Design and Verification of Time-Critical Byzantine Fault-Tolerant Systems

Vincent Rahli – University of Birmingham

Joint work with David Kozhaya (ABB) & Jeremie Decouchant (TU Delft)

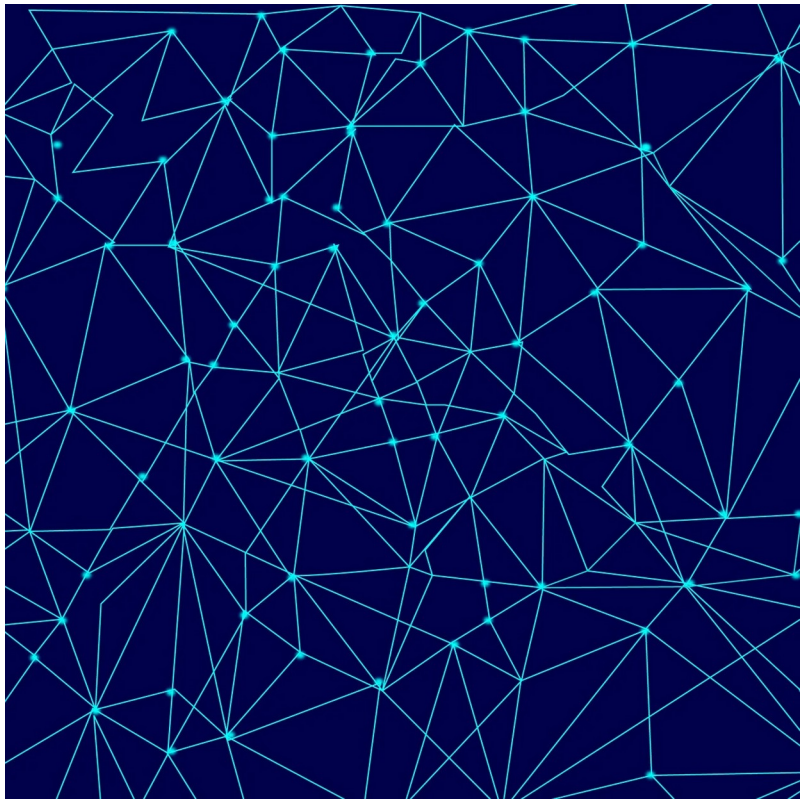
Distributed Systems

Why?

