Vous devez écrire un simulateur implémentant la technique décrite dans ce document ainsi qu'un rapport. Le rapport contiendra une brève description du problème (1/4 du rapport max), une brève description de ce que vous avez implémenté (1/4 du rapport max) ainsi qu'une évaluation des performances du protocole étudié. La troisième séance de TP sera consacrée aux projets. Le travail est à effectuer par groupe de 4 maximum. Le travail est **à rendre pour le 11 mars**.

Consignes :
- Le simulateur est à écrire dans un langage de votre choix parmi C/Java/C++ (si vous préférez le faire dans un autre langage, merci de me le demander).
- Vous devez rendre une archive contenant un **Makefile** ou des **instructions simples** permettant de compiler votre code. Le code doit compiler du premier coup. Un code qui ne compile pas vaut 0.
- Le rapport devra faire **4 pages maximum** en police 10pts (-2pt par page supplémentaire).
- Le rapport est à rédiger dans le format de votre choix (latex, openoffice,...). Ne rendre qu'une version pdf.

Quelques remarques :
- Les différentes questions posées (sur les graphiques ou l'évaluation de performance) ne sont qu'une indication pour vous aider à rédiger votre rapport. Vous êtes encouragés à vous poser d'autres questions et à critiquer vos résultats.
- La limite de pages doit vous pousser à être clair et concis et à faire des choix sur les résultats que vous souhaites présenter.
- Penser à justifier les choix que vous avez faits et à discuter les résultats (est-ce le bon indicateur ? les intervalles de confiance sont-ils bons ? . . .)
- Penser à inclure des graphiques clairs et informatifs (voir la *checklist for good graphics*).
- Environ 2/3 des points sera consacré au rapport, 1/3 au code lui-même (fonctions implémentés, qualité et clarté du code,...)

Projet : Instabilité d'Aloha

On se propose d'étudier la performance du protocole de communication Aloha. On s'intéressera en particulier aux problèmes d'instabilité. Le protocole Aloha est le premier protocole à accès multiple (ce protocole fait partie des protocoles MAC : Medium Access Control). Il a été mis en place pour un réseau de communication par paquets sur un réseau sans fil sur les îles Hawai. L'objectif est de réussir à partager une fréquence unique pour toutes les communications.

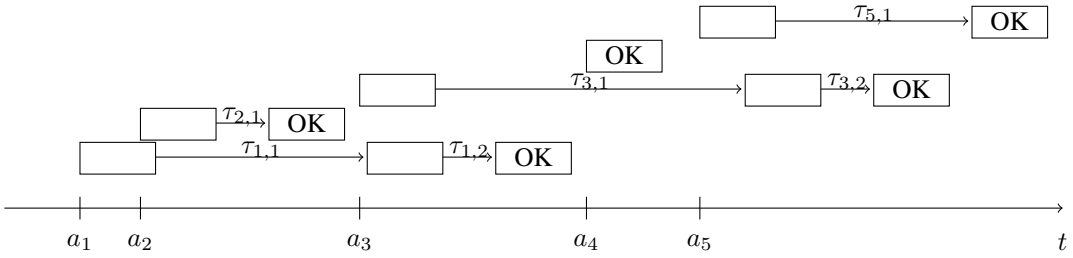
Aloha est un protocole distribué. Quand un utilisateur du réseau a un paquet à transmettre, il émet un signal dont la durée dépend de la taille du paquet qu'il veut émettre. S'il est le seul à émettre un signal pendant ce temps, les données sont transmises avec succès. Si une autre station tente d'émettre un paquet pendant ce temps, il y a alors collision et les deux paquets sont perdus. Les deux stations en question attendent alors un temps (aléatoire) avant de retenter de transmettre leur paquet.

Chaque fois que deux ou plusieurs paquets ont été en collision, les informations correspondantes sont corrompues et perdues. Par conséquent, il doivent être ré-émis. Cette re-émission entraîne un surcoût dans l'utilisation de la bande passante, par conséquent le débit réellement utile (utilisé pour les paquets sans collisions), est inférieur à 100 %.

Les clients arrivent chacun leur tour dans le système. Quand un client arrive dans le système, il fonctionne selon l'algorithme suivant :

1. Tenter d'émettre.
2. S'il n'y a pas de collision, quitter le système.
3. S'il y a collision, attendre un temps aléatoire τ et recommencer à l'étape 1.

La figure ci-dessous illustre le fonctionnement du système. Les carrés représentent un paquet en train de transmettre. Quand deux carrés sont l'un au dessus de l'autre, il y a collision. Les dates $a_1, a_2, \ldots$ représentent les dates d'arrivées du client 1,2,. . .



On peut voir sur le shéma :
- Le client 1 arrive à la date a_1 et commence à émettre.
- Le client 2 arrive à la date a_2 et commence à émettre mais comme 1 n'a pas fini, il y a collision.
- Chacun a donc perdu son paquets et les deux clients attendent un temps aléatoire $\tau_{1,1}$ et $\tau_{2,1}$.
- Quand 2 recommence à émettre, il est seul. Son paquet est donc accepté.
- Quand 1 recommence à émettre, 3 est lui aussi en train d'émettre et il y a donc collision. Les client 1 et 3 attendent alors $\tau_{1,2}$ et $\tau_{3,1}$.

Exercice 1. Simulation

On utilise les notations et les hypothèses suivantes :
- a_n est la date d'arrivée du $n$ième client.
  - On suppose que le temps entre deux arrivée $a_{n+1} - a_n$ est aléatoire est suit une loi exponentielle de paramètre λ.
- Si le client i a subit j collisions, il attend un temps τ_{ij}.
  - τ_{ij} est aléatoire et suit une loi exponentielle de paramètre μ.
- Le temps pour émettre un paquet est constant égal à 1 seconde.
- On suppose que le système est initialement vide.

a. Écrire un simulateur à événement discret simulant le système décrit. Le simulateur sortira ses résultat en mode texte.

Exercice 2. Quelques courbes

a. Tracer l'évolution du nombre de clients dans le système en fonction du temps pour différentes valeurs de λ (.1, .5, 1 et 2 par exemple) et de μ.

b. Étudier la courbe.

Quelques soient les paramètres du système, on devrait observer deux phases dans le comportement du système. Au début le système marche à peu près puis au bout d'un moment, le nombre de paquets se met à grandir.

c. Pour différentes valeurs de λ, μ, tracer le nombre de paquets transmis effectivement en fonction du temps. Que peut on observer quand le temps devient grand ?

d. On fixe $\lambda = 1/5$. Calculer le nombre total moyen de paquets transmis dans le système en fonction de μ. Même question en fixant $\mu = 2$ et en faisant varier λ.

Exercice 3. Évaluation de performance

a. Caractériser le passage du régime stable au régime instable : au bout de combien de temps cela se passe-t-il ? Il y a-t-il une valeur seuil sur le nombre de clients ?

b. Afin de stabiliser le système, on peut imaginer plusieurs méthode :
- un système de remise à 0 du système : quand le nombre de paquets dépasse un serveur central envoie un message à tout le monde pour leur dire de détruire leur paquet en attente.
- Limiter le nombre d'utilisateurs

En avez vous d'autres à proposer ? Pouvez vous évaluer ces méthodes ?

c. Si à chaque fois qu'une connexion détecte une collision, elle décide d'attendre un temps 2 fois plus grand, est-ce que cela change vos résultats ?