Université Paul Sabatier
Master ISTR — Spécialité ASTR

Travaux Pratiques avec Network Simulator 2 (ns-2)

UE Réseau et Commande

Année 2014-2015

Yann Labit, Pascal Berthou, et Guthemberg Silvestre
prenom.nom@laas.fr

1 Introduction

Internet est en train de devenir le réseau universel pour tous les types de données, du transfert simple de fichiers binaires jusqu'à la transmission de la voix, de la vidéo ou d'informations interactives en temps réel. De plus, Internet croît très rapidement, en taille (nombre d'utilisateurs, d'ordinateurs connectés, etc.) et en complexité, en particulier à cause de la nécessité d'offrir de nouveaux services et d'optimiser l'utilisation des ressources de communication pour améliorer la Qualité de Service (QdS) offerte aux utilisateurs. La robustesse de l'Internet d'aujourd'hui dépend fortement du mécanisme de contrôle de congestion de TCP (Transmission Control Protocol) ; protocole de Transport le plus utilisé. La grande variabilité des types de données et donc des applications déployées sur l'Internet rend impensable de laisser l'utilisateur créer/intégrer lui-même le contrôle adéquat de congestion. Par conséquent, il existe un grand nombre de mécanismes de gestion active des files d'attente appelés AQM (Active Queue Management). Ces techniques permettent d'optimiser la gestion des files d'attente des routeurs qui constituent le réseau. La gestion de file d'attente active permet d'améliorer les transferts de données en termes de délais, d'utilisation de liens, de bande passante disponible, de taux de pertes de paquets et d'équité du système.

Les séances de Travaux Pratiques ont comme objective principal comprendre la gestion des files d'attentes d'un réseau sur lequel de nombreux flux circulent. Dans un premier temps, un réseau basique est simulé pour faciliter la prise en main du logiciel et des scripts permettant d'analyser le comportement des différentes applications et protocoles de Transport. Ensuite, plusieurs politiques de gestion de files d'attente seront simulées et analysées. Ces méthodes devront être comparées entre elles et avec un mécanisme de gestion de files provenant de la théorie de la commande.

2 La simulation d'un réseau et l'analyse des résultats

2.1 Objectives

Dans cette première séance, vous allez simuler un réseau avec cinq nœuds avec ns-2. Vous allez apprendre comment exploiter les données générées par ns-2 afin de pouvoir les analyser. Les files d'attente simulées dans cette séance sont du type DropTail.

Plusieurs mesures permettent de connaître l'état d'un réseau et le comportement des flux applicatifs dans ce réseau. ns-2 génère des données brutes : les événements du réseau avec les dates de mise en attente, d'émission, de suppression et de réception des différents paquets circulant sur le réseau simulé. Différents scripts vous sont fournis pour transformer ces données brutes en mesures compréhensibles et similaires aux mesures que l'on pourrait obtenir dans la réalité. Pour chaque simulation, vous devrez observer les mesures suivantes :

- Le débit d'un ensemble de flux à un certain point (l'arrivée de préférence) : somme de la quantité de données reçues divisée par la durée d'observation (`http://homepages.laas.fr/gdasilva/scripts/format_throughput_data.pl` et `http://homepages.laas.fr/gdasilva/scripts/plot_throughput.p`),
- Les statistiques de la file d'attente au niveau du goulet d'étranglement (`http://homepages.laas.fr/gdasilva/scripts/format_queue_data.pl` et `http://homepages.laas.fr/gdasilva/scripts/plot_queue.p`),
- Pour les flux TCP, la fenêtre de congestion des flux (`http://homepages.laas.fr/gdasilva/scripts/plot_congestion_window.p`).

Ces valeurs sont à observer dans ce TP, mais aussi dans les suivants. Vous devrez alors définir une méthode pour analyser le comportement des flux grâce à ces mesures et qui permette de mieux comprendre la simulation du réseau.

2.2 Simulation d'un réseau

Le réseau simulé dans cette partie comporte trois clients distincts et un seul serveur (voir Figure **??**). Les chemins entre les clients et le serveur ne sont pas disjoints ; les trois émetteurs étant connectés au même routeur. Dans un tel réseau, une congestion peut apparaître au niveau du routeur si la somme de la bande passante des trois premiers liens est supérieure à la bande passante entre le routeur et le serveur.

Le routeur agit alors comme un goulet d'étranglement. Cette congestion peut être observée de différentes manières :

- Comportement instable des communications (débit, délai, gigue, ...),
- Réactions de la congestion par la fenêtre de congestion des flux TCP,

