CHALMERS
UNIVERSITY OF TECHNOLOGY

Distributed Computing and Systems

Network-wide Consensus

Utilizing the Capture Effect in Low-power Wireless Networks

Beshr Al Nahas
Chalmers

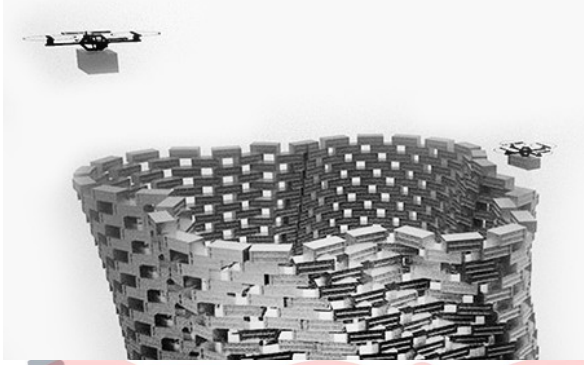
Simon Duquennoy
RISE SICS

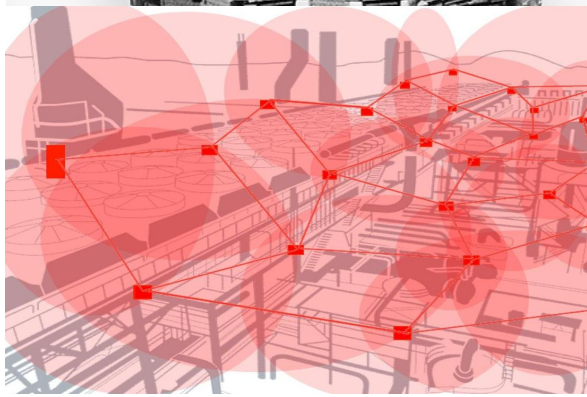
Olaf Landsiedel
Chalmers

November 6, 2017
Delft, ACM SenSys

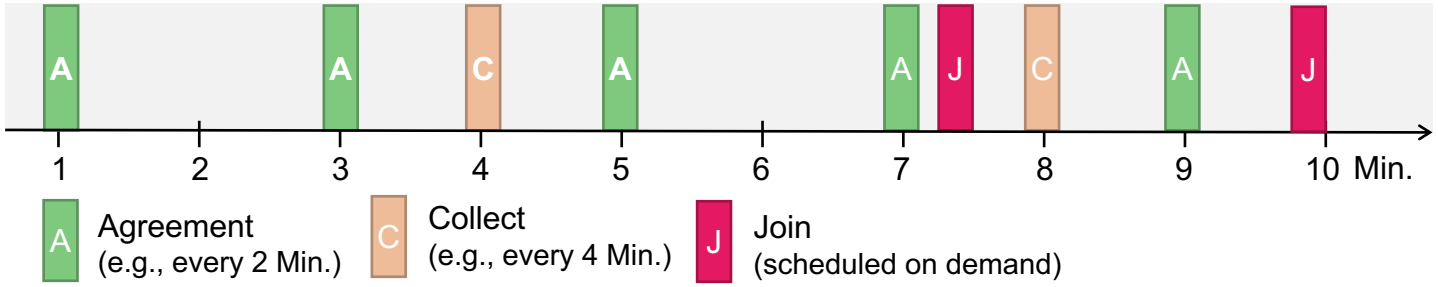
Consensus is Important but Hard

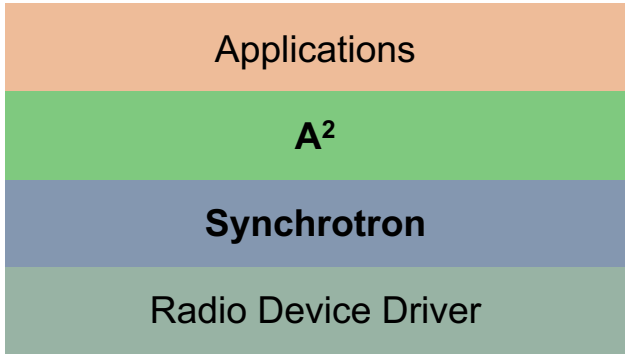
- Essential building block for mission-critical CPS / IoT
  - Vehicles or UAV agree on maneuvers
  - Process control: ensure consistent settings of actuators
- A classic protocol: Two-phase Commit [J. Gray 1979]
  - Propose a transaction (one-to-all)
  - Collect votes (all-to-one)
  - Disseminate decision (one-to-all)
  - Acknowledgment (all-to-one)
- Challenges: Consensus is heavy
  - Talk to every node back and forth several times
  - Resource constrained
  - Larger scale than done in the data centers





Contribution: Efficient Agreement at Scale





- A^2: Agreement in the Air
  - Network-wide consensus
    - Two-phase commit
      475 milliseconds - 180 nodes
  - Group membership
  - In-network processing

- Synchrotron
  - Reliable time-slotted MAC layer
  - All-to-all communication
  - Build on Chaos protocol
    - Utilizes the capture-effect
    - Multihop communication
    - No routing, link estimation, …

Outline

Motivation

CONSENSUS

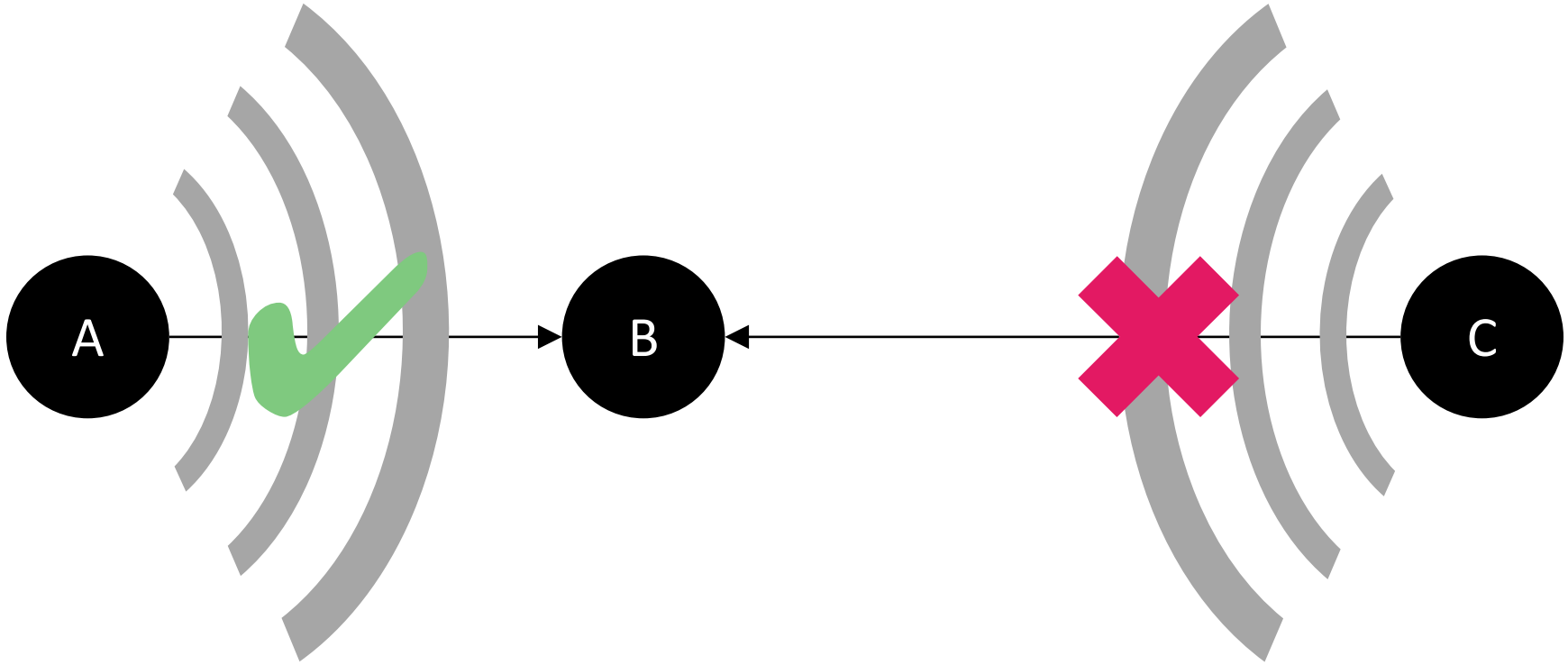
- A^2 Services – Agreement as a service
- Group Membership

MAC layer: Synchrotron

Evaluation

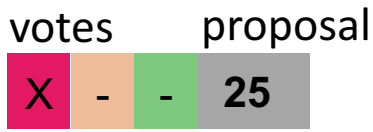
Conclusion

Concurrent Wireless Transmissions
(of different data)

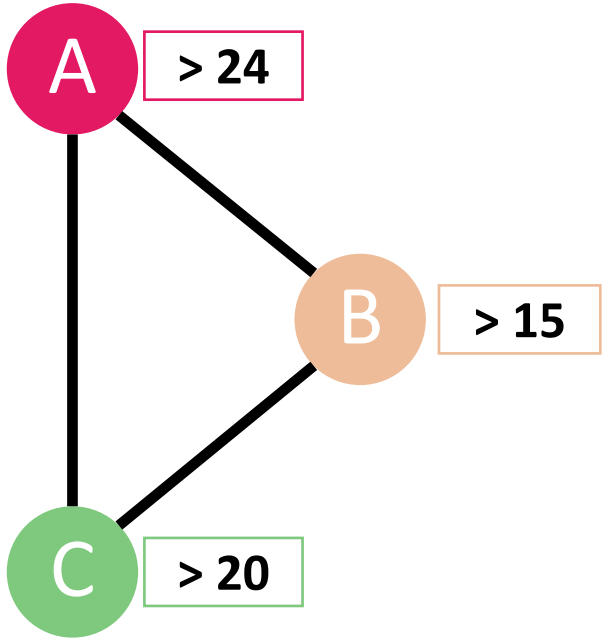


- Receive stronger signal of concurrent transmissions
  - Preambles overlap: 802.15.4 @2.4GHz: 5 bytes = **160 *µs***
  - Threshold: roughly **3dB**
  - Known as: Capture Effect

Consensus with Two-phase Commit



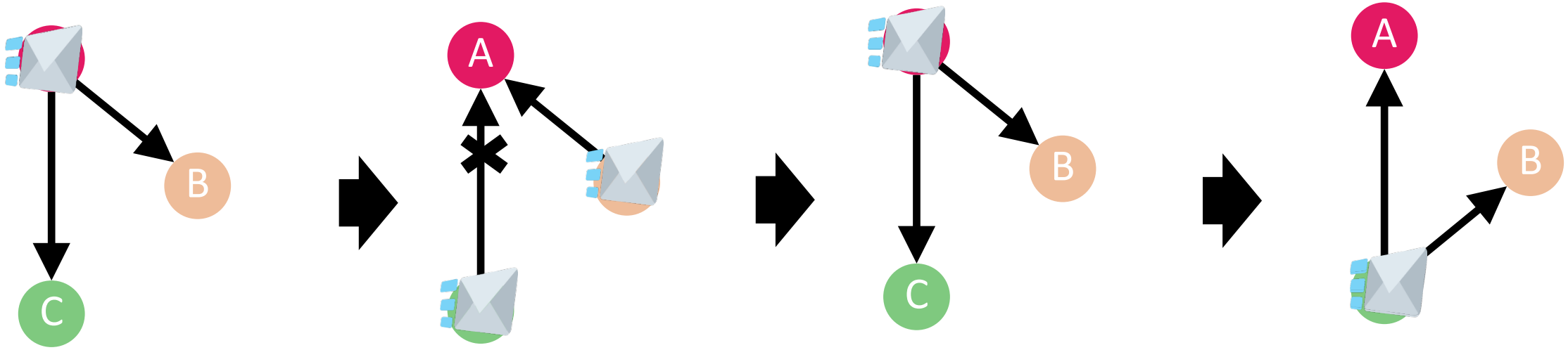
1. Collect votes
2. Disseminate decision



