M2 – SAR –
23/11/2007

Luciana Arantes, Pierre Sens

ARA

Exercice1 : Point de reprise (1 heure)

On considère l'algorithme de Chandy-Lamport dont le principe est le suivant :

- Le coordinateur diffuse un *marker* et fait son point de reprise
- Réception du marker sur p sur le canal c:
  - Si p n'a pas encore sauvegardé son état :
    - Rediffusion du marker et sauvegarde d'état

- Tout message reçu sur le canal c entre la sauvegarde et la réception du marker sur c est sauvé (message en transit)

1.1

Illustrez une configuration aboutissant à une incohérence si les canaux ne sont pas FIFO.

Soit la configuration suivante (les traits pointillés représentent les messages de type marker, les traits pleins les messages applicatifs, les rectangles noirs les points de reprise)

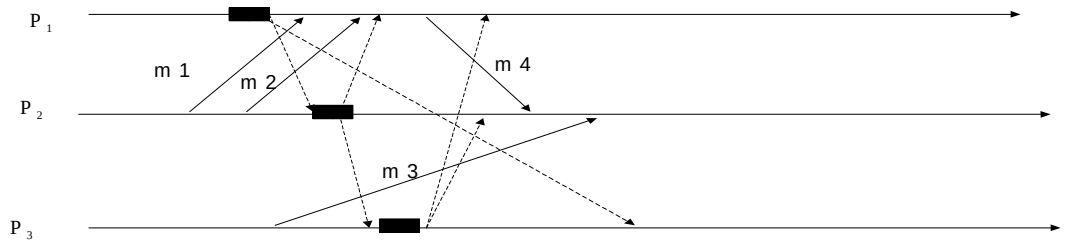

Figure 1

1.2

Quels seront les messages en transit sauvegardés sur le support stable

1.3

On souhaite améliorer l'algorithme pour réduire le nombre de messages à sauvegarder.

Proposez une version modifiée de l'algorithme qui limite le nombre de messages sauvegardés. La version modifiée ne doit pas générer de messages supplémentaires par rapport à l'algorithme initial.
Indication : l'idée de base est de retarder les sauvegardes des points de reprise de la façon suivante :

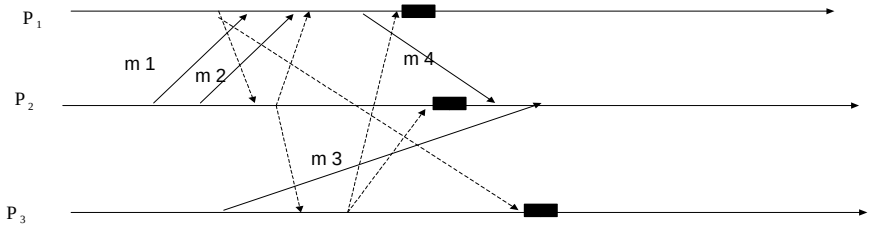

Figure 2

1.4

Dans la configuration de la figure 2, quels sont les messages en transit à sauvegarder.

1.5

Modifiez votre algorithme pour le message m4 ne soit pas sauvegardé et refaites le schéma.

Exercice 2 : Protocole de Diffusion Fiable Uniforme (45 min)

Considérez la spécification de la diffusion fiable uniforme :

- **Validité :** si un processus correct diffuse le message *m*, alors tous les processus corrects délivrent *m*
- **Accord uniforme** : si un processus (**correct** ou **fautif**) délivre le message *m*, alors tous les membres corrects délivrent *m.*
- **Intégrité uniforme**: Un message *m* est délivré au plus une fois à tout processus (**correct** ou **fautif**), et seulement s'il a été diffusé par un processus.

Supposez aussi l'existence d'un détecteur de défaillance **parfait** sur chaque processus *p* . Si un processus *q* tombe en panne, *p* est a terme informé et il n'y a pas de fausses suspicions:

Processus p :

upon <crash, q>

$correct_p = correct_p \setminus \{q\}$

Les canaux sont faibles et les processus sont susceptibles de subir de pannes franches. Le système possède au début N processus corrects : $\Pi = \{p_1, p_2, .. p_n\}$

2.1

Nous voulons offrir le pseudo code de l'algorithme de diffusion fiable uniforme. Est-ce qu'un processus peut délivrer un message *m* dès qu'il le reçoit ? Justifiez votre réponse.

2.2

Complétez le corps de la primitive **Unif_real_broadcast (m)** afin d'offrir une diffusion fiable uniforme et le code lorsqu'un processus reçoit un message (**upon event** recv(m, p_j)) afin d'offrir une délivrance du message *m* selon la spécification ci-dessus décrite. Vous pouvez ajouter d'autres variables locales si vous trouvez nécessaire ainsi que d'autres fonctions et/ou événements ou même changer le code du événement <carsh,q>.

Processus p_i :

Variables locales :

$correct_{pi} = \Pi$; /* Π = ensemble de tous les processus du système */

. . .

Unif_real_broadcast (m)

. . .

upon event recv(m, p_j) **do**

. . .

Real_deliver(m) /* délivrer le message */

upon <crash, p_j>

$correct_{pi} = correct_{pi} \setminus \{ p_j \}$

2.3

Considérez que *f* est le nombre maximal de pannes pouvant se produire (f < N). Est-ce que votre algorithme pourrait être optimisé ? Justifiez votre réponse sans donner un nouveau pseudo code.

Exercice 3 : Protocole de Diffusion Atomique et Causal (15 min)

Soit un groupe *fermé* composés de trois processus P_1, P_2 et P_3. Considérez que :

- A t= 0, P_1 diffuse dans le groupe le message m_1 et P_2 diffuse m_3
- A la réception de m_1, P_2 diffuse le message m_2 dans le groupe.

3.1

Quels sont les ordres de délivrance de messages possibles si on utilise :

- ABCAST (diffusion atomique) :
- CBCAST (diffusion causal) ;
- ACBCAST (diffusion atomique causal) ?