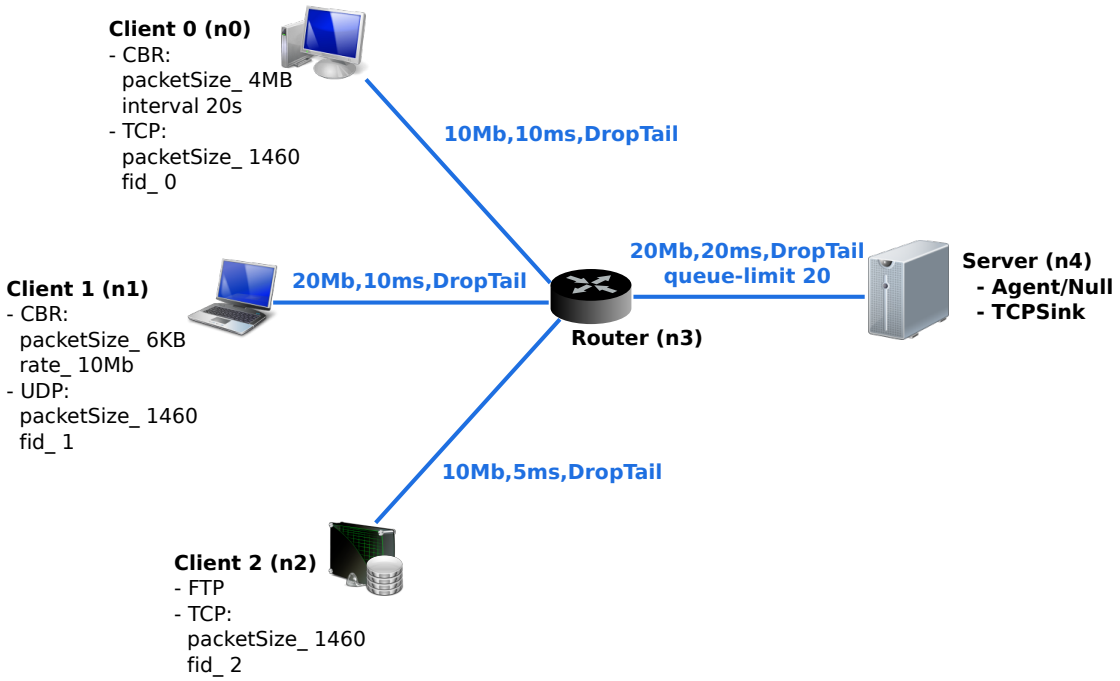

Figure 1: Réseau avec trois nœuds clients, un routeur et un serveur. Un squelette du script Tcl est disponible sur `http://homepages.laas.fr/gdasilva/scripts/squelette_tp.tcl`

- File d'attente du routeur saturée.

Ces différents phénomènes vont être observés dans cette partie du TP. Vous devrez les interpréter pour comprendre quelle est la réaction de TCP face à cette situation et quelles en sont les conséquences sur le réseau ainsi que sur les communications.

La taille de la file sera modifiée seuleument pour le lien entre le router et le serveur. Pous les trois autres liens, ns-2 utilisera la taille par défaut (voir le fichier `ns-allinone-RELEASE/ns-x.xx/tcl/lib/ns-default.tcl`). La Figure **??** représente un schéma résumé de l'exécution et du traitement de donnée d'une simulation.

2.3 Exercices pratiques

2.3.1 Simulation

Compléter le script de simulation [1] afin de pouvoir simuler l'expérience voulue. Lancez ns en donnant en paramètre le script à simuler. Une fois la simulation terminée, vous pouvez lancer nam pour visualiser le réseau et les flux qui y circulent.

```
ns simulation_definition.tcl
```

[1] `http://homepages.laas.fr/gdasilva/scripts/squelette_tp.tcl`

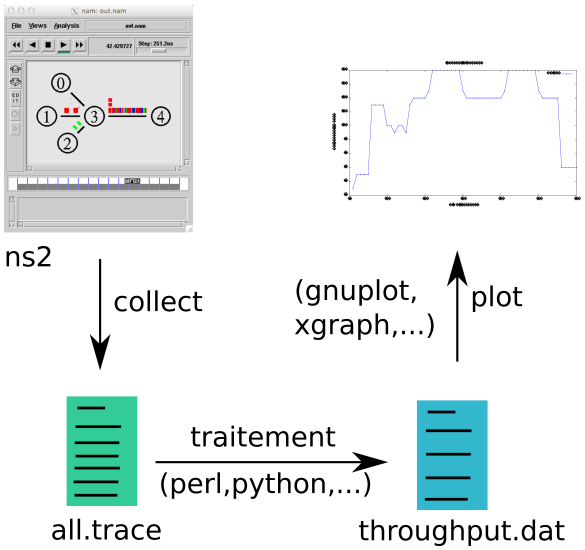

Figure 2: Exécution et traitement de données d'une simulation.

Dans un premier temps, déterminez quelle doit-être la valeur de la bande passante entre le routeur et le serveur pour qu'une congestion apparaisse au niveau du routeur ?

2.3.2 Lecture des traces et calcul des valeurs

Utilisez les scripts listés en Section **??** pour calculer le débit des flux, des statatistiques d'une file d'attente et d'une fenêtre de congestion.

```
perl format_throughput_data.pl all.trace \
    4 1 1000000 > throughput.dat
perl format_queue_data.pl queue.trace \
    0.2 > queue.dat
```

Comment ces deux script ont traité les traces de ns-2? Quels sont les données dans les fichiers générés?

2.3.3 Affichage des mesures

Vous pouvez maintenant visualiser les résultats à l'aide de gnuplot.

```
gnuplot plot_throughput.p
gnuplot plot_queue.p
gnuplot plot_congestion_window.p
```

2.3.4 Analyse des résultats

- Que pouvez-vous dire sur TCP lorsqu'il est seul sur le réseau ? Le comportement de TCP vous parait-il conforme à la théorie ? Quelles phases pouvez-vous distinguer sur

les graphes ?

- Quel est le comportement de TCP en présence d'autres flux ? Les valeurs obtenues pour le débit et le délai vous semblent-elles logiques comparées aux valeurs choisies comme caractéristiques du réseau ? Concluez en faisant une synthèse de vos observations.

- Comment pouvez-vous vérifier de manière pratique que la congestion du réseau est due au goulet d'étranglement ?

