LAB 05: ABD QUORUMS

ALGORITMOS E SISTEMAS DISTRIBUÍDOS (ASD)

Alex Davidson

October 2023

a.davidson@fct.unl.pt

- Use your source code in Phase 1 of the project to implement a read/write register, on top of an ABD Quorum design.

- More details at https://github.com/MEI-ASD-2023/lab05-abd-quorum

- Will be useful for Phase 2 of the project

LAB FOR TODAY

- Particular case of state-machine replication

- Only supports two functions: read$()$, write(v)

- write(v) returns ACK upon termination

- read$()$ returns previously written value, or initial value v_0

READ/WRITE VARIABLE REPLICATION

- Quorum-based system

- **Assumption (Liveness):** $2f + 1$ correct processes, for f possible failures

- **Assumption:** Asynchronous

- **Assumption:** Reliable (pp2p) links

- Generalises to multiple values

Identifiers for values, and parallel executions

ABD (ATTIYA, BAR-NOY, DOLEV) ALGORITHM

State for process i ($\mathtt{st}_i$):

- v_i: value, initially v_0
- t_i: pair $(\mathtt{sn}_i, \mathtt{id}_i)$, initially $(0, 0)$
- $t_i > t_j$ iff $(\mathtt{sn}_i > \mathtt{sn}_j) \vee ((\mathtt{sn}_1 = \mathtt{sn}_2) \wedge (\mathtt{id}_1 > \mathtt{id}_2))$

Phase 1:

1. Send message read-tag$()$ to all replicas
2. Wait for a quorum Q ($|Q| > 2f+1$) of replies
3. Let $\mathsf{sn}_{max} = \mathsf{max}(\{\mathsf{sn}_i\}_{t_i \in Q})$

Phase 2:

1. Send message write$(t_v = (\mathsf{sn}_{max}+1, \mathsf{id}_j), v)$ to all
2. Wait for a quorum Q ($|Q| > 2f+1$) of ACK
3. Return ACK (write considered terminated)

write(v) ISSUED BY PROCESS j

- Upon reception of `read-tag()`: return t_i

- Upon reception of `write`(t_v, v):
  - if $t_v > t_i$: $t_i = t_v$; $v_i = v$
  - Return ACK

- Upon reception of `read()`: Return t_i, v_i

REPLICA PROCESS i

1. Send read() to all replicas

2. Wait for a quorum Q of replies $\{(t_i, v_i)\}_{i \in Q}$

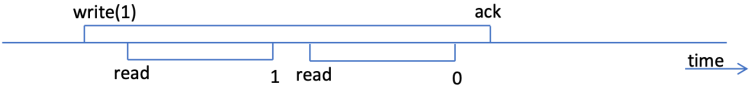
3. Let $(t_{max}, v_{max}) = \max(\{t_i, v_i\}_{i \in Q})$

4. Return v_{max}

Question: Does this ensure **atomicity**?

READ ALGORITHM: NAÏVE ATTEMPT

Nope



This execution can occur because t_{max} may be derived from a non-completed write.

READ ALGORITHM: NAÏVE ATTEMPT

Solution: add a write-back(p) hase

1. **Phase 1**: Proceed as in naïve case
2. **Phase 2**: Send write(t_{max}, v_{max}) to all replicas
3. Wait for a majority quorum of ACK
4. Return v_{max}

ABD READ ALGORITHM

Termination ($\leq f$ faults):

- Assuming well-formed messages
- Follows from properties of underlying links

ABD READ: TERMINATION

- Serialisation points exist in **Phase 2** of read() and write()
- Writes necessarily happen **before** reads
- Reads return most recent value by a completed write(), otherwise read() forces write() completion.
- Subsequent write() ops will not be impacted by forced previous writes
- Serialisation points for write() are defined between start and end of operation

ABD READ: ATOMICITY PROOF SKETCH

- “Sharing Memory Robustly in Message-Passing System” (JACM 1995)
- Edsger W. Dijkstra Prize in Distributed Computing: 2011
- Available here

MORE DETAILS