- Better performance
- More storage
- Higher resilience

Distributed problems

- Broadcast
- Consensus



Properties\Abstractions

- Agreement
- Validity

Models

- Synchronous
- Asynchronous
- Partially synchronous

Cyber-Physical Systems

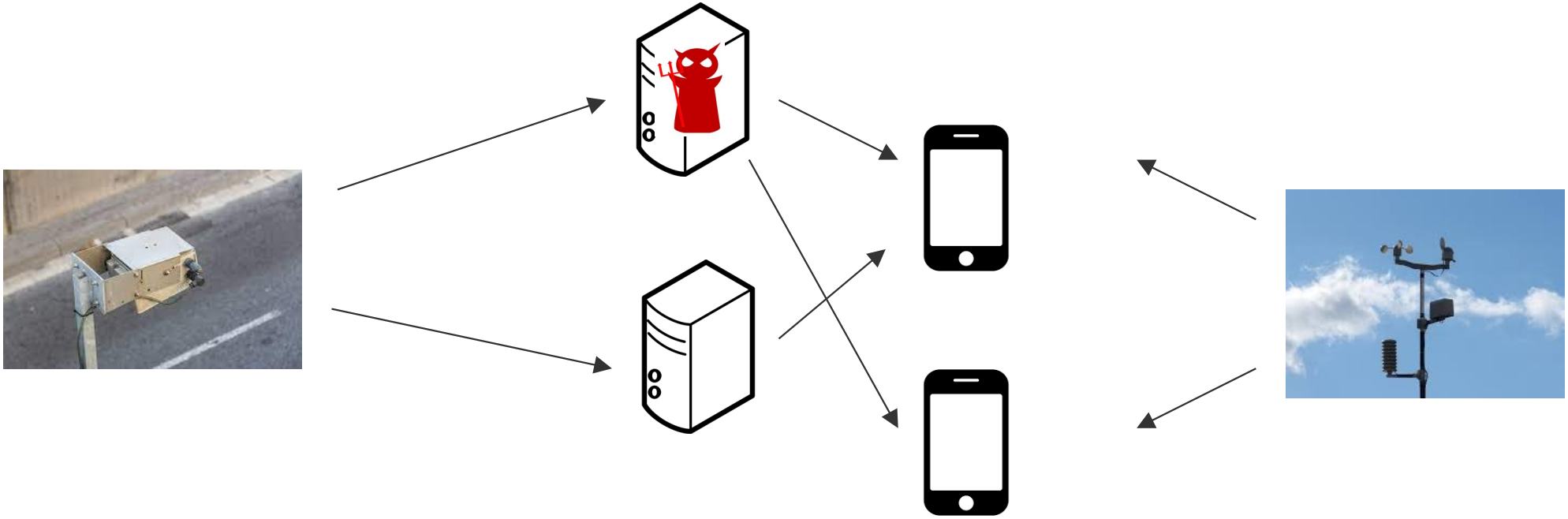
Specific requirements

Energy efficient devices
Real-time constraints

Models

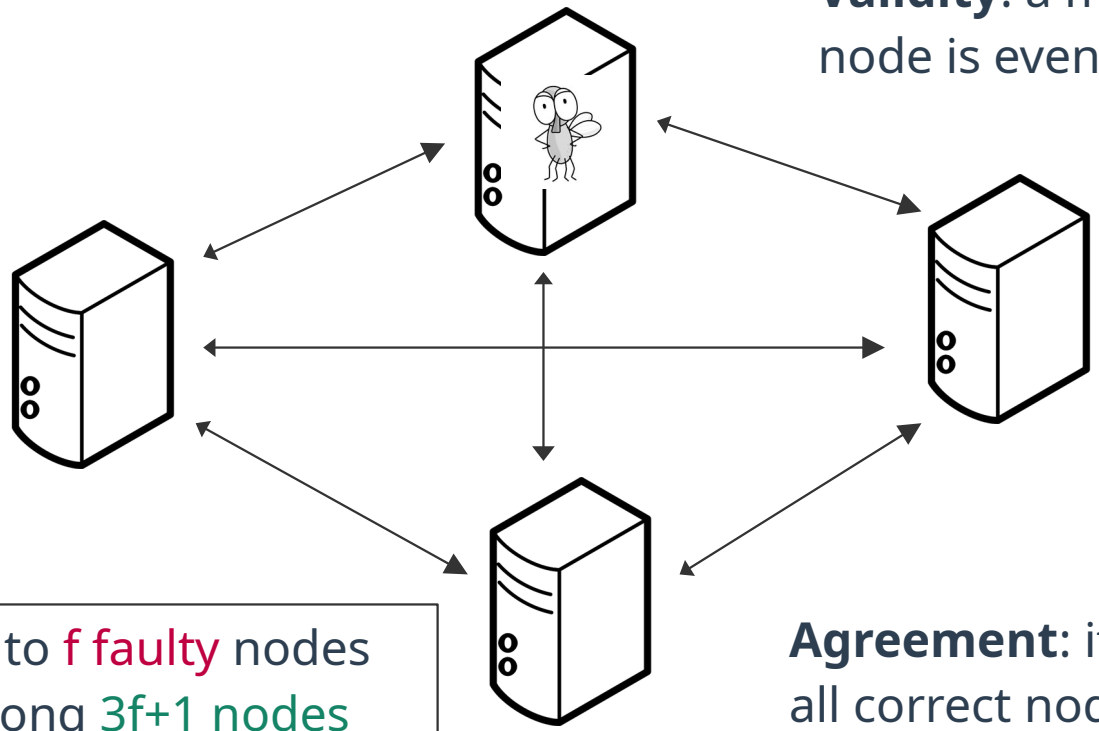
Varying quality of the communication infrastructure due to scale & heterogeneity