2.3.5 Impact de la surcharge du réseau sur le protocole de Transport TCP

Diminuez la bande passante du lien routeur-serveur et refaites l'expérience. Ensuite, augmentez le taux de transfert do Client 1. Vous pouvez également changer la charge du réseau en modifiant les bandes passantes de clients vers le routeur, ou bien modifier le comportement du protocole CBR (*e.g.* intervalle d'envoi au lieu de taux d'envoi). Éventuellement, augmentez la durée de simulation pour mieux visualiser les résultats.

- Quelle valeur minimale faut-il prendre pour que le débit soit uniquement limité par les liens en amont ? Les fluctuations des différents paramètres observés sont dues à des remplissages et des débordements de buffer dans le routeur. Augmentez puis diminuez le niveau maximum de la file d'attente et observez l'impact sur les flux.

- Que pouvez-vous conclure de cette solution ?

2.4 Conclusion

Synthétisez les observations que vous avez faites au cours de cette séance. Votre attention a du normalement se porter sur le comportement de TCP en présence de congestion dans le réseau et sur l'impact de la politique de file d'attente DropTail.

L'analyse de cette première séance doit vous permettre de préparer au mieux la séance suivante.

3 Comparaison de différentes files

3.1 Objectifs

Nous avons vu dans le TP précédent que la taille de la file d'attente avait un impact fort sur les communications. La politique DropTail utilisée est la plus simple qui existe, lorsque la file est pleine, elle refuse les nouveaux paquets et/ou supprime les derniers ajoutés. Network Simulator propose l'utilisation de différentes politiques de files d'attentes dans les liens terrestres :

- Random Early Discard (RED) : les paquets sont supprimés suivant une certaine probabilité. Cette probabilité augmente entre deux seuils définis par l'utilisateur.
- Deficit Round Robin (DRR) : un file d'attente par flux est crée et un ordonnanceur pioche dans les files une après les autres,
- Stochastic Fair Queuing (SFQ) : implémentation de l'algorithme Fair Queuing,
- Class-Based Queuing (CBQ) : différencie les flux suivant leur classe (flow id),
- Des paramètres peuvent modifier le comportement des files, par exemple DropFront : Lorsque la file est pleine, les paquets les plus anciens sont supprimés,

Les précisions nécessaires à l'utilisation des ces politiques de file d'attente sont détaillées en ligne [2]. Pour plus de détails sur le fonctionnement ou sur les paramètres de ces files, vous pouvez regarder les fichiers sources dans `ns-allinone-RELEASE/ ns-x.xx/queue/` ou le fichier contenant les valeurs par défaut prises par ns dans `ns-allinone-RELEASE/ ns-x.xx/tcl/lib/ns-default.tcl`.

Le réseau simulé dans cette séance possède les caractéristiques définies dans Figure **??** (vous pouvez utiliser le script de la dernière séance comme base).

Les politiques testées dans cette séance seront comparés celle utilisée dans la première séance : DropTail. Reprenez votre précédent script, modifiez-le et faites une simulation. Observez les résultats et sauvegardez-les pour pouvoir y revenir par la suite.

3.2 Random Early Discard

Dans cette partie, vous allez mettre en place une politique RED dans la file d'attente du routeur. Cette politique introduit une probabilité de supprimer un paquet suivant le niveau de la file d'attente. Cette probabilité vaut 0 lorsque la file est vide. À partir d'un seuil dit "minimal", cette probabilité augmente et atteint sa valeur maximale pour un seuil dit "maximal". Au delà du seuil maximal, la probabilité de suppression vaut 1. La Figure**??** présente l'évolution de cette probabilité suivant la taille de la file d'attente.

[2] `http://www.isi.edu/nsnam/ns/doc/node69.html`

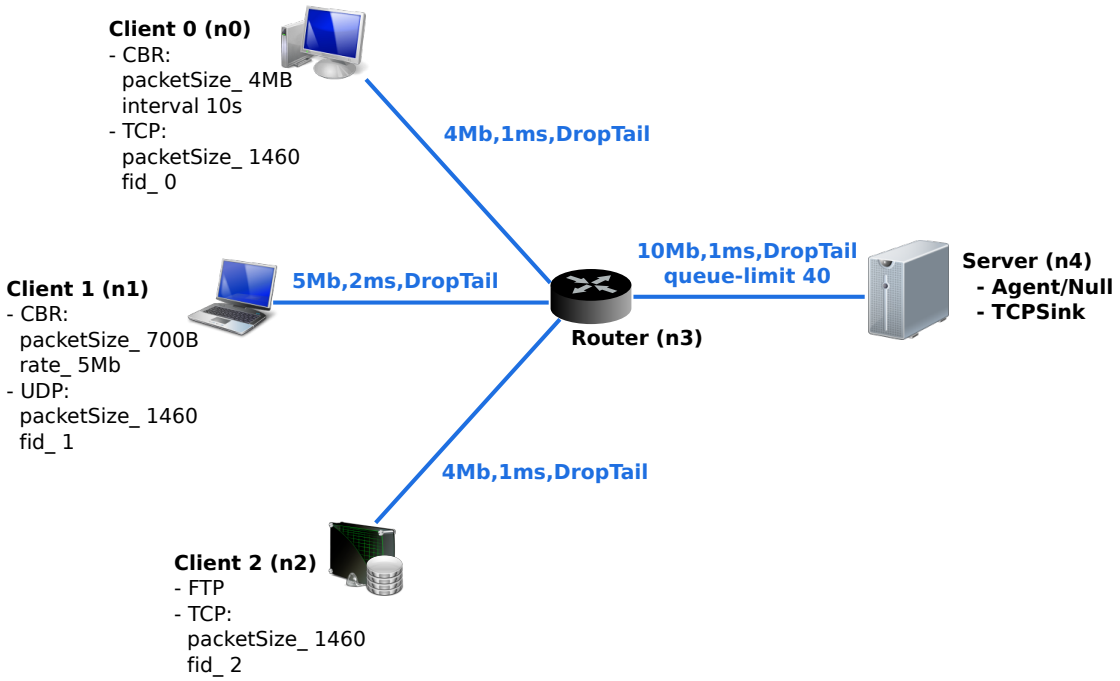

Figure 3: Nouveau réseau avec trois nœuds clients, un routeur et un serveur. Un squelette du script Tcl est disponible sur `http://homepages.laas.fr/gdasilva/scripts/squelette_tp.tcl`