- Example: Single hop for simplicity – Same on multihop
- Three nodes, fully connected
- Goal: consensus on decision
  - One proposes a number
  - Others agree or disagree based on thresholds
- Assumption: members are known

Two-phase Commit: (1) Voting

1. Collect votes
2. Disseminate decision



- A initiates a round
- By sending its proposal
- Receivers: vote

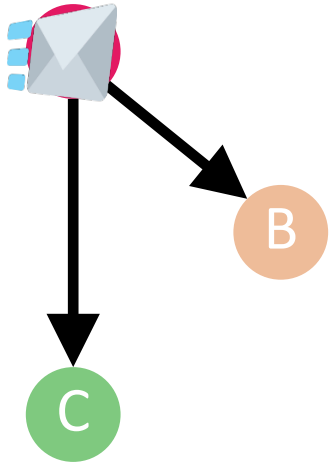
- B, C send concurrently
- A receives the stronger one (capture effect) and combines with local state

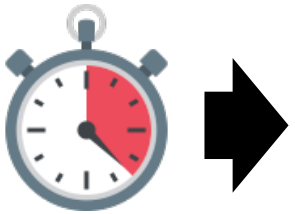
- A learned new information
- After the third round
  - C has learned the final vote

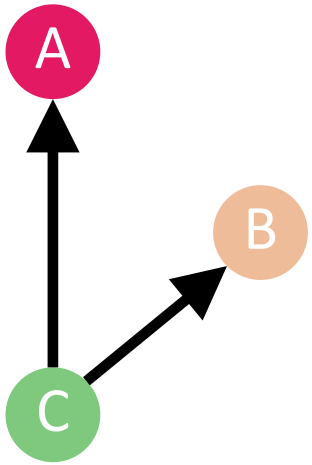
- C learned new info: send
- A, B receive
- Network converges
- A realizes everybody agree

Two-phase Commit: (2) Commit

1. Collect votes
2. Disseminate decision







	Ack		Committed value
A	X	-	- 25c
B	X	X	- 25c
C	X	-	X 25c

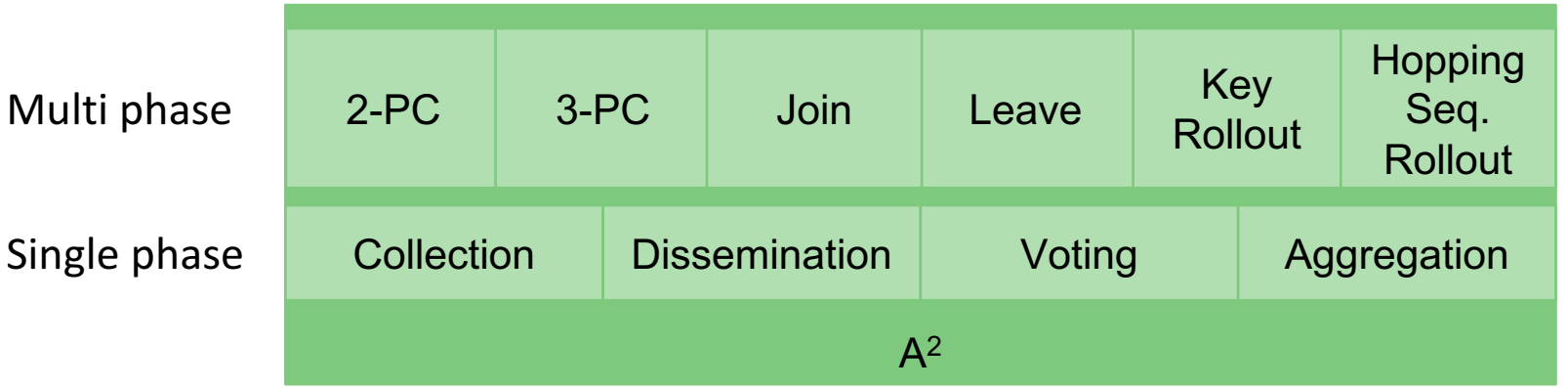
- A: commits and disseminates
- B, C receive and commit

X	X	X	25c
X	X	X	25c
X	X	X	25c

- A, B, C merge and spread the new info they learn until all flags are set

More Consensus

- Join: network consensus on the nodes in the network, and their IDs
- Recipe: 2-phase operation
  - Phase 1: Collect the addresses of joining nodes
    - Initiator: Commit on who joined and assign IDs
  - Phase 2: Disseminate
    - Stop when nodes know their IDs, as indicated by the progress flags
- Build complex services using the recipe (single or multi-phase rounds)



Outline

Motivation

Consensus

- A2 Services – Agreement as a service
- Group Membership

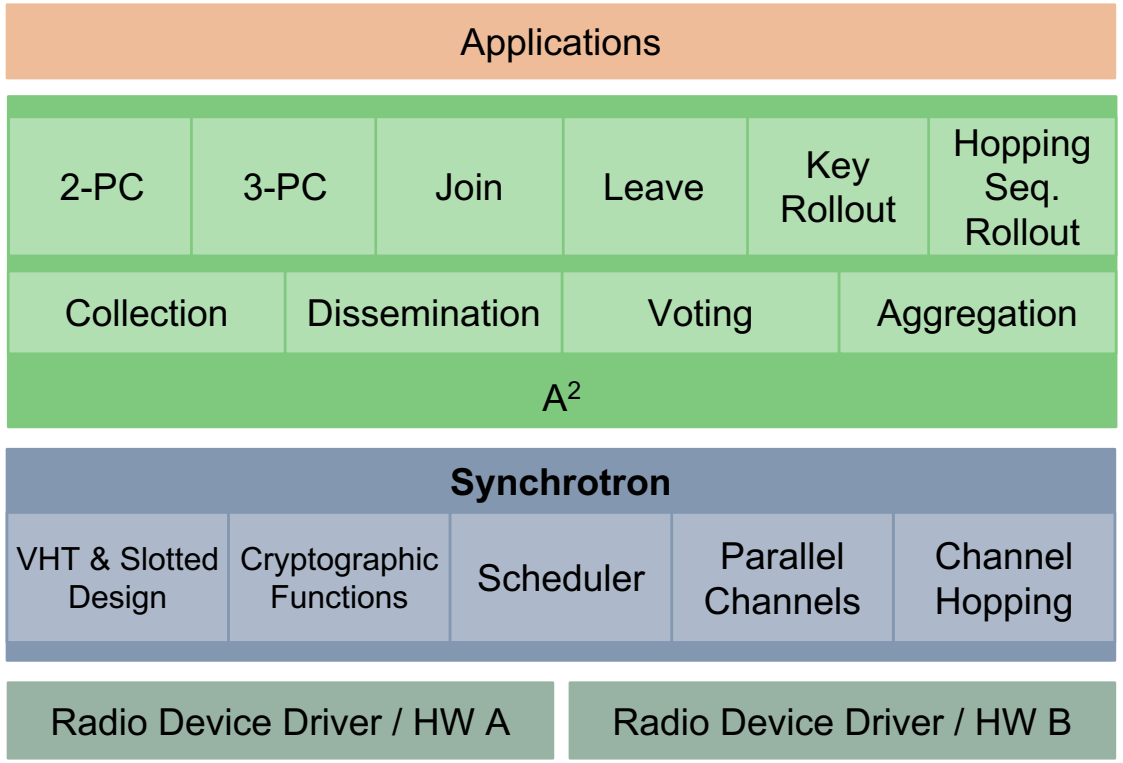
MAC LAYER: SYNCHROTRON

Evaluation

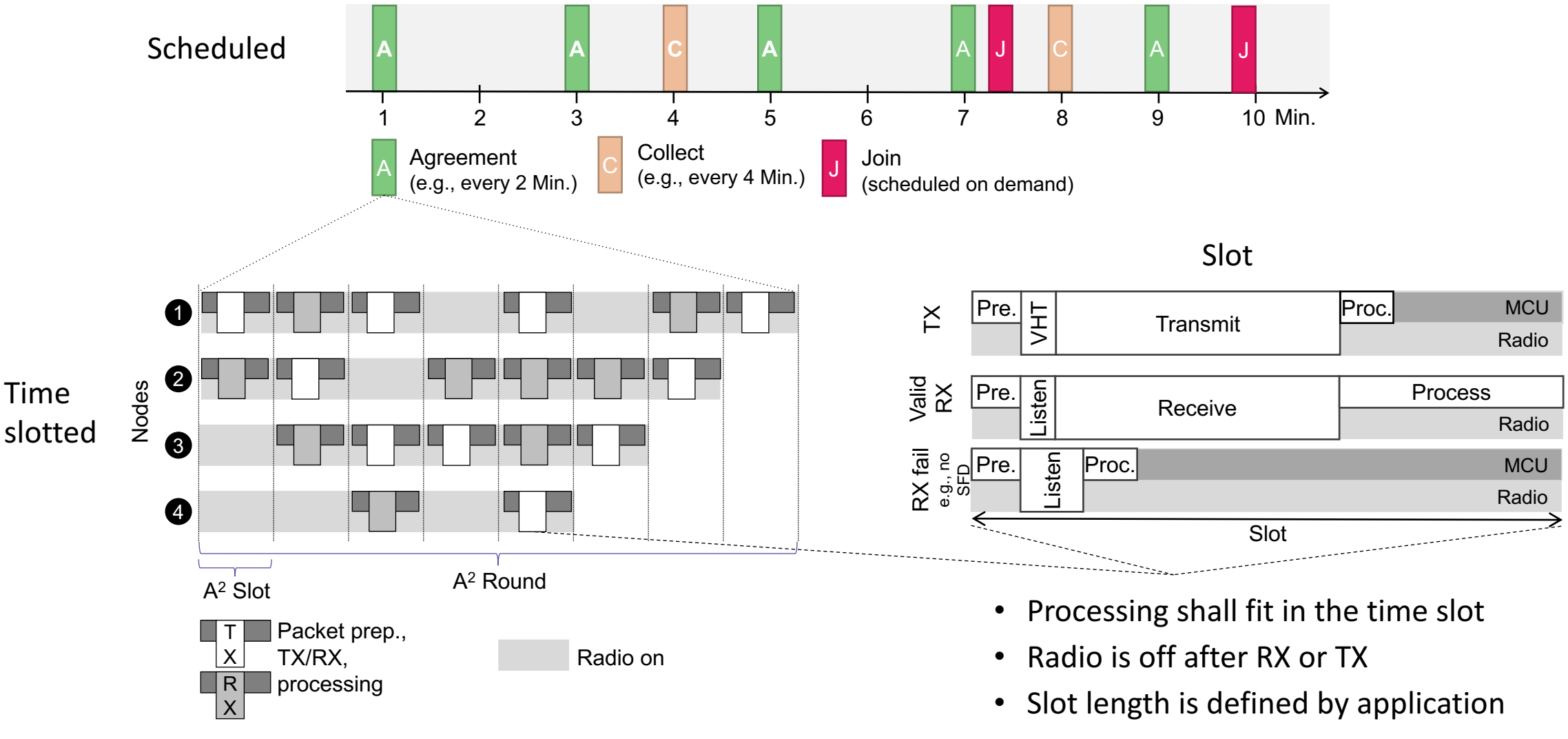
Conclusion

MAC Layer: Synchrotron

- Reliable synchronous transmissions kernel
- Time-slotted design
- Flooding-based (capture-effect)
  - No need for routing
- Robust: Multichannel
- Speedup: Parallel Channels



