9. ASYNCHONOUS AGREENMENT

Magnus Jensen

- Motivation
- Unscheduled Broadcast
  - Betingelser for asynkron protokol
    - Kommandoer
    - Egenskaber
  - Bracha Broadcast
    - Ideen
    - Protokollen
    - Analyse
- Asyncronous Byzantine Agreement
  - Weak Agreement
    - Protokollen
  - Fra Weak Agreement til Non-Terminating

Note: den her er ikke så lang, men den er svær og meget der skal skrives på tavlen. Så der burde være nok fyld. Tager den for lang tid, så tag de sidste punkter fra.

ASYNCHONOUS AGREENMENT

1. Motivation

I den Asynkrone Agreement model; har vi ikke en betegnelse for tid - faktisk bruger vi det slet ikke.

> *Ingen tid*

Det har sine fordele og ulemper

> *(+) Ingen synkrone ure*

hvilket kan være helt problem i sig selv

> *(-) Ikke skelne langsom, fra: ikke sendt*

Så vi kan beskylde en part for at være ond fordi den aldrig sendte en besked. Hvorfor vi har **Eventuel Delivery** af beskeder.

> *(-) ingen runder*

En anden konsekvens af, at vi ikke kan bruge tid - er at vi ikke har nogen betegnelse om at arbejde i runder. Så hvad gør vi så?

2. Unscheduled Broadcast

Så når vi ikke har tid, og **ikke kan arbejde i runder**; så kan vi ikke vide noget om** - hvornår at **en anden** part har planlagt at **sende en besked**.

Dette kalder vi for et Unscheduled Broadcast, eller så fint på dansk - et uplanlagt broadcast.

2.1 Betingelser for asynkron protokol

Men hvordan kan vi så udføre en handling?

For enhver asynkron protokol gælder følgende:

> - *Der er aktiveringsregler*
> - *Et tidspunkt venter for n - t beskeder*
> - *Et tidspunkt venter for t + 1 beskeder*
> - *n > 3t for stort flertal af ærlige parter.*

2.1.1 Funktioner

> - *Broadcaster B kan få: (BROADCAST, P1, id, m)*
> - *Pj kan outputte (DELIVER, P1, ID, m).*

2.1.2 Egenskaber

> - ***Validity 1**: Hvis en korrekt Pi levere, og B er korrekt, blev B bedt om det*
> - ***Agreement**: Alle korrekte levere det samme*
> - ***Validity 2**: Hvis B er korrekt og får input m, vil alle korrekte Pi levere m*
> - ***Propagation** (flooding): Hvis en korrekt P levere m, vil alle korrekte levere m*

2.2 Bracha Broadcast

2.2.1 Ideen

Når en broadcaster B flooder med en besked M, er ideen at:

> - *Ærlige parter EKKO beskeden rundt*

> - *Når EKKO ses fra et flertal, erklæres beskeden KLAR til offentliggørelse*
> - *Når KLAR ses fra et flertal, offenliggøres beskeden*

2.2.2 Protokollen

> - *B skal broadcaste m, så send den til alle*
> - *Besked fra B, ekko den til alle andre, medmindre man har set id'et før*
> - *Ekko fra n - t parter, send klarmelding*
> - *KLAR fra t + 1 parter, send KLAR*
> - *KLAR fra n - t parter, så udskriv.*

2.2.3 Analyse

Validity 2:

- m broadcastes
- Alle ærlige parter sender EKKO til alle
- Alle ærlige parter får n - t EKKO og sender et KLAR
- Alle ærlige parter får KLAR og udskriver

Validity 1:

- Hvis B aldrig fik besked på at broadcaste, så sendes maksimalt t ekko.
- Siden t EKKO eller KLAR ikke er nok til at udskrive, vil ingen ærlig P udskrive.

Propagation:

Vi starter med at kigge på, hvordan det kan ske, at en korrekt P har outputtet. Det vil ske fordi:

- P har modtaget KLAR fra n - t parter, hvilket er minimum 2t + 1
- t kan være korrupte; så de t + 1 har også sendt en klarmelding til alle andre
- Så alle ærlige parter modtager minimum t + 1 KLAR; hvorfor alle levere m.

Agreement:

Vi bruger lidt af samme logik som før, og kigger på hvad der skal til for at en Pi vil levere en m.

> - *t onde*

> - *3t + 1 ærlige*

- Pi har modtaget KLAR fra 2t + 1 parter
- T kan være korrupte; så t + 1 ærlige sendte KLAR til Pi
- En Pj der levere, vil derfor også have modtaget klarmelding fra t + 1 parter
- Vi ved ikke hvilke parter der er onde; så minimum 1 ærlig part har sendt samme m til begge, hvorfor de levere det samme.

3. Asyncronous Byzantine Agreement[1]

Hurtig gennemgang.

Async Byzantine Agreement, er en afstemning der kan foregå i et distribueret system asynkront - og parterne skal nu tælle stemmer og blive enige om en beslutning.

- Hver afstemning har et id.
- Hver part kan få en stemme i input *(VOTE, id, Vi)*, hvor Vi ∈ {0, 1}
- Hvert part kan outputte en beslutning *(DECISION, id, Di)* hvor Di ∈ {0, 1}

3.1 Weak Agreement

En protokol der opretholder async byzantine agreement, er Weak Agreement.

Den siges at være svag, og grundet at vi tillader en ærlig part at være i tvivl om hvad for en beslutning den skal tage og at der heller ej behøver at være enighed om man er i tvivl; altså kan en korrekt part outputte "?".

Protokollen virker ved:

> *Weak Agreement: n > 5t*

3.1.1 Protokollen

- Alle sender v til alle andre
- Alle venter på v fra n - t parter
  - Hvis n - 2t af v = 0, output d = 0
  - Hvis n - 2t af v = 1, output d = 1
  - Ellers d = ?

3.2 Fra Weak Agreement til Non-Terminating

Det kan være et problem, at det rent faktisk er en svag-enighed der gives i den forestående protokol.

- Kør Weak Agreement med input Vi og lad outputtet være Di
- Hvis Di != ? så sæt Vi = Di
- Hvis Di = ? så sæt Vi lig et uniformligt tilfældigt bit
- Start forfra

---

1. Egenskaber udenladt, se /noter/readme.md