Le réglage des divers paramètres (seuil minimum, seuil maximum, probabilité, ...) peut être fait par l'utilisateur même si ns comprend des valeurs par défaut. Les valeurs utilisées sont choisies en fonction :

- Du réseau où la file d'attente est implémentée : dimensionnement, type de flux ...
- De la position du nœud dans ce réseau : routeur, points d'accès, nœud utilisateur, ...
- Du comportement souhaité : délai faible, stabilité débit et/ou délai, débit important, ...

Vous pouvez regarder en ligne [3] ainsi que le code source de ns-2 pour déterminer sur quels facteurs vous allez agir et quels sont les noms des variables associés. Ouvrez le fichier `ns-allinone-RELEASE/ ns-x.xx/tcl/lib/ns-default.tcl` pour trouver les valeurs par défaut d'une file d'attente RED. Les seuils sont réglés automatiquement si leur valeur n'est pas spécifiée dans le script de simulation. Le réglage se fait en fonction de la bande passante du lien, de la taille moyenne des paquets et du temps de service souhaité.

[3] `http://www.isi.edu/nsnam/ns/doc/node69.html`

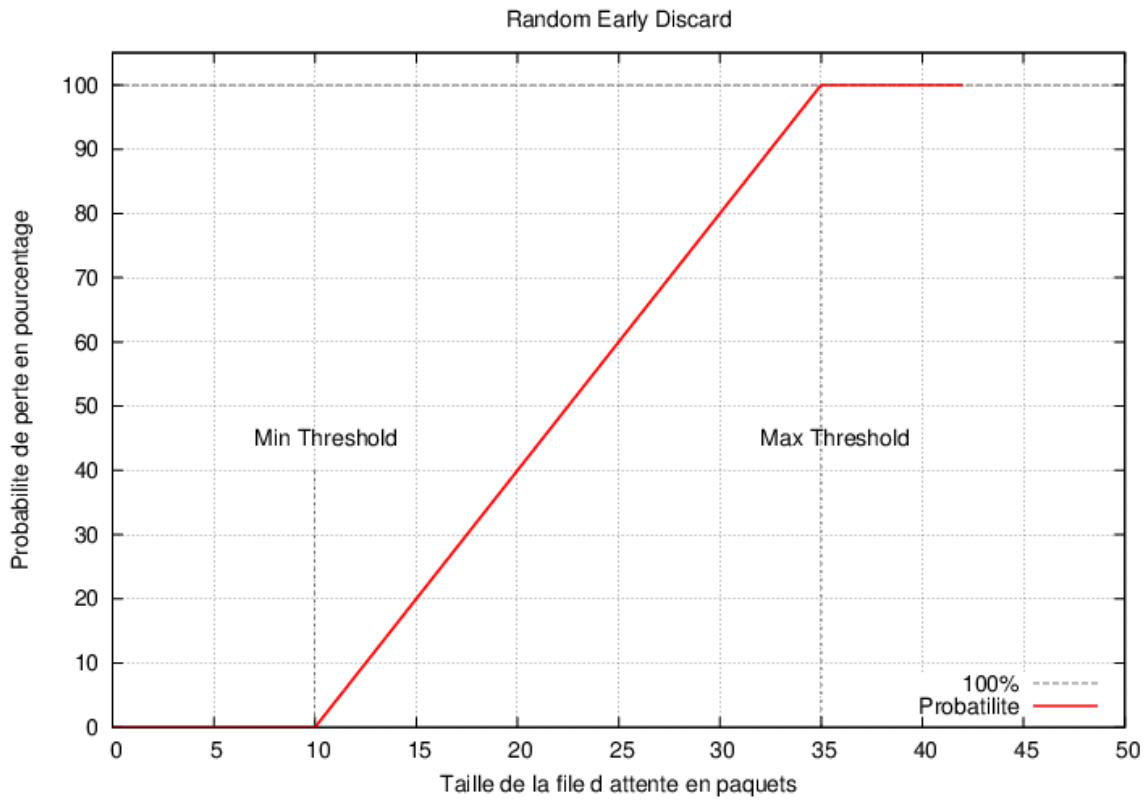

Figure 4: Probabilité de suppression d'un paquet en fonction de la taille de la file.

$$Threshold_{Min} = \max(5, \frac{TargetDelay \times \frac{Bandwidth}{8 \times MeanPacketSize}}{2}) \quad (1)$$

$$Threshold_{Max} = 3 \times Threshold_{Min} \quad (2)$$

Les équations **??** et **??** sont utilisées par ns-2 pour déterminer les valeurs des seuils de RED. Les paramètres TargetDelay et MeanPacketSize sont des paramètres de RED. Pour l'exercice pratique, il faudra vérifier les valeurs des principaux paramètres du RED, à savoir `thresh_`, `maxthresh_`, et `linterm_`.

