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 : Medium Access Control : exponential backoff.

Lorsque deux personnes tentent de communiquer sur un même canal en même temps, il y a collision entre les deux et les données des deux utilisateurs sont généralement perdues. Pour palier à ce phénomène, de nombreux protocoles de communication comportent une partie chargée de répartir l'accès au canal de communication. On parle de protocole "MAC" (Medium Acces Control). Ces protocoles reposent soit sur une gestion centralisée des conflits (un serveur central décide de qui parle quand) soit décentralisée : quand une machine détecte un conflit, elle d'elle même de quand ré-émettre en fonction de ce qu'elle observe. Ce dernier mode est largement privilégié dans les réseaux de données actuels. Il présente à la fois l'avantage d'être plus facile à mettre en place (il n'y a pas besoin de mettre en place un serveur central) mais l'inconvénient d'être souvent moins efficace.

Dans ce projet, on va s'intéresser à l'algorithme Exponential Backoff, notamment utilisé dans des protocoles comme Ethernet ou 802.11. Le principe est simple : quand une station détecte une collision, elle choisit de ré-émettre le paquet au bout d'un temps choisit aléatoirement (qui vaut en moyenne τ). Si la station détecte une deuxième collision, elle attend un temps aléatoire de moyenne double 2τ puis 4τ s'il y a une deuxième collision, $8\tau \ldots$

On considère le modèle suivant :
- On considère que les stations arrivent dans le système avec 1 paquet à transmettre et un état 1. Le temps d'arrivée entre deux stations est distribué selon une loi exponentielle de paramètre λ.
- Quand une station arrive, elle tente d'émettre. La durée d'émission d'un paquet est de 1 seconde.
- Si les émissions de deux stations se recouvrent (même pendant 1 ms), il y a collision et les deux paquets sont perdus.
- Lorsqu'une station subit une collision, si elle est dans l'état i, elle décide de ré-émettre son paquet dans t secondes, où t est une valeur aléatoire distribuée selon une loi exponentielle de paramètre $2^i\tau$. Elle passe ensuite dans l'état $i + 1$.
- S'il n'y a pas de collision, la station quitte le système.

Exercice 1. Simulation

Écrire un programme permettant de simuler l'algorithme décrit ci dessus. Vous pouvez soit décider de simuler un nombre d'arrivée de paquets fixés, soit un temps fixé $T_{\max}$. Les paramètres λ, τ, $T_{\max}$ seront à faire varier.

Exercice 2. Étude graphique

Dans un premier temps, on fixera $\tau = 1$. On pourra ensuite essayer de faire varier τ pour voir si on obtient d'autres résultats.

a. Tracer l'évolution du nombre de stations ayant un paquet à émettre au cours du temps pour différentes valeurs de λ (par exemple .2, .5, 1, 2).

b. On note $n(t)$ le nombre de paquets transmis avec succès à l'instant t. Tracer $n(t)/t$ en fonction de temps pour plusieurs valeurs de λ.

c. Normalement pour une valeur de λ donnée, $n(t)/t$ tend vers une valeur, notée $d(\lambda)$. Tracer $d(\lambda)$ en fonction de λ.

d. On note $E(\lambda)$ le nombre d'émissions de paquets par seconde (comprenant à la fois les émissions qui sont des succès et les collisions) pour un λ fixé. Tracer $E(\lambda)$ et $E(\lambda)/\lambda$.

e. On note $W(\lambda)$ le temps d'attente d'une station entrant dans le système (*i.e.* le temps total entre le moment où la station arrive dans le système et le moment où la station a transmis son paquet avec succès. Tracer la distribution de $W(\lambda)$ ainsi que sa moyenne pour différentes valeurs de λ.

Exercice 3. Évaluation de performances

a. Quand λ est trop grand, le système n'arrive pas à traiter suffisamment de clients : le système devient instable. Quelles sont les conditions de stabilité du système ?

b. Si il y avait une contrôleur qui pouvait décider qui émet quand, le système aurait un débit de 1 paquet par seconde au maximum. Ici, on voit que c'est moins. Quelle est l'efficacité maximale du système ?

c. Dans l'algorithme, une station qui subit une collision attend un temps exponentiel de moyenne $2^i\tau$. On remplace ce temps d'attente par une valeur tirée au hasard uniformément parmi $0, \tau \ldots 2^{i+1}\tau$, cela affecte-t-il les performances du système ?