Université de Batna 2 **Batna le 11 Janvier 2018**
Faculté des Mathématiques et d'Informatique
Département d'Informatique
Section : Master 1 IAM
Durée : 90 minutes

EXAMEN FINAL
MODULE : INFORMATIQUE REPARTIE

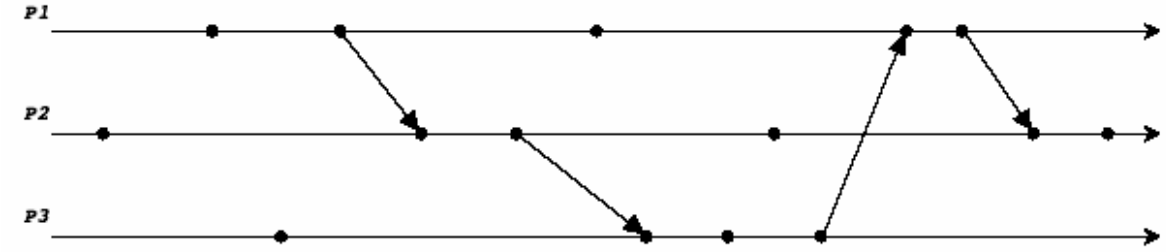
Exercices 01

Q1 : Démontrer que les horloges de Lamport ne sont pas équivalentes à la causalité.

Q2 : Démontrer qu'avec les horloges de Lamport , on ne peut pas avoir un ordre total sur les événement d'un calcul distribué.

Exercices 02

On considère trois processus coopérants qui réalisent un algorithme quelconque et s'échangent des messages. On s'intéresse à l'exécution de la figure ci-dessous. Toutes les horloges sont initialisées 0.



Q1 : Donnez les horloges logiques (Lamport) associées à chacun des événements, et les valeurs d'horloge véhiculées par les messages.

Q2 : Donnez les horloges vectorielles (Mattern) associées à chacun des événements, et les valeurs d'horloge véhiculées par les messages.

Exercice 03:

On considère le problème de la k-exclusion mutuelle qu'est une généralisation du problème de l'exclusion mutuelle. Cette généralisation consiste à imposer que, sur les ***N*** processus du système réparti, seuls **k≥1** processus peuvent être en section critique simultanément. (Dans l'exclusion mutuelle normale **k= 1**).

Pour traiter ce problème, nous proposons de modifier l'algorithme de Ricart et Agrawala. La modification proposée est la suivante : on attend toujours ***(N – 1)*** autorisations, mais les processus qui sont en section critique (ou qui attendent d'y rentrer) peuvent distribuer au maximum ***(K – 1)*** autorisations avant de stocker les demandes dans la file d'attente comme dans l'algorithme original.

Chaque processus ***Pi*** qui demande l'entrée en section critique initialise donc un compteur ***compteur$_i$*** à 1. Ce compteur représente le nombre de processus qui ont demandé l'entrée en section critique après lui et qu'il a autorisé à y rentrer (lui compris).

Lorsqu'un processus ***Pj*** demande l'autorisation d'entrer en section critique à un processus ***Pi*** qui est en section critique (ou en attente de section critique), il y a deux possibilités :

1- Soit la demande de ***Pj*** précède celle de ***Pi***, auquel cas ***Pi*** envoie une permission à ***Pj*** sans modifier ***compteur$_i$.***

