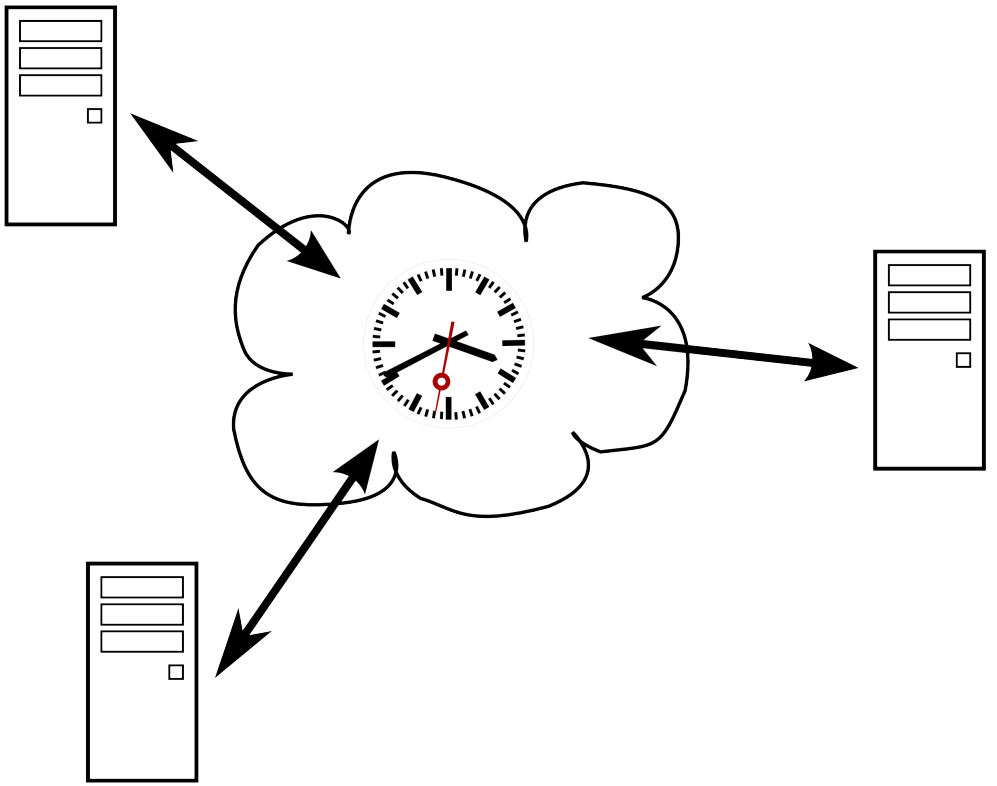
The Flooding Time Synchronization Protocol

by Maróti et al., 2004

Speaker: Ivo Steinmann

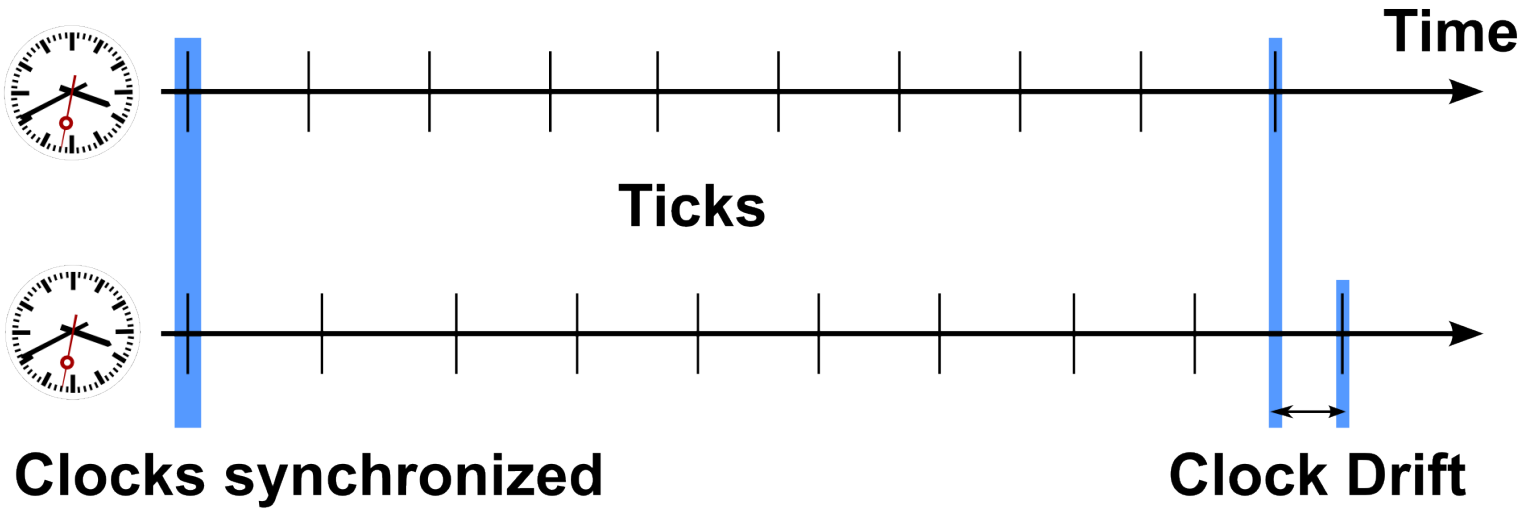
Time Synchronization

- Internal clocks of computers may differ
- Network with coherent view on time



Hardware Clocks

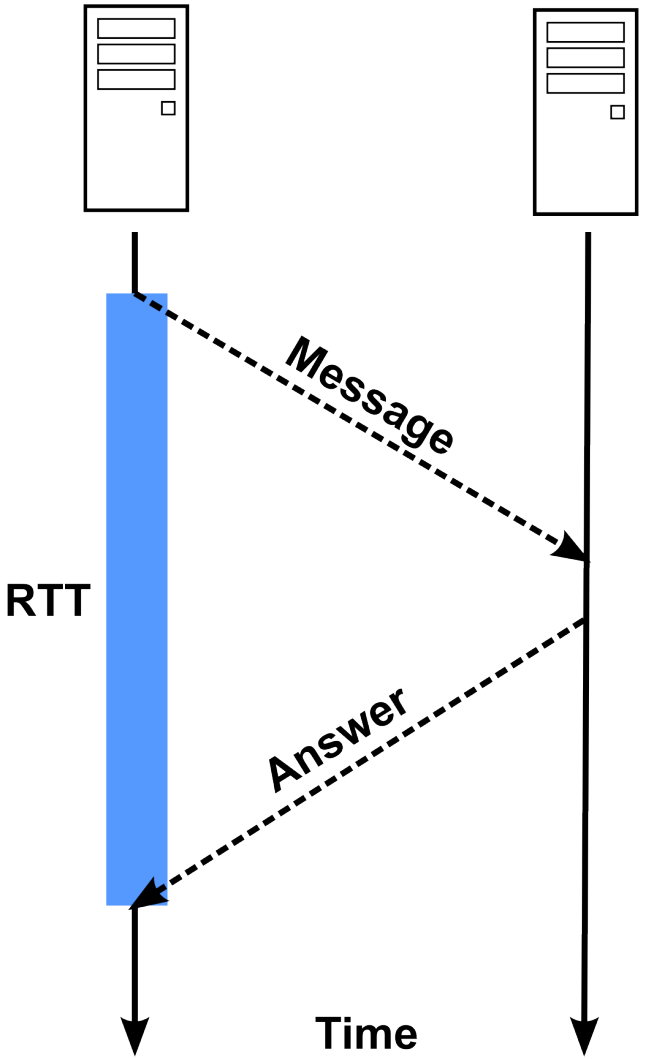
- The same hardware clock can have different clock drift rates at different occasions.
- Sensitive to temperature and power changes
- Resynchronization necessary after a certain time



Round Trip Time (RTT)

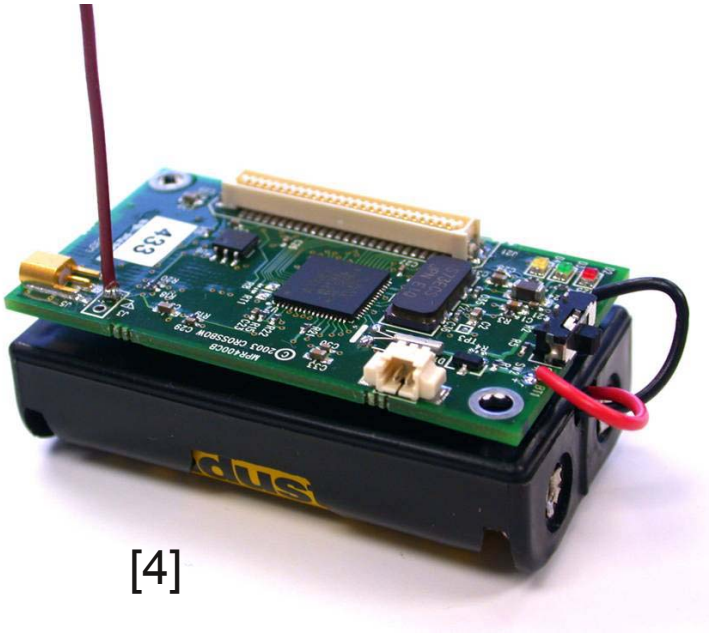
- The length of time it takes for a message to be sent plus the length of time it takes for an acknowledgment (answer) of that signal to be received
- Also known as ping time
- With the RTT it is possible for a host to compute the message delivery time to another host
- Used in Network Time Protocol (NTP)
- The receiver can compute the time of the sender by

```
offset = RTT / 2
sender_time = msg.timestamp + offset
```