- En se basant sur les caractéristiques de notre réseau et sur les valeurs par défaut de ces deux paramètres, déterminez les valeurs utilisées par RED pour le seuil minimal et le seuil maximal (pour le calcule, suppose que la taille moyenne des paquets est de 983 octets et que le délai cible ne doit pas être supérieur à 20 fois la valeur du délai du lien entre le routeur et le serveur).

Faites une première simulation en utilisant ces valeurs et analysez vos résultats.

- Quels commentaires pouvez-vous faire par rapport à la simulation précédente avec DropTail ? Les gains obtenus vous semblent-ils satisfaisants ? Pour quelle raisons ?

- Certains paramètres doivent être changés pour que la configuration de la file d'attente soit en accord avec les caractéristiques du réseau et les flux présents. Quels sont ces paramètres et quelles doivent-elles être leurs valeurs ? Refaites la simulation en réglant ces paramètres. Quels changements observez-vous ?

- Vous allez maintenant modifier les seuils pour qu'il y ait moins de suppression de paquets dans le routeur. Quelles valeurs pouvez-vous prendre ? Modifiez ces valeurs dans le script de simulation et refaites la simulation. Quel est l'effet de ce changement sur les communications ? Comment auriez-vous pu obtenir le même résultat sans régler directement les seuils ?

3.3 Deficit Round Robin

Dans cette partie, vous allez utiliser la politique de file d'attente Deficit Round Robin implémentée dans ns-2. Cette politique consiste à différencier les flux et les stocker dans différentes files d'attente. Les flux peuvent être différenciés suivant leur nœud et aussi leur port éventuellement. Une représentation est donnée par la Figure **??**.

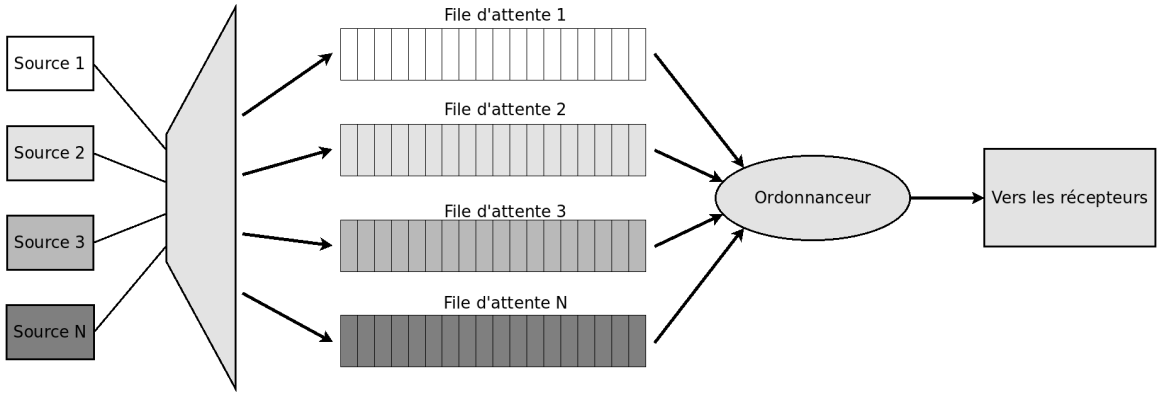

Figure 5: Illustration de la politique de file d'attente Deficit Round Robin.

Le Round Robin consiste à parcourir des files d'attente et à les servir une après les autres. Les Deficit Round Robin utilise un quantum pour déterminer la quantité de données qui peut être envoyée à chaque cycle. Si le quantum n'est pas consommé entièrement, un compteur déficitaire enregistre la différence entre la valeur du quantum et la quantité de données envoyées. Les réactions suivant les données contenues par les files sont donc les suivantes :

- Paquets en attente de taille inférieure au quantum : une quantité de donnée égale ou inférieur au quantum peut être envoyée et le compteur enregistre la différence.

- Paquets en attente de taille supérieure au quantum : rien n'est envoyé et le compteur est incrémenté de la valeur du quantum.

- Pas de paquets en attente : le quantum n'est pas utilisé mais le compteur est mis à 0.

Lors du cycle suivant, la valeur du compteur est ajoutée au quantum pour permettre l'envoi de paquets de plus grande taille ou la récupération de la part du quantum précédent non-utilisée. Avec ns-2, l'utilisateur peut régler la valeur du quantum, le nombre de files (buckets) et la taille totale de la file d'attente.

Comme pour RED, des détails sur DRR et sur les paramètres à utiliser sont donnés en ligne [4]. Les valeurs pas défaut sont stockés dans le même fichier "ns/tcl/lib/ns-default.tcl".

- Effectuez une simulation sur le réseau précédent avec une politique DRR dans le routeur. Que constatez-vous sur le comportement des flux ?

- Comparez ces résultats avec ceux obtenus précédemment avec DropTail et RED. Quelle politique vous semble la plus performante et pourquoi ? Le comportement du réseau vous semble-t-il plus proche des simulations faites avec DropTail ou avec RED ?

- Le Round Robin étant basé sur un principe simple ; le service de chaque file successivement. Par exemple, le Weighted Round Robin permet de mettre un poids à chaque file d'attente et de les servir suivant ce poids. Quel peut-être l'intérêt d'une telle politique ? Donnez un exemple d'une situation où cette file améliorerait les performances du réseau ou des applications.

