Master Informatique 2022 - 2023

Rapport Mif19

TP – Simulations avec Network Simulator (NS2)

Par Samuel YANG et Anh-Kiet VO

Rendu le 15/06/2023

Professeur : Mr HADDAD Mohammed
Université : Claude Bernard Lyon 1, 69100 Villeurbanne

Table des matières

I. Premier pas

Question 1

La procédure "finish" est responsable de la finalisation de la simulation au bout de 5 secondes. Elle est lancée au bout de 5 secondes d'exécution et permet de générer un fichier trace.

Question 2

La principale différence entre TCP (Transmission Control Protocol) et UDP (User Datagram Protocol) réside dans leur mode de fonctionnement :

- TCP est un protocole orienté connexion, fiable et orienté vers les flux de données. Il garantit la livraison des données dans l'ordre et sans perte, en gérant la retransmission des paquets perdus et le contrôle de congestion.
- UDP, en revanche, est un protocole sans connexion, non fiable et orienté vers les datagrammes. Il ne garantit pas la livraison des données ni leur ordre. Il est souvent utilisé pour des applications où une latence réduite est primordiale, telles que la voix sur IP, la diffusion en continu et les jeux en ligne.

Question 3

Un flux FTP (File Transfer Protocol) circule sur la couche de transport TCP. FTP utilise TCP pour établir une connexion fiable entre le client et le serveur, assurant ainsi un transfert de fichiers précis et sécurisé.

Question 4

Avant de communiquer, les agents doivent être attachés pour indiquer à NS-2 (le simulateur de réseau utilisé dans le script) qu'ils sont responsables de l'envoi et/ou de la réception de données sur des nœuds spécifiques. L'attachement des agents à des nœuds spécifiques permet au simulateur de suivre et de gérer les flux de données entre les nœuds du réseau simulé.

Question 5

La taille par défaut d'un paquet CBR (Constant Bit Rate) dans ce script est définie à 1000 octets. Cela est indiqué à la ligne suivante :

- $cbr set packet_size_ 1000

Cependant, il est important de noter que la taille du paquet peut être modifiée en fonction des besoins spécifiques de la simulation.

II. Topologie du réseau et routage dynamique

Question 1

Les petits paquets que l'on peut voir sur NAM correspondent aux paquets échangés entre les nœuds du réseau lors de la simulation. Le nœud communique alors sa table de routage avec tous ses voisins afin de les renseigner sur les modifications de la topologie du réseau. Ils seront échangés au début avec le protocole de routage "Distance vector" qui nécessitera la connaissance de toutes les tables de routage dans son voisinage. Ces paquets sont échangés à différents moments de la simulation, en fonction de la planification des événements spécifiée dans le script.

Question 2

A

Pour déterminer le nombre de chemins potentiels entre un point de départ et une destination dans une grille composée de 9 nœuds, il est possible d'appliquer une méthode de recherche de chemins telle que l'algorithme de recherche en profondeur (DFS) ou l'algorithme de recherche en largeur (BFS). Cela permettra d'explorer l'ensemble des chemins envisageables.

B

Dans une grille de 9 nœuds, il existe divers chemins possibles entre n1 et n9. Le choix du chemin le plus court dépendra de la métrique utilisée pour évaluer la distance entre les nœuds, telle que le nombre de sauts ou la longueur des liens.

D

Dans le cas le plus défavorable en matière de défaillance des liens, l'algorithme DV peut être confronté à la situation où tous les liens le long du chemin entre n3 et n8 sont coupés. Cela signifie que tous les liens reliant n3 à n8 deviennent inaccessibles, ce qui peut entraîner une perte de connectivité entre ces deux nœuds.

III. Échanges de messages

Initialement, le nœud n0 envoie un ping avec une valeur de 42 à n1. Une fois reçu, n1 répond en envoyant un paquet de taille 100 avec un compte à rebours de valeur 5 à n0. Lorsque n0 reçoit le paquet, la valeur du compte à rebours diminue de 1, puis n0 le renvoie à n1. Ce processus se répète jusqu'à ce que la valeur du compte à rebours atteigne zéro. Une fois le compte à rebours terminé, n0 envoie le message "ignore this message please" avec une taille de segment maximale (MSS) de 500. Ensuite, n1 envoie à nouveau un message "ping" avec un MSS de 828, et n0 répond avec un pong de MSS 100. Tout cela se déroule dans notre fonction qui permet l'envoi du pong après la réception du ping.

```
# The "process_data" instance procedure is what will process received data
# if no application is attached to the agent.
# In this case, we respond to messages of the form "ping(###)".
# We also print received messages to the trace file.
Agent/UDP instproc process_data {size data} {
    global ns
    $self instvar node_
    # note in the trace file that the packet was received
    $ns trace-annotate "[$node_ node-addr] received {$data}"
    # if the message was of the form "ping(###)" then send a response of
    # the form "pong(###)"
    if {[regexp {ping *\(([0-9]+)\)} $data entirematch number]} {
        $self send 100 "pong($number)"
    } elseif {[regexp {countdown *\(([0-9]+)\)} $data entirematch number] && $number > 0 } {
        incr number -1
        $self send 100 "countdown($number)"
    }
}
```

Figure 1: Procédure "process_data" qui traite les données reçues par l'agent UDP

IV. Réseaux sans fil

Dans cette partie nous avons pu faire un fichier C qui permettra de concevoir un générateur de mouvements aléatoires pour ce type de réseaux. Le fichier générateur se nomme « setdest.exe ».