Time Synchronization in Sensor Networks

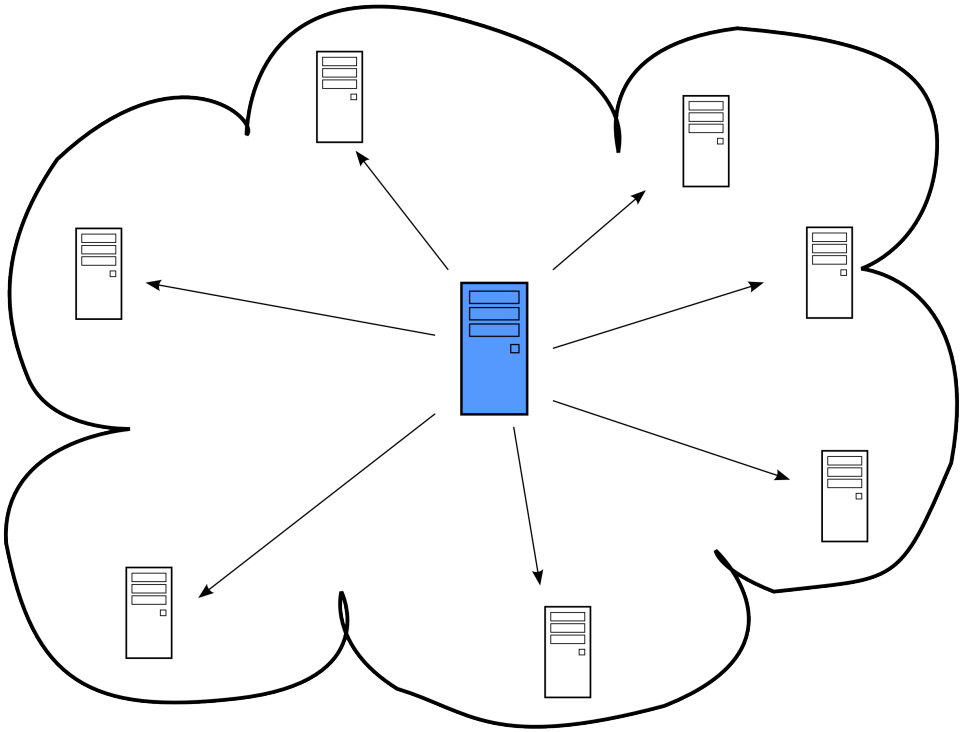
- Cloud of small devices (Embedded Systems)
- Limited computational power
- Limited available energy
- Non-deterministic wireless communication
- Robust to topology changes and node failures



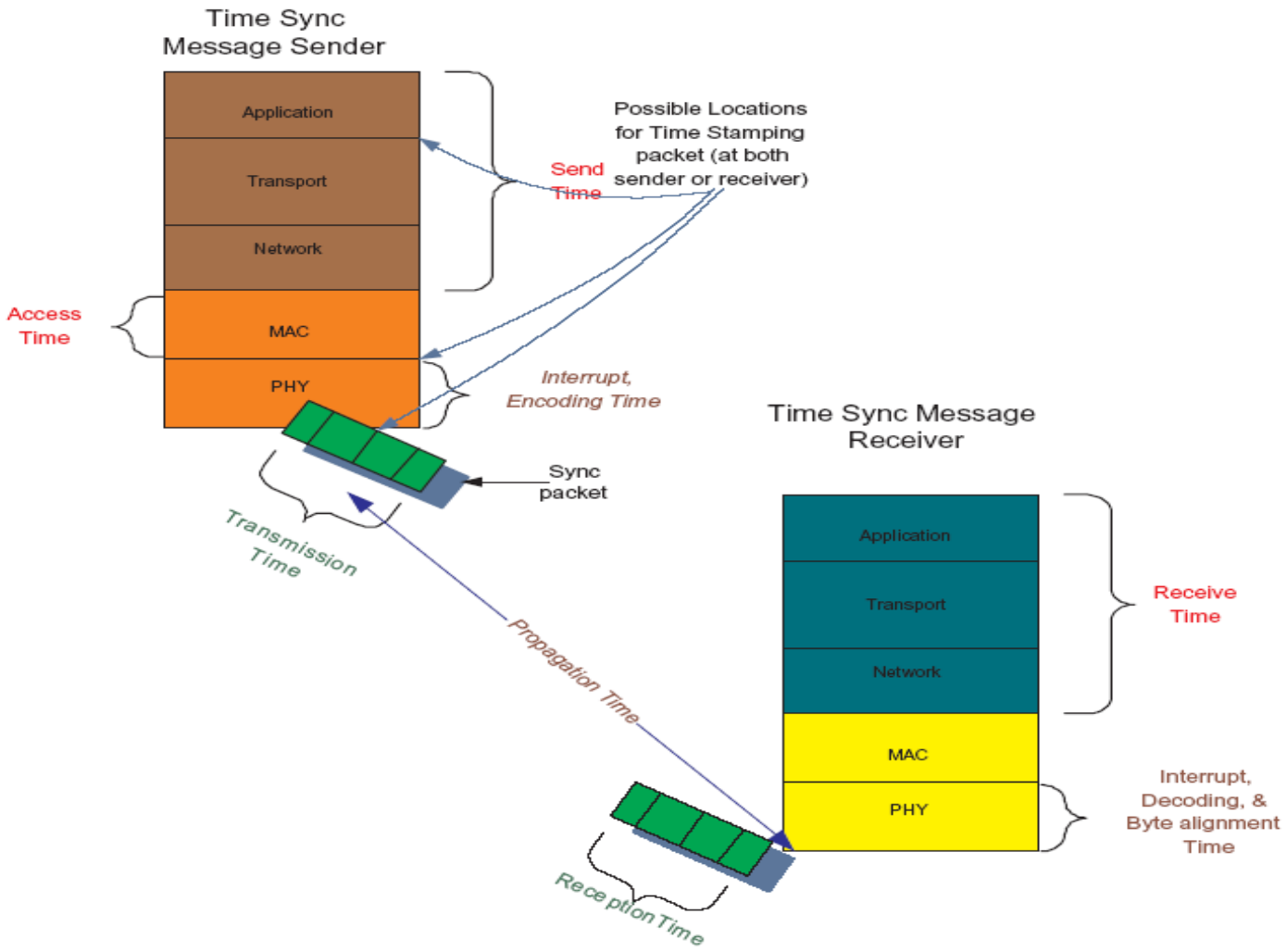
[4]

Broadcasting

- Broadcast messages with embedded timestamps to the neighbour nodes
- One way communication: The message delivery time of a broadcasted message cannot be calculated (no ack. message)
- Another method is required to compute the message delivery time
- But
  - it is energy efficient
  - it scales better



Uncertainties in Radio Message Delivery (1)



[2]

Uncertainties in Radio Message Delivery (2)

- **Send Time** – Depending on the system call overhead of the operating system and on the current processor load
- **Access Time** – message stays in the radio device and waiting for access to the transmit channel
- **Interrupt Handling Time** – the delay between the radio chip raising and the microcontroller responding to an interrupt
- **Transmission Time** – depending on the length of the message and the speed of the radio
- **Propagation Time** – depending on distance between sender and receiver (less than one microsecond for ranges under 300m)
- **Encoding Time** – for the radio chip to encode and transform a part of the message to electromagnetic waves
- **Byte Alignment Time** – differenent byte alignment of the sender and receiver. Is depending on processor power.

