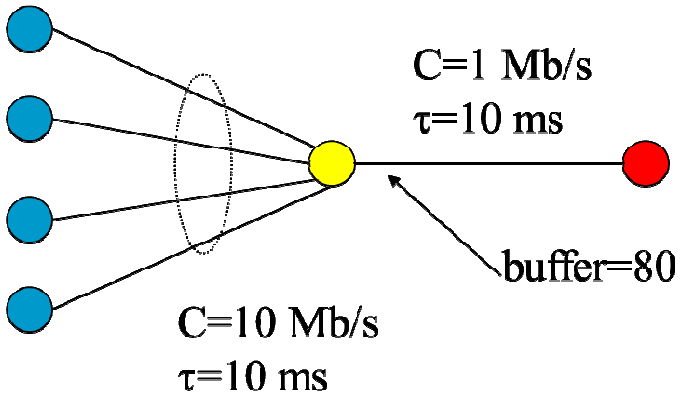
TP 2: Le protocole TCP et le Routage

Ce TP a pour but d'utiliser le simulateur Network Simulator (NS2) pour comprendre le fonctionnement du protocole TCP, en particulier le mécanisme de fenêtre glissante, le contrôle de flux et congestion. Ensuite, nous allons étudier le routage dynamique (Distance Vector).

Ex. 1 : TCP

Considérons la topologie suivante, avec 5 liens full-duplex et 4 sources FTP/TCP. Chacun des liens à gauche (entre chaque nœud bleu et le nœud jaune) a une capacité de 10 Mbit/s et un délai de propagation de 10ms. Le lien à droite (entre le nœud jaune et le nœud rouge) a une capacité de 1 Mbit/s, un délai de propagation de 10 ms et un buffer qui peut contenir au maximum 80 paquets.

La commande *$ns queue-limit $Node1 $Node2 80* permet de fixer la taille d'un buffer.



La durée totale de la simulation est de 2 [s].
Il faut créer 4 Agent/TCP/RFC793edu (un pour chaque nœud bleu, avec paramètres window_ = 20 et packetSize_ = 1000 bytes), et bien sur un Agent/TCPSink pour chacun d'entre eux (tous connectes au nœud rouge). Ensuite, il faut créer 4 applications FTP pour chacun de 4 Agents TCP RFC793edu et les faire démarrer au début de la simulation.

- Question 1.1 : calculer le Round Trip Time (RTT) pour chacune des 4 connections (il est bien évidemment le même), et a partir de cela le débit de transmission moyen (R=window_*packetSize_/RTT)
- Question 1.2 : est-ce que le débit total (4*R) est supérieur ou pas à la capacité du lien à droite (avec buffer = 80) ?
- Question 1.3 : est-ce qu'on observe des pertes de paquets dans le réseau ? Pourquoi ? Pour visualiser avec *nam* ce qui se passe dans la file d'attente (buffer), on peut utiliser la commande :
$ns duplex-link-op $n1 $n2 queuePos 0.5
(C'est preferable de colorier les 4 flux TCP avec des couleurs differentes, avec des commandes telles que $ns color 1 Blue et $agent1 set class_ 1)

Ex. 2 :

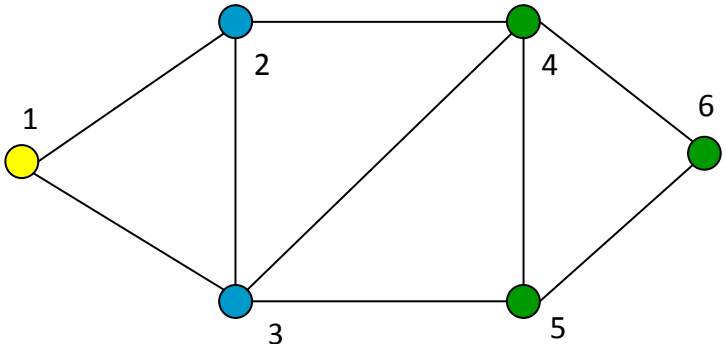
Considérons maintenant une variante de la topologie de l'Ex. 1, avec notamment :

- Le buffer du lien à droite (entre le nœud jaune et le nœud rouge) peut contenir 10 paquets
- La fenêtre *window_* est = 5 paquets

- Question 2.1 : calculer le débit de transmission moyen (R=window_*packetSize_/RTT) de chaque source TCP.
- Question 2.2 : est-ce qu'on observe des pertes de paquets dans le réseau ? Pourquoi ?

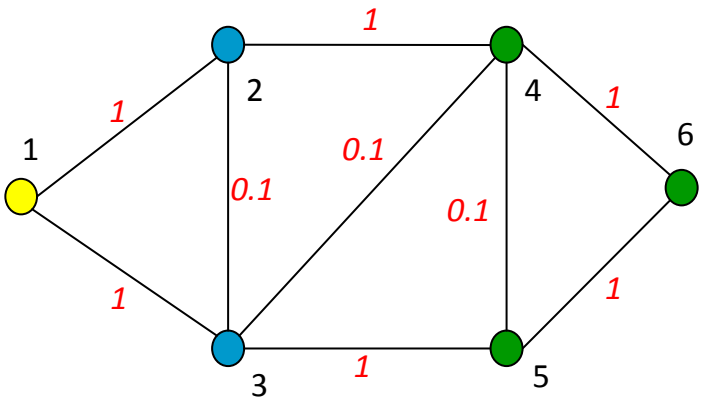
Ex. 3 :

Considérons tout d'abord la topologie suivante, avec 6 nœuds.
- Tous les liens ont une capacité de 10 Mbit/s et un délai de propagation de 20 ms.
- 3 agents UDP gèrent le trafic, notamment :
    1) entre le nœud 1 (source de trafic) et le nœud 4 (destination)
    2) entre le nœud 1 (source de trafic) et le nœud 5 (destination)
    3) entre le nœud 1 (source de trafic) et le nœud 6 (destination)
- A ces 3 agents UDP correspondent 3 applications CBR, toutes avec le même débit de transmission, 3 Mbit/s, et la taille de paquet, 5000 bytes.
- La durée totale de la simulation est de 1 s.



- Question 3.1 : Vérifier c'est quoi le parcours suivi par les 3 flux CBR dans le réseau (c'est mieux de colorier les flux avec des couleurs différentes).

-Maintenant, changer le cout de 3 liens comme illustré dans la figure qui suit (par default, le cout d'un lien est égal à 1 ; ici le cout de trois liens devient 0.1). La commande *$ns cost $Node1 $Node2 0.1* peut être utilisée.



- Question 3.2 : Vérifier c'est quoi le nouveau parcours suivi par les 3 flux CBR dans le réseau.

- Finalement, on revient à la première topologie (celle où tous les liens on cout = 1).
- Ici, on utilise un protocole de type *Distance Vector* (tel que RIP, par exemple), avec les commandes (au tout debout du script)

set ns [new Simulator]
$ns rtproto DV

- A l'instant 0.3 s, le lien entre les nœuds 3 et 4 tombe en panne (utiliser la commande suivante)
$ns rtmodel-at 0.3 down $Node3 $Node4

- Question 3.3: Vérifier le parcours des paquets avant et après l'instant t= 0.3 s.