

## TP : Simulation de protocole de rendez-vous et de Broadcast

Les objectifs de ce TP sont :

- de réaliser une synchronisation entre plusieurs machines.
- d'évaluer l'influence des communications sur le temps de synchronisation.
- de réaliser des transferts de données efficaces.

Tous les fichiers nécessaires à la réalisation du TP sont accessibles par le lien broadcast sur la page : <http://moais.imag.fr/membres/jean-noel.quintin/Teaching.html>

### 1 Prise en main de simgrid

Simgrid est un outil pour la simulation. Cet outil permet de calculer les temps de communications sur un réseau en prenant en compte les contentions. De plus, la simulation est rapide et facile à mettre en place en comparaisons des expériences réelles.

Pour simuler le comportement du réseau, Simgrid utilise un fichier de description la plate-forme souhaitée. Dans la suite, nous vous fournissons des fichiers de plate-forme et un générateur pour observer le comportement de vos algorithmes.

Nous vous proposons, d'utiliser l'interface MSG de simgrid. Cette interface permet de spécifier le code exécuté par chacune des machines. L'utilisateur code une ou plusieurs fonctions prenant des paramètres de type argv et argc (comme un main classique d'un programme). En plus du fichier de plate-forme, l'interface MSG utilise un fichier de déploiement, qui spécifie les fonctions exécutées par chacune des machines et les paramètres fournis aux fonctions.

Dans l'archive fourni du TP, vous trouverez :

- un exemple appelé sendrecv.c qui réalise l'envoi d'un message entre deux machines.
- un exemple appelé communication\_asynchrone.c qui réalise l'envoi asynchrone de messages entre plusieurs machines.
- la librairie Simgrid avec la documentation.

Il suffit de taper make pour compiler l'exemple. Pour plus d'information sur Simgrid, la documentation est mise à votre disposition dans l'archive ou sur le site web :

<http://simgrid.gforge.inria.fr>.

### 2 Synchronisation entre processeurs

Dans cette partie du TP, nous vous demandons de créer un point de rendez-vous entre les processeurs à partir des différents algorithmes vus en cours.

**Question 1** *Programmez un exemple dans lequel une machine définie et connue des autres, reçoit un message de chacune des machines, et leurs en renvoi un.*

**Question 2** *Programmez l'algorithme dans lequel chaque machine envoie un message à toutes les machines puis attend de recevoir un message de chaque machine.*

**Question 3** *Dans cette question, chaque machine est un noeud d'un arbre binaire. Dans un premier temps, chaque machine attend un message de chacun de ses fils s'ils existent. Une fois les deux messages reçus, chaque machine envoie un message à son père et attend sa réponse. Pour finir chaque machine envoie un message à ses fils.*

*Testez cet algorithme.*

**Question 4** *Observez le comportement de chacun des algorithmes.*

*Détaillez le temps d'exécution et le nombre de messages envoyés.*

### 3 Broadcast

Dans cette partie, une machine a toutes les données à transférer sur toutes les autres machines. Inspirez vous des algorithmes présentés dans la figure broadcast.gif pour en programmer un.

Pour plus d'information sur les algorithmes implémentés dans BitTorrent vous pouvez lire : Multicast Optimizing Bandwidth