3.4 PI

En NS2, PI est une politique de file d'attente du type AQM basée sur la loi de commande qui a le même nom. Afin d'éviter le phénomène de congestion d'un élément du réseau à partir de la vision bout en bout des échanges TCP, il a été 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.

Le modèle de communication choisi pour l'AQM PI est basé sur les travaux de Hollot et Misra (C. Hollot, V. Misra, D. Towsley and W. Gong, "On Designing Improved Controllers for AQM Routers Supporting TCP Flows", INFOCOMM 2001)[**?**]. Cf l'article, listé dans les références bibliographiques, pour les détails. Comme préparation pour la dernière séance de travaux pratique, veuillez:

- Lire la référence du modèle de communication choisi pour l'AQM PI [**?**];

- Tester l'efficacité de l'AQM PI en changent des paramètres comme le `a_`, `b_`, `w_`, et `qref_`.

[4] `http://www.isi.edu/nsnam/ns/doc/node69.html`

Facultatif : Vous pouvez simuler les autres politiques de files d'attente présentes en ligne [5] et implémentées dans ns-2. Quelle politiques vous paraissent adaptées au réseau simulé ?

3.5 Conclusion

Faites une synthèse des travaux faits pendant ces deux premières séances. Vous devrez expliquer l'impact des politiques de file d'attente sur le réseau en vous basant sur l'analyse de vos simulations. Cette conclusion doit vous permettre de préparer la troisième et dernière séance appliquée aux outils de l'Automatique pour la QdS d'un réseau de communication.

4 Une introduction au réseau sans fil

4.1 Introduction

Cette section introduit d'une façon succincte une simulation en réseau sans fil. La topologie consiste de deux nœuds mobiles node_(0) and node_(1). Pendant la simulation, les nœuds bougent sur une surface de 500mX500m. Les nœuds démarrent de deux extrémités de cette surface. Ils bougent en se rapprochant pendant la première moitié de la simulation, puis ils s'éloignent pendant la deuxième moitié. Il y a une connexion TCP entre les nœuds. En fonction des mouvements des nœuds, il y a des variations de puissantes du signal qui peuvent causer des pertes de paquets.

Ce scénario a été basée sur le tutoriel du NS-2 [6] et elle contient une introduction succincte des simulation en réseaux sans fil. Le script décrit ci-dessous est téléchargeable sur `http://www.isi.edu/nsnam/ns/tutorial/examples/simple-wireless.tcl`.

4.2 Les étapes d'une simulation de réseau sans fil simple

Pour simuler un réseau sans fil, il faut spécifier une série de nouveau paramétrés sur NS-2, comme la couche de liaison (LL), une interface file (IfQ), la couche MAC, les canaux pour envoie et réception de données etc. La liste simplifiée de paramètres est la suivante :

```
# ==========================================
# Define options
# ==========================================
set val(chan) Channel/WirelessChannel;# channel type
set val(prop) Propagation/TwoRayGround;#propagation model
set val(netif) Phy/WirelessPhy;# network interface type
set val(mac) Mac/802_11;# MAC type
set val(ifq) Queue/DropTail/PriQueue;# interface queue type
set val(ll) LL;# link layer type
```

[5] `http://www.isi.edu/nsnam/ns/doc/node69.html`

[6] http://www.isi.edu/nsnam/ns/tutorial/nsscript5.html

```
set val(ant) Antenna/OmniAntenna;# antenna model
set val(ifqlen) 50;# max packet in ifq
set val(nn) 2;# number of mobilenodes
set val(rp) DSDV;# routing protocol
```

En suite, nous devons créer l'objet qui maintient le registre des mouvements des nœuds dans les frontière de déplacement, et aussi définir la surface carrée qui définie ces frontières.

```
# set up topography object
set topo       [new Topography]

$topo load_flatgrid 500 500
```

Puis, il faut créer l'objet godn God (General Operations Director), qui est utilisé pour maintenir les informations globales sur l'environnement simulé.

```
#
# Create God
#
create-god $val(nn)
```

Après, nous créons les nœuds mobiles. En effet, nous configurons les nœuds avant de les créer. Cela consiste à definir le type adressage (plate, hiérarchique,...), le type de protocole de routage adhoc, la couche de liaison, MAC, IfQ, etc.

```
#
#  Create the specified number of mobilenodes [$val(nn)] and
#  "attach" them to the channel.
#  Here two nodes are created : node(0) and node(1)

# configure node

        $ns_ node-config -adhocRouting $val(rp) \
                         -llType $val(ll) \
                         -macType $val(mac) \
                         -ifqType $val(ifq) \
                         -ifqLen $val(ifqlen) \
                         -antType $val(ant) \
                         -propType $val(prop) \
                         -phyType $val(netif) \
                         -channelType $val(chan) \
                         -topoInstance $topo \
                         -agentTrace ON \
                         -routerTrace ON \
                         -macTrace OFF \
                         -movementTrace OFF
```

```
        for {set i 0} {$i < $val(nn) } {incr i} {
                set node_($i) [$ns_ node]
                $node_($i) random-motion 0;# disable random motion
        }
```

Maintenant que les nœuds sont crées, il faut définir les positions de départ et les paramètres pour les déplacements.

```
#
# Provide initial (X,Y, for now Z=0) co-ordinates for mobilenodes
#
$node_(0) set X_ 5.0
$node_(0) set Y_ 2.0
$node_(0) set Z_ 0.0

$node_(1) set X_ 390.0
$node_(1) set Y_ 385.0
$node_(1) set Z_ 0.0

#
# Now produce some simple node movements
# Node_(1) starts to move towards node_(0)
#
$ns_ at 50.0 "$node_(1) setdest 25.0 20.0 15.0"
$ns_ at 10.0 "$node_(0) setdest 20.0 18.0 1.0"

# Node_(1) then starts to move away from node_(0)
$ns_ at 100.0 "$node_(1) setdest 490.0 480.0 15.0"
```