[1]

Typical Latencies

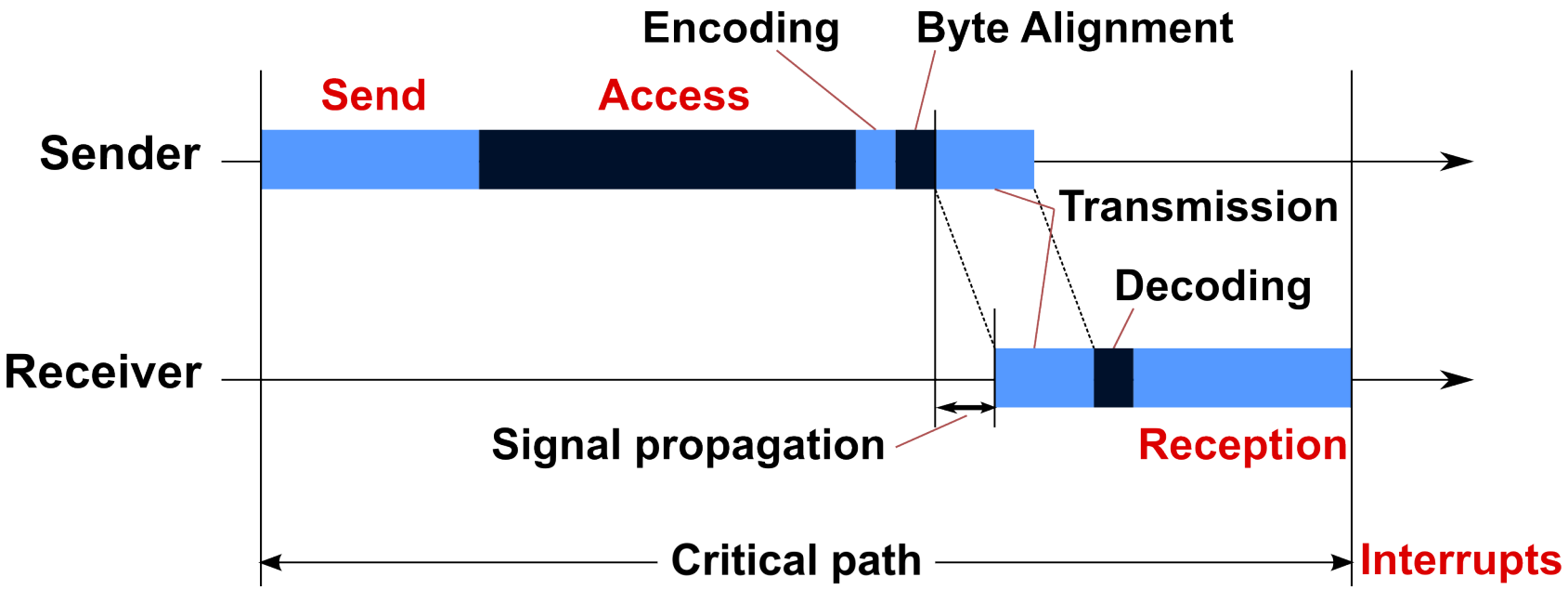
deterministic
non-deterministic

Send and Receive	0 – 100 ms	nondeterministic, depends on the processor load
Access	10 – 500 ms	nondeterministic, depends on the channel contention
Transmission / Reception	10 – 20 ms	deterministic, depends on message length
Propagation	< 1µs for distances up to 300 meters	deterministic, depends on the distance between sender and receiver
Interrupt Handling	< 5µs in most cases, but can be as high as 30µs	nondeterministic, depends on interrupts being disabled
Encoding plus Decoding	100 – 200µs, < 2µs variance	deterministic, depends on radio chipset and settings
Byte Alignment	0 – 400µs	deterministic, can be calculated

[1]

