Low-Power System Design

227-0781-00L

Fall Semester 2019

Jan Beutel

Plan for Today

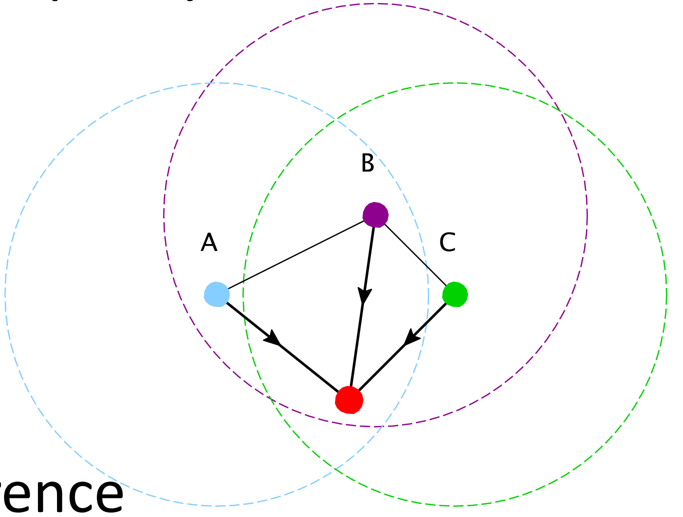
- MAC Layer Techniques
  - Constructive interference: A-MAC
  - Network Flooding: GLOSSY
  - MAC Layer Timestamping
- Recent Research
  - Evaluating Concurrent Transmissions
- Network Time Synchronization
  - Basics, Fundamental Effects
  - Protocol Examples
- Slides contain material from P. Dutta, F. Oesterlind, F. Ferrari, A. Schaper and R. Wattenhofer

Low-Power System Design

MAC LAYER TECHNIQUES – CONSTRUCTIVE INTERFERENCE

Wireless Interference

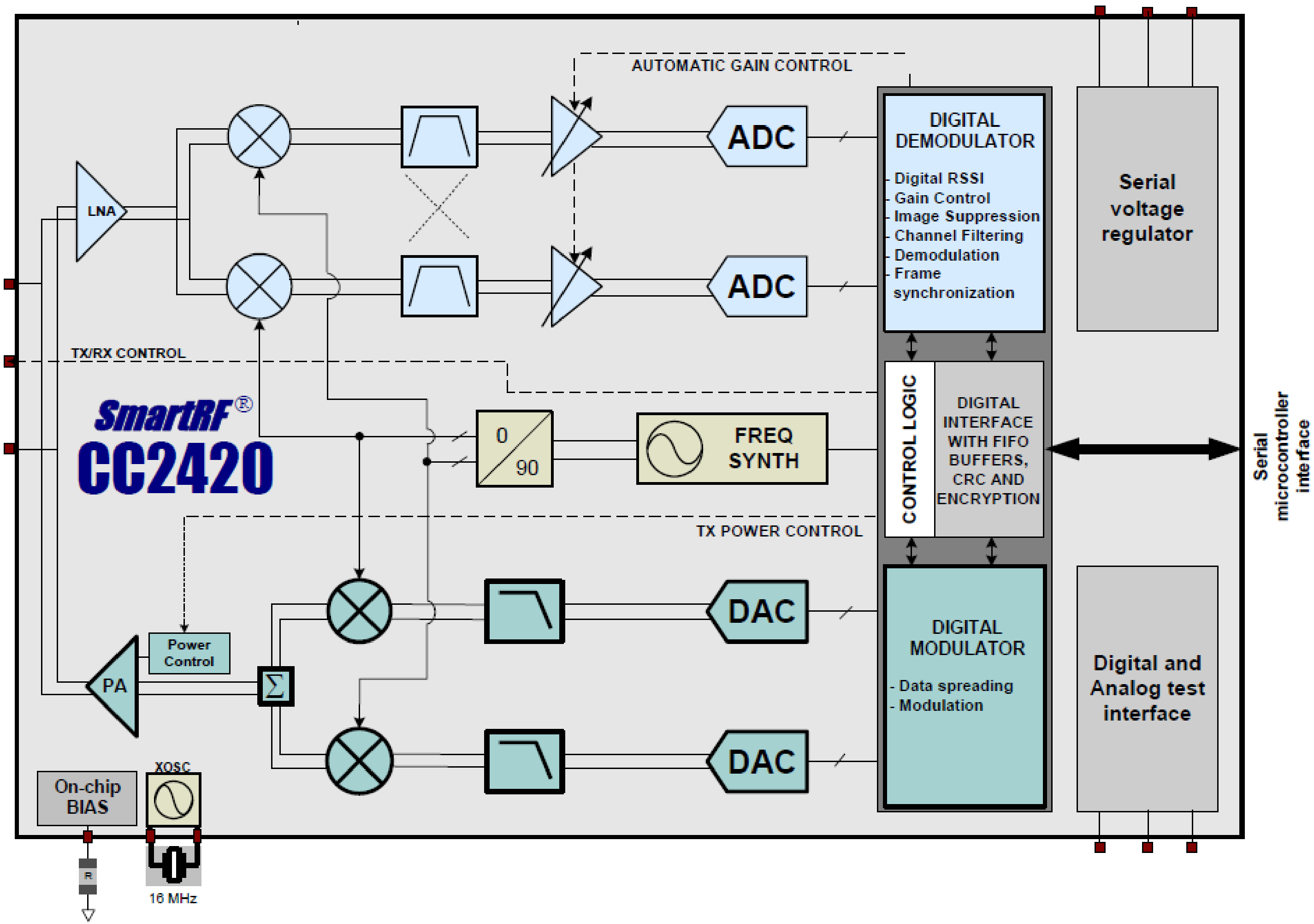
- Spatially close wireless stations transmit signals at the same time and with the same frequency



