Informe del seminari: Opty

Alejandro Carol
Héctor Mañosas
Xavier Sedó

5 de Desembre, 2013

1 Introducció

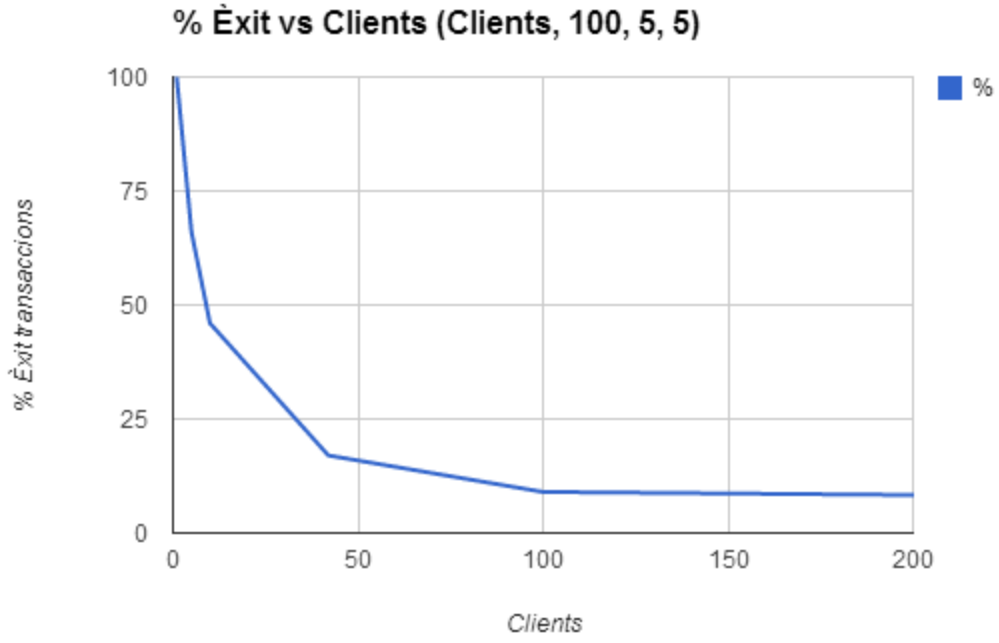
L'objectiu d'aquesta pràctica és aprendre i implementar l'algoritme de transacció del servidor usant el control de concurrència optimista. També implementarem una estructura de dades actualitzable en Erlang.

2 Feina feta

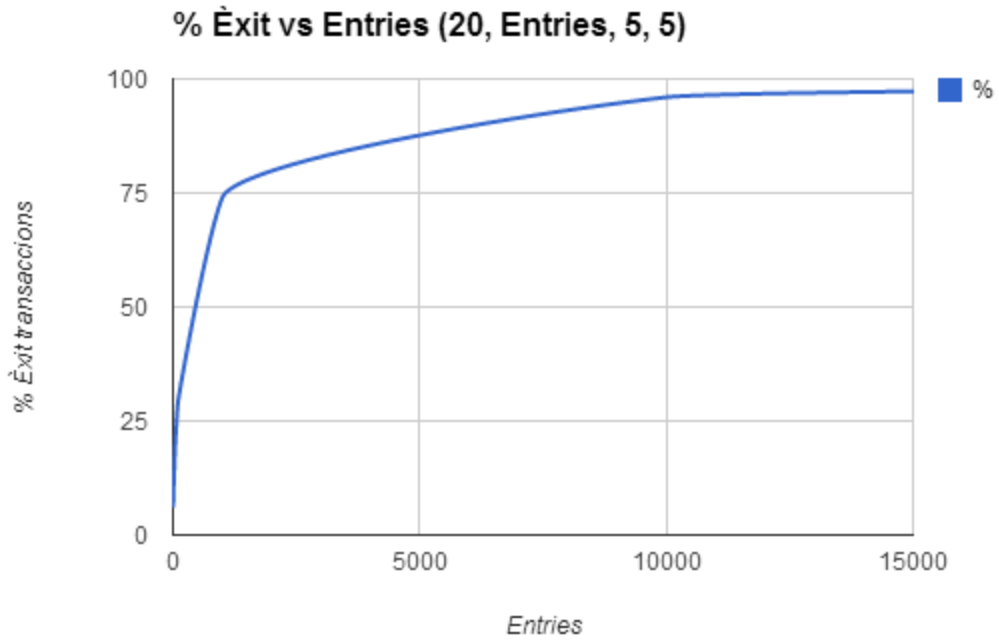
Finalitzar la implementació de l'algorisme Opty, incorporant el codi que faltava als mòduls: *entry*, *handler*, *validator* i *server*.
També adaptar el codi per a que el *Server* s'executi en una màquina i els clients en una altra diferent.