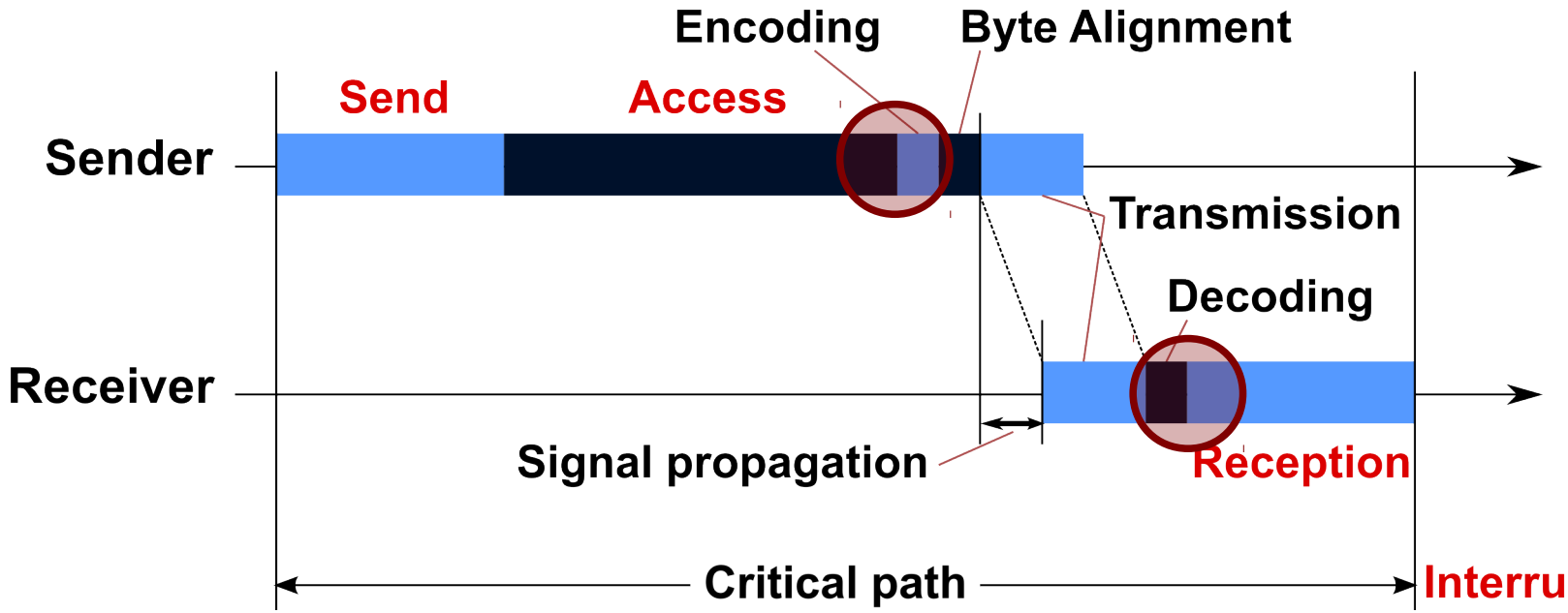
Communication Latencies

- Variations in latency (”jitter”) impact synchronization accuracy

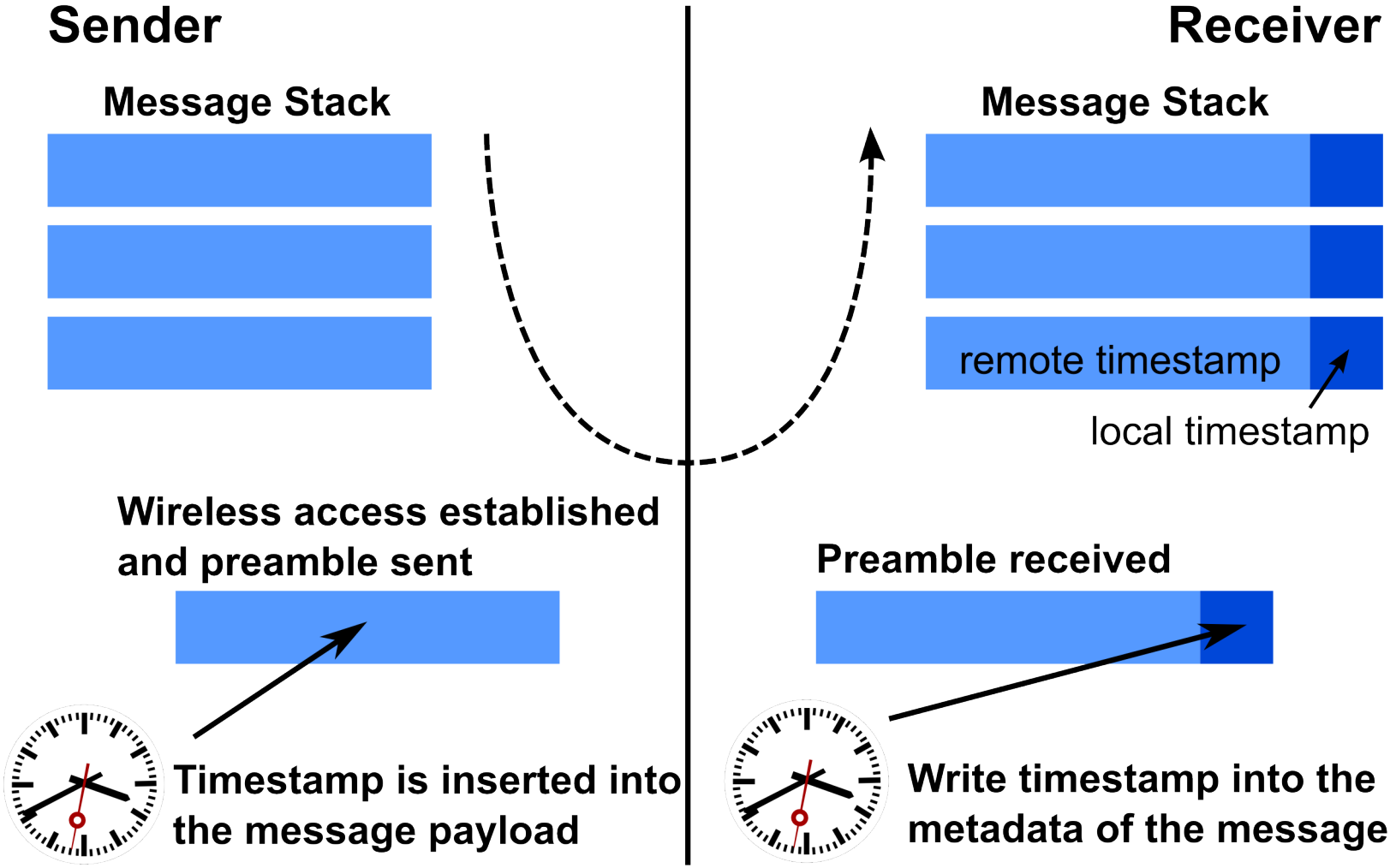


MAC Layer Timestamping (1)

- Sender – Read clock and write the timestamp just before sending the first byte of the radio message
- Receiver – Read clock just after receiving the first byte of radio message
- Interrupt Handling Time is the only uncertainty that stays

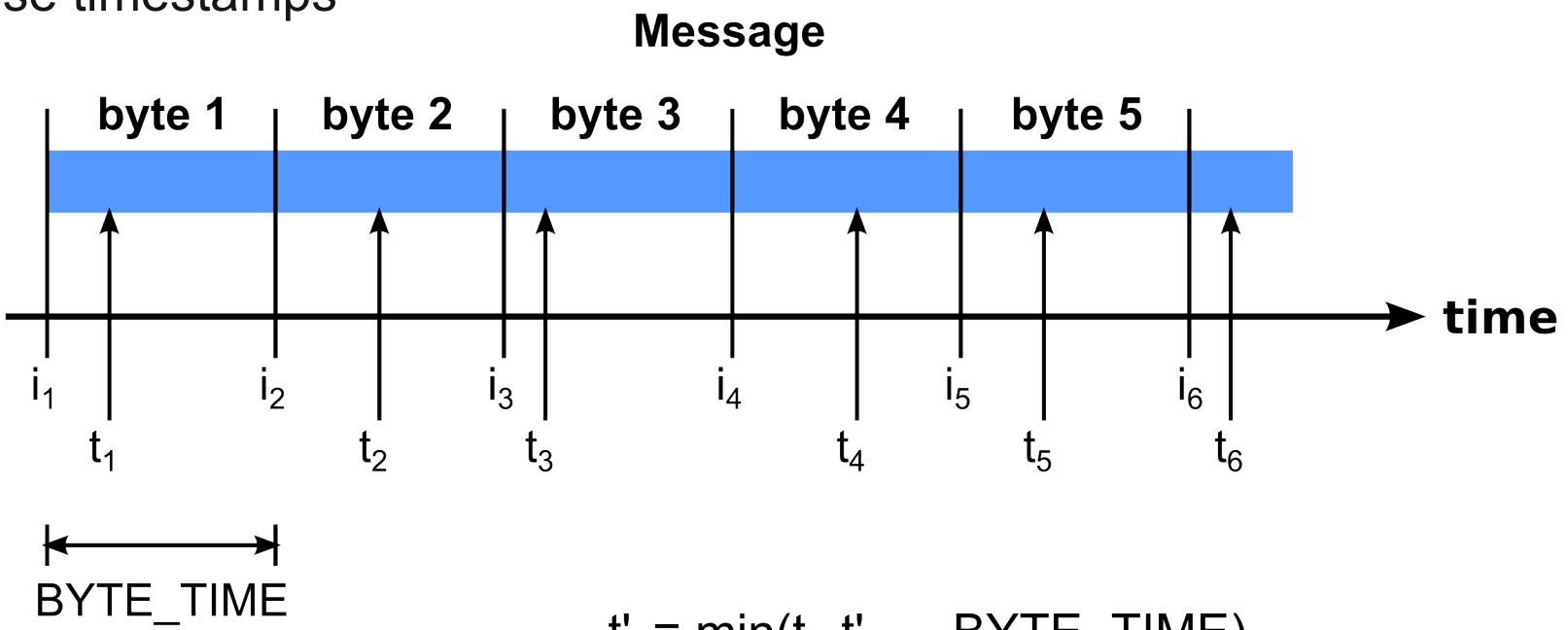


MAC Layer Timestamping (2)



Byte-wise Timestamping

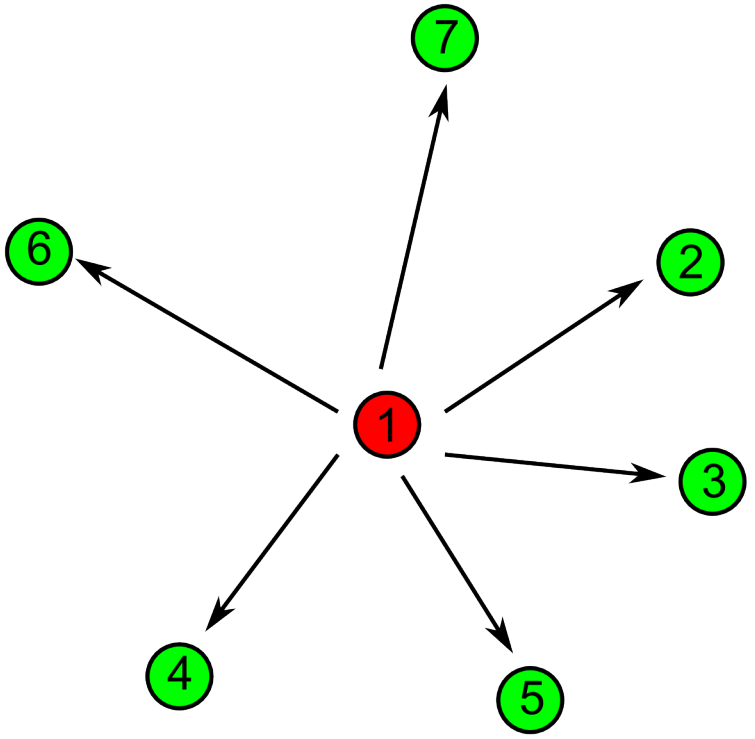
- Mica2 platfrom uses a byte-oriented radio chip. This means that on each byte transmission or receipt an interrupt is raised
- The timestamps of the first six bytes are used to estimate the arrival time of a packet → compensate the interrupt latency
- A single timestamp for this packet is then calculated by taking the average of these timestamps



