre nous nous sommes focalisé sur le cadre applicatif de ce mémoire, à savoir le mécanisme de contrôle de congestion mis en place par TCP pour éviter la saturation des nœuds intermédiaires. A partir d'une vision bout en bout, le protocole TCP utilise le signal de perte de paquet, alors synonyme d'un phénomène de congestion dans le réseau, pour ajuster son débit d'émission. Nous avons montré qu'il est possible de jouer sur le taux de perte par le mécanisme d'AQM des routeurs pour agir indirectement sur la taille des flux TCP. Ceci nous conduit naturellement à la question qui nous intéresse qui consiste à formuler le problème du contrôle de congestion et de régulation du trafic en un problème de stabilité du point de vue de l'automatique. Nous avons présenté aussi dans ce chapitre le

modèle dynamique du comportement de TCP qui a ouvert les portes à l'utilisation de techniques issues de l'automatique pour la régulation du trafic TCP.

Au troisième chapitre, nous nous sommes intéressés à la présentation de la théorie des systèmes à retard. Nous avons fait quelques définitions concernant la stabilité ainsi que certains théorèmes usuels. Nous avons soulevé les points spécifiques qui nous sont utiles pour la suite du mémoire. Plus particulièrement, nous sommes intéressés par l'approche de *Lyapunov-Krasovskii* pour l'étude de stabilité des systèmes linéaires avec retard et les critères de stabilisation en termes d'inégalités matricielles linéaires (LMIs). Nous avons présenté aussi la modélisation incertaine des systèmes linéaires à retard en utilisant des polytopes pour faire face aux problèmes d'incertitudes et de variations des paramètres.

Dans le dernier chapitre, nous nous sommes focalisés sur la partie applicative qui fait également l'objet d'une part importante des travaux de ce mémoire. Nous avons proposé le modèle fluide non linéaire régissant l'évolution des débits d'émission de différentes sources TCP partageant un lien. Ce modèle est ensuite linéarisé autour d'un point d'équilibre. Le choix de ce point de fonctionnement est essentiel car il conditionne, à l'équilibre, la QdS de la communication. Puis, sur les bases de la méthodologie développée au précédent chapitre, nous avons construit un mécanisme d'AQM, réalisé à partir d'un retour d'état. Des simulations sous Matlab/Simulink et des comparaisons avec le comportement du *Random Early Detection* (RED) attestent de la validité de l'ensemble de la démarche et montrent que notre mécanisme de contrôle de congestion est en mesure de réguler le trafic TCP et assurer la QdS prescrite.

Comme perspectives, plusieurs points sont à approfondir et ouvrent la voie vers de nouveaux travaux de recherche :

Nous avons considéré le cas ou les sources sont homogènes, c’est-à-dire, elles ont un même retard (RTT) et la même taille de la fenêtre de congestion. Nous souhaitons que dans des travaux futurs, les sources seront considérées comme étant hétérogènes, c'est-à-dire, chacune aura son propre retard et sa propre taille de fenêtre de congestion, ça d’une part. D’autre part, les autres paramètres du réseau sont aussi incertains et peuvent être variant dans le temps, donc ça serait intéressant de les inclure dans l’étude polytopique. En parlant de retard, nous avons considéré le cas ou le retard est connu et constant, donc le problème revenait à l’étude d’un système linéaire avec un retard constant, mais dans le cas ou le retard est variant, une étude des systèmes à retards variant dans le temps serait nécessaire afin de minimiser le pessimisme de l’analyse.

Bibliographie

[1] http://www.cisco.com/web/learning/netacad/index.html

[2] W. R. Stevens. *TCP/IP règles et protocoles*. Addison-Wesley France, Paris, France, 1995. Titre original : TCP/IP Illustrated : the protocols, Addison-Wesley Publishing Company, 1994.

[3] A. Tanenbaum. *Computer networks*. Prentice-Hall, Englewood Cliffs, USA, 1989.

[4] J. Postel. Transmission control protocol - DARPA internet program protocol specification. RFC 793, September 1981.