Real-Time Byzantine Reliable Broadcast (RTBRB)



Goal: distribute data reliably despite arbitrary faults

Real-Time Byzantine Reliable Broadcast (RTBRB)



Up to f faulty nodes among 3f+1 nodes

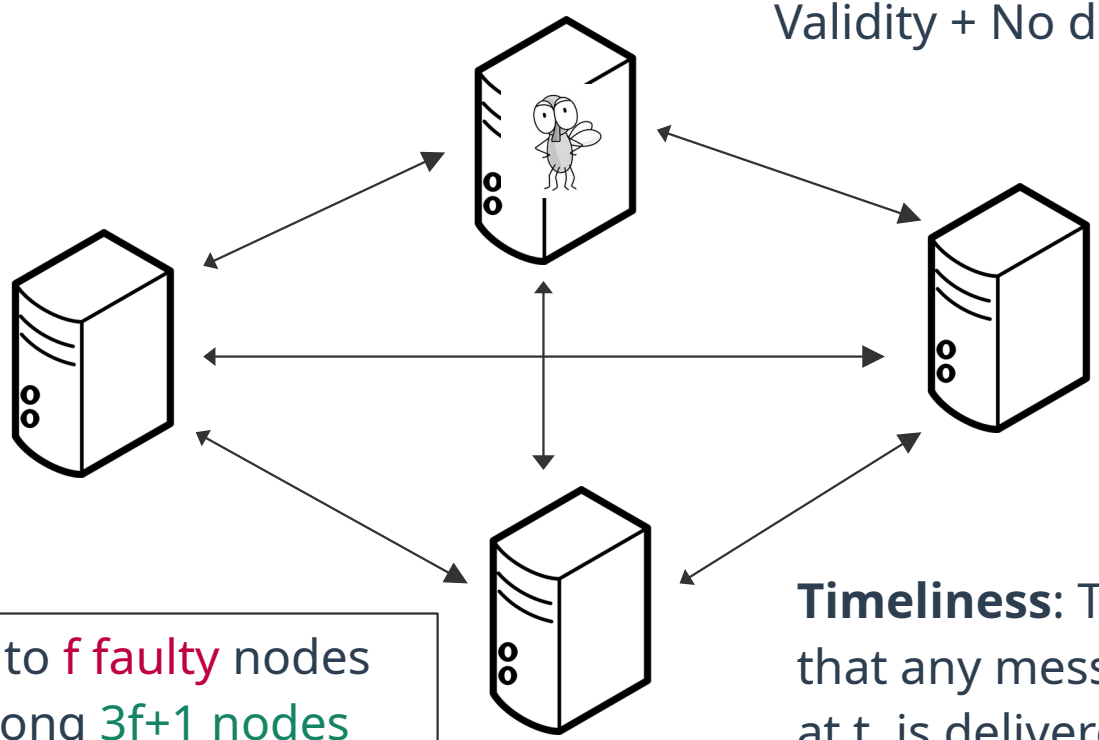
Validity: a message broadcasted by a correct node is eventually delivered by a correct node

No duplication: no correct process delivers a message more than once

Integrity: if a correct node delivers a message sent by a correct sender, then that sender broadcasted it

Agreement: if a correct node delivers a message, all correct nodes eventually deliver that message

Real-Time Byzantine Reliable Broadcast (RTBRB)