$$t'_i = \min(t_i, t'_{i+1} - \text{BYTE\_TIME})$$
$$t'_6 = t_6$$

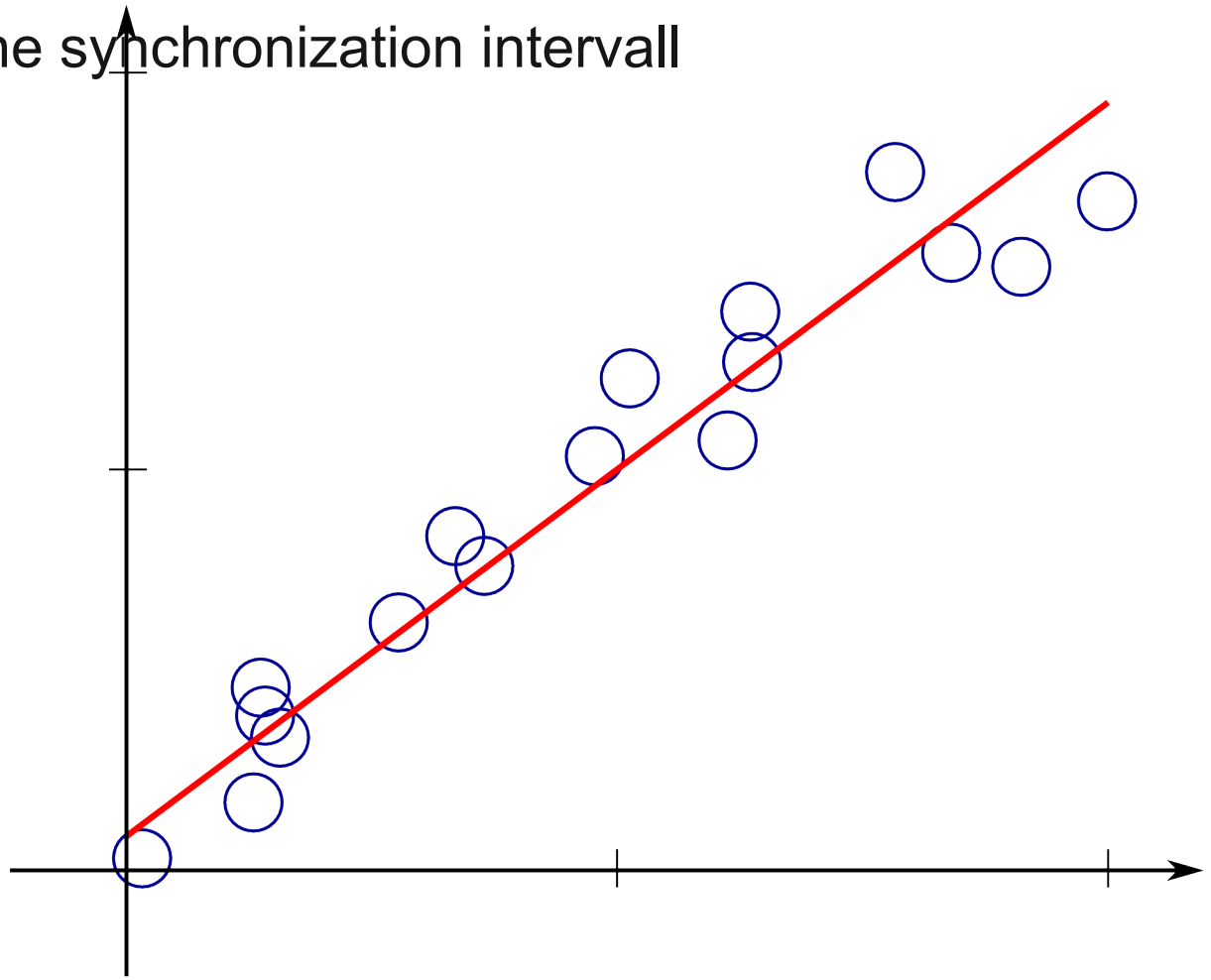
FTSP

- FTSP uses Broadcasting
- FTSP uses MAC Layer Timestamping with byte-wise timestamping to get rid of all uncertainties in radio message delivery



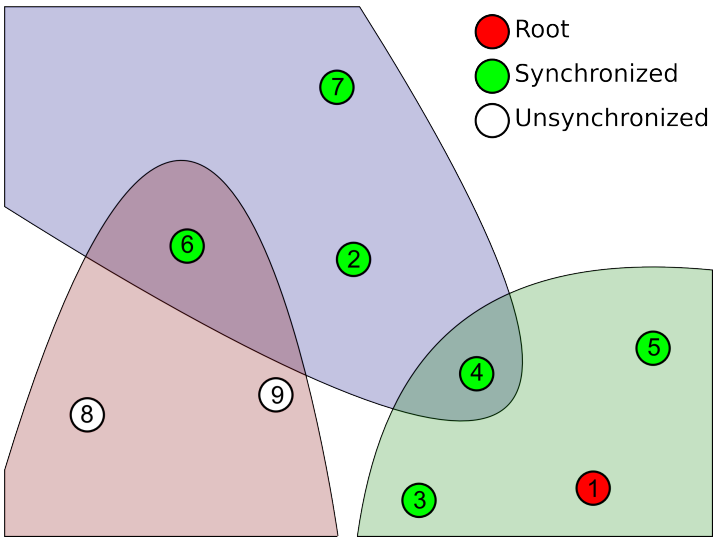
FTSP – Clock Drift Management

- The offsets between two clocks changes in a linear fashion
- This drift can be computed using linear regression
- Increase the synchronization intervall

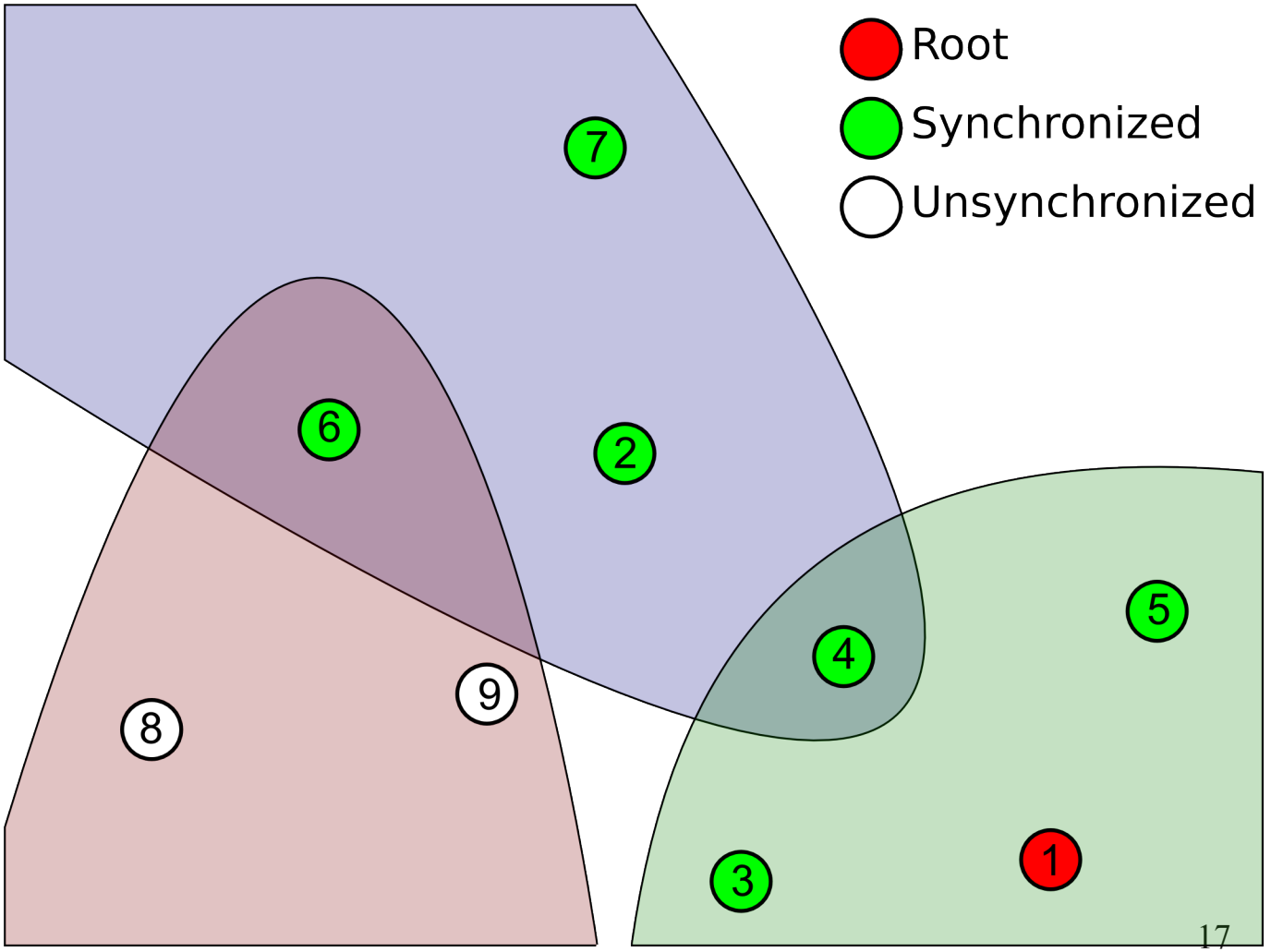


FTSP – Multi-hop time synchronization

- Node properties
  - Unique ID
  - Root ID (Root node where this node is synchronized to)
  - Highest sequenze number
  - Time (local clock)
  - Regeression table
- Message properties
  - Root ID
  - Sequenze number
  - Timestamp

