NTNU
Det skapende universitet

Institutt for datateknikk og informasjonsvitenskap

Løsningsforslag til
Eksamensoppgave i TDT4190 Distribuerte systemer

Faglig kontakt under eksamen: Jon Olav Hauglid
Tlf.: 93 80 58 51

Eksamensdato: Lørdag 8. juni 2013
Eksamenstid (fra-til): 9.00 – 13.00
Hjelpemiddelkode/Tillatte hjelpemidler: D. Ingen trykte eller håndskrevne hjelpemidler tillatt. Bestemt, enkel kalkulator tillatt.
Annen informasjon: Oppgavesettet inneholder 7 oppgaver. Det er angitt i prosent hvor mye hver (del-)oppgave teller ved sensur. Gjør rimelige antagelser der du mener oppgaveteksten er ufullstendig og skriv kort hva du antar. Lykke til!

Målform/språk: Bokmål
Antall sider: 2
Antall sider vedlegg: 0

Kontrollert av:

_____

Dato Sign

Oppgave 1 – Distribuerte objekter og fjernkall (5 % på a, 10 % på b)

a) Forklart kort forskjellene på protokollene ”forespørsel” (request), ”forespørsel-svar” (request-reply) og ”forespørsel-svar-bekreftelse” (request-reply-acknowledge reply).

Ved «forespørsel» sender klienten bare en forespørsel til tjeneren. Klienten vet ikke om tjeneren får meldingen og utfører operasjonen. Tjeneren vet heller ikke om den går glipp av forespørsler.
Ved «forespørsel-svar» sender klienten en forespørsel til tjeneren og får et svar tilbake. Klienten kan sende forespørselen på nytt til den får svar. Ved bruk av sekvensnummer kan man unngå at operasjonen utføres flere ganger. Alternativt kan man benytte idempotente operasjoner for å unngå problemer. Tjeneren sender svar, og hvis det er sekvensnummer i forespørselen kan den unngå å utføre operasjonen flere ganger dersom den cacher svaret den sender til klienten slik at den kan sende svaret uten å utføre operasjonen.
Ved «forespørsel-svar-bekreftelse» sender klienten, i tillegg til meldingene i «forespørsel-svar», en bekreftelse på at den har mottatt svaret. Dette er ikke nødvendig for at operasjonen skal bli korrekt utført, men det gjør det mulig for tjeneren å slette cachede svar når den vet at klienten har mottatt dem. Det trengs ikke svar på bekreftelsen siden tjeneren ellers kan la svarcachen tømmes når den har blitt for gammel.

b) Hvorfor kan man ikke gjøre call by reference med fjernkall? Hvordan kan man erstatte call by reference i fjernkall?

Call by reference sender referanser, dvs. pekere, til prosedyren. Siden den kallende og den kalte prosedyren i et fjernkallsystem kan være i forskjellige adresserom (f.eks. forskjellige maskiner), gir ikke minnepekere nødvendigvis mening for mottakeren.
Som erstatning for call by reference, kan klienten sende en fjernpeker til et objekt som tillater fjernkall.

Oppgave 2 – Likemannsnettverk (”Peer-to-peer networks”) (5 % på a, 10 % på b)

a) Vil et søk i et ustrukturert peer-to-peer nettverk gi flere meldinger enn et søk i et strukturert peer-to-peer nettverk? Anta like mange peers. Begrunn svaret.

I ustruktuerte nettverk sendes søket til alle kjente peers som igjen sender til kjente peers etc. I et strukturert nettverk sendes søket som en melding som går stegvis nærmere den peer der man vet at svaret ligger (hvis svar finnes). Dermed blir det færre meldinger i strukturerte nettverk.

b) Anta et Pastry-nettverk med 12 bits GUID. Det finnes 20 peers med GUID lik 014, 200, 204, 207, 245, 2ED, 2F0, 444, 5AC, 5EF, 75C, 811, AAA, AAB, B12, B32, CC5, D11, F45, FEC. Pastry bruker to typer rutingtabeller: Prefikstabell og løvnodetabell. Anta at hver løvnodetabell inneholder 4 peers. Lag prefiksrutingtabellen og løvnodetabellen for peer 200.

0	014				444	5AC		75C	811		AAA	B12	CC5	D11		F45
1					245										2ED	2F0
2					204			207								

For flere av rutene er det flere mulige valg. F.eks. kan 5AC erstattes med 5EF.

Løvnodetabellen vil inneholde FEC, 014 – 204, 207 – dvs de 4 med nærmest GUID (husk at et Pastry-nettverk er logisk organisert som en ring, derav FEC).

Oppgave 3 – Sikkerhet (10 % på a, 5 % på b)

a) Anta at du deltar i et gruppearbeid og skal sette opp en tjener som alle i gruppa skal kunne logge seg på ved hjelp av asymmetriske nøkler ("public key"). Hvilke nøkler trengs og hvor skal de lagres? Beskriv hvordan pålogging kan skje.

Det er flere mulige løsninger her. For eksempel:
Alle i gruppa trenger hver sin private og offentlige nøkkel. Alle offentlige nøkler lagres på tjeneren. Alle beholder sin private nøkkel. Ved pålogging kan tjeneren lage en sesjonsnøkkel og kryptere den med brukerens offentlig nøkkel. Dette sendes så til brukeren som bare kan få tak i sesjonsnøkkelen med sin private nøkkel. Nonce kan brukes for å unngå at meldinger kan snappes opp av tredjepart og sendes på nytt senere.
En løsning bør ikke innebære sending av private nøkler over nett. Ei heller et felles nøkkelpar som brukes av alle deltakere da det f.eks. gjør det vanskelig å fjerne tilgang til enkeltmedlemmer senere.

b) Hva er formålet med blokkchiffer-kjeding ("cipher block chaining")?