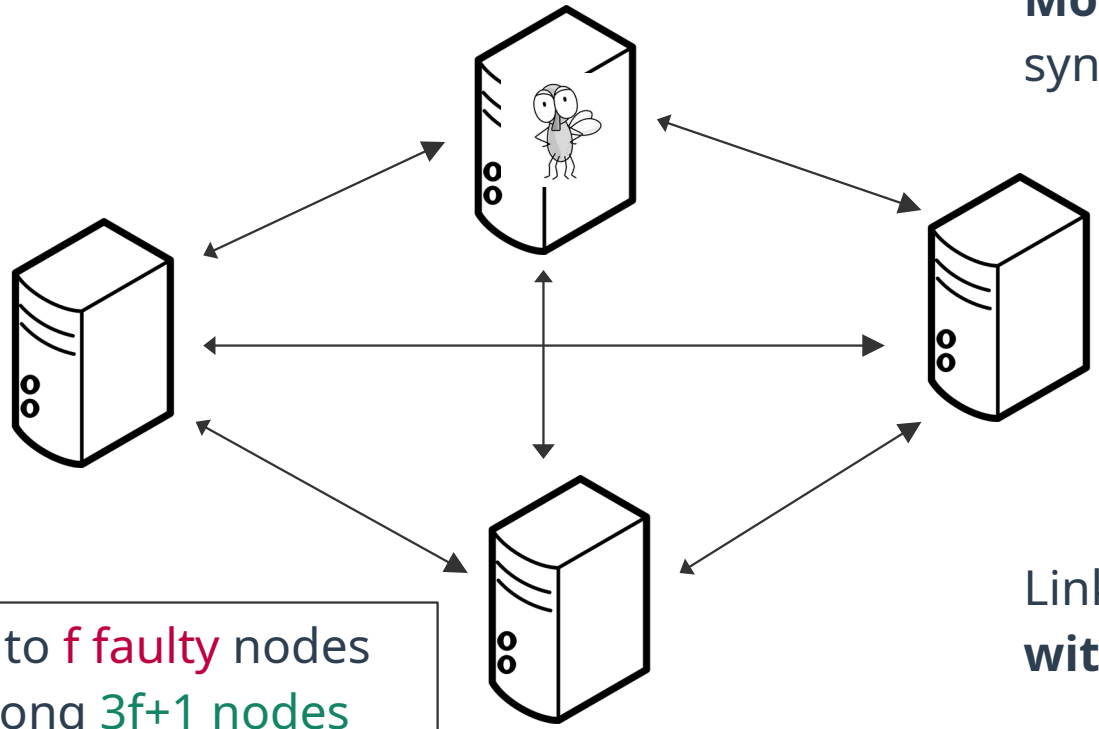


Validity + No duplication + Integrity + Agreement

Up to f faulty nodes among 3f+1 nodes

Timeliness: There exists a known bound D such that any message broadcasted by a correct node at t, is delivered by correct nodes by t+D