3 Experiments

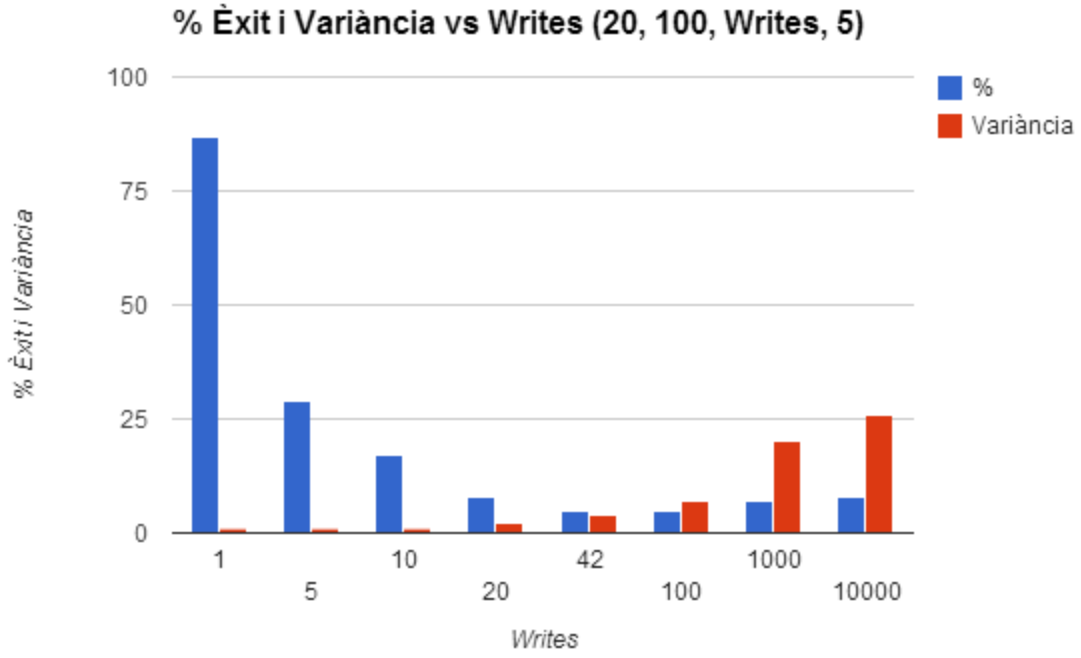
Primera part
i) Provar variacions en el nombre de clients:



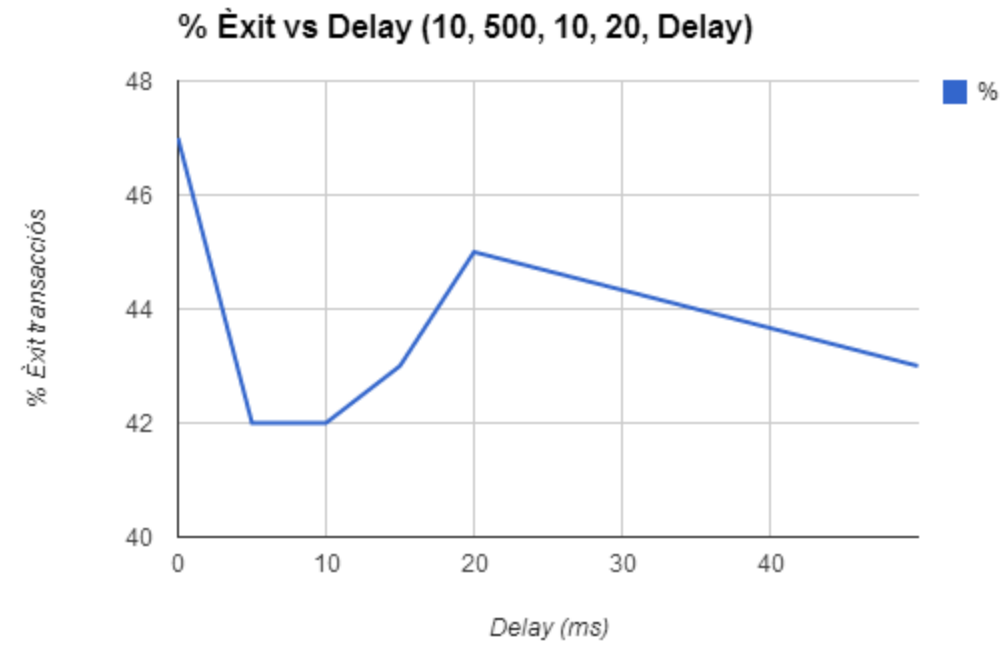
ii) Provar variacions en el nombre d'entries