Stations A, B, and C transmit signals to a common receiver R

- Destructive interference
  - Interference generally reduces the probability that a receiver correctly detects the information
- Constructive interference
  - A receiver detects with high probability the superposition of the signals generated by multiple transmitters

IEEE 802.15.4 Uses DSSS Modulation

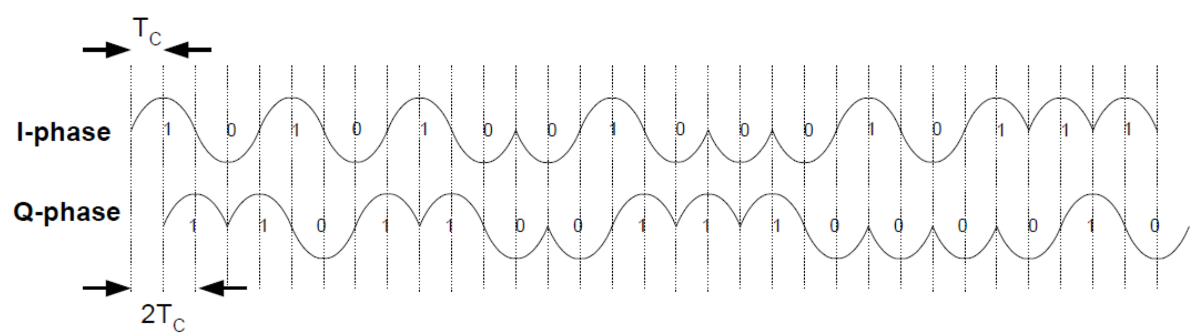


[Texas Instruments]

IEEE 802.15.4 Modulation

- 1 Byte is divided into 2x 4-Bit Symbols
- Each Symbol is mapped to a pseudo-random noise (PN) sequence with 32 chips (2 MChips/sec)
- Offset-Quadrature Phase Shift Keying (O-QPSK) with half-sine chip shaping (equivalent to MSK modulation)

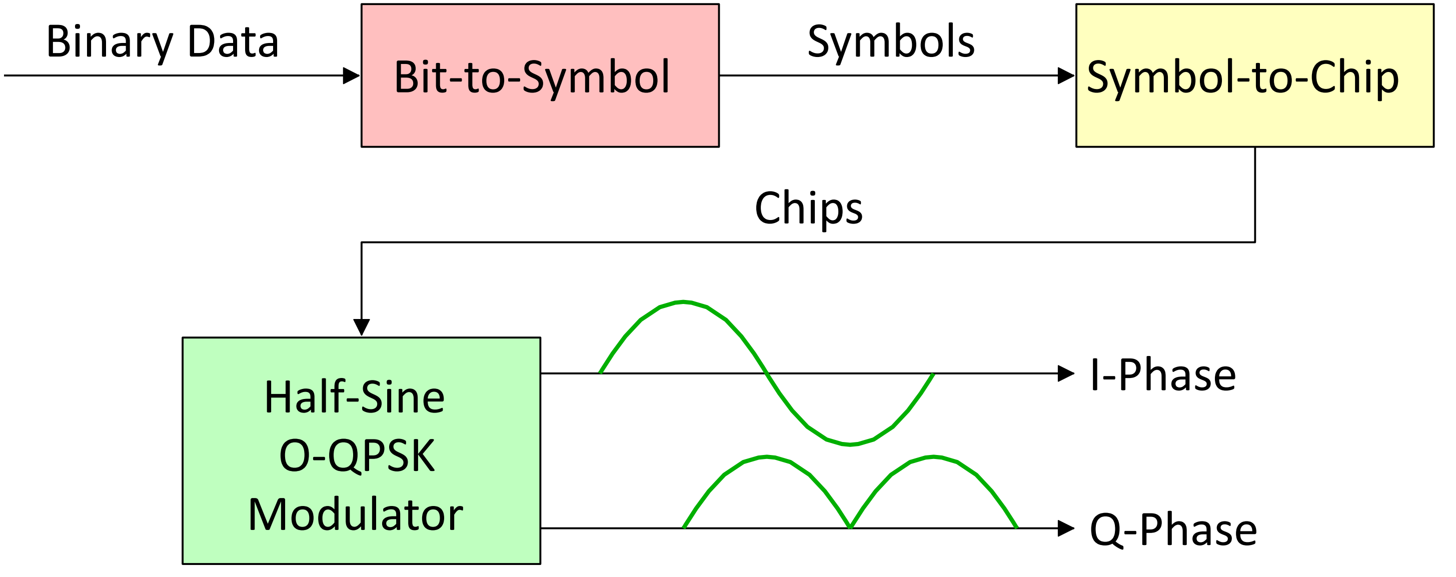
Symbol	Chip sequence ($C_0, C_1, C_2, \ldots, C_{31}$)
0	1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0
1	1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0
2	0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0
3	0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1
4	0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1
5	0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0
6	1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1
7	1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1
8	1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1
9	1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1
10	0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1
11	0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0
12	0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0
13	0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1
14	1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0
15	1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0



- PN sequences introduce randomization and redundancy

