

TP Election à base de vagues écho

Question 1

L'algorithme d'élection est à insérer dans le code du serveur.

Le serveur a besoin d'analyser le contenu des messages, pour notamment savoir si le message reçu est un Jeton d'élection ou un jeton gagnant.

Les clients n'ont pas à être importunés avec les messages d'élection. (Ils ne participent pas).

Question 2

L'algorithme de vague écho est écrit sous la forme orientée controle dans notre polycopié.

Le voici sous sa forme événement :

```
Init:
Init p : #p, Init rec, lrec : 0, Init state : indecis, Init father, win, caw : Undefined

Pour lancer l'élection
# some init
Init win, father : undefined
Init rec, lrec : 0
Init state : indecis
send Jeton( p ) to all neighbours

On reception( m )
if m contain Jeton( r )
    if caw == undefined
        #First message received by a non candidate process. We need to do some init
        # init for new wave
        Init rec, lrec : 0, Init state : indecis, Init father, win : Undefined
    endif
    if caw == undefined || r < p
        caw = r
        rec = 0
        father = emmeteur( m )
        send m to all neighbours except father
    endif
    if caw == r
        increment rec
        if rec == #Neigh
            if caw == p
                send GAGNANT( p ) to all neighbours
            else
                send Jeton( r ) to father
            endif
        endif
    endif
else
    # m contain Gagnant( r )
    if lrec < #Neigh
        if lrec == 0 || r != p
            send GAGNANT( r ) to all neighbours
        endif
        lrec ++
        win = r
        if lrec == #Neigh
            if win == p
                state = gagnant
                launch actions
            else
                state = perdant
            endif
            # to allow us to launch a new election
            caw = undefined
        endif
    endif
endif
```

Nous échangeons des messages de la classe ElectionToken. Nous savons que nous avons à faire à un message ElectionToken car le message est précédé de 4 octets (ajoutés à l'envoi) qui précisent son type.

Les actions sont lancés par le gagnant en fin d'élection.

Dans notre cas, il lui incombe d'informer chaque serveurs des autres serveurs présents dans la topologie, ainsi que de collecter la liste des pseudos afin de pouvoir envoyer les join / leave notifications liées au changement de topologie.

Question 3

Voici nos variables :

- rec : nombre de jetons reçus pour la vague en cours
- caw : identifiant du processus de la vague en cours
- lrec : nombre de réponses reçues pour la vague gagnante
- state : état électoral du serveur
- win : identifiant du vainqueur
- father : un moyen de communication avec son parent direct
- #Neigh : nombre de voisins directs

Question 4

L'algorithme est lancé quand :

- On le demande explicitement sur l'entrée clavier du serveur
- On a un changement de notre topologie serveur : ajout ou retrait d'un ou plusieurs serveurs

Question 5

Nous nous basons sur l'adresse du socket du serveur pour identifier nos serveurs de manière unique sur notre réseau. Pour comparer ces adresses de socket, nous comparons simplement la chaîne de caractère obtenue par l'appel à la méthode `toString()`.

Le gagnant est celui qui a la plus grande chaîne de caractère (au sens de `compareTo()`).

Question 6

Il existe une méthode d'envoi de messages : `NetManager.sendInterServerMessage`

Il existe une méthode permettant d'envoyer un message à tous nos voisins : `State.broadcastInterServerMessage`

Il existe une méthode pour envoyer un message à tous nos voisins sauf un : `State.broadcastInterServerMessageWithoutFather`

Les deux dernières méthodes s'appuient sur la première.

Pour gérer l'accès à la machine à état en exclusion mutuelle, nous allons identifier les méthodes appelé via les commandes clavier, et protéger les ressources accédées via des mutex.

Question 7

`#Neigh` est récupéré via la méthode `State.getNbConnectedServers()` qui récupère la cardinalité de l'ensemble des serveurs auxquels nous sommes connectés.

Question 8

J'autorise plusieurs exécutions de l'algorithme.

Question 9

Je l'ai testé sur les topologies suivantes :

X-----X

X---X
| /
| /
| /
| /
| /
X



X-----X-----X

X-----X-----X-----X

Je relance une élection à chaque changement de topologie. Il suffit de déconnecter un serveur, par exemple dans la topologie triangle, et les voisins lancent alors une élection en même temps (donc 2 candidats).