TDT4190
Tino Lazreg
Øving 4

Implementasjon:

Timeout:

Her har jeg endret to metoder, lock(int transactionId) i Resource klassen, og acquireLock(ResourceAcess resourceAccess) i Transaction.

lock(int transactionId) venter nå i en random tid mellom 1 og 5 sekunder, hvis ressursen er låst av en annen transaksjon. Før jeg endret det, så ventet den evig, som medførte en vranglås.

acquireLockResourceAcess resourceAccess) ble endret til å avbryte transaksjonen om ressursen ikke blir tildelt, altså at vi har fått en timeout.

Edge Chasing:

Jeg har endret på de samme metodene som i timeout ved å legge inn en sjekk for PROBING_ENABLED. Jeg har en egen klasse Probe, som arver fra thread. Hver transaksjon som starter vil starte en ny tråd gjennom Probe, som kaller sendProbe(probeList) metoden til Server. Probe setter serveren i konstruktøren, og har en LinkedList for å lagre serverId. Listen inneholder identifikatorene til alle transaksjonene som probingkjeden har vært innom. Hver gang sendProbe blir kalt så sjekker den om denne transaksjonen ligger i listen. Hvis den fins, så vet man at det fins en sykel i vent-for-grafen, og dette betyr at det fins en vranglås. Jeg løser vranglåsen med å si ifra til ressursen, som avbryter transaksjonen som oppdaget vrangåsen, og ressursene som den transaksjonen hadde tilegnet seg blir frigjort. NotifyResource() er metoden som håndterer dette, og den kaller notify på den ressursen det gjelder. Notify() gjør sånn at den originale tråden til transaksjonen vil fortsette. Siden transaksjonen ikke får tilgang på ressursen den vil, så vil den få en timeout og bli avbrutt.

Hvis sendProbe ikke finner sin egen ID, så vil den kalle sendProbe() til serveren som har ressursen vi venter på. Hvis man kommer til en transaksjon som har ikke venter på noen (waitingFor == null), så vil tråden bli avsluttet, siden det betyr at det ikke fins en vranglås i systemet.

Diskusjon:

Timeout:

Jeg venter et tilfeldig antall sekunder mellom 1 og 5, og dette kan medføre at løsningen er treg. Jeg kunne ventet et mindre antall sekunder, før en eventuell timeout, men dette kan igjen medføre til at mange flere transaksjoner blir avbrutt enn nødvendig. Implementasjonen er veldig enkel, og vil ikke føre til mer nettverkstrafikk. I motsetning til edge chasing, vil ikke probe-meldinger bli sendt ut, så dette gjør timeout til en mye enklere løsning.

Edge Chasing:

Fordelen med edge chasing er at vi ikke regner med en tilfeldig timeout tid, men heller får en timeout når en mulig vranglås er oppdaget. Dette fører til at vi får mange færre unødvendige avbrytelser enn med timeout. Edge chasing er vanskeligere å implementere, siden man må passe på at probe-meldinger blir sendt riktig. Noen svakheter med implementasjonen min er at siden flere

meldinger kan bli sendt samtidig, så kan det hende at flere tråder oppdager samme vranglås samtidig, og derfor så kansellerer de begge transaksjonene sine. Fantomvranglåser er også et problem, siden programmet vil tro at det er en ekte vranglås, siden jeg ikke har en sjekk når en vranglås oppdages. Jeg bare går ut i fra at det er en ekte vranglås, og avbryter transaksjonen. Det som kan hende er at vranglåsen jeg oppdaget, allerede har blitt løst av en annen tråd, eller at ressursen har blitt ledig, og at jeg derfor unødvendig avbryter transaksjonen.