TD/TP Diffusion causale, horloge de Mattern et délivrance causale
Durée estimée : 3h

Ce TP fait suite au TP précédent. Si ce dernier n'a pas été fait, tout se retrouve dans le présent sujet. L'objectif est de mettre en oeuvre l'algorithme de la diffusion causale et l'horloge de Mattern pour assurer la délivrance causale.

Indications pour la réalisation de ce TP :

- Le graphe d'interconnexion est quelconque et est bidirectionnel. Utiliser votre programme P_{config} mis en oeuvre lors des précédents TP. Si vous n'avez pas un tel programme, commencer par une configuration manuelle du réseau en impliquant jusqu'à 4 processus. Cette alternative ne vous dispense en aucun cas de la nécessité d'avoir un programme automatisant la construction d'un réseau quelconque.
- N'importe quel processus peut diffuser un message à n'importe quel moment. Pour faire simple, un processus P_i effectuera une diffusion d'un message toutes les $intervalle_i$ secondes (donnée passée en paramètre du processus P_i).
- Choisir des structures avec des types simples pour les messages échangés.
- Utiliser l'horloge de Mattern pour estampiller les évènements du système. Seuls les évènements de type envoi et réception sont à estampiller. Remarque : la diffusion d'un message nécessitera plusieurs opérations élémentaires d'envoi, toutes doivent avoir la même estampille (voir le cours).
- Pour matérialiser la délivrance et faire la différence avec l'évènement de réception, un processus P_i affichera une notification de l'arrivée d'un message, mais n'affichera ce message qu'au moment de sa délivrance. Par conséquent, un message qui arrive dans le bon ordre sera affiché immédiatement après la notification de réception.
- Produire une trace d'exécution pertinente.
- Une fois les premiers tests de votre système effectués, produire une situation d'arrivée de deux messages dans le désordre et donc la remise dans l'ordre des messages (délivrance causale). Vous pouvez vous inspirer de l'exemple du cours avec 3 processus. Remarque : provoquer un tel test pourrait naturellement nécessiter d'adapter votre programme.
- Si vous souhaitez atteindre un niveau plus avancé, faire des tests en provoquant la panne d'un processus qui aurait commencé la diffusion mais ne l'aurait pas terminée. L'exécution doit pourvoir se poursuivre sans erreur après la disparition de ce processus et tous les processus restants doivent avoir reçu le message que ce dernier a commencé à diffuser.

Important : décomposer la réalisation de ce travail en plusieurs étapes et de manière incrémentale. Cela sera bien plus efficace.

Pour rendre ce TP plus attractif et être plus sensibilisé à un contexte réparti, travailler en groupe impliquant plusieurs personnes, chacune doit implémenter un processus P_i et l'exécuter sur son poste de travail. Une contrainte doit toutefois être respectée : définir un protocole d'échange commun pour que les différents processus se "comprennent".