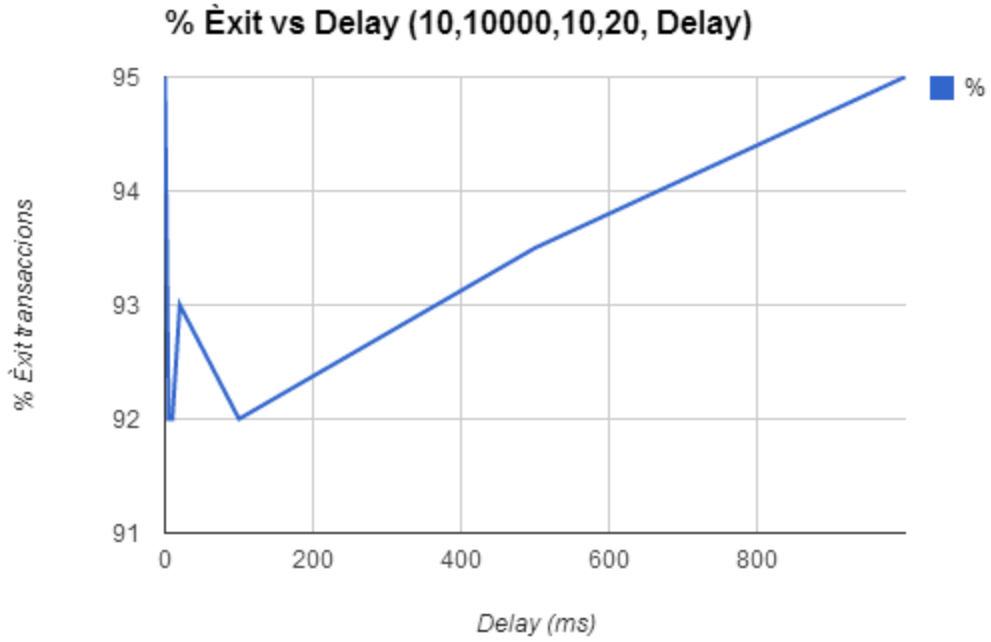


iii) Provar variacions en el nombre de transaccions de write

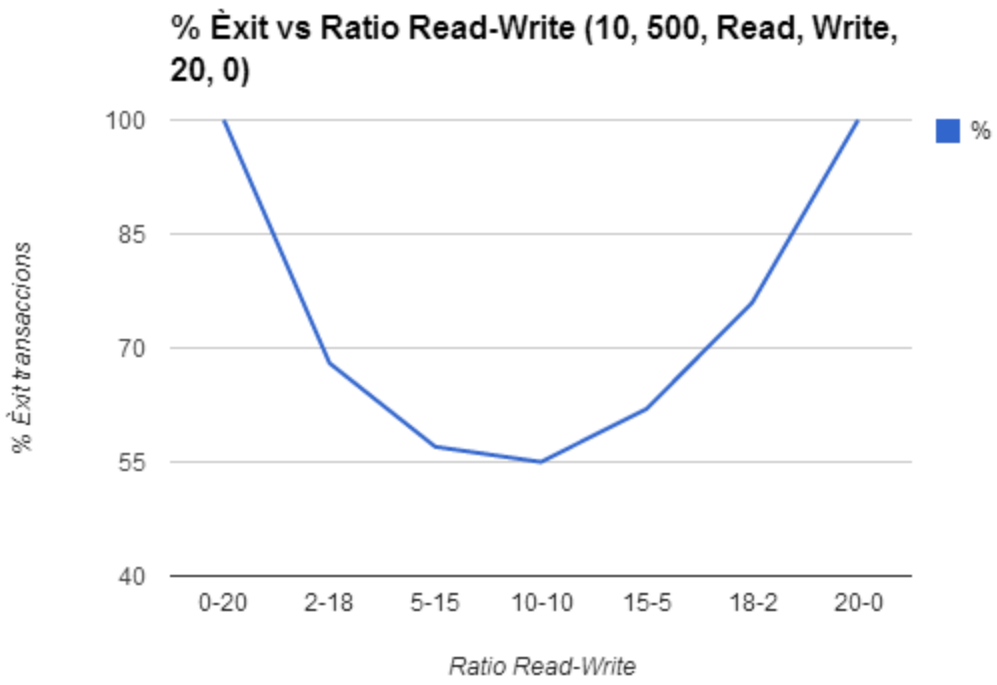


iv) Provar variacions en la durada de les transaccions (sleep)

