

Université de Mons
Faculté Des Sciences
Département d'Informatique

Projet réseau - Selective-repeat

Professeur :

Bruno QUOITIN

Assistants :

Jeremy DUBRULLE

Auteurs :

Théo GODIN 210170

François VION 210809

Groupe :

23

Table des matières

1	Construction et exécution	1
2	Applications	1
3	Selective-repeat	1
3.1	Envoi d'un message	1
3.2	Pipelining	1
3.3	Envoi d'un segment	1
3.4	Timeout et RTO	2
3.5	Réception d'un acquittement	2
4	Contrôle de congestion	2
5	Bugs rencontrés	2
6	Outils	2

1 Construction et exécution

La compilation du projet se fait en exécutant la commande :

```
$ javac reso/examples/selectiverepeat/Demo.java
```

Et l'exécution :

```
$ java reso.examples.selectiverepeat.Demo [x] [y]
```

(où x (entier) est le pourcentage de perte de paquets aléatoire et y (entier) le bit rate).

Une fois l'exécution terminée les fichiers log et csv sont générés à la racine du projet.

2 Applications

Nous avons décidé de suivre l'exemple ping-pong et de créer deux classes application, une qui envoie les données et une autre qui les reçoit. L'app sender reçoit les données à envoyer en paramètre de son constructeur et passe ensuite ces données à la méthode sendMessage du protocol. Tandis que l'app receiver, au lancement de l'application, ne fait qu'écouter.

3 Selective-repeat

3.1 Envoi d'un message

Un message peut être envoyé par une application via la méthode sendMessage de notre protocol. Cette méthode va séparer ce message en plus petits messages de 8 caractères qui vont pouvoir ensuite être envoyés en pipelining.

3.2 Pipelining

La méthode pipeliningSend s'occupe d'envoyer des segments tant que la window n'a pas atteint sa taille maximale.

3.3 Envoi d'un segment

Afin d'envoyer un segment, la méthode send envoie ce segment à la destination, l'ajoute à l'arrayList représentant la window et finalement crée et démarre un timer afin de gérer les timeout pour ce segment.

Du côté du receveur, la méthode sendAck permet d'envoyer un acquittement de même sequence number que le segment reçu.

3.4 Timeout et RTO

Si un timer arrive à sa fin, le segment associé est ré-envoyé et un nouveau timer est créé. La durée du timer est calculée de la même manière que vue au cours. Si l'on a un timeout, la durée de celui-ci est doublée et si l'envoyeur reçoit un ACK, les valeurs pour R, SRTT, devRTT et RTO (durée du timer) sont recalculées avec les formules indiquées dans le cours.

3.5 Réception d'un acquittement

Lors de la réception d'un acquittement par l'envoyeur, si le segment correspondant est présent dans la window, celui-ci est marqué comme acquitté. Ainsi, au prochain appel de la méthode `verifyWindow`, la fenêtre sera avancée pour chaque paquets ayant reçu son acquittement.

4 Contrôle de congestion

Afin de gérer le contrôle de congestion, la taille de la fenêtre est modifiée au cours du processus afin d'avoir une taille optimale tout au long de l'envoi. Cela se fait lorsque l'envoyeur reçoit un ACK ou lors d'un timeout. Lors de la réception d'un ACK, on double la taille de la fenêtre si celle-ci est inférieure au `ssthresh` (fixé à 100 au départ). Si elle est inférieure, on l'incrémente de 1. Lors d'un timeout, la taille de fenêtre est fixée à 1 et le `ssthresh` est fixé à la moitié de la dernière valeur de fenêtre. Dans le cas où la fenêtre avait une taille de 1, le `ssthresh` est fixé à 1.

5 Bugs rencontrés

Nous avons remarqués grâce aux logs que le receveur recevait des paquets acquittés. Après avoir testé chaque méthode envoyant des messages, la source du bug n'a toujours pas été découverte car aucune d'entre elles n'envoie de message acquitté. Ce bug a donc été contourné en ajoutant au protocole l'attribut `actor` permettant de savoir si l'application courante est l'envoyeur ou le receveur, ce qui est testé lors de la réception d'un message.

6 Outils

Notre projet comporte un système de logs implémenté dans la classe `Tools`. La méthode `log` nécessite 3 paramètres :

1. le temps
2. l'acteur
3. l'action

Ces 3 informations sont ensuite enregistrées dans le fichier `selectiverepeat.log` à la racine du projet.

Nous avons également implémenté la méthode `plot` afin de suivre l'évolution de la fenêtre. Cette méthode permet de générer un fichier csv contenant les données de l'évolution de la fenêtre en fonction du temps.