En suite, nous pouvons définir le trafic TCP d'échange entre les noeuds.

```
# Setup traffic flow between nodes
# TCP connections between node_(0) and node_(1)

set tcp [new Agent/TCP]
$tcp set class_ 2
set sink [new Agent/TCPSink]
$ns_ attach-agent $node_(0) $tcp
$ns_ attach-agent $node_(1) $sink
$ns_ connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns_ at 10.0 "$ftp start"
```

Finalement, il faut définir le début et la fin de la simulation et la lancer.

```
#
# Tell nodes when the simulation ends
#
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at 150.0 "$node_($i) reset";
}
$ns_ at 150.0 "stop"
$ns_ at 150.01 "puts \"NS EXITING...\" ; $ns_ halt"
proc stop {} {
    global ns_ tracefd
    $ns_ flush-trace
    close $tracefd
}

puts "Starting Simulation..."
$ns_ run
```

5 Réseaux de communication et AQM de la théorie de la Commande (PI) : Simulations et validations via MATLAB/SIMULINK/NS-2

5.1 Objectives

Les deux premières séances dédiées à NS-2 vous ont permis d'appréhender le logiciel de simulations de réseaux (NS-2) et ainsi permettre de préparer la troisième et dernière séance appliquée aux outils de l'Automatique pour la QdS d'un réseau de communication. Dans les TP précédents, vous avez mis en place plusieurs politiques de file d'attente (RED en particulier) et analysé les intérêts et les inconvénients de chacune. Pendant cette séance, vous utiliserez vos connaissances en Automatique pour mettre en place une politique de file d'attente basée sur un correcteur Proportionnel Intégrale.

Pour implémenter un correcteur dans une file d'attente, il faut adapter les nécessités et les possibilités de l'Automatique au monde des réseaux informatiques. Dans notre situation, on souhaite maîtriser le flux de paquets arrivant dans le routeur. Ainsi, en stabilisant la file d'attente autour d'un niveau "idéal", il devient possible d'obtenir de meilleures performances en terme de débit mais aussi en terme de délai.

Il est nécessaire pour mettre en place un tel système de connaître le fonctionnement de l'entité générant les paquets et de pouvoir la modéliser. Le générateur de flux dans un réseau est le protocole de Transport. Comme vous avez pu le constater dans les séances précédentes, TCP contient un mécanisme de contrôle du flux et de la congestion. Lorsque des pertes de paquets ou des transmissions trop longues sont détectées, TCP réduit sa fenêtre de congestion et diminue ainsi son débit d'émission. Un correcteur implanté entre le nœud émetteur et le nœud récepteur est donc capable d'influer sur la connexion TCP (donc le flux de paquets) en choisissant de laisser passer ou de supprimer un paquet.

5.2 Modèle de réseau de communications

Rappels :

- 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, 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.

- Afin d'éviter le phénomène de congestion d'un élément du réseau à partir de la vision bout en bout des échanges TCP, il a été 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'hypothèse fondamentale de cet algorithme est de considérer qu'une perte de paquets est synonyme de congestion. En effet, les pertes accidentelles, dûes à un problème physique de la ligne de communication ou encore aux erreurs bits non corrigées par la couche liaison sont négligeables.

- Algorithme AIMD (Figure **??**).

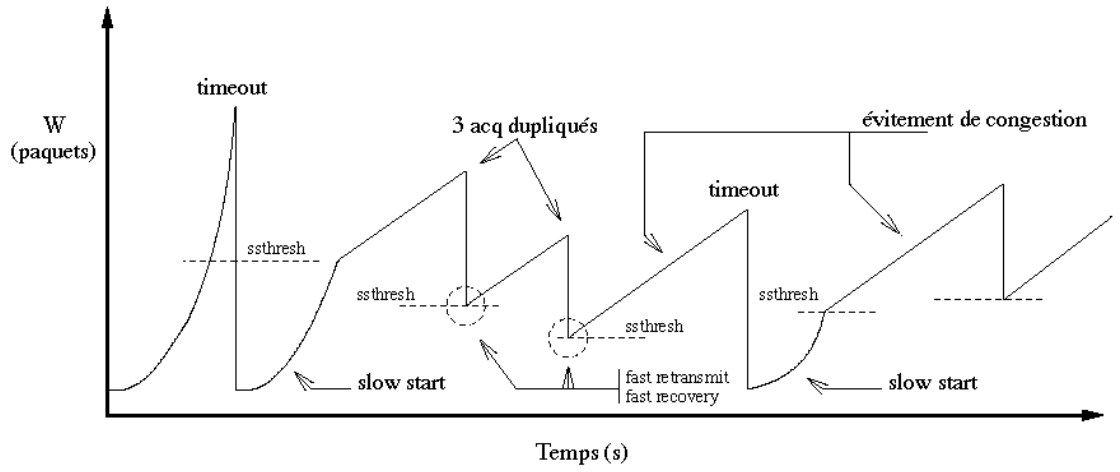

Figure 6: Exemple d'évolution de la fenêtre de congestion W.

La philosophie de TCP est de "sonder" le réseau en augmentant progressivement la taille de la fenêtre de congestion. Puis, suivant l'information suggérant la perte d'un paquet, elle est diminuée brutalement afin de sortir rapidement de l'état de congestion.