V. Analyse du fichier trace

Afin d'éviter toute congestion, il est possible de fixer le débit de nos nœuds à 0,5 Mo/s en utilisant la commande "$cbr0 set rate_ 0.5 Mo/s"

SENT BLUE PACKETS : 523
LOST BLUE PACKETS : 0
SENT RED PACKETS : 520
LOST RED PACKETS : 0

En mettant le débit de nos noauds à 2.0 mbps, nous obtenons :

SENT BLUE PACKETS : 689
LOST BLUE PACKETS : 23
SENT RED PACKETS : 669
LOST RED PACKETS : 0

Il est observé que le flux 1 subit une perte de paquets plus importante que le flux 2, avec respectivement 23 paquets perdus et aucun paquet perdu. Sur les 689 paquets reçus, le flux 1 en perd 23. Le calcul (689-23) / 689 = 0.966 révèle une perte de paquets de 3.4% pour ce flux. Voici le résultat obtenu à l'aide de la commande awk après la transition du deuxième flux en mode ftp.

SENT BLUE PACKETS : 1212
LOST BLUE PACKETS : 76
SENT RED PACKETS : 71
LOST RED PACKETS : 0

Il est observé que le protocole FTP envoie considérablement moins de paquets que le protocole CBR, mais il n'y a aucune perte de paquets. Cette situation est attribuable au fait que FTP utilise le protocole TCP, qui garantit la livraison des paquets, tandis que CBR utilise le protocole UDP, qui ne garantit pas la transmission des paquets mais offre une vitesse relative plus élevée.

VI. Files d'attente

Question 1

La fonction `sendpacket` est une procédure qui envoie des paquets de données à intervalles aléatoires. Voici une analyse de cette fonction :

- La variable `time` est initialisée avec la valeur actuelle du temps dans la simulation.
- La fonction `$InterArrivalTime value` génère un temps aléatoire inter-arrivée en utilisant une distribution exponentielle.
- La fonction `$pktSize value` génère une taille aléatoire de paquet en utilisant une distribution exponentielle.
- Les paquets sont envoyés par l'agent source (`$src`) avec la taille définie.
- La fonction `sendpacket` est programmée pour être rappelée à un moment futur déterminé par le temps actuel et le temps inter-arrivée généré. Ainsi, de nouveaux paquets sont envoyés de manière récurrente avec des intervalles aléatoires

Question 2

Le débit moyen peut être calculé en utilisant la formule : débit moyen = (taille des paquets) / (temps de réponse moyen). Le résultat que j'obtiens est 7.11.

Question 3

Le temps de réponse peut être calculé en utilisant la formule : temps de réponse = temps moyen passé dans la file d'attente + temps moyen de service. J'obtiens un temps de réponse de 0.125176.

Question 4

La longueur moyenne de la file est le débit multiplié par le temps de réponse. Le résultat obtenu est : 0.89.

Question 5

Le taux d'occupation de la file peut être calculé en utilisant la formule : taux d'occupation = (temps moyen de service) * (débit d'arrivée moyen). On obtient un taux d'occupation de 0.85.

Question 6

L'intensité du trafic peut être calculée en utilisant la formule : intensité du trafic = (débit d'arrivée moyen) / (débit de service moyen). Le résultat est 0.653061.

Question 7

Le taux de rejet peut être calculé en utilisant la formule : taux de rejet = (nombre de paquets rejetés) / (nombre total de paquets arrivés). Le taux de rejet et donc de 0.0165975.

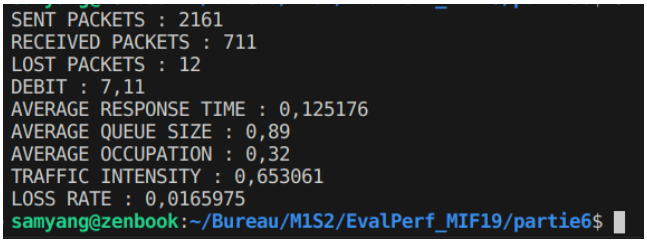

Figure 2: Utilisation de mon script

Question 8

Pour M/M/1/∞, j'ai modifié le paramètre qsize à 100000. Nous avons alors ces valeurs :

- Débit moyen : 7.23 paquets/secondes
- Temps de réponse moyen : 0.149378 secondes
- Longueur moyenne de la file : 1.08 paquets
- Taux d'occupation moyen : 0.85
- Intensité du trafic : 0.653061
- Taux de perte : 0

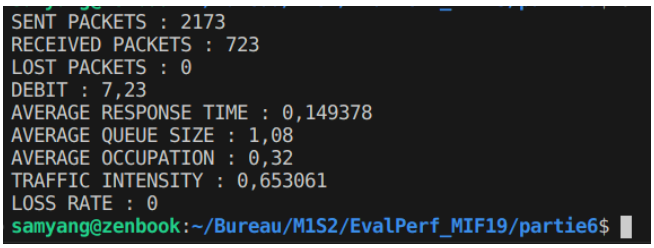

Figure 3: Utilisation de mon script avec qsize=100000

VII. Évaluation de la mobilité et Mouvement aléatoire

Question 1

La fonction Ax(t) calcule la distance moyenne entre un nœud x et tous les autres nœuds à un instant t, représentant le mouvement relatif du nœud x par rapport aux autres.

La mobilité moyenne Mx d'un nœud x sur une période de temps T mesure les changements de distances entre les positions du nœud x à différents instants, reflétant ses mouvements relatifs au fil du temps.

La mobilité moyenne dans le réseau, Mob, est la moyenne des mobilités individuelles de tous les nœuds, évaluant la mobilité relative globale du réseau en considérant les mouvements relatifs de tous les nœuds.

Table des Illustrations