

Informe del seminari: Paxy

Alejandro Carol
Héctor Mañosas
Xavier Sedó

19, de Novembre, 2013

1 Introducció

L'objectiu d'aquesta pràctica és aprendre i implementar l'algoritme de Paxos per obtenir el consens en un sistema distribuït i millorar-ne la robustesa.

A la primera part de la pràctica es tracta d'entendre el funcionament de l'algoritme, sobretot conceptes com *voted*, *accepted*, *round*, *acceptor*, *proposer*... i experimentar amb ell.

A la segona part de la pràctica hem d'afegir tolerància a fallades mitjançant la persistència.

2 Feina feta

Finalitzar la implementació de l'algorisme Paxos, incorporant el codi restant al mòdul *acceptor* i *proposer*.

També hem dividit la funcionalitat del paxy en dues funcions "start_acceptor_node/2" i "start_proposer_node/0", que permet comprovar-ne el correcte funcionament distribuït, mitjançant dos nodes diferents.

Afegir tolerància a fallades emprant el mòdul *pers*, que afegeix persistència a disc, i modificant el codi del mòdul *acceptor* per aprofitar-la.

3 Experiments

i) Intenta introduir retards en l'*acceptador* (per a la instància, just després de rebre els missatges *prepare* i/o *accept*) i veure si l'algoritme encara acaba.

Hem provat amb delays de 20 ms a 3s i en tots els casos l'algorisme ha acabat.

ii) Intenta ignorar l'enviament de missatges *sorry* comentant les corresponents sentències a l'acceptador. Es podria arribar a un acord quan no s'envien missatges *sorry*?

Sí, s'arriba a un acord de totes maneres.

iii) Intenta aleatoriament la caiguda de missatges *promise* i/o *vote* en l'acceptador. Reporta l'algoritme respostes contradictòries?

Si no es perden molts paquets (< 50%) l'algoritme funciona bé, si se'n perden molts dona alguns problemes.

iv) Què passa si incrementem el nombre d'*acceptors* i *proposers*?

Que l'algoritme triga més a acabar degut a que els proposers triguen molt més a arribar a un consens, tot i que un cop ho han fet els acceptors es posen d'acord ràpidament.

v) Divideix el mòdul paxy en dos parts i fes les adaptacions necessàries per activar els *acceptors* i fer-los córrer en una màquina i els *proposers* en una altra.

Hem modificat el codi del paxy afegint les funcions “start_acceptor_node/2” i “start_proposer_node/0”.

“start_acceptor_node” té com a paràmetres un Seed i l’Id del procés que haurà d’iniciar els proposers, i quan ha engegat els acceptors avisa al node dels proposers i li envia la informació amb la que haurà d’engegar els proposers.

Tolerància a fallides

Simula una ruptura i reinicia utilitzant el procediment crash i observa com es recupera l’acceptor amb èxit:

Com podem observar l’acceptor c s’ha reincorporat després del crash conservant la informació que tenia.

```
1. erl (beam.smp)

[Acceptor d] set gui: voted {11,zorro} promise {11,zorro} colour {0,0,255}
[Proposer kilgore - Round] set gui: Round {12,kilgore} Proposal {0,255,0}
[Acceptor f] set gui: voted {12,bill} promise {12,kilgore} colour {0,0,255}
[Acceptor g] set gui: voted {12,bill} promise {12,bill} colour {0,0,255}
[Acceptor c] set gui: voted {12,bill} promise {12,kilgore} colour {0,0,255}
[Acceptor e] set gui: voted {12,bill} promise {12,kilgore} colour {0,0,255}
[Acceptor d] set gui: voted {12,bill} promise {12,kilgore} colour {0,0,255}
[Proposer kilgore - Collected] set gui: Round {12,kilgore} Proposal {0,0,255}
[Acceptor d] set gui: voted {12,kilgore} promise {12,kilgore} colour {0,0,255}
[Acceptor e] set gui: voted {12,kilgore} promise {12,kilgore} colour {0,0,255}
[Acceptor g] set gui: voted {12,bill} promise {12,kilgore} colour {0,0,255}
[Acceptor b] set gui: voted {12,bill} promise {12,kilgore} colour {0,0,255}
[Acceptor c] set gui: voted {12,kilgore} promise {12,kilgore} colour {0,0,255}
4> paxy:crash(c, 1233).
Acceptor with name c crashed
Acceptor with name c is up again
true
[Acceptor f] set gui: voted {12,kilgore} promise {12,kilgore} colour {0,0,255}
[Proposer kilgore - After Ballot] [a,b,c,d,e,f,g] decided {0,0,255} in round {12,kilgore}
[Acceptor b] set gui: voted {12,kilgore} promise {12,kilgore} colour {0,0,255}
[Acceptor d] set gui: voted {11,zorro} promise {12,bill} colour {0,0,255}
[Acceptor g] set gui: voted {12,kilgore} promise {12,kilgore} colour {0,0,255}
[Proposer willard - Round] set gui: Round {12,willard} Proposal {0,0,255}
[Acceptor d] set gui: voted {12,bill} promise {12,bill} colour {0,0,255}
[Acceptor d] set gui: voted {12,bill} promise {12,kilgore} colour {0,0,255}
[Acceptor e] set gui: voted {12,kilgore} promise {12,willard} colour {0,0,255}
[Acceptor a] set gui: voted {12,kilgore} promise {12,willard} colour {0,0,255}
[Acceptor f] set gui: voted {12,kilgore} promise {12,willard} colour {0,0,255}
[Acceptor d] set gui: voted {12,kilgore} promise {12,kilgore} colour {0,0,255}
[Acceptor g] set gui: voted {12,kilgore} promise {12,willard} colour {0,0,255}
[Proposer willard - Collected] set gui: Round {12,willard} Proposal {0,0,255}
[Acceptor g] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
[Acceptor b] set gui: voted {12,kilgore} promise {12,willard} colour {0,0,255}
[Acceptor c] set gui: voted {12,kilgore} promise {12,willard} colour {0,0,255}
[Acceptor a] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
[Acceptor d] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
[Acceptor f] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
[Acceptor c] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
[Proposer willard - After Ballot] [a,b,c,d,e,f,g] decided {0,0,255} in round {12,willard}
[Acceptor d] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
[Acceptor f] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
[Acceptor c] set gui: voted {12,willard} promise {12,willard} colour {0,0,255}
```

Paxos Algorithm

Proposers	Acceptors
Proposer 1 Round: {11,zorro} Proposal: {0,0,255}	Acceptor 1 Round voted: {13,kurtz} Cur. Promise: {13,kurtz}
Proposer 2 Round: {12,bill} Proposal: {0,0,255}	Acceptor 2 Round voted: {13,kurtz} Cur. Promise: {13,kurtz}
Proposer 3 Round: {12,willard} Proposal: {0,0,255}	Acceptor 3 Round voted: {13,kurtz} Cur. Promise: {13,kurtz}
Proposer 4 Round: {12,kilgore} Proposal: {0,0,255}	Acceptor 4 Round voted: {13,kurtz} Cur. Promise: {13,kurtz}
Proposer 5 Round: {13,kurtz} Proposal: {0,0,255}	Acceptor 5 Round voted: {13,kurtz} Cur. Promise: {13,kurtz}
	Acceptor 6 Round voted: {13,kurtz} Cur. Promise: {13,kurtz}
	Acceptor 7 Round voted: {13,kurtz}

Altres millores

Canvia el codi dels procediments collect/4 i vote/2 per implementar la millora i prova com funciona.

Una vegada implementat i provat no hem pogut observar cap canvi en la velocitat de trobar el consens. Això possiblement es deu a que treballem amb una quantitat d’acceptors força reduïda, i segurament si la quantitat d’acceptors fos d’algun ordre de magnitud més gran seria possible percebre una millora.

4 Opinió personal

Paxos ha estat un algorisme força més desafiant d'implementar que els anteriors, on la dificultat principal era més entendre el seu funcionament, més que barallar-nos amb el llenguatge.

Hem vist un detallet a millorar, que és que a la pàgina 10 de la documentació hi ha un `if ?sorry`, on faltaria un cas `"true ->"` per evitar que un procés mori si definim `sorry` com a `false`.