- 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.

Dans le second cas, on distingue deux types d'indication de congestion : les indications par Time Out et les indications par acquittements dupliqués (3DupAck)

- Si la source n'a pas de réponse du destinataire ($\equiv$ TO) : W $\leftarrow$ 1.
- Si la source reçoit trois acquittements identiques ($\equiv$ 3DupAck) : W $\leftarrow$ W/2.

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

- Au cours de cette séance, nous nous intéressons 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. Les ordinateurs utilisent le protocole TCP pour communiquer et tenant compte du comportement du protocole décrit précédemment, il s'agit de résoudre le problème de partage de ressource sous une certaine contrainte de Qualité de Service (comme l'équité ou un service différencié entre les sources).
- Le modèle de communication choisi est basé sur les travaux de Hollot et Misra (C. Hollot, V. Misra, D. Towsley and W. Gong, "On Designing Improved Controllers for AQM Routers Supporting TCP Flows", INFOCOMM 2001)[**?**]. Cf l'article, listé dans les références bibliographiques, pour les détails. Le modèle "espace d'état" est le suivant :

$$\begin{cases} \dot{W}(t) = \dfrac{1}{R(t)} - \dfrac{W(t)}{2}\dfrac{W(t-h)}{R(t-h)}p(t-R(t)) \\ \dot{q}(t) = -C + \dfrac{N(t)}{R(t)}W(t) \end{cases} \tag{3}$$

- Un AQM est placé comme le montre la figure suivante.

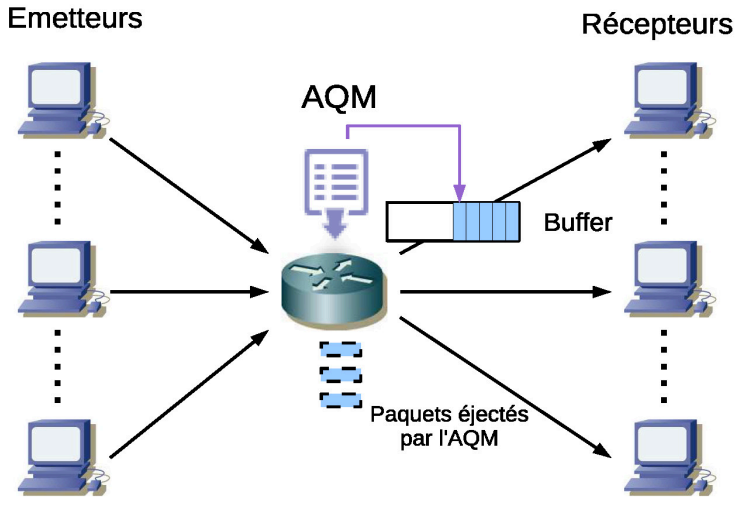

Figure 7: Placement de l'AQM.

Questions :

1. A partir de l'article référencé ci-dessus, identifier les entrées, les sorties du modèle de communication ainsi que les unités employées par les caractéristiques du modèle.

2. Justifiez physiquement les différents termes du modèle "Espace d'État" de Hollot et Misra. Question 3 : Proposer un schéma fonctionnel permettant de boucler le système et satisfaire des niveaux de performances.

3. Etablir un schéma-bloc du modèle sous Simulink ; Toutes les données numériques sont paramétrées dans MATLAB.

4. Mettre en place une simulation du modèle non linéaire. Conclure quant aux performances du système (jeu de valeurs à choisir soit à partir de la référence en ligne [7], soit à partir de l'exemple).

5. Ecrire le modèle linéarisé du modèle de Hollot et Misra autour d'un point de fonctionnement (Wo, qo, Ro).

6. Etablir un schéma-bloc du modèle linéarisé sous Simulink ; Toutes les données numériques sont paramétrées dans MATLAB.

7. Mettre en place une simulation du modèle linéarisé. Conclure quant aux performances du système (jeu de valeurs à choisir soit à partir de la référence en ligne [**?**]). Comparer avec le modèle non linéaire.

8. Ecrire sous SIMULINK la synthèse d'un AQM de type RED puis PI (Proportionnel et Intégral), à partir de la référence de Hollot et Misra.

9. Mettre en place une simulation du modèle linéarisé bouclé avec le RED puis bouclé avec le PI. Conclure quant aux performances du système (jeu de valeurs à choisir soit à partir de la référence en ligne [**?**]). Comparer avec le modèle non linéaire bouclé.

10. Discrétiser le PI et trouver l'équation récurrente qui sera implémentée dans NS-2.

11. A partir de ce niveau, l'étude se poursuit avec NS-2, pour évaluer les performances de la loi de commande PI et la comparer à l'AQM de type RED. Sous NS-2, compléter le fichier correspondant au PI, puis effectuer une simulation identique réalisée en Section **??**. Pour cela, compléter les valeurs des coefficients du correcteur PI, la fréquence d'échantillonnage. Comparer les performances avec l'AQM RED, d'un point de vue W, q, pertes.

12. Modifier les paramètres précédents (PI, Fe) et évaluer les performances.

[7] `http://www.isi.edu/nsnam/ns/doc/node69.html`

References

[1] *ns2 documentation: Different types of queue objects.* http://www.isi.edu/nsnam/ns/doc/node69.html, 11 2014.

[2] C. V. Hollot, V. Misra, D. Towsley, and W.-B. Gong, *On designing improved controllers for aqm routers supporting tcp flows*, in INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, vol. 3, IEEE, 2001, pp. 1726–1734.