Probabilistically Synchronous System Model



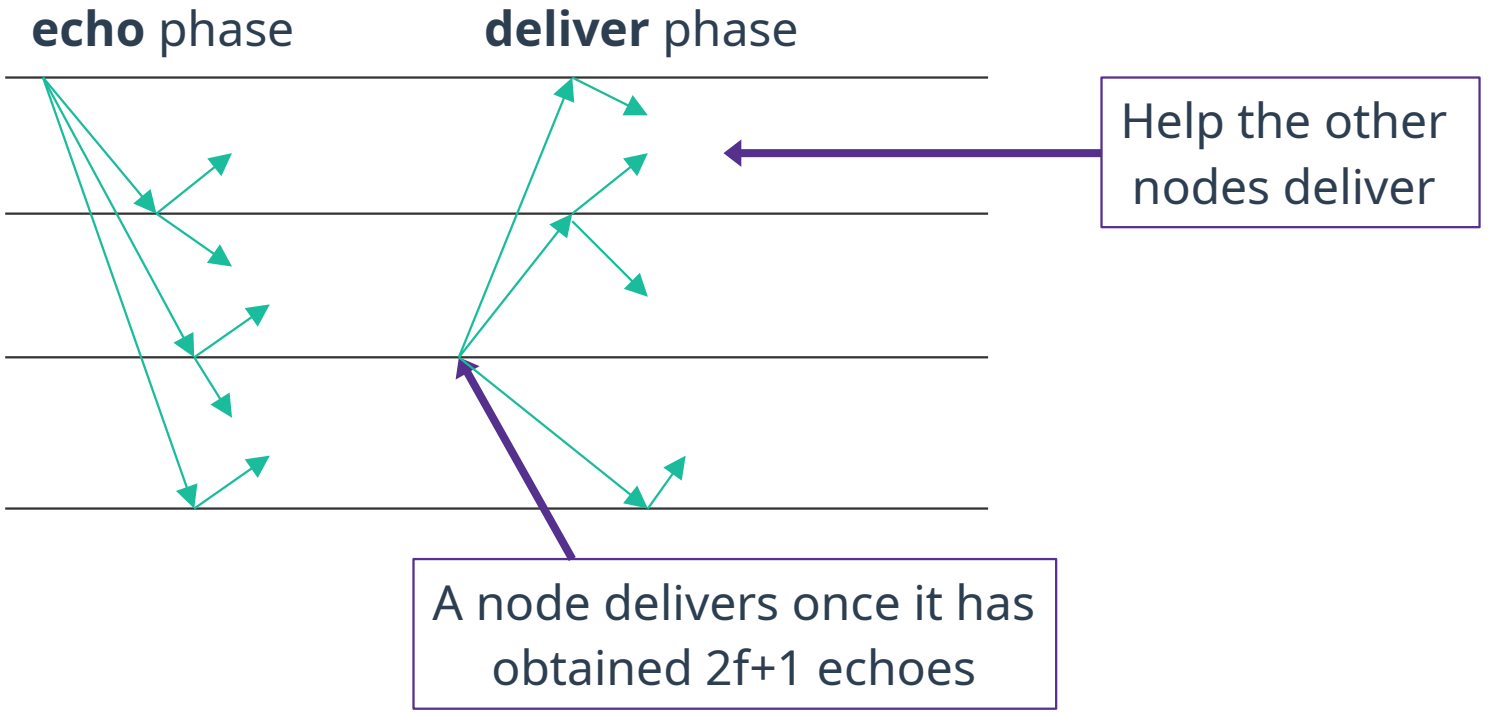
More realistic than synchronous/asynchronous