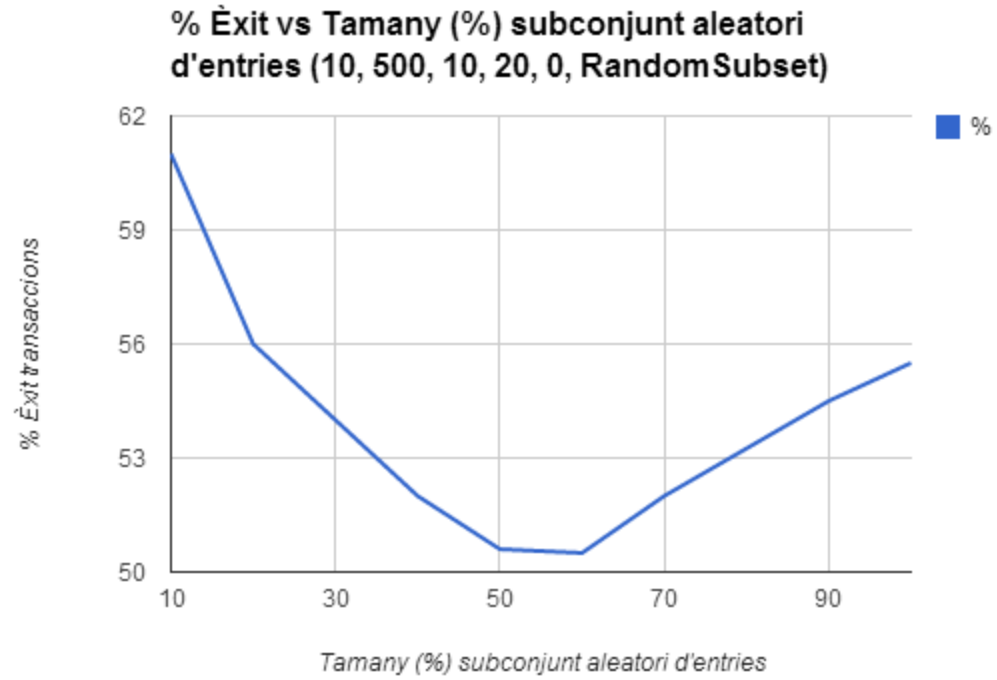




v) Diferents ratios d'operacions de lectura i escriptura per transacció (per defecte és 1)