[5] V. Jacobson. Congestion avoidance and control. In *ACM SIGCOMM*, pages 314–329, Stanford, CA, August 1988.

[6] V. Jacobson. Modified, TCP congestion avoidance algorithm,. end2end-interest mailing list, April 1990.

[7] M. Allman, V. Paxson, and W. Stevens. TCP congestion control. RFC 2581, April 1999. URL : http ://www.ietf.org/rfc.html.

[8] B. Braden, D. Clark, and J. Crowcroft. Recommendations on queue management and congestion avoidance in the internet. RFC 2309, April 1998.

[9] S. Ryu, C. Rump, and C. Qiao. Advances in active queue management (AQM) based TCP congestion control. *Telecommunication Systems*, 4 :317–351, 2004.

[10] S. Floyd and V. Jacobson, “Random early detection gateways for congestion avoidance,” *IEEE/ACM Transactions on Networking*, vol. 1,pp. 397–413, Aug. 1993.

[11] S. Athuraliya, D. Lapsley, and S. Low, “An enhanced random early marking algorithm for internet flow control,” in *IEEE INFOCOM*, pp. 1425–1434, Dec. 2000.

[12] W. e. a. Feng, “Blue: A new class of active queue management algorithms,” University of Michigan, Tech. Rep. CSE-TR-387-99, 1999.

[13] S. Kunniyur and R. Srikant, “Analysis and design of an adaptive virtual queue (avq) algorithm for active queue management,” in *Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM)*, San Diego, CA, USA, pp. 123–134, Aug 2001.

[14] M. Christiansen, K. Jeffay, D. Ott, and F.D. Smith, “Tuning RED for web traffic,” in *Proceedings of ACM/SIGCOMM*, 2000.

[15] M. May, T. Bonald, JC. Bolot, “Analytic Evaluation of RED Performance,” in *Proceedings of IEEE/INFOCOM*, 2000.

[16] Teunis J. Ott, T. V. Lakshman, L. H.Wong, "SRED: Stabilized RED," in *Proceedings of IEEE/INFOCOM*, 1999.

[17] W. Feng, D. Kandlur, D. Saha, and K. Shin, "Blue: A New Class of Active Queue Management Algorithms," Tech. Rep., UM CSE-TR-387-99, 1999.
[18] D. Lin and R. Morris, "Dynamics of random early detection," in *Proceedings of ACM/SIGCOMM*, 1997.

[19] W. Feng, Dilip D. Kandlur, Debanjan Saha, and Kang G. Shin, "A Self- Configuring RED Gateway," in *Proceedings of IEEE/INFOCOM*, 1999.

[20] V. Firoiu and M. Borden, "A Study of Active Queue Management for Congestion Control," in *Proceedings of IEEE/INFOCOM*, 2000.

[21] C. V. Hollot, V. Misra, D Towsley, and W. Gong. On designing improved controllers for aqm routers supporting TCP flows. In *IEEE INFOCOM*, volume 3, pages 1726–1734, April 2001.

[22] C. V. Hollot, V. Misra, D Towsley, and W. Gong. Analysis and design of controllers for AQM routers supporting TCP flows. *IEEE Trans. on Automat. Control*, 47 :945–959, June 2002.

[23] V. Misra, W. Gong, and D. Towsley. Stochastic differential equation modeling and analysis of TCP windowsize behavior. Technical Report ECE-TR-CCS-99-10-01, University of Massachusetts, October 1999.

[24] V. Misra, W. Gong, and D Towsley. Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED. In *ACM SIGCOMM*, pages 151–160, August 2000.

[25] D. Agrawal and F. Granelli. Redesigning an active queue management system. In *IEEE Globecom*, volume 2, pages 702–706, December 2004.

[26] C. V. Hollot and Y. Chait. Nonlinear stability analysis for a class of TCP/AQM networks. In *IEEE Conference on Decision and Control*, pages 2309–2314, December 2001.