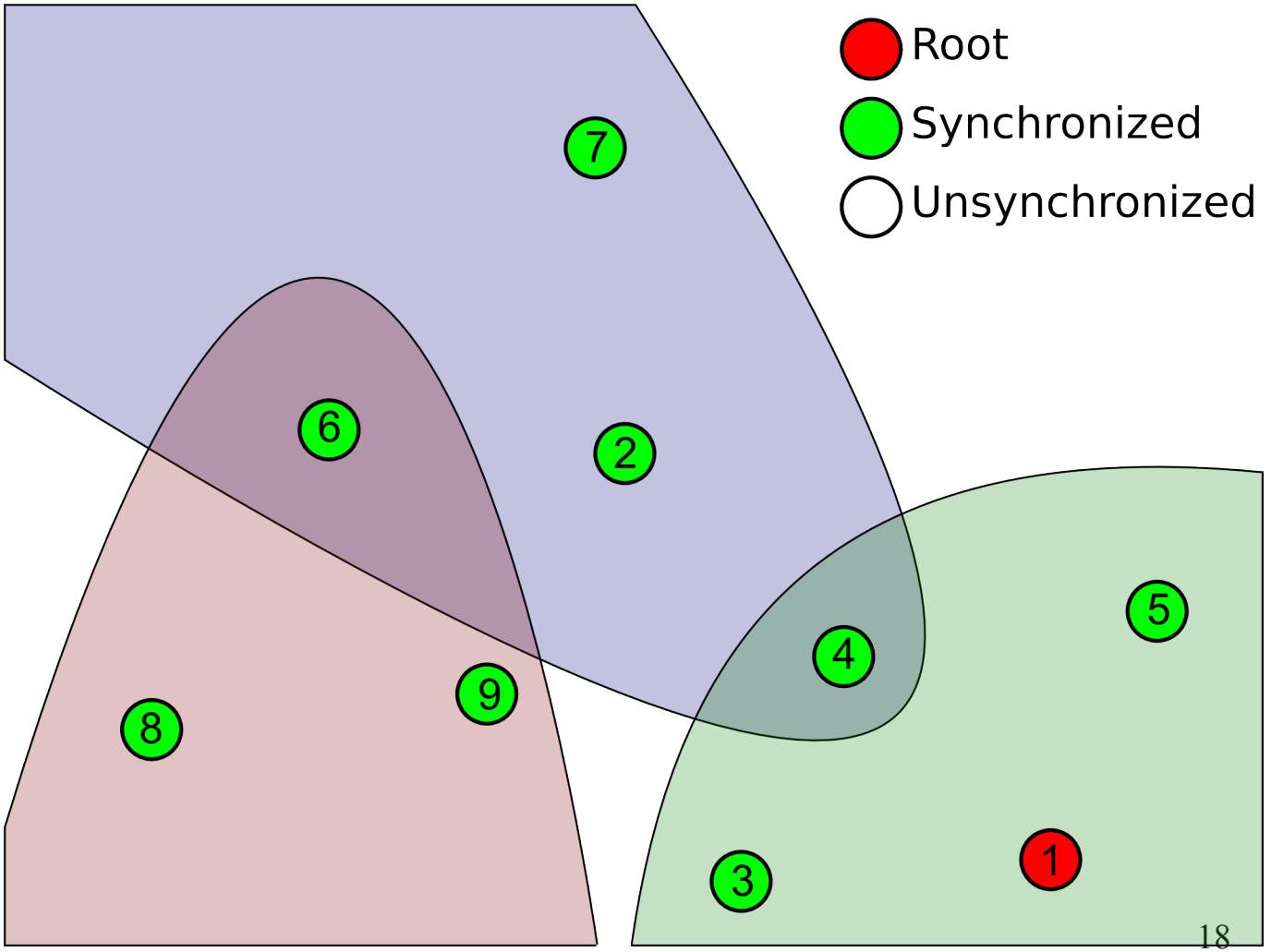


FTSP – Multi-hop time synchronization (1)



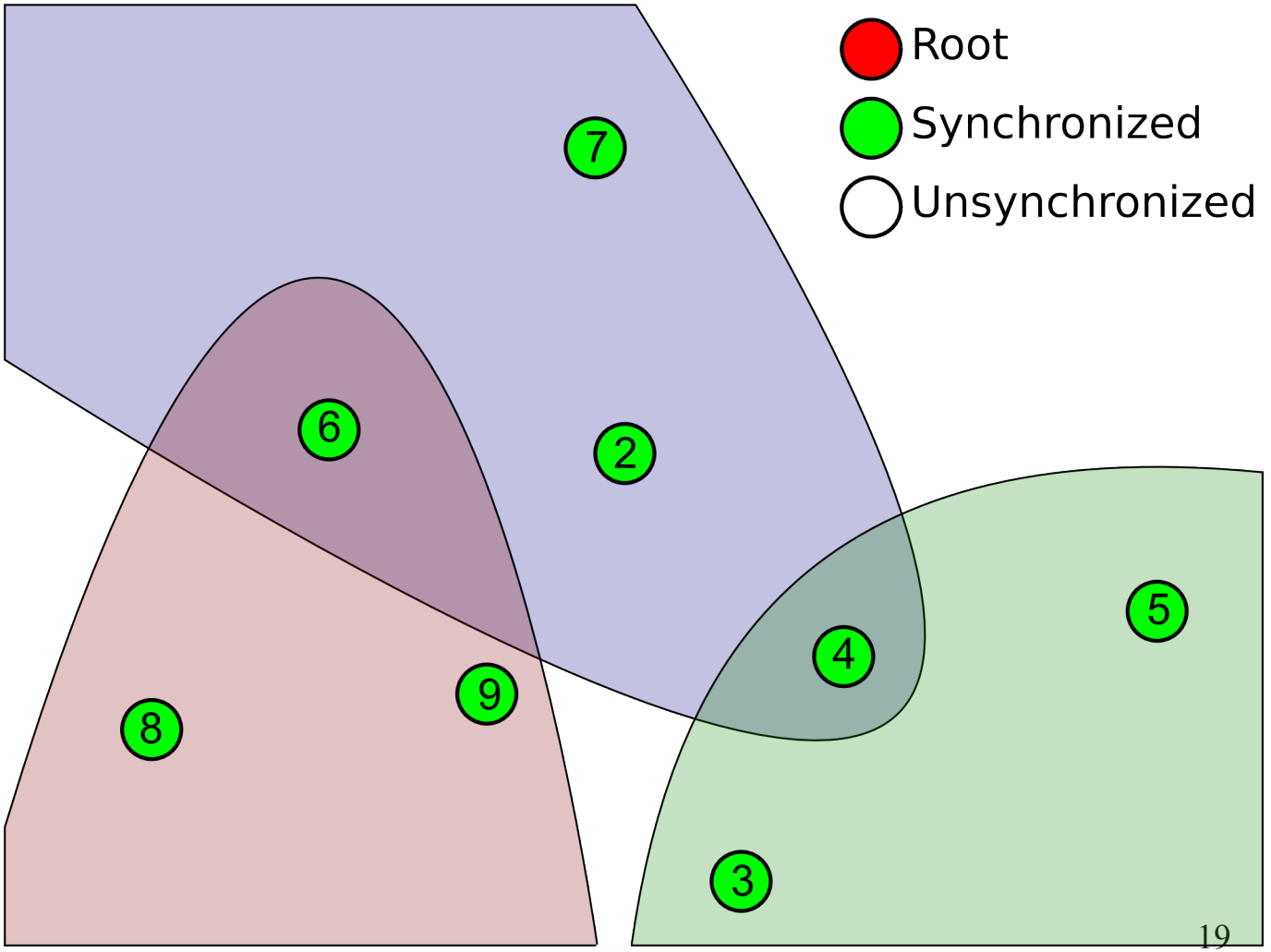
FTSP – Multi-hop time synchronization (2)