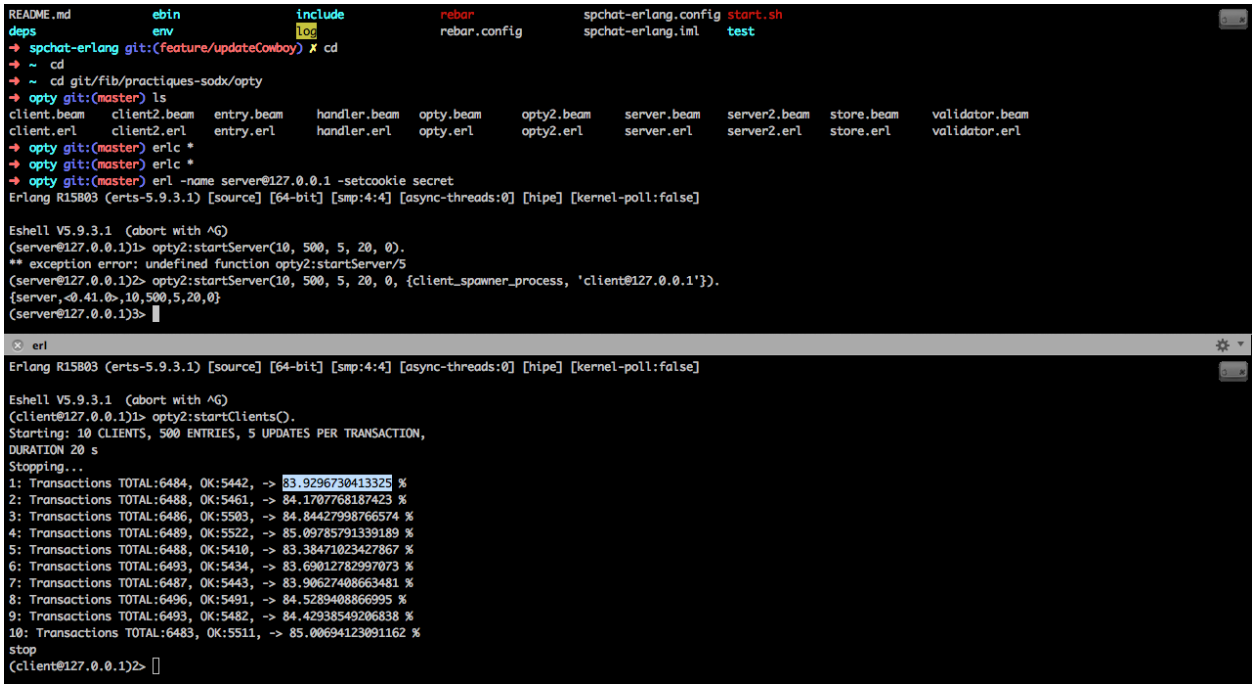


vi) Diferent percentatge d'entrades accedides respecte el nombre total d'entrades.



Segona part

Dividir el mòdul *Split* en dos parts i fer les adaptacions necessàries per a permetre que el servidor s'executi en una màquina i el clients en una altra diferent.



4 Preguntes obertes

Quin és l'impacte de cada un d'aquests paràmetres en la taxa d'èxit (per exemple en el percentatge de transaccions fetes respecte el total)?

Respecte la variable client, podem observar que quan augmenta el nombre de clients la taxa d'èxit disminueix considerablement. Això passa perquè augmenta la probabilitat de conflicte i per tant, també el nombre de transaccions fallides.

Respecte el nombre d'entries, podem observar que a poc que s'augmenti el nombre d'entries la taxa d'èxit augmenta considerablement. Així doncs, la probabilitat de col·lisió disminueix.

Respecte el nombre writes, podem observar que amb poc augment del nombre de writes, la taxa d'èxit disminueix considerablement. A partir de cert punt, a igual nombre de writes que d'entries, la taxa d'èxit deixa de sentit ja que la variança dels resultats obtinguts augmenta fins al punt de convertir la mitja en irrellevant.

Respecte el delay, no observem cap tendència clara, tot i que en teoria augmentant el delay hauria de disminuïr la taxa d'èxit, ja que augmenta la probabilitat que un altre client faci un write a un dels entries que seran consultats durant el temps marcat pel delay.

Respecte al ratio read-write, es pot veure de forma clara que si només hi han read o només writes, no pot haver cap conflicte i per tant, la taxa d'èxit és del 100%. Per altra banda, també veiem que la taxa mínima d'èxit passa quan el nombre de read i writes és el mateix.

Respecte el tamany del subconjunt aleatori d'entrades accedit respecte el total, es pot observar com es produeixen dos fenòmens. El primer és que en disminuïr molt el tamany dels subconjunts,10%-40%, aquests es tornen més disjunts entre ells i per tant augmenta la taxa d'èxit, el segon, que consisteix en què la taxa disminueix es produeix en disminuir el tamany de forma moderada (50-100 %) on els subconjunts no son prou disjunts per a la relació clients*writes i nombre d'entries però en canvi un possible solapament entre subconjunts fa que la probabilitat de fer un read d'un entrie modificat augmenti i per tant la taxa d'èxit baixi.

És la mateixa taxa d'èxit per als diferents clients?
No hem detectat grans variacions quan hi ha un nombre prou gran de transaccions.

Si executem això en un xarxa distribuïda Erlang, on s'executa el handler?
Tal com ho tenim implementat és el client qui s'encarrega d'iniciar el handler, pel que hauran d'executar-se al mateix node. Sembla ser que aquesta és la millor opció, ja que la relació amb els clients és 1:1 i el handler i el client s'enviaran força missatges (mentre que el handler no es passarà tants missatges amb les entries o el validator en comparació).

5 Opinió personal

La documentació proporcionada és adient per realitzar la pràctica i està ben estructurada. Creiem que una millora possible, seria proporcionar el codi en un zip per a que no haguem de copiar el codi del enunciat de la pràctica, que resulta una mica farragós.

Ens ha sobtat bastant la manera com parem als clients, un per un, ja que causa uns resultats dels experiments força estranys, perquè els últims clients en rebre la senyal d'stop segueixen executant transaccions amb menys clients al sistema, provocant que la taxa d'èxit augmenti. Hi ha algun motiu per això?

El codi és el següent:

```
stopClients([Pid|L]) ->
      Pid ! {stop, self()},
      receive
            {done, Pid} -> ok
      end,
      stopClients(L).
```