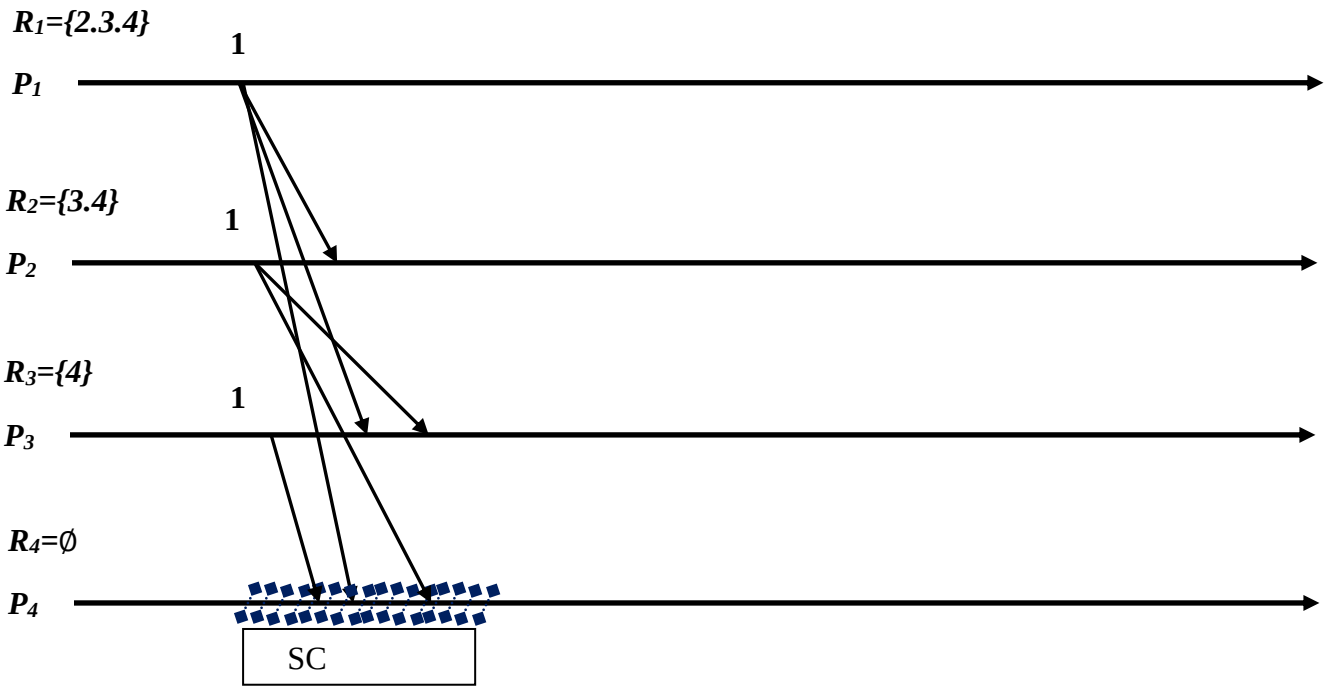
2- Soit la demande de ***Pi*** précède celle de ***Pj*** :

- Si ***compteur$_i$*** **< K** alors ***Pi*** envoie une permission à ***Pj*** et incrémente ***compteur$_i$***.
- Si ***compteur$_i$*** **= K** alors ***Pi*** met la demande de ***Pj*** dans sa file d'attente comme dans l'algorithme original.

Q1 : En se basant sur l'algorithme de Ricart et Agrawala, écrire un algorithme qui prend en charge cette modification.

Q2: Est-il possible de résoudre le problème de la k-exclusion mutuelle en modifiant l'algorithme de Carvalho-Roucairol ? Si oui, expliquer le principe à suivre pour modifier l'algorithme de Carvalho-Roucairol.

Exercice 03:



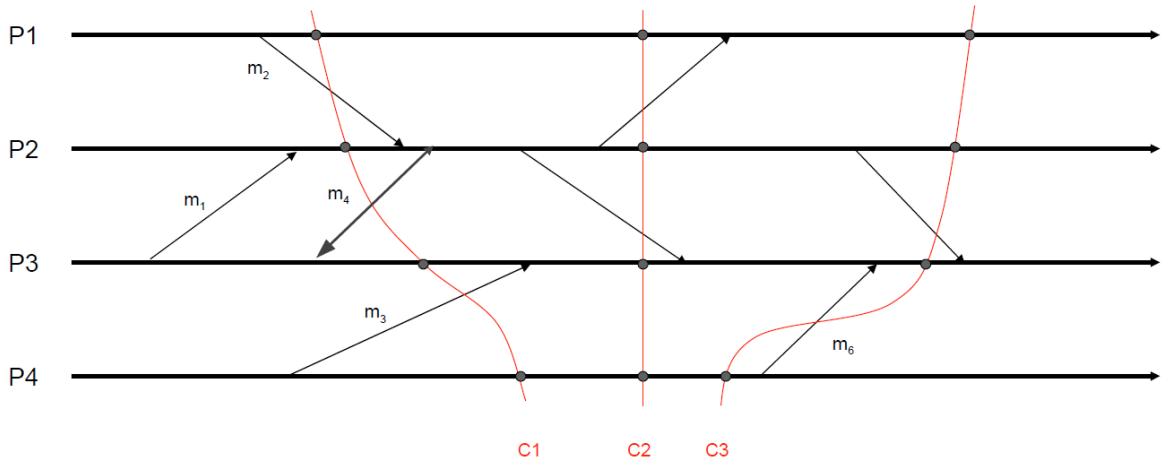
Exercice 03:

Exercice 04:

Q1 : Pourquoi l'état global d'un système reparti ne peut-il se réduire à la somme des états locaux des processus qui le composent ? Donner un exemple concret.

Q2 : Définir la notion de coupure cohérente. Pour l'exécution d'un calcul réparti proposé dans la figure ci-dessous, identifier les coupures cohérentes. Justifier votre réponse.

Q3 : Expliquer pourquoi l'algorithme de Chandy et Lamport 85 pour déterminer un état global se termine.



GOOD LUCK