Links are reliable and synchronous **with high probability**

Up to f faulty nodes among 3f+1 nodes

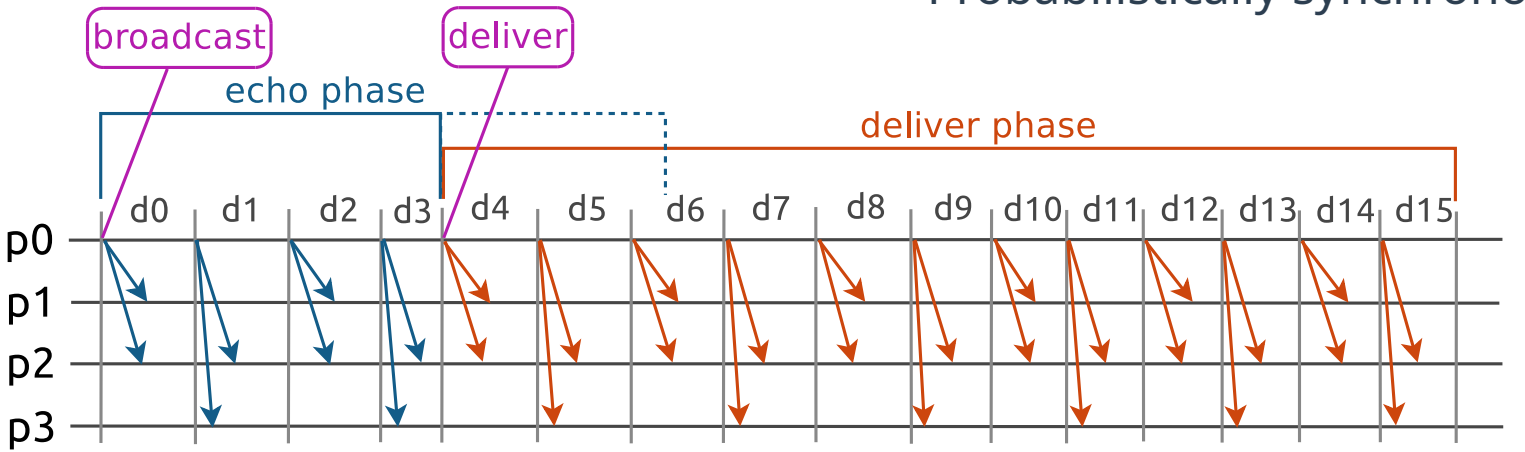
Pistis – a RTBRB Protocol

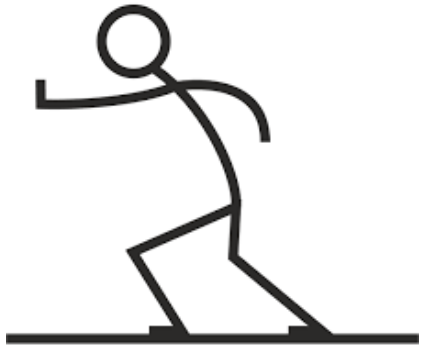
Similar to Bracha's Byzantine reliable broadcast protocol



Pistis – Pushing Messages

Probabilistically synchronous → “Synchronous”

