Conception des systèmes répartis

3ième Année Informatique et Mathématiques Appliquées
Durée : 2 heures (Documents autorisés : Précis et notes de cours)

Décembre 2000

Remarque préliminaire : Toutes les questions valent 2 points.

1 Causalité (8 points)

On considère le chronogramme suivant :

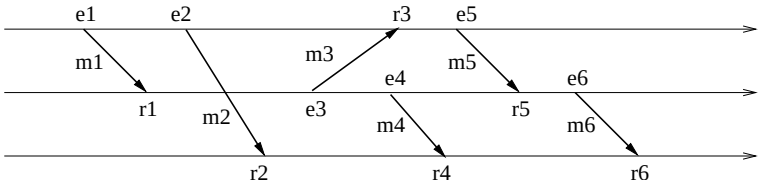

FIG. 1 – Chronogramme 1 : Causalité

Questions

1. Parmi les e_i, trouvez deux événements d'émission qui ne sont pas causalement liés.
2. Précisez si les événements r_4 et r_5 sont causalement liés ou pas. Justifiez votre réponse.
3. Trouvez l'ensemble des événements de réception parmi les r_i vérifiant les deux propriétés suivantes : ils sont postérieurs à r_2 dans le temps absolu MAIS ils ne sont pas causalement liés à r_2 (autrement dit on n'a pas $r_2 \prec r_i$).
4. Dans un protocole point à point d'ordre causal (ordonné), les messages m_2 et m_4 peuvent-ils être délivrés au moment de leur réception, c'est-à-dire aux instants r_2 et r_4. Justifiez votre réponse.

2 Datation(10 points)

2.1 Horloges de Lamport

On considère le schéma de datation par des horloges de Lamport. La classe *date* ci-dessous rappelle la représentation d'une date logique sous la forme d'un couple $(site, compteur)$:

```
class Date {   // définition et comparaison de date
   protected int s ; protected int cpt ;
   boolean Avant ( Date d )  {
      return (this.cpt < d.cpt) ||( (this.cpt == d.cpt) && (this.s < d.s)) ;
   }
}
```

Chaque site possède une horloge locale H_s gérant une date logique sur laquelle les opérations d'incrémentation *Top* et de recalage *Recaler* peuvent être exécutées :

```
class Horloge extends Date {
   // Création de l'horloge
   Horloge( int où )  { s = où ; cpt = 0 ; }

   // Lecture-Incrémentation de l'horloge
   Date Top()  throws Exception
       { Date dc = (Date) super.clone(); this.cpt++; return dc ; }

   // Recalage de l'horloge
   void Recaler( Date d ) { if (this.Avant(d) )  this.cpt = d.cpt + 1 ; }
}
```

Enfin, les actions de mise à jour de ces horloges sur les différents types d'événements sont les suivantes :

Type d'événement sur un site s	Action mettant en jeu l'horloge du site s
Événement interne sur s	$H_s.Top()$
Émission sur s de m	$dm = H_s.Top()$; envoi de $< dm, m >$
Réception sur s de $< dm, m >$	$H_s.Recaler(dm)$

FIG. 2 – Tableau des actions de mise-à-jour des horloges

Question

5. Décorez le chronogramme (3) en précisant la valeur de chaque horloge locale aux instants marqués par des points d'interrogation et en précisant la date surchargeant chaque message ;

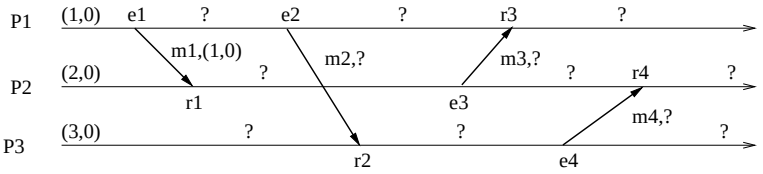

FIG. 3 – Chronogramme 2 : Datation

2.2 Horloges simplifiées

On essaie de définir un mécanisme de datation avec seulement un compteur local par site (on élimine la composante site des horloges de Lamport). On veut que le mécanisme garantisse cependant :

$$\forall e, e' \ : \ e \prec e' \ \Rightarrow \ d_e < d_{e'}$$

Questions

6. Modifiez les classes Java *Date* et *Horloge* pour ce mécanisme simplifié de datation globale.
7. Précisez, sous forme d'un tableau similaire au tableau (2.1), les actions de datation.
8. Quelle propriété est cependant perdue par rapport aux horloges de Lamport ? Justifiez votre réponse.

2.3 Horloges de Mattern

Les horloges de Mattern permettent de dater les événements d'un calcul réparti en assurant la propriété fondamentale exprimée sous forme de l'équivalence suivante :

$$\forall e, e' \ : \ e \prec e' \ \equiv \ d_e < d_{e'}$$

On suppose que, pour une application donnée, seuls les événements internes aux sites et les émissions ont besoin d'être datés. On envisage donc de ne plus comptabiliser les événements de réception en supprimamt le top lors de l'opération de recalage. Autrement dit, l'opération de recalage devient :

```
void Recaler( DateV D ) {
   for (int i=0 ; i < cpt.length ;i++)
         this.cpt[i] = Math.max(this.cpt[i],D.cpt[i]) ;
   //  en commentaire : this.cpt[s]++ ;   // Top implicite
   }
```

Question

9. Cette modification garde-t-elle la propriété fondamentale des horloges de Mattern ? Justifiez votre réponse.

3 Prise de cliché (2 points)

L'algorithme de Chandy et Lamport permettant de prendre un cliché cohérent d'un calcul réparti utilise l'hypothèse de canaux FIFO.

Question

10. Proposez une adaptation de l'algorithme permettant de lever cette hypothèse. Autrement dit, comment obtenir un cliché cohérent même avec des canaux non FIFO ?