Operation



- Processing shall fit in the time slot
- Radio is off after RX or TX
- Slot length is defined by application

Outline

Motivation

Consensus

- A2 Services – Agreement as a service
- Group Membership

MAC layer: Synchrotron

EVALUATION

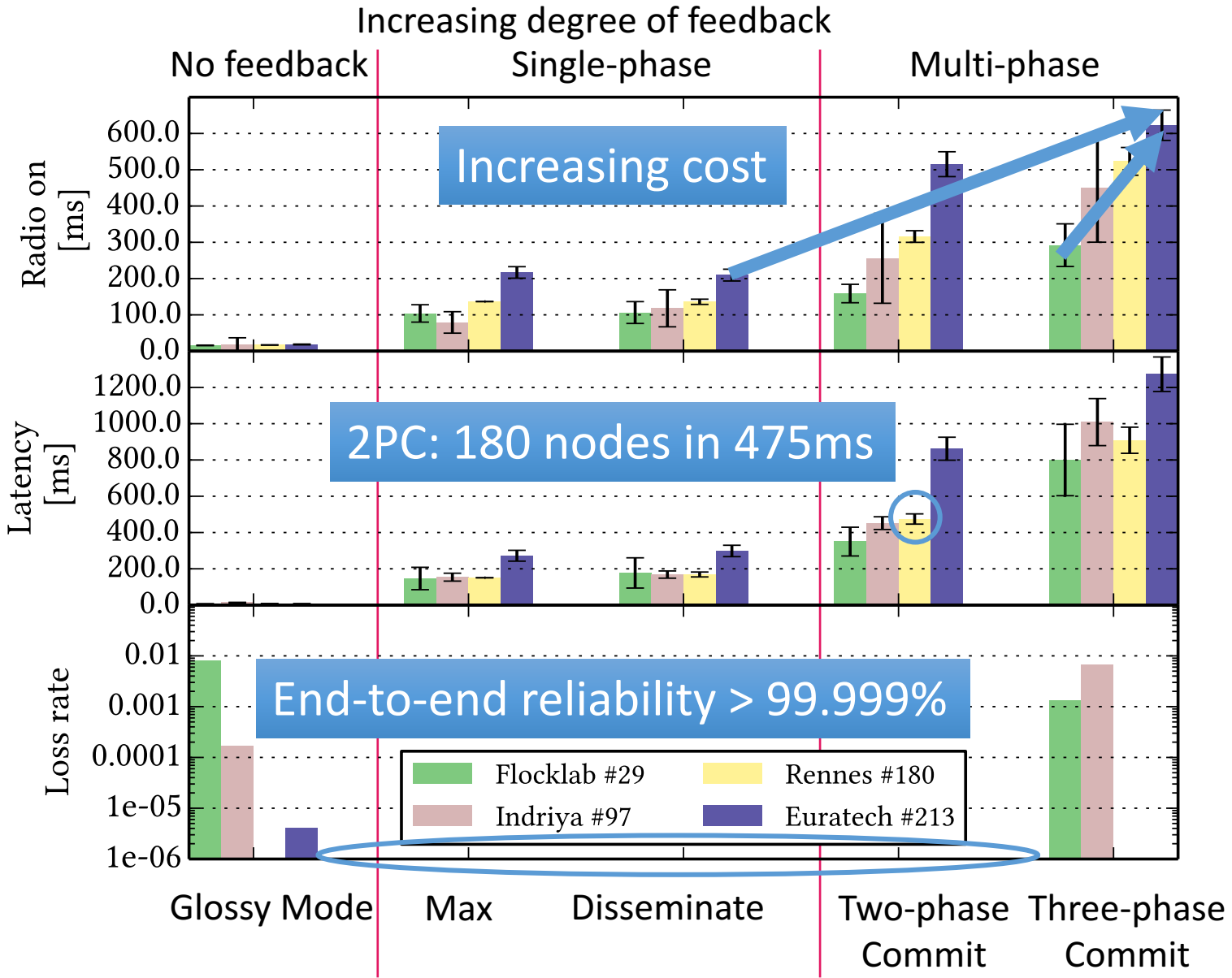
Conclusion

Cost of Consensus

- Compare
  - A² services
  - Increasing complexity
- Scenario
  - Periodic run
- Four testbeds
- 29-213 nodes

Flocklab #29 Rennes #180
Indriya #97 Euratech #213

- TelosB
  - CC2420 – 2.4GHz IEEE 802.15.4



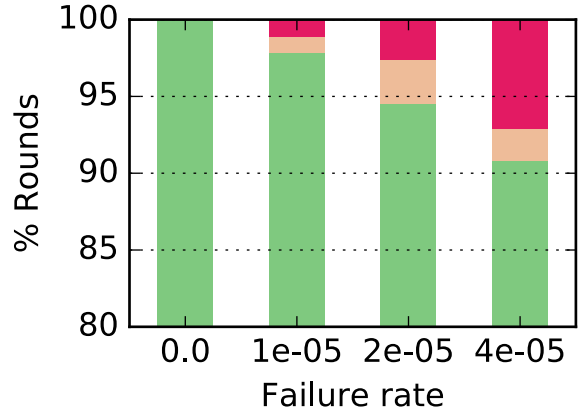
CRC problems

- Testbed with 200 nodes
  - Joined more than 200 nodes
    - Non-existing node IDs!
  - Nobody is sending the strange IDs
- Reason: CRC collisions (802.15.4: 16bit CRC)
  - Broken packets (invalid data) with valid CRCs
    - Note: A^2 send many packets
    - Probability of collision: 2^{-16}= 15 per million
  - Workaround: HW cryptographic checksum – AES Message Integrity Check
    - Now: two checksums: 16bit CRC and 16bit cryptographic MIC
    - Reduces the probability of collisions

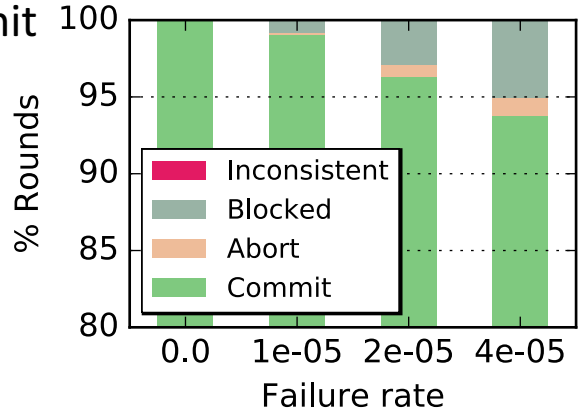
Consistency of Consensus Primitives

- Run transactions that we expect to commit
  - Vote, two-phase commit, three-phase commit
  - Fuzz the protocol: Inject random failures (programmatically)
- Possible outcome of each transaction
  - Inconsistent: some nodes abort, others commit
  - Blocked: to recover later
  - Consistent: all abort or commit
- Fundamental tradeoff:
  Consistency, Availability, Partition tolerance
  - **Impossible** to guarantee when failures can happen [FLP]
  - 2PC: block to handle failure
  - 3PC: abort to handle failure, but sometimes inconsistent

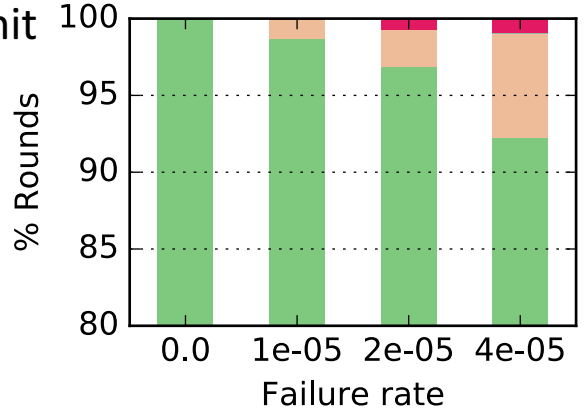
Vote



2-phase commit



3-phase commit



Outline

Motivation

Consensus

- A2 Services – Agreement as a service
- Group Membership

MAC layer: Synchrotron

Evaluation

LIMITATIONS & CONCLUSIONS

Limitations

- Point-to-point communication over an all-to-all protocol is costly
- Support on different radio technologies is not simple
  - Capture-effect support?
  - Tight synchronization
- Scaling
  - Packet Capture performance declines with density
  - Bitmap for votes and progress flags

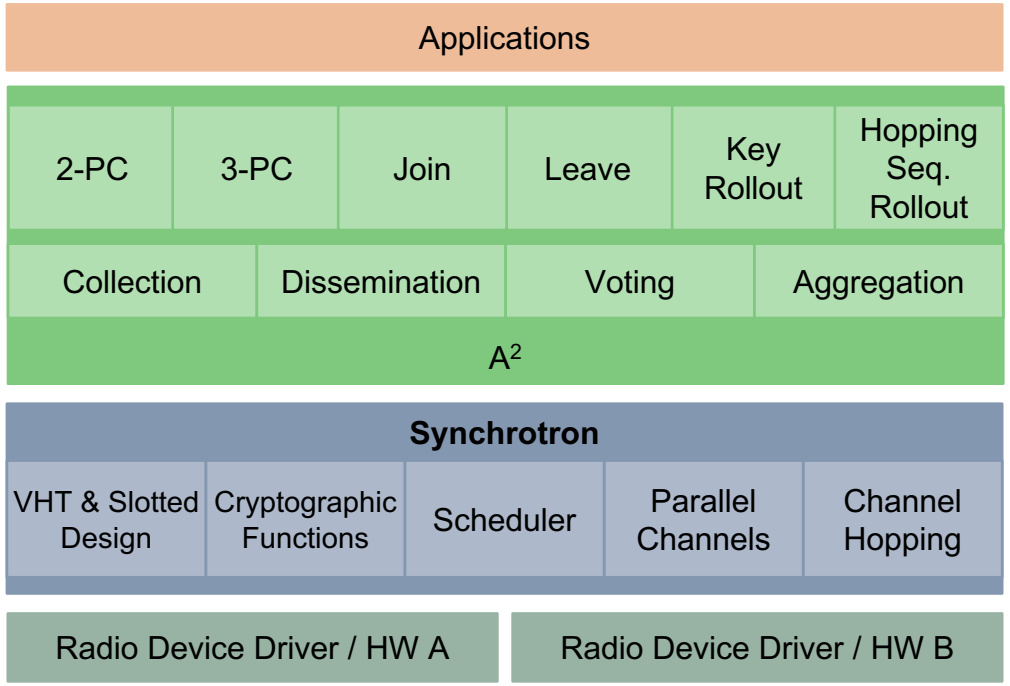
Conclusions

- A^2 & Synchrotron
  - Network-wide primitives
  - Consensus: two/three phase commit
  - Group Membership
- No traditional network stack
  - No Routing, link estimation, …
  - Flooding based - capture effect
- Results
  - High end-to-end reliability
    > 99.999%
  - Network-wide consensus
    2PC: 475ms with 180 nodes
- Ongoing work
  - Leader election (**poster, today 18.00**)
  - Paxos

- Contiki implementation
- Open source: BSD
- Try it: https://github.com/iot-chalmers/a2-synchrotron

Thanks! Questions?

