Fault Tolerant Consensus – Wrapup I

University of Auckland

16 Oct 2020

1. Synchronous network model

2. Stopping failures
3. EI

4. Side by side

5. Byzantine agreement with authentication

Synchronous network model

- All these algorithms are still based on the synchronous network model
-
-

Impossibility of Asynchronous Distribu
Single Faulty Process

- Solutions for the asynchronous model use randomisation, failure detectors (partially synchronous model)

Stopping failures model

- Much simplified version of the Byzantine agreement
-

- No possibility to send confusing messages
  (i.e. different messages to different directi

- The problem can be solved for any
  (not only when $3F \leq N - 1$)

The Stopping agreement conditions – vs ~~Byz~~

- Termination: all non-faulty processes eventually decide
-
- Validity: if all ~~non-faulty~~ processes st value $v \in V$, then v is the only one pos

- If the processes start with different initial va decision could be any of these (as long as it is consistent)

EIGStop

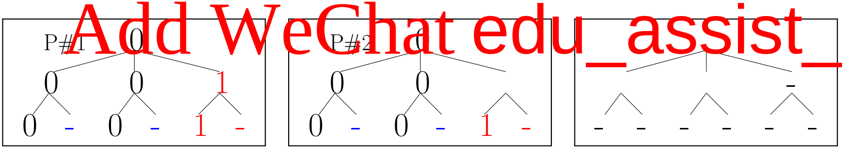
- EIG tree as in the EIGByz, $F + 1$ messaging rounds
  - recall: F can be as high as $N - 1$ (not at most $(N - 1)/3$

  - all values at all levels! not just leaves
  - nulls discarded! not assumed v_0

- If W is singleton, $W = \{v\}$, then the decision is v
- Otherwise, if W is mixed, $W = \{0, 1\}$, then the decision is v_0
  - no voting! no tie breaking

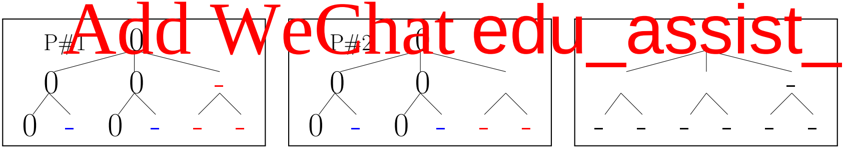
EIGStop example – assuming $v_0 = 1$; nulls as -

- Process #1 : init 0; decision $v_0 = 1$
-
-



EIGStop example – assuming $v_0 = 1$; nulls as -

- Process #1 : init 0; decision 0
-
-

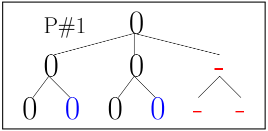


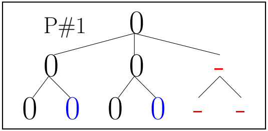
EIGStop example – assuming $v_0 = 1$; nulls as -

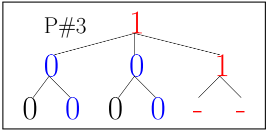
- WHAT IF scenario – NOT supported by this EIGStop protocol

- What if P#3 fails before sending any 1st round out-message

- Process #2 : init 0; decision 0
- Process #3 : init 1 : decision $v_0 =$







OptEIGStop

- Each process sends out only two messages

- The first time it learns about a different val
- Arbitrary choice, if there are more

EIGStop vs EIGByz vs 3PC – assuming $v_0 = 0$

- X indicates a faulty process, which fails from start, before sending any 1st round message

Initial	EIGStop	EIGByz	3PC
0 1 1 1	0		
1 1 1 1	1		
0 0 0 X	0		
0 0 1 X	0	0	0
0 1 1 X	0	0	0
1 1 1 X	1	1	0

EIGStop vs EIGByz vs 3PC – assuming $v_0 = 1$

- X indicates a faulty process, which fails from start, before sending any 1st round message

Initial	EIGStop	EIGByz	3PC
0 1 1 1	1		
1 1 1 1	1		
0 0 0 X	0		
0 0 1 X	1	1	0
0 1 1 X	1	1	0
1 1 1 X	1	1	0

Complexity

- EIGStop
  - Rounds: $f+1$

    Messages: $((f+1)n^2)$ messages
- [illegible]
  - Rounds: $f+1$
  - Messages: $\mathcal{O}((f+1)n^2)$ messages
- 3PC:
  - Rounds: $\mathcal{O}(f+1)$
  - Messages: $\mathcal{O}(fn)$ messages

Byzantine agreement with authentication

- Assume that each process digitally signs its messages in a total safe way, e.g. based on a reliable unbreakable PKI/DSA...

- Is this reasonable?
-
-

  itself is hacked or even turns into a Byzantine p
- Anyway, assuming that such digital signa
  Byzantine faulty nodes are not able to wreak
  havoc than a stopped process

- EIGStop can be adapted to solve the (slightly different) Byzantine agreement with authentication
- Faster/better/more general algorithms possible...