Université Denis Diderot Paris 7
UFR d'informatique
M2-algorithmique répartie

Examen d'algorithmique répartie (22 mars 2010)

(documents autorisés-1 page)

Soit $\Pi = \{p_1, \ldots, p_n\}$ un ensemble de processus communiquant par envoi, réception de messages. La communication est point à point, synchrone et sans perte de messages.

On suppose que les processus s'exécutent en *rondes synchronisées* : à chaque ronde r ($r = 1, \ldots, r = k, \ldots$) :

- chaque processus p envoie un message (le message de p pour la ronde r) vers tous les autres
- chaque processus reçoit les messages de la ronde r
- chaque processus change d'état suivant les messages reçus et détermine le message à envoyer à la ronde suivante.

Pour une ronde r, on notera, s_p^r le message envoyé par p à la ronde r, R_p^r les messages reçus par p dans la ronde r. Un processus peut tomber en panne (crash) au cours d'une ronde r, dans ce cas, certains processus peuvent recevoir son message s_p^r (plus précisément : $s_p^r \in R_q^r$) et d'autres peuvent ne pas le recevoir (plus précisément : $s_p^r \notin R_q^r$). Si p ne tombe pas en panne dans la ronde r on dira que p est *vivant* dans la ronde r, on notera $Viv(r)$ l'ensemble des processus vivants dans la ronde r, $Crash(r)$ sera l'ensemble des processus qui ne sont pas vivants dans la ronde r : $Crash(r) = \Pi - Viv(r)$. On ne considère ici que des pannes crash, aussi, on aura : $p \in Crash(r) \Rightarrow \forall r' \geq r : p \in Crash(r')$. Pour une exécution donnée $Correct$ est l'ensemble des processus qui sont vivants pour toutes les rondes.

On s'intéresse ici à des problèmes de *décision*. Pour cela, on suppose que chaque processus p a une valeur initiale v_p. Chaque processus a d'autre part une variable D_p qui ne peut être écrite qu'une seule fois. On dira qu'un processus p *décide* v s'il écrit v dans D_p. Un *algorithme de consensus* est un algorithme de décision qui assure les propriétés suivantes :

- **Terminaison :** tous les processus corrects décident,
- **Intégrité :** si un processus p décide v alors il existe $q \in \Pi$ tel que $v = v_q$,
- **Accord :** si p et q décident alors ils décident la même valeur.

On considère le protocole suivant où tous les processus échangent toutes les informations qu'ils ont.

Plus précisément, chaque p maintient une variable locale Val_p qui contient un ensemble de valeurs et :

- Initialement, $Val_p = \{v_p\}$
- A chaque ronde r, chaque processus p envoie Val_p ($s_p^r = Val_p$) et ajoute à Val_p les messages qu'il a reçus dans la ronde ($Val_p := Val_p \cup R_p^r$)

On notera Val_p^r la valeur de Val_p à la fin de la ronde r.

1. En supposant qu'il y a quatre processus p_1, p_2, p_3 et p_4, décrire une exécution dans laquelle p_1 et p_2 sont les seuls processus corrects et à la fin de la ronde 2, $Val_1 \neq Val_2$.
2. Montrer que si $Viv(r) = Viv(r+1)$ alors pour tout $(p,q) \in Viv(r+1)$, on a $Val_p^{r+1} = Val_q^{r+1}$.
3. Montrer que si $r \leq r'$ alors pour tout $p \in \Pi$, $Val_p^r \subseteq Val_p^{r'}$.
4. Montrer que si $Viv(r) = Viv(r+1)$ alors pour tout $r' > r$, pour tout $(p,q) \in Viv(r)$, $Val_p^{r'} = Val_q^{r'}$.
5. Montrer que pour tout r on a $\bigcup_{p \in \Pi} Val_p^r \subseteq \{v_1, \ldots, v_n\}$.
6. Si on suppose qu'au plus t processus peuvent tomber en pannes (quelque soit l'exécution, $n - |Correct| \leq t$), montrer que pour tout $(p,q) \in Viv(t+1)$ on a $Val_p^{t+1} = Val_q^{t+1}$. On rappelle le principe suivant, si on met k objets dans $k+1$ boîtes au moins une boîte sera vide !
7. En déduire un algorithme de consensus en supposant qu'au plus t processus peuvent tomber en panne. Montrer que l'algorithme proposé vérifie bien les propriétés de terminaison, d'intégrité et d'accord.
8. Donner une exécution de cet algorithme telle qu'au bout de t rondes ($t < n+1$) pour au moins 2 processus, p et q, on a $Val_p^t \neq Val_q^t$.
9. Si on ne connaît ni le nombre n de processus ni le nombre t de processus qui peuvent tomber en panne, comment pourrait-on procéder pour obtenir un algorithme de consensus.