Exercices Dea RSD – TC4
Partie Algorithmique répartie
2002/2003
Correction partielle Planche d'exercices 1

F. Baude

Exercice 1

Pour faire court ! La première version de Chang et Roberts permet que si les processus ne se réveillent pas tous, alors un leader est élu, mais, celui-ci a le numéro maximum parmi le sous -ensemble de processus qui ont pris part à l'élection. Pour la deuxième version, on peut constater que l'algorithme obligera tous les processus à prender part à l'élection; ainsi, le processus leader sera effectivement celui de plus grand numéro. Les 2 versions sont évidemment correctes, si il ne s'agit que d'élire un unique processus, quel que soit, finalement, son numéro !

Exercice 3

A partir de l'algorithme Echo présenté dans l'article de Mattern, écrire un algorithme permettant de résoudre l'élection de leader dans un graphe quelconque.

Voici un rappel de l'algorithme Echo de Mattern, puis, sa modification pour résoudre le problème de l'élection de leader dans un graphe quelconque.

3.1 L'algorithme ECHO

L'algorithme Echo est un algorithme distribué que l'on peut utiliser pour traverser un graphe connexe quelconque de manière parallèle et pour diffuser des données d'un initiateur à tous les processeurs. L'idée de base est qu'un seul processeur se déclenche et inonde la topologie avec une info et veut recevoir la confirmation que `tous` ont reçu l'info.

Cet algorithme peut servir à synchroniser des processeurs. Il généralise Chang & Roberts pour n'importe quelle topologie (physique ou logique).

Remarque :

- lorsque l'info fait le tour complet de l'anneau, on est bien dans le cas Chang & Roberts.
- c'est aussi un **algo total** (fait intervenir tous les processus).

Idée de l'algo Echo :

Le processus initiateur envoie un message à tous ses voisins. Ceux qui reçoivent l'info pour la première fois la transmettent eux-même à leurs voisins (moins le

père), mais aussi, mémorisent de quel lien vient l'info (père) pour pouvoir renvoyer un acquittement (un écho).
⇒ Cela correspond à la construction d'un arbre couvrant orienté sur la topologie.

Dès qu'un processus reçoit une info, il devient **engagé**, et propage l'info.
Lorsqu'un processus a reçu un écho ou une info de tous ses voisins, ce processus (autre que initiateur) est **désengagé**, et renvoie écho à son père.
Chaque processus a une variable initiateur initialisée à FAUX.

Un processus ayant reçu un message (info ou écho) de tous ses voisins, sait que tous ses voisins ont reçu l'info, et donc fait remonter écho vers son père.
L'algorithme Echo est terminé lorsque l'initiateur a reçu un message (info ou écho) de tous ses voisins. A ce moment là, il est sûr que tous les processus ont eu l'info.

P$_i$

I_i : {¬engagé et leader = moi} // au plus un processus exécute son I_i

initiateur := $Vrai$;
engagé := $Vrai$;
$Nl = 0$; // compte le nb de voisins
envoyer $< info >$ aux voisins.

X_i : {un message est arrivé de P} // message est de type info ou echo

si ¬engagé alors // mess = info
engagé := $vrai$;
$Nl := 0$;
$PRED := P$;
envoyer $< info >$ à $\{voisins - PRED\}$
$Nl := Nl + 1$;
si $Nl = |voisin|$ alors
engagé := $Faux$;
si initiateur alors $< terminé >$
sinon, envoyer $< echo >$ à $PRED$

Pour cet algorithme on suppose qu'un seul processus devient initiateur. Il suffit d'avoir élu précédement un `leader`.

Cet algorithme peut être utilisé pour :

- Construire un arbre couvrant sur une topologie quelconque.
- Faire de la synchronisation entre les processus ↪ application de type maître-esclave.
- Pour mettre en œuvre la détection de la terminaison.

Complexité

- Nombre total de messages avec **un** initiateur

  Au plus 2 messages transitent par lien. Si chaque processus a au plus E voisins (il y a N processus), le nombre total de messages est $2 \times E \times N$.

- Complexité en temps

  Le nombre maximum d'enchaînement de messages est au pire, $2 \times N$

3.2 Application de Echo pour l'élection du leader

Un algorithme basé sur echo peut élire un leader. Il suffit d'appliquer une purge des messages que l'on reçoit et qui portent comme identificateur une identité de processus n'ayant aucune chance d'être élu.

Idée : On suppose que tous les processus se réveillent spontanément. Chaque processus commence un algo echo en envoyant son identité. P_i prend en compte un message portant l'identité K seulement si $K > i$.
Si P_i a reçu un message portant l'identité j ($j > i$), P_i ne prend en compte un message portant l'identité K que si $K > j$.
De ce fait, seulement l'identité la plus grande est prise en compte par tous les processus, et donc seulement le processus d'identité la plus grande termine l'algo echo.
A ce moment là, il va informer les autres par un message $< leader-trouvé, moi >$.
Et les autres vont terminer dès qu'ils ont reçu un message $< leader-trouvé, moi >$ de tous leurs voisins.

Algorithme Echo :

P_i

I_i : {¬engagé } //tous le font

engagé = $vrai$;
$Nl = 0$; // compte le nb de voisins
envoyer $< chercher - leader, i >$ aux voisins;
$max := i$;

X_i : {message provenant de p arrive}∧ {¬terminé}∧ {engagé} //message de type info ou echo

case of :
$message - type =< chercher - leader, K >$
si $K > max$ alors // nouveau max
$max := K$;
$PRED := P$;
$Nl := 0$;
envoyer $< chercher - leader, K >$ aux $\{voisins - PRED\}$
si $K \geq max$ alors // 2 cas : mess indiquant un nouveau max, ou identité de k avait déjà été reçue
$Nl := Nl + 1$;
si $Nl = |voisin|$ alors

```
si K = i   alors                                   // c'est moi le max
        envoyer < chercher − leader,i > aux voisins;
        leader − trouvé := vrai;
        leader := i;
        Nlead := 0
sinon
        envoyer < chercher − leader,K > à PRED;
message − type :=< chercher − leader,K >;
si ¬leader-trouvé   alors                  // je viens de connaître le leader
        leader − trouvé := vrai;
        leader := K;
        Nlead := 0;
        envoyer < leader − trouvé,K > aux voisins;
Nlead := Nlead + 1;
si Nlead = |voisin|   alors   terminé := vrai
```

Exemple : Je vous laisse appliquer cet algo sur le graphe donné dans l'énoncé de l'exercice.
4 sites : a, b, c, f, et les canaux de communication bi-directionnels suivants : ab, ac, af, cf. Supposez que l'ordre lexicographique des identités associées aux sites permet de déterminer le max (à savoir ici, f). Supposez que tous les sites se réveillent spontanément.

Complexité :
Si on ne compte pas les messages qui finalement seront détruits, on a au plus $2 \times E \times N$ messages (algo echo initié depuis le processus de plus grand numéro), plus la diffusion finale (i.e. $2 \times E \times N$).
C'est la situation la plus favorable.