- {8,9} synchronized



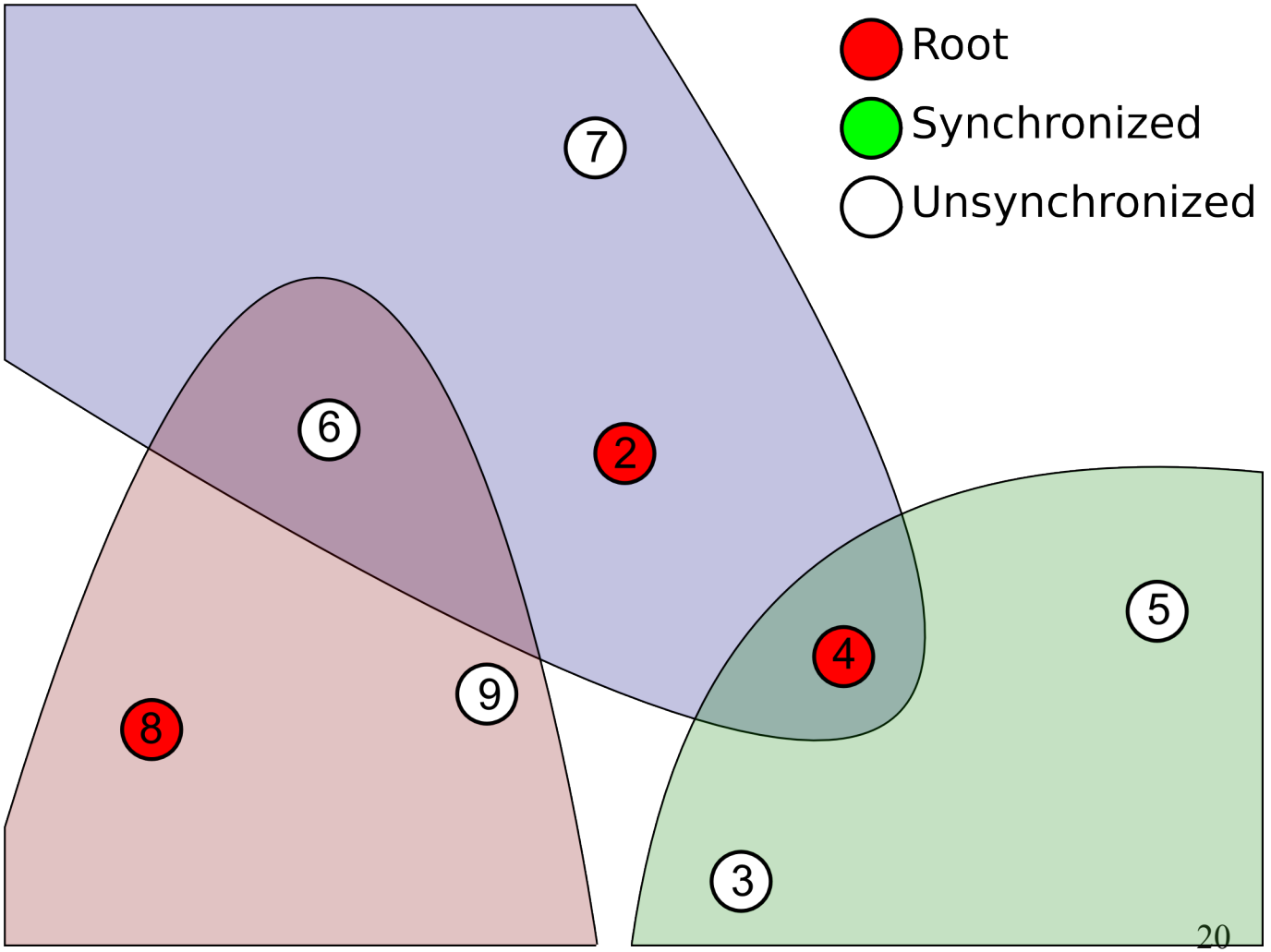
FTSP – Multi-hop time synchronization (3)

- {1} failed (root)



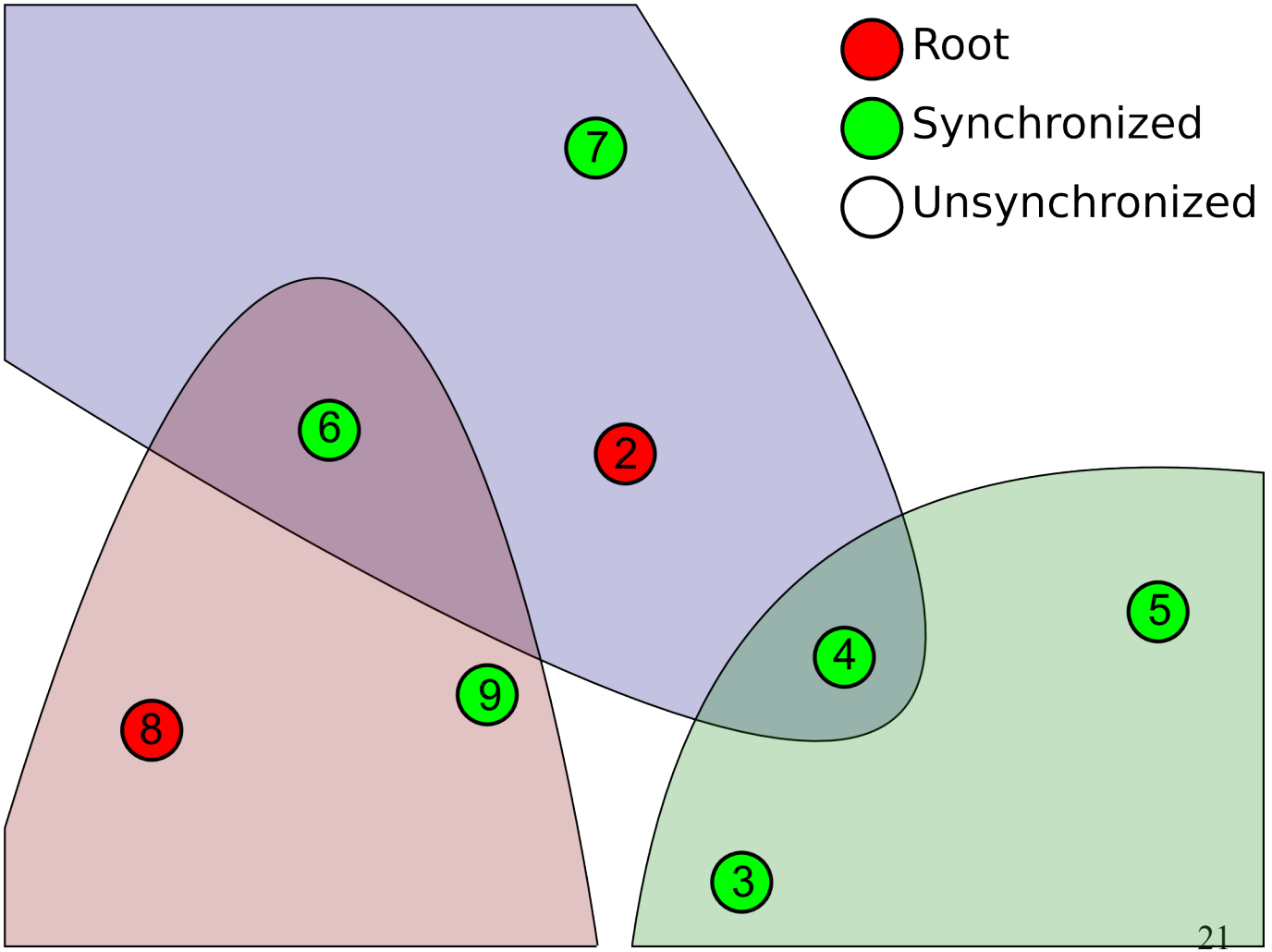
FTSP – Multi-hop time synchronization (4)

- {2},{4},{8} become root
- {3},{4},{6},{7},{8} out of sync



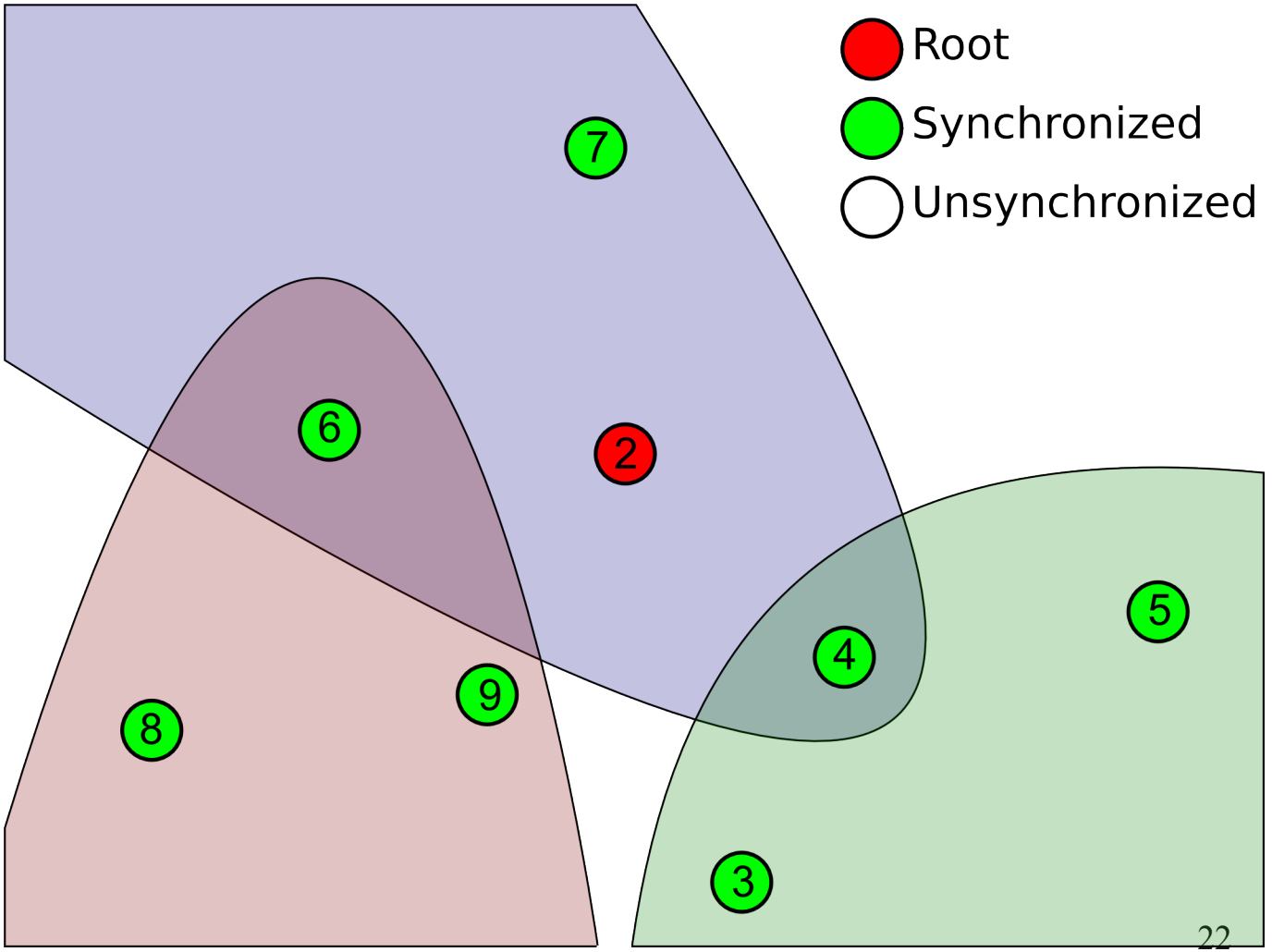
FTSP – Multi-hop time synchronization (5)