Normalt sett krypteres data blokk for blokk. Hvis blokker krypteres uavhengige av hverandre, vil det være mulig å kunne gjenkjenne repterende mønstre etter kryptering og dermed kunne knekke krypteringen med statistisk analyse. Blokkchiffer-kjeding unngår dette ved at etterfølgende blokker krypteres avhengig av hverandre. For eksempel med at en kryptert blokk XOR'es med neste blokk.

Oppgave 4 – Tid og global tilstand (5 % på a, 10 % på b)

a) Et distribuert system består av tre prosesser p1-p3 og vektorklokker skal brukes for å holde orden på logisk tid. Hvor mange vektorelementer vil vektoren for prosess 1 bestå av? Hva er reglene for hvordan de blir oppdatert?

Vektorklokker har like mange elementer som det er prosesser i systemet. Hver verdi forteller hvor mange hendelser prosessen har kunnet blitt påvirket av fra hver av prosessene i systemet.

Rett før hver lokale hendelse hos p1 økes V1[1] med 1.
Hver gang p1 mottar en melding med vektor Vm settes V1[j] = max(V1[j], Vm[j]).

b) Hva er et "consistent cut"? Representerer et "consistent cut" en global tilstand som et distribuert system nødvendigvis må ha vært i? Begrunn svaret.

Et cut er et union av prefiks av hver prosess sin lokale historie. Et konsistent cut er slik at hvis hendelsen B er med i cut'et og A ➔B, så er A også med. Dvs. at hvis konsekvens er med, må årsak også være med. Slik at mottaket av en melding kan ikke være med uten at sendingen av meldingen også er med.

Et konsistent cut representerer en global tilstand som et distribuert system kan ha vært i, men ikke nødvendigvis må ha vært i. Anta et distribuert system med to prosesser som hver har sin lokale hendelse. Et cut som bare inneholder den ene hendelsen er akkurat like konsistent som et cut som bare inneholder den andre hendelsen, ettersom to lokale hendeler hos to forskjellige prosesser ikke har skjedde-før relasjonen mellom seg. Men med mindre de to hendelsene skjedde nøyaktig samtidig må systemet faktisk ha vært i en tilstand der bare den ene av hendelsene har skjedd.

Oppgave 5 – Koordinering og enighet (10 % på a, 10 % på b)

a) Avstemnings-algoritmen for distribuert gjensidig utelukkelse (”mutual exclusion”) er basert på at hver prosess har et avstemningssett med prosesser som må spørres før tilgang kan gis. Hvordan må avstemningssett være for at de skal gi korrekt oppførsel? Hvordan bør de være for å være mest mulig rettferdige og effektive? Begrunn svaret.

Korrekt: Settene som brukes av alle par av prosesser må overlappe. Dermed unngår man at et sett gir adgang til en prosess samtidig som et annet sett gir adgang til en annen prosess.
Rettferdig og effektivt: Like stort for alle deltakere (et lite sett er raskt å spørre mens et stort sett tar tid, så like stort er rettferdig), alle deltakere med i like mange sett (lastdeling er også rettferdig) og setteme så små som mulig (det er effektivt å spørre få).

b) Pålitelig multikast ved hjelp av IP multicast bruker gjerne en "hold-back queue" der meldinger som er mottatt av en gitt node kan vente før de bli levert ("delivered") til en prosess. Hva oppnås med denne køen?

Det ordner meldinger fra en gitt avsender slik at de blir levert i samme rekkefølge som de ble sendt. Dette gir FIFO ordning av meldinger. Dessuten blir det enklere å merke om noen meldinger mangler slik at negativ acknowledgement kan sendes.

Oppgave 6 – Google case study (10 %)

Anta at du skal telle hvor mange ganger forskjellige ord forekommer i en mengde dokumenter. Forklar steg for steg hvordan dette kan gjøres ved hjelp av MapReduce.

MapReduce er et rammeverk for parallellprosessering. Det baserer seg på en antagelse om at store operasjoner kan deles opp i mindre deler der hver del blir prosessert uavhengig (map) før delresultatene slås sammen hver for seg (reduce). Så det er parallelitet både under utføring av map og under utføring av reduce.
Eksempel: Telling av ordforekomster i dokumenter. Del opp dokumentene. Map: Hver gang ord finnes, skriv <ord 1> til logg. Alle <ord, 1> for et gitt ord samles (ord er nøkkel). Reduce: tell opp antall <ord, 1> for et gitt ord.

Oppgave 7 – NoSQL (10 %)

NoSQL-datamodeller baserer seg gjerne på aggregater. Forklar hva slike aggregater er og diskuter kort fordeler og ulemper med denne måten å organisere data på.

Et aggregat er relaterte data som behandles som en enhet. Mens normalisering av data, som man typisk gjør i et relasjonsdatabasesystem, fører til at man lagrer adresse og postnr/-sted separat, kan man med aggregering velge å lagre dem sammen. Det medfører dobbeltlagring

(den funksjonelle avhengigheten postnr => poststed lagres mange ganger) noe som gjør oppdatering/consistency mer vanskelig. Men samtidig blir lesing raskere siden man slipper join. Det kan også være enklere å gjøre valg mhp. replikering/sharding siden aggregering gjerne gjøres etter bruksmønster.