[27] S. Manfredi, M. di Bernardo, and F. Garofalo. Robust output feedback active queue management control in TCP networks. In *IEEE Conference on Decision and Control*, pages 1004–1009, December 2004.

[28] K. B. Kim. Design of feedback controls supporting TCP based on the state space approach. *IEEE Trans. on Automat. Control*, 51 (7), July 2006.

[29] Dugard, L. and Verriest, E. (1997). Stability and control of time-delay systems. LNCIS 228 of Lecture Notes on Control and Information Sciences, Berling-Verlag, New York.

[30] Kolmanovskii, V., Niculescu, S.I., and Richard, J.P. On the lyapunov-krasovskii functionals for stability analysis of linear delay systems. Int. J. Control, 72, 374_384, 1999

[31] JP. Richard, Time-delay systems: an overview of some recent advances and open problems, Automatica 39 pages1667 – 1694, 2003.

[32] Ivanescu. D. (2000) On the stability, stabilization of Time-delay Systems : Theory and Applications (in French) Phd. Thesis, Laboratoire D'Automatique de Grenoble, INPG, 8 Septembre.

[33] X. Li, X. and de Souza, C. Criteria for robust stability and stabilization of uncertain linear systems with state delay. Automatica, 33, 1657_1662, 1997.

[34] Shimanov. On stability in the critical case of a zero root for systems with time lag. J. Appl. Math. Mech, 24, 653_668, 1960.

[35] Fridman, E. (2001b). New lyapunov-krasovskii functionals for stability of linear retarded and neutral type systems. System and Control Letters, 43, 309_319.

[36] Han, Q.I. Robust stability of uncertain delay-differential systems of neutral type. Automatica, 38, 719_723, 2002.

[37] K. Gu, V. L. Kharitonov, and J. Chen. *Stability of Time-Delay Systems*. Birkhäuser Boston. Control engineering. 2003.

[38] E. Fridman and U. Shaked. An improved stabilization method for linear time-delay systems. *IEEE Trans. on Automat. Control*, 47 :1931–1937, November 2002.

[39] F. Gouaisbaut and D. Peaucelle. A note on stability of time delay systems. In 5th *IFAC Symposium on Robust Control Design (ROCOND'06)*, Toulouse, France, July 2006.

[40] N. Olgac and R. Sipahi. An exact method for the stability analysis of time-delayed linear time-invariant (lti) systems. *IEEE Trans. on Automat. Control*, 47(5) :793–797, 2002.

[41] Hale, J.K. and Lunel, S.M. (1991). Introduction to functional differential equations. Applied Math. Sciences, 99, Springer-Verlag, New York

[42] Krasovskii, N. Stability of motion. Stanford University Press, 1963.

[43] M. Wu, Y. He, J. H. She, and G. P. Liu. Delay-dependent criteria for robust stability of time-varying delay systems. *Automatica*, 40 :1435–1439, 2004.

[44] S. Xu and J. Lam. A survey of linear matrix inequality techniques in stability analysis of delay systems. *International Journal of Systems Science*, 39(12) :1095–1113, December 2008.

[45] P. Park. A delay-dependent stability criterion for systems with uncertain time-invariant delays. *IEEE Trans. on Automat. Control*, 44(4) :876–877, Apr 1999.

[46] Y. S. Moon, P. Park, W. H. Kwon, and Y. S. Lee. Delay-dependent robust stabilization of uncertain state-delayed systems. *Int. J. Control*, 74(14) :1447–1455, 2001.

[47] K. Gu. An integral inequality in the stability problem of time-delay systems. In *Proceedings of the 39th IEEE Conference on Decision and Control*, volume 3, pages 2805–2810, December 2000.

[48] Razumikhin, B. On the stability of systems with a delay. Prikl. Math. Meh., 20(4), 500_512, 1956.

[49] Niculescu, S.I. (1996). Sur la stabilité et la stabilisation des systèmes linéaires à états retardés. Thèse de Doctorat, Laboratoire INPG, France.

[50] http://www.cse.cuhk.edu.hk/~cslui/csc5480.html