- {4} left root mode

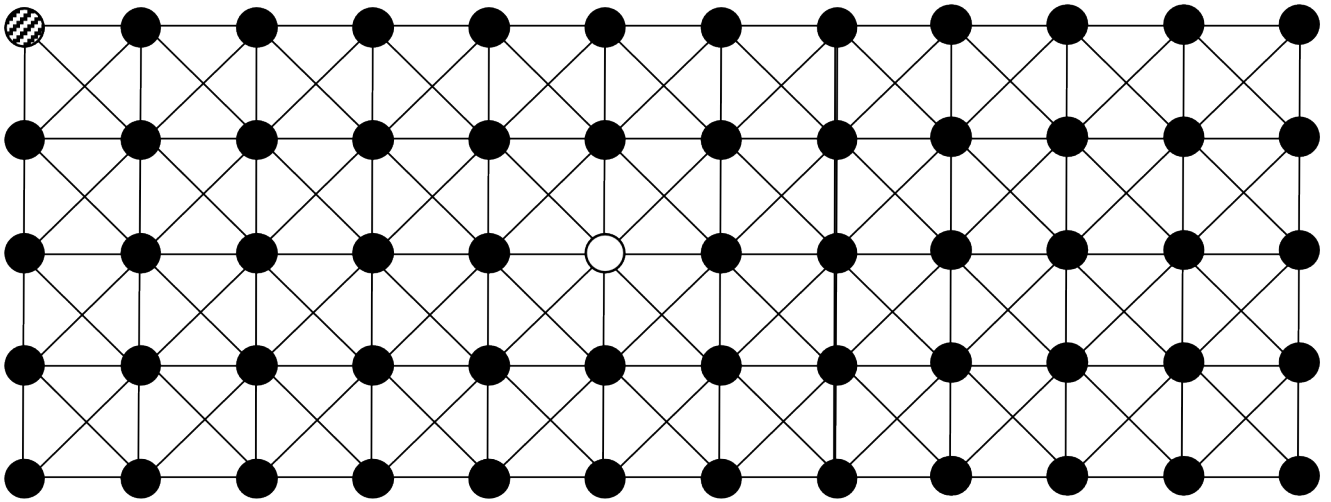


FTSP – Multi-hop time synchronization (6)

- {8} left root mode



FTSP – Experiment Layout

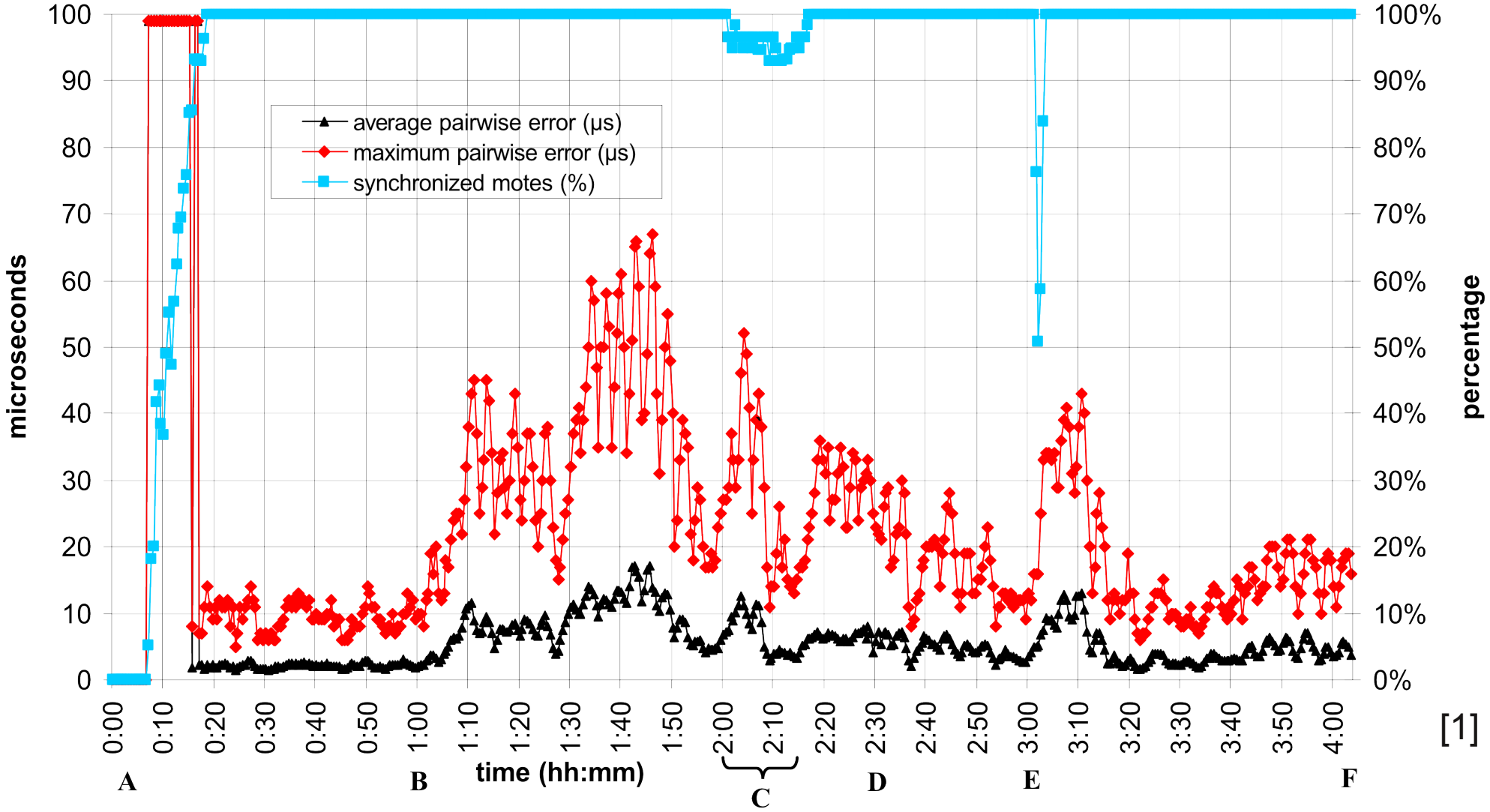


[1]

○ ID_1 – first leader ◍ ID_2 – second leader

The layout and links of the experimental setup. Each node can Only communicate with its (at most 8) neighbours.

FTSP – Experiment Result



[1]

A) turn on all motes
B) switch off root with ID1
C) randomly select nodes and reset one by one with 30 second period
D) turn off motes with odd IDs
E) turn on motes with odd IDs

Questions?

References

- [1] The Flooding Time Synchronization Protocol (Maróti, Kusy, Simon, Lédeczi) [Paper]
- [2] The Flooding Time Synchronization Protocol (Maróti, Kusy, Simon, Lédeczi) [Presentation]
- [3] Time Synchronization for Networked Embedded Systems (Kay Römer)
- [4] ETH Sensor Network Museum (http://www.snm.ethz.ch)

Accuracy

- Application dependent
- High (Microseconds)
  - Goldengate Bridge (Vibrations)
  - Earthquake monitoring
- Medium (Seconds)
  - Minefield
- Low (Minutes)
  - Glacier