Network-wide Consensus
Utilizing the Capture Effect in Low-power Wireless Networks



CHALMERS
UNIVERSITY OF TECHNOLOGY

Distributed Computing and Systems

Network-wide Consensus

Utilizing the Capture Effect in Low-power Wireless Networks

Beshr Al Nahas
Chalmers

Simon Duquennoy
RISE SICS

Olaf Landsiedel
Chalmers

November 6, 2017
Delft, ACM SenSys

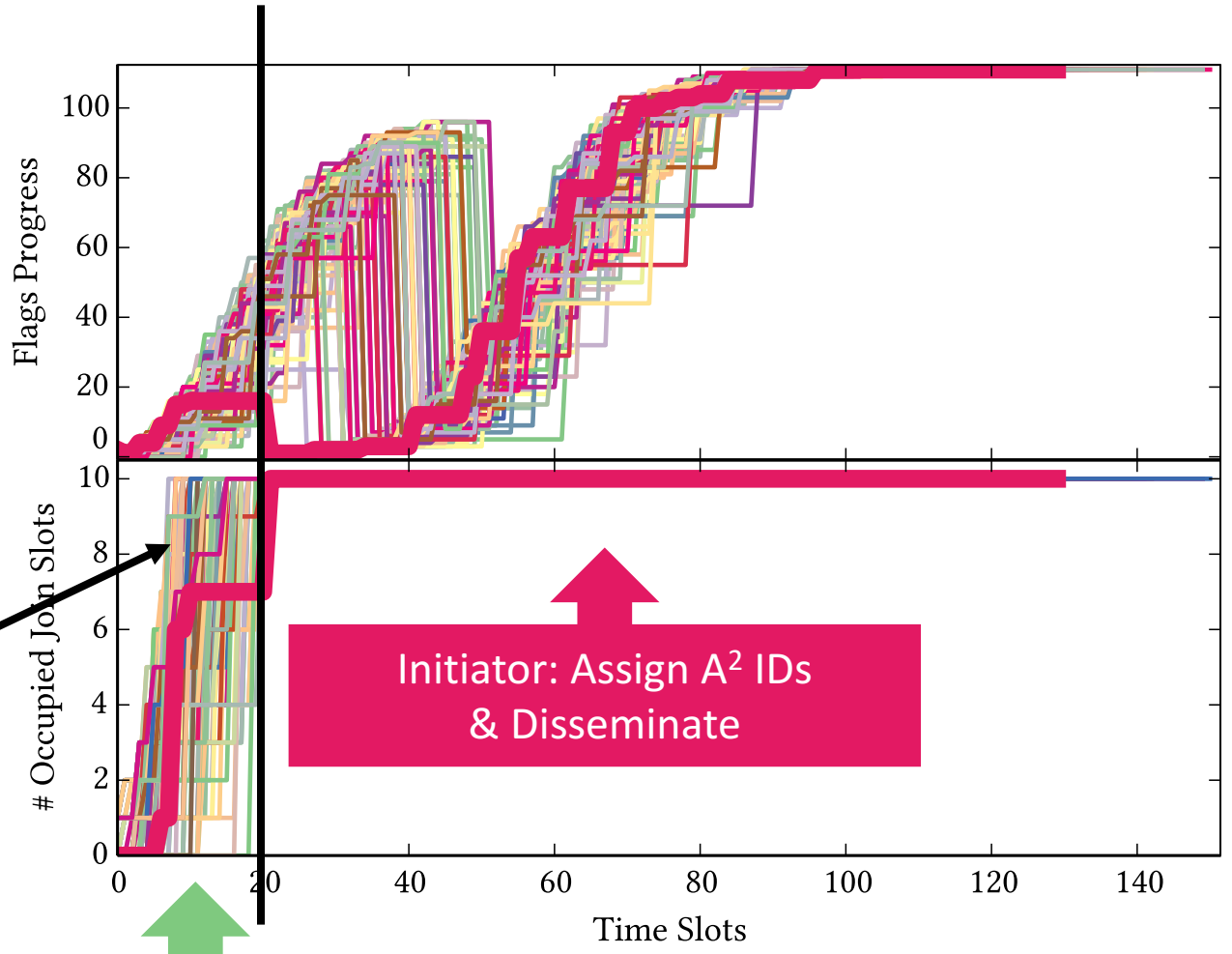
Join Performance

- Group Membership: Join
- Euratech: 212 nodes
  - Join 10 nodes per round
    - Always 10

→ 22 rounds

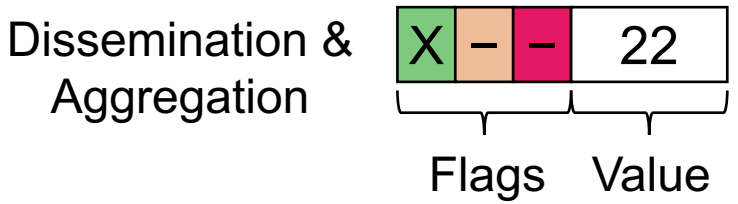
→ 23 seconds

Flags | Phase | Node IDs of joining nodes | Network IDs assigned to joining nodes

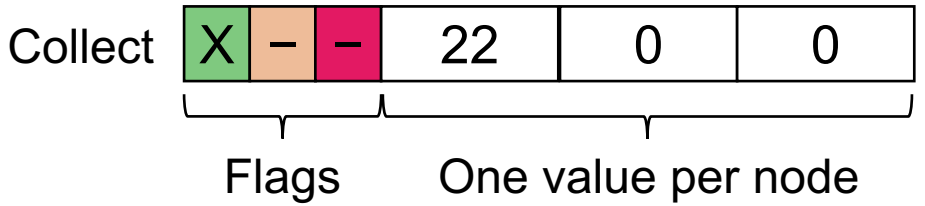


All-To-All Communication

- **Aggregation**; e.g., max value: Flags? Values?
  - Flags: show completion, one flag per node
  - Value: max value

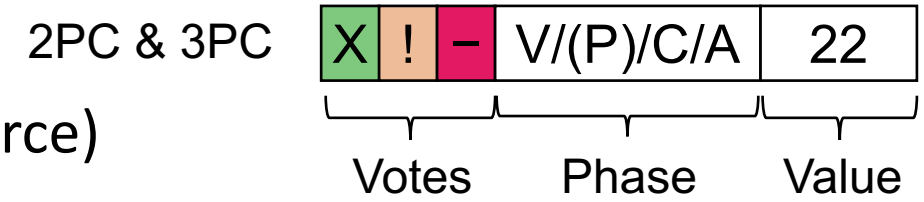


- **Data dissemination**: Flags? Values?
  - Flags: acknowledgment, one per node
  - Value: disseminate, set by source - does not change during round
- **Data collection**: Flags? Values?
  - Flags: show completion, one per node
  - Value: one value per node: collect data all-to-all



All-To-All Communication

- **2-Phase Commit**: Flags? Values?
  - One node makes a proposal, others accept (or not)
    - Two phases: first ready to commit, then commit (or abort)
  - Flags: show completion, two flags per node
    - First flag: vote: ready to commit
    - Second flag: commit
  - Value: value to commit on, set by proposer (source)
  - 3-Phase Commit: 3 flags per node

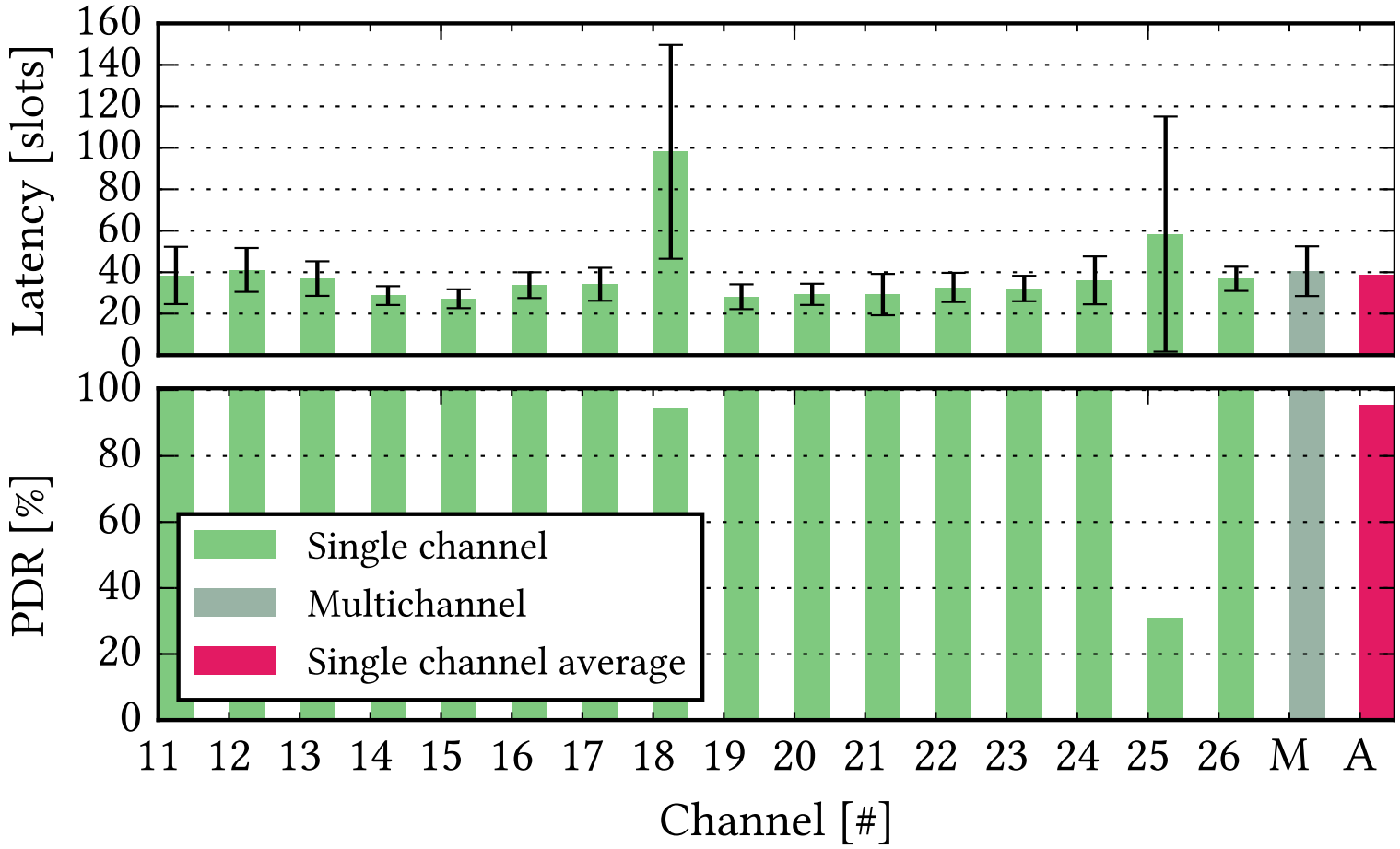


Evaluation

- Four public testbeds – University campuses
  - Flocklab 29 nodes – 1 floor – ETH
  - Indriya 97 nodes – 3 floors – Singapore
  - IoTLab Rennes 180 nodes – 1 hall – INRIA
  - IoTLab Euratech 213 nodes – 1 hall – INRIA
- Implementation on Contiki OS
  - CPU 16bit MSP430 – Radio CC2420 IEEE 802.15.4