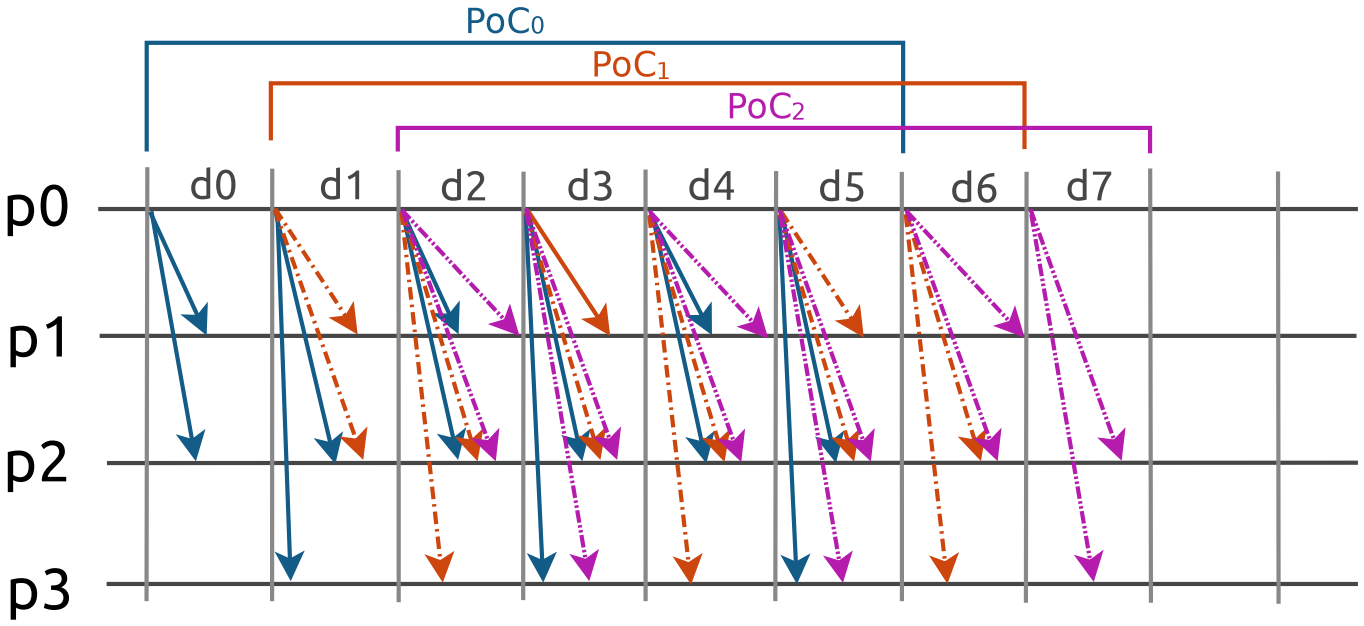




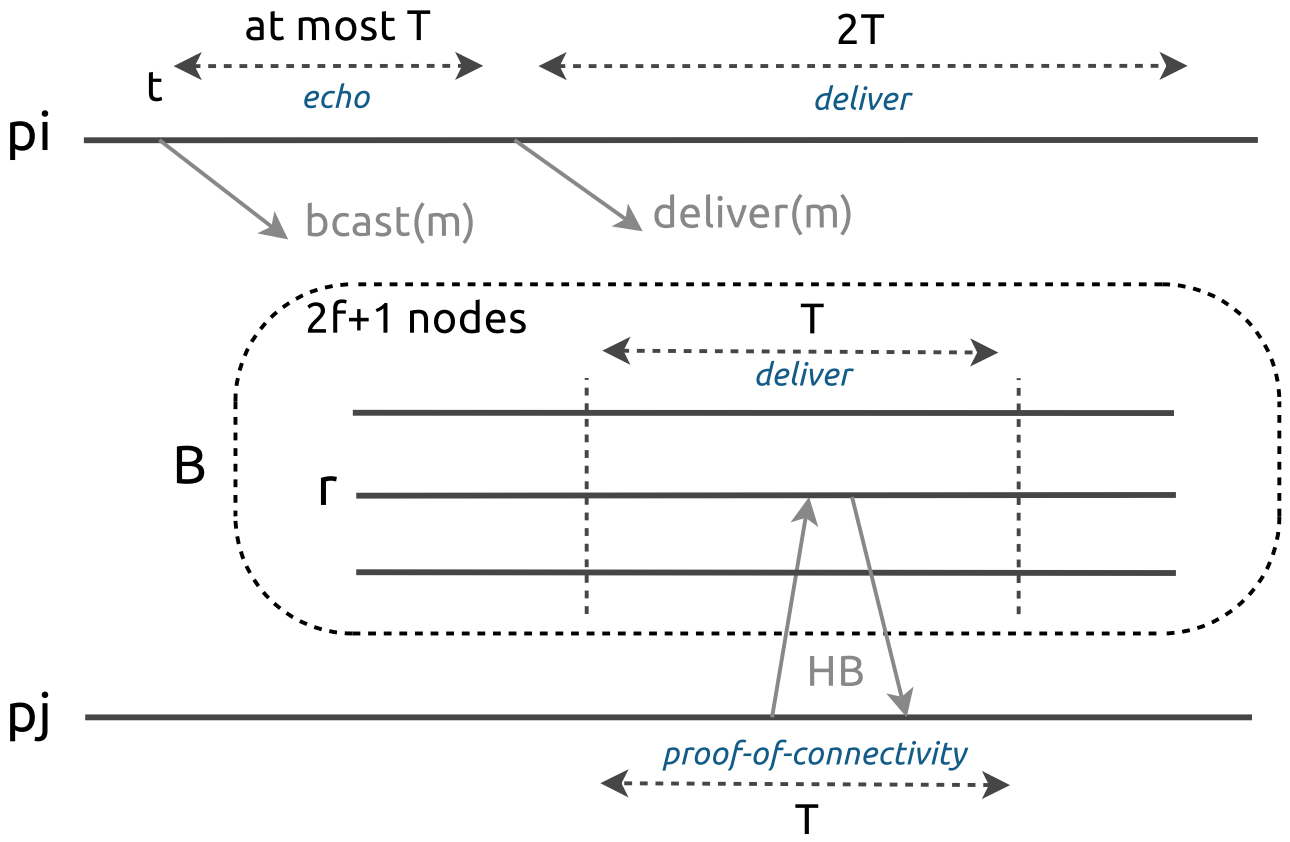
1. **Repeat** sending in case a transmission fails
2. Send to X (here 2f+1) out of 3f+1 nodes to avoid too many messages
3. If a node does not gather enough replies to its message, it becomes **passive**

Pistis – Pulling Messages (Proof-of-Connectivity)

To guarantee **timeliness**, nodes regularly **pull** messages



Formalization in Coq



We are hiring a postdoc! Contact: V.Rahli@bham.ac.uk

Questions?

Pistis: suite of real-time protocols

Formal verification of BFT protocols

Design of BFT protocols (blockchain)