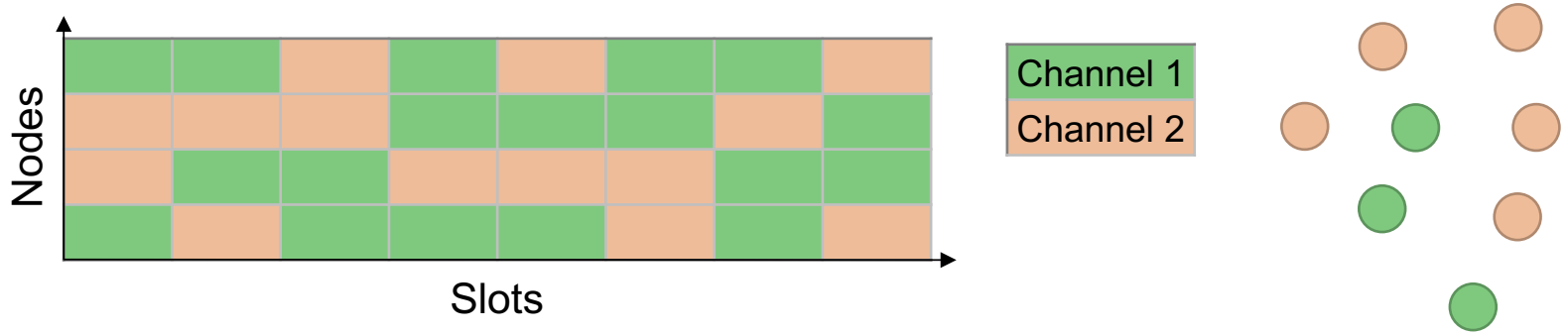
Channel Hopping

- Change the channel every time-slot
- Flocklab:
  29 nodes



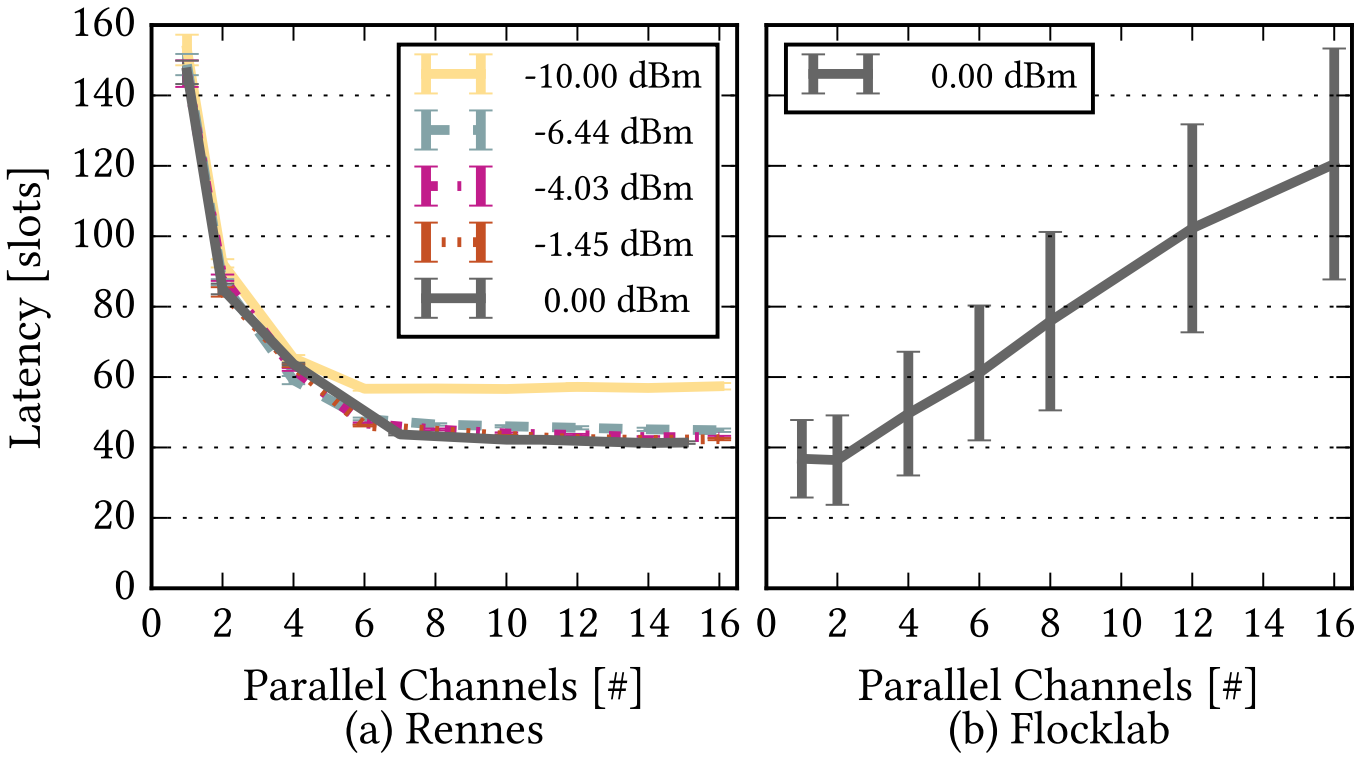
Speeding up: Parallel Channels

- Capture-effect performance drops with density
  - i.e., packets are corrupted if too many nodes send
- Use channels in parallel
- Every nodes switch to random channel
  - → Nodes form dynamic clusters
  - → Spatial reuse → Data converges quicker



Parallel Channels

- Combination of TX power and number of channels
- Dense testbed:
  Rennes – 180 nodes
  - Achieve x 2-3 lower latency (→ power)
- Sparse testbed:
  Flocklab – 29 nodes
  - No effect on latency



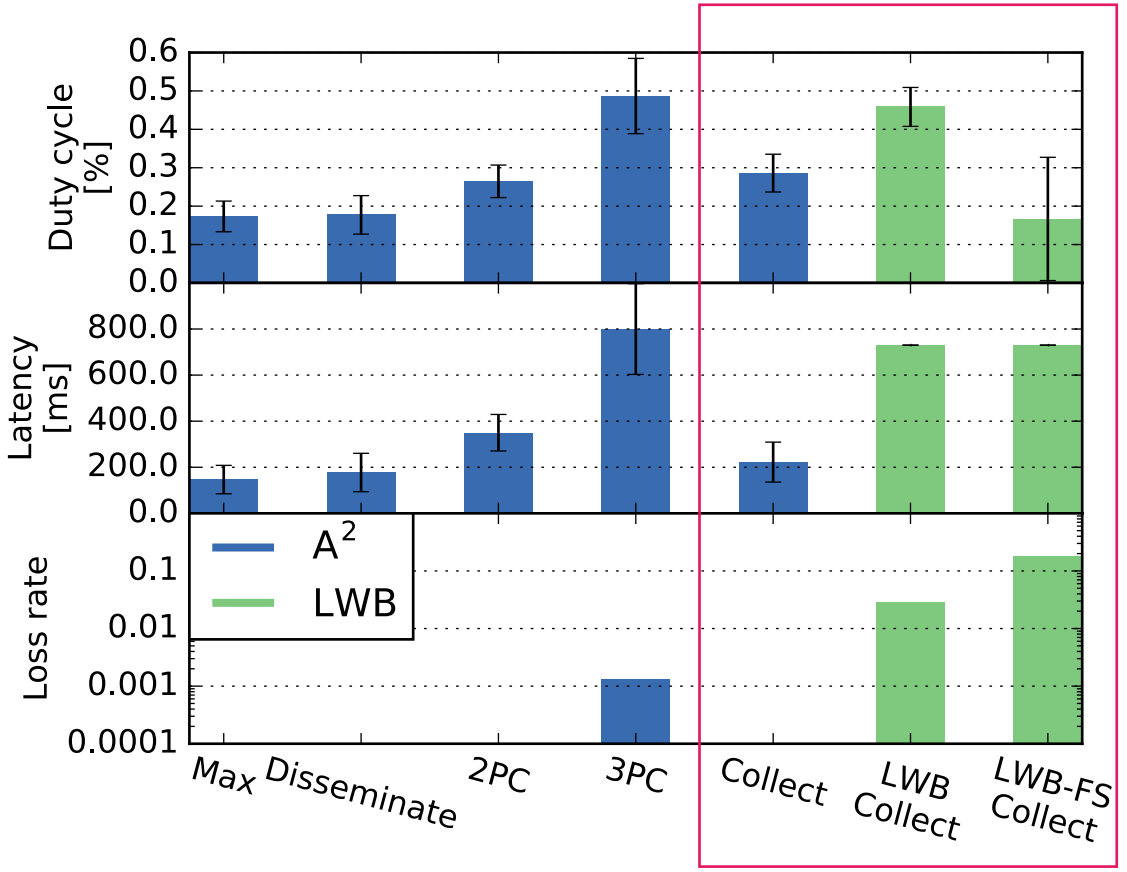
Reliability

- Two 180 nodes testbeds
- Running Max
- 2 x 3.7 million points collected successfully without a single loss (end-to-end)

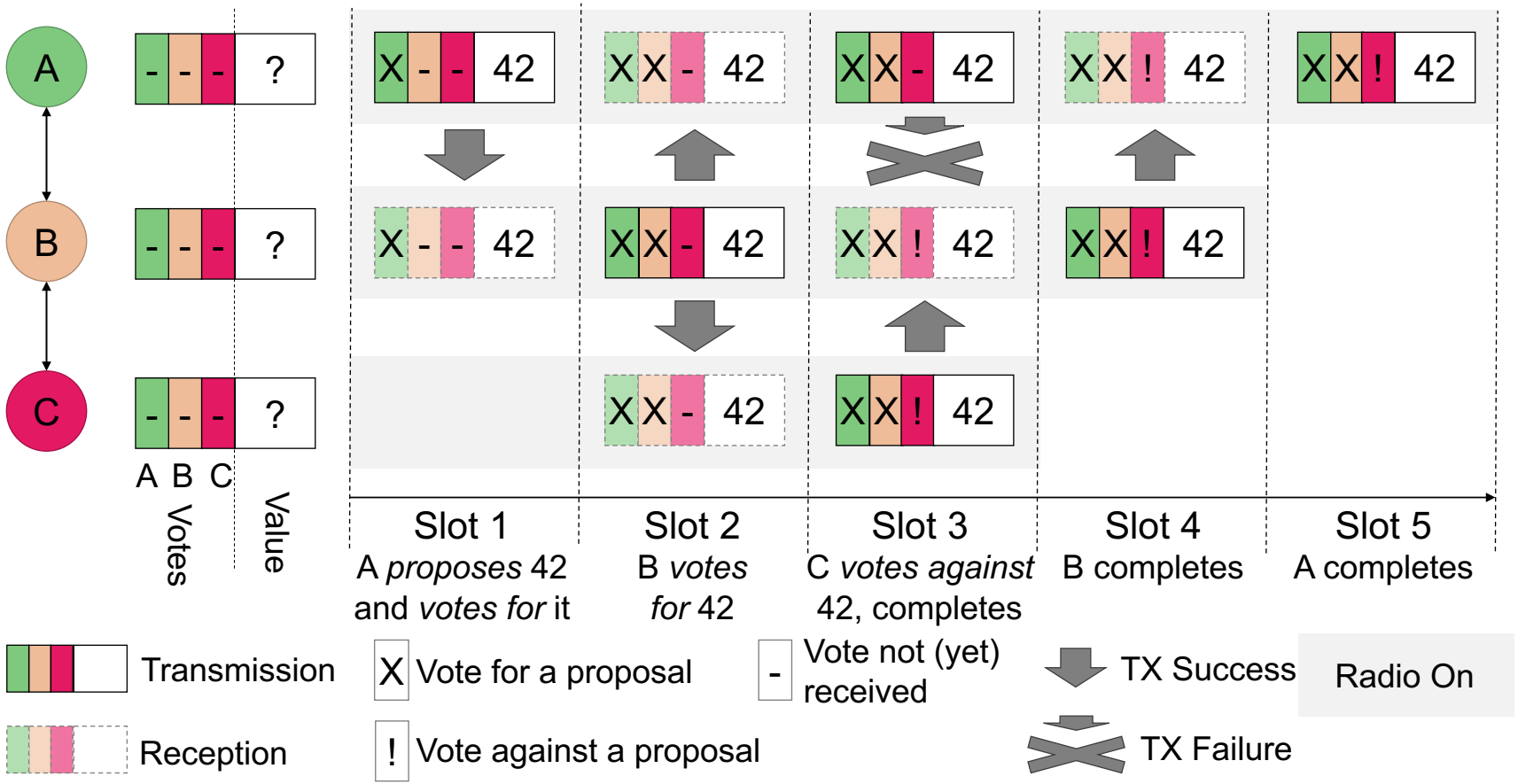
Comparison to LWB

- 29 nodes testbed
- 1 minute interval

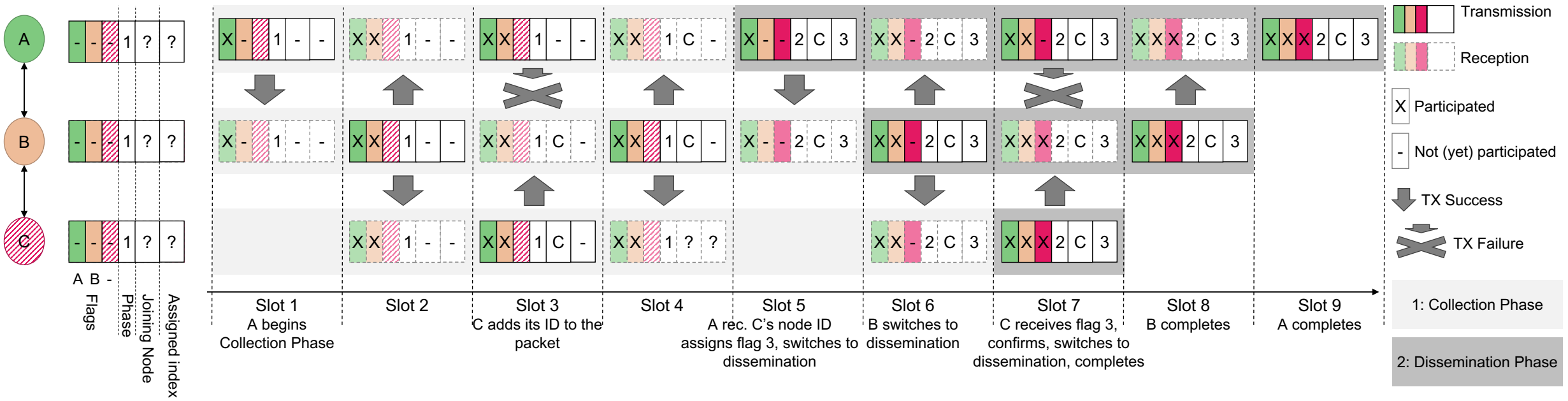
- Lower latency
- No losses
- Low duty cycle



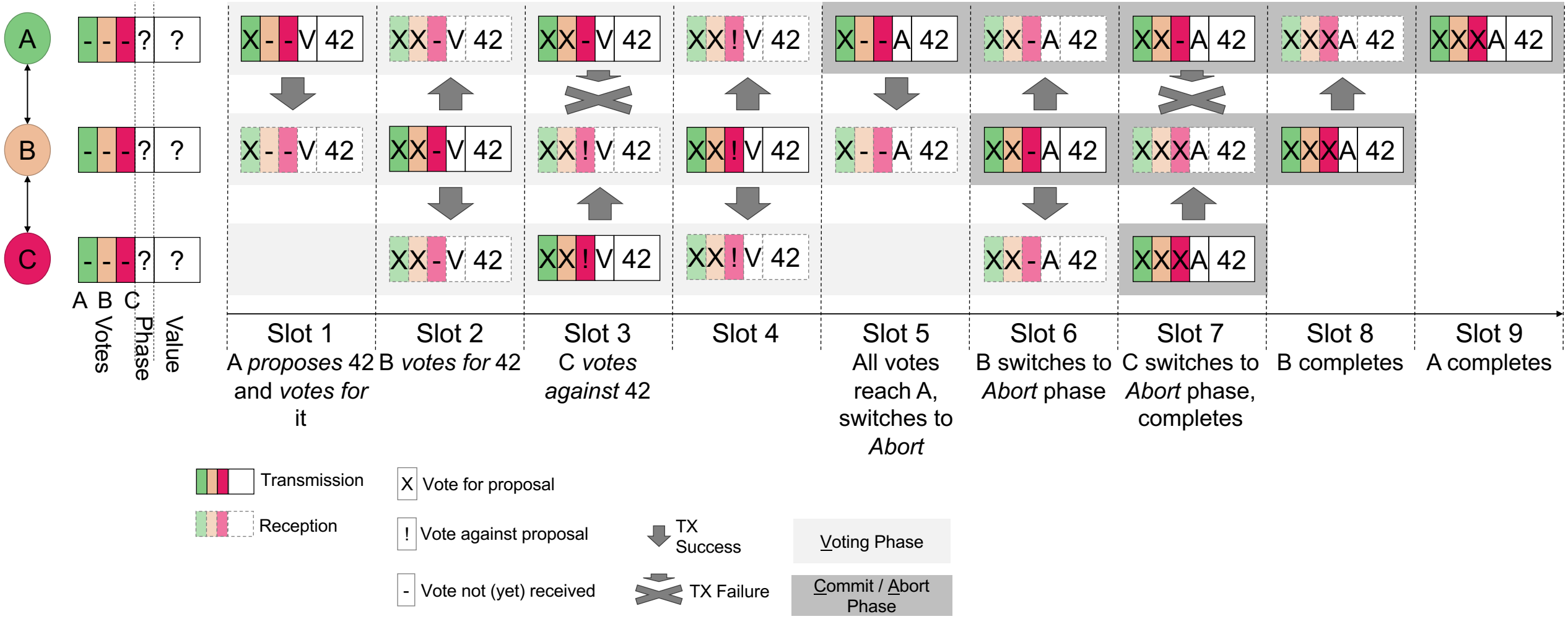
Voting


A
B
C
- - - ?
A B C
Votes
Value
X - - 42
X X - 42
X X ! 42
Slot 1
A *proposes* 42 and *votes for* it
Slot 2
B *votes for* 42
Slot 3
C *votes against* 42, completes
Slot 4
B completes
Slot 5
A completes
Transmission
Reception
X Vote for a proposal
! Vote against a proposal
- Vote not (yet) received
TX Success
TX Failure
Radio On

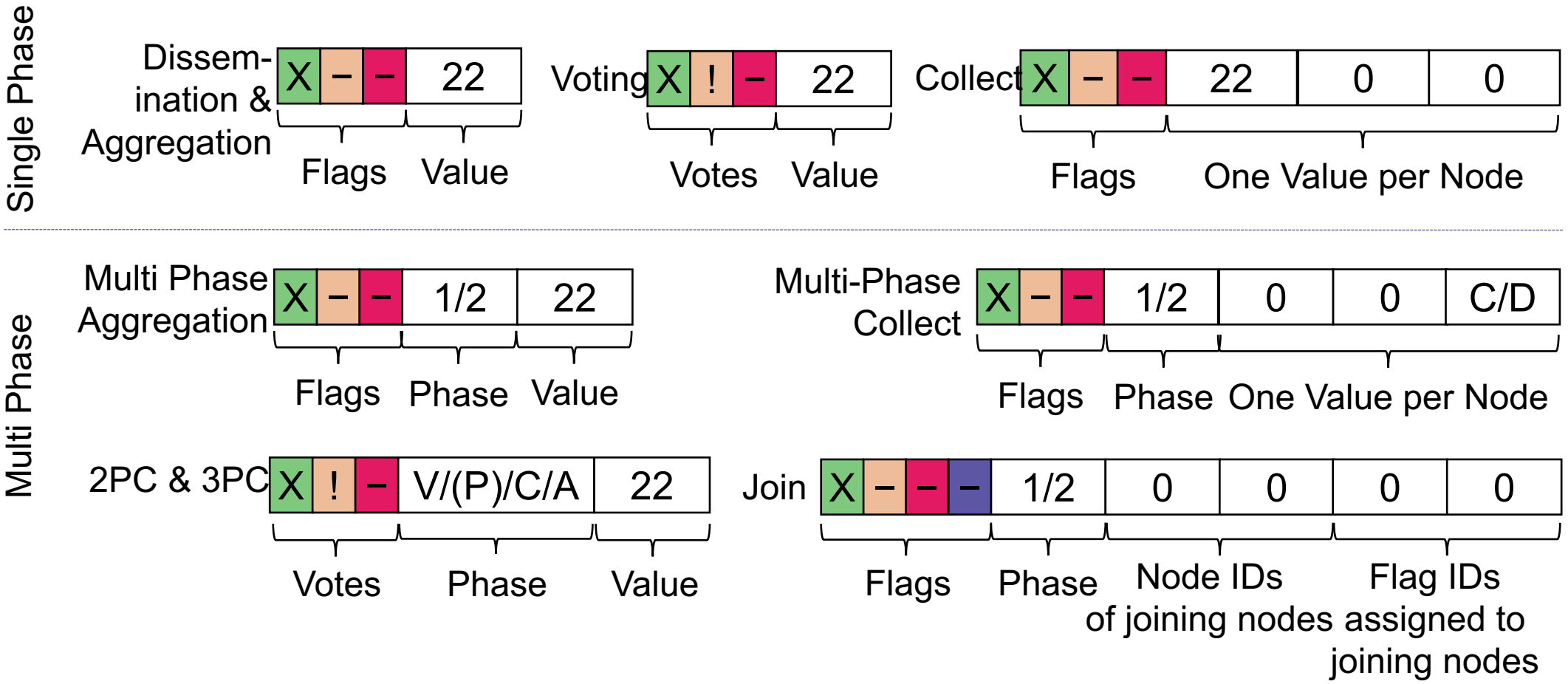
Join


A
B
C
- - - 1 ? ?
A B -
Flags
Phase
Joining Node
Assigned index
Slot 1
A begins Collection Phase
Slot 2
Slot 3
C adds its ID to the packet
Slot 4
Slot 5
A rec. C's node ID assigns flag 3, switches to dissemination
Slot 6
B switches to dissemination
Slot 7
C receives flag 3, confirms, switches to dissemination, completes
Slot 8
B completes
Slot 9
A completes
Transmission
Reception
X Participated
- Not (yet) participated
TX Success
TX Failure
1: Collection Phase
2: Dissemination Phase

Two-phase Commit

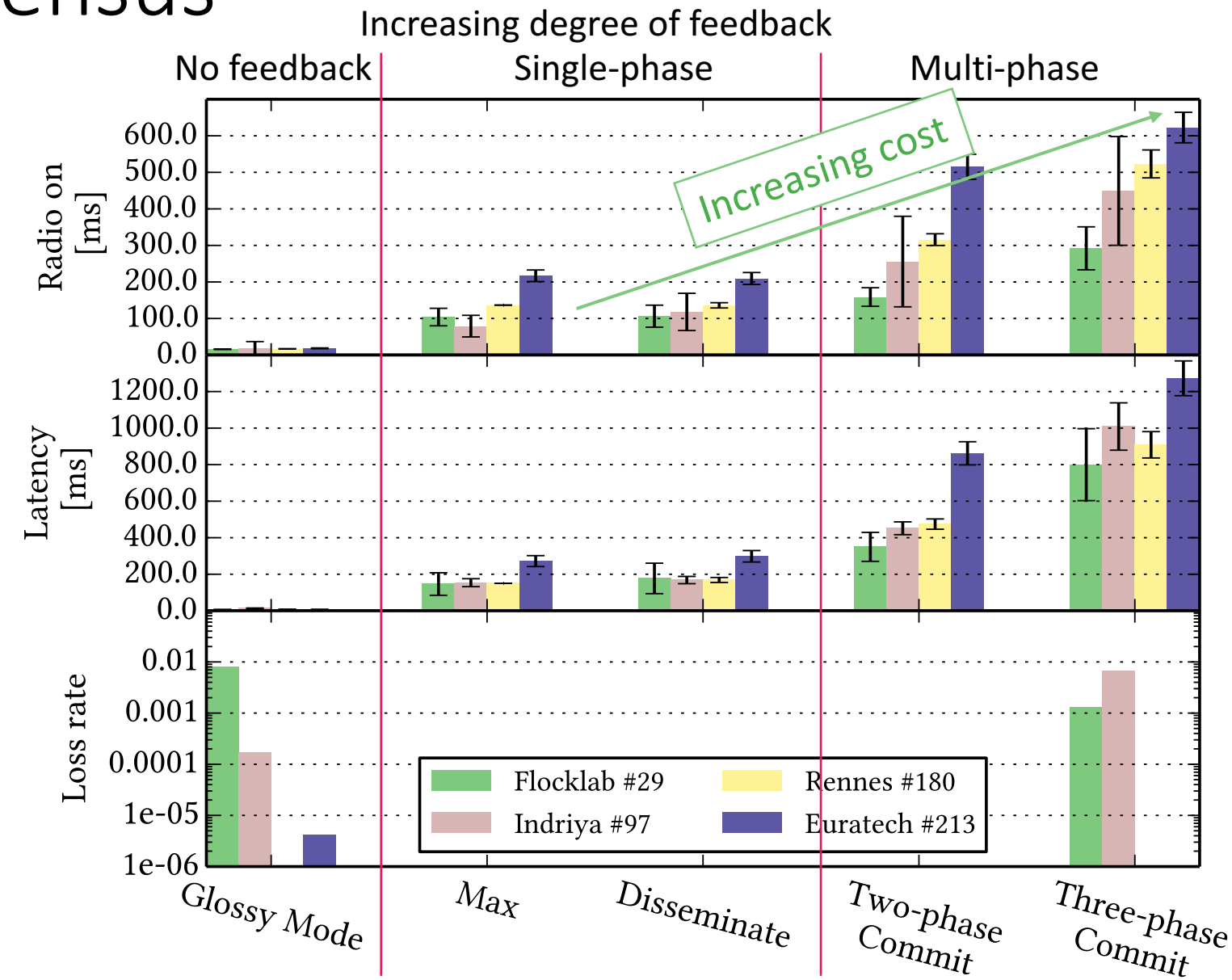


All-To-All Communication

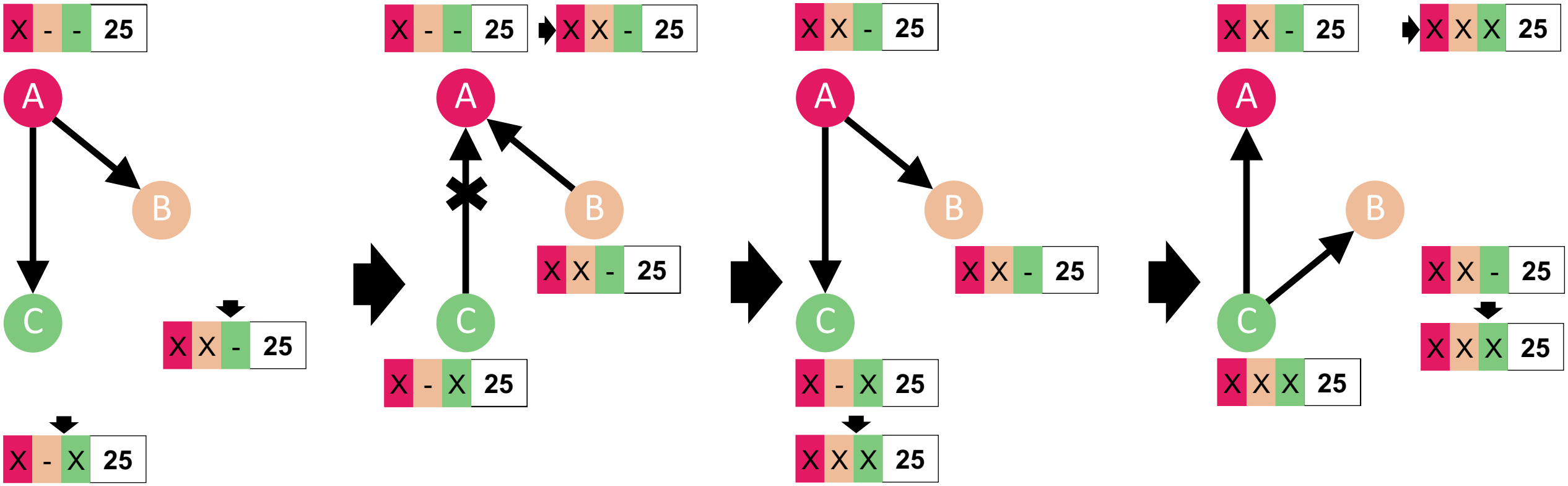


Cost of Consensus

- Four testbeds
- 29-213 nodes
- TelosB
  - CC2420 – 2.4GHz IEEE 802.15.4



Two-phase Commit: (1) Voting



- A initiates a round
- By sending its proposal
- Receivers: vote

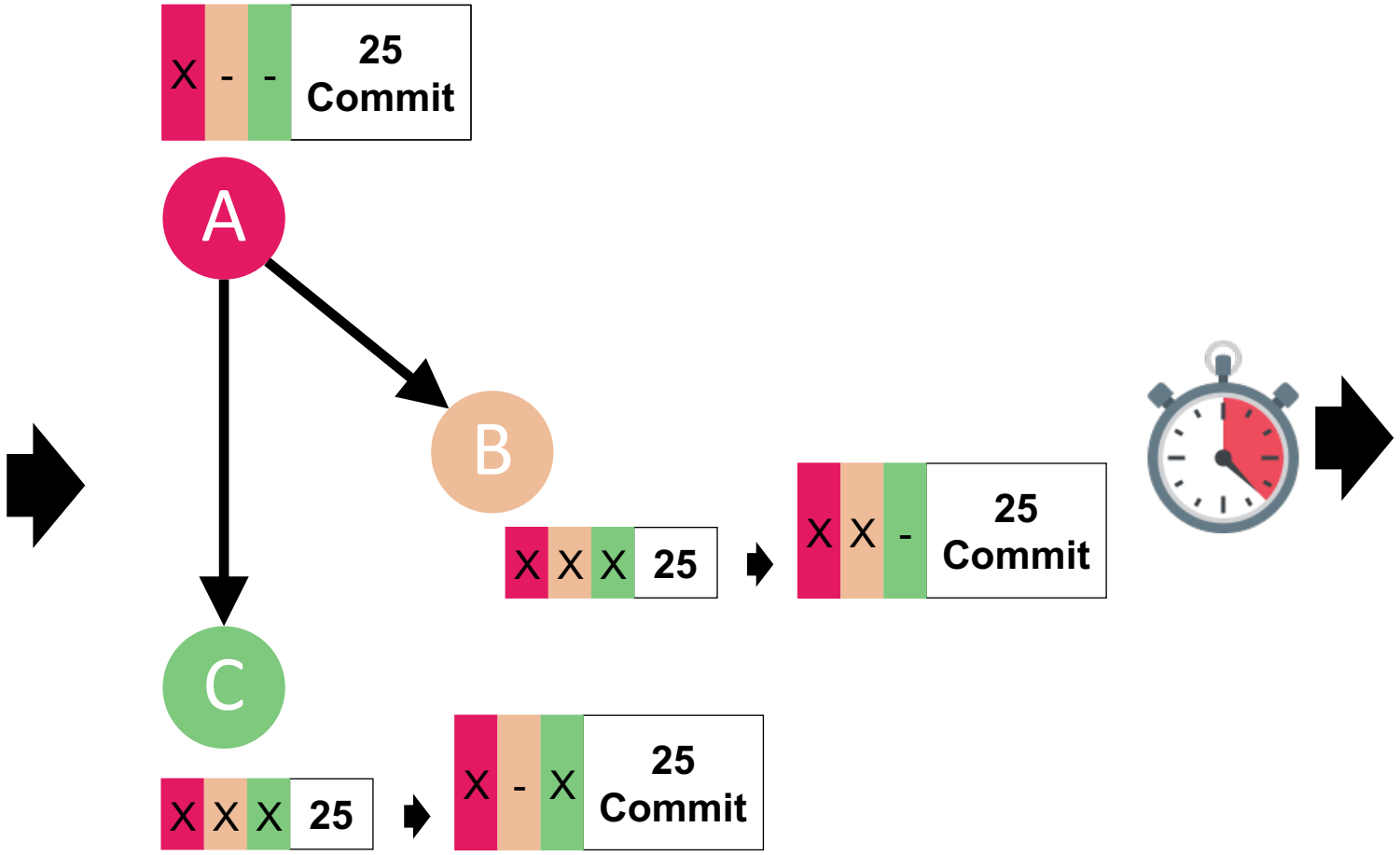
- B, C send concurrently
- A receives the stronger one (capture effect) and combines with local state

- A learned new information
- After the third round
  - C has learned the final vote

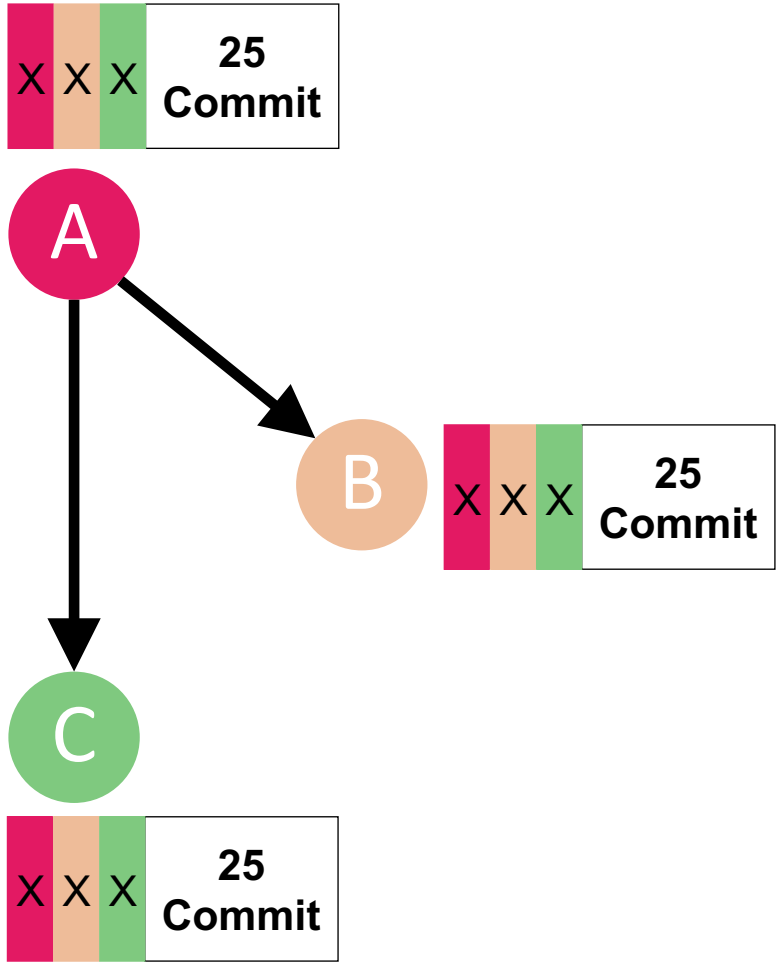
- C learned new info: send
- A, B receive
- A realizes everybody agree

Two-phase Commit: (2) Commit

1. Collect votes (all-to-one)
2. Disseminate decision (one-to-all)



- A: commits and disseminates
- B, C receive and commit



- A, B, C merge and spread the new info they learn until all flags are set

Leadership

poster

Bootstrap (Ongoing)

- Deploy a network and elect leader
  - To be
    - Chaos/A2 initiator
    - LWB initiator
    - RPL root
    - …
- Here: Bootstrap Chaos/A2 without initiator
  - Elect the initiator
  - Ensure a single initiator

Design 1

- Bootstrapping and Clustering
  - Listen and randomly timeout if no leader
  - Propose self as leader
  - Listeners join the proposed leader as in A2

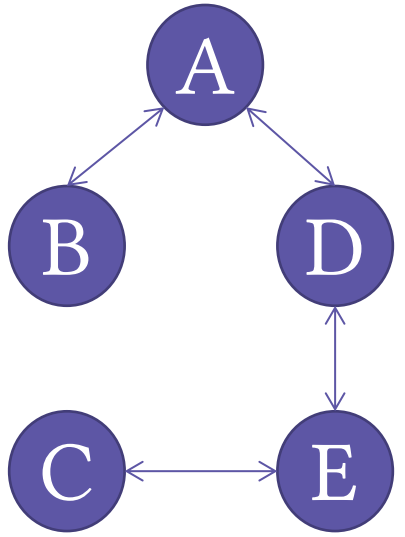
Design 2

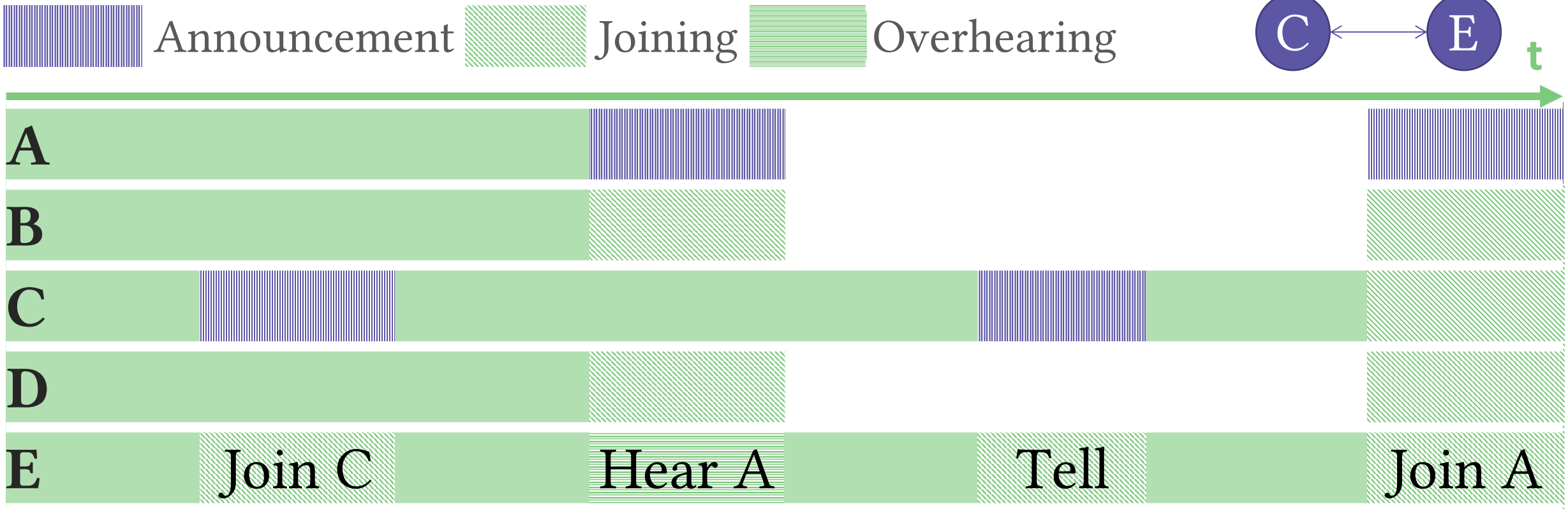
- Leader Election
  - Quorum stability condition: Cluster size > 50%
  - Nodes keep listening to find bigger clusters
    - Inform current cluster about the biggest heard
    - Cluster members agree on the biggest
      - Leave current and join the biggest
  - Only the majority cluster survives

Design 3

- Failure Recovery
  - Upon leader failure, nodes stop hearing packets
  - The random timeout mechanism kicks in
  - Restart the process and elect a new leader

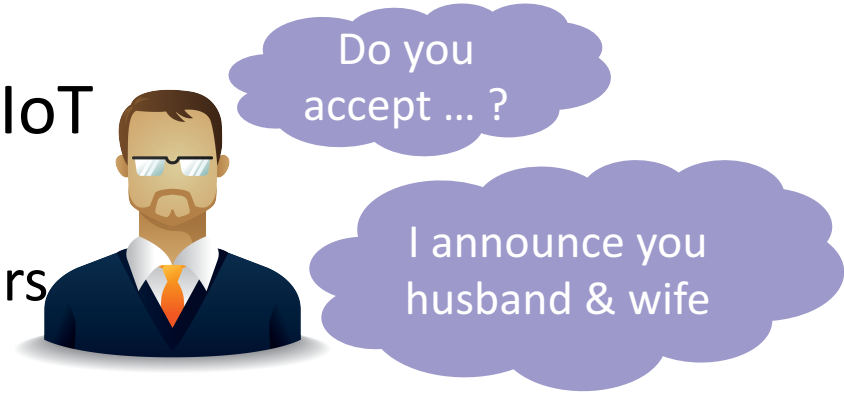
Example

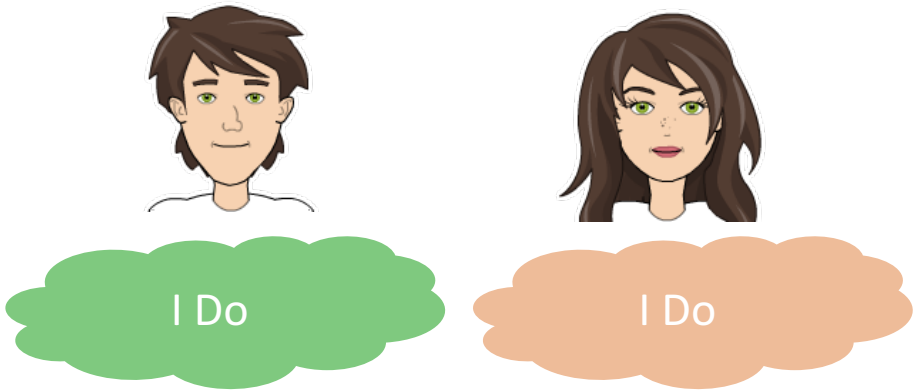

A
B
D
C
E


Announcement
Joining
Overhearing
t
A
B
C
D
E
Join C
Hear A
Tell
Join A

Consensus is hard

- Essential building block for mission-critical CPS / IoT
  - Vehicles or UAV agree on Maneuvers
  - Process control: ensure consistent settings of actuators
- Two-phase Commit Protocol*
  - Propose a transaction (one-to-all)
  - Collect votes (all-to-one)
  - Disseminate decision (one-to-all)
  - Acknowledgment (all-to-one)
- Challenges
  - Consensus is heavy: Talk to every node back and forth
  - Larger scale than done in the data centers





* Jim Gray, "Notes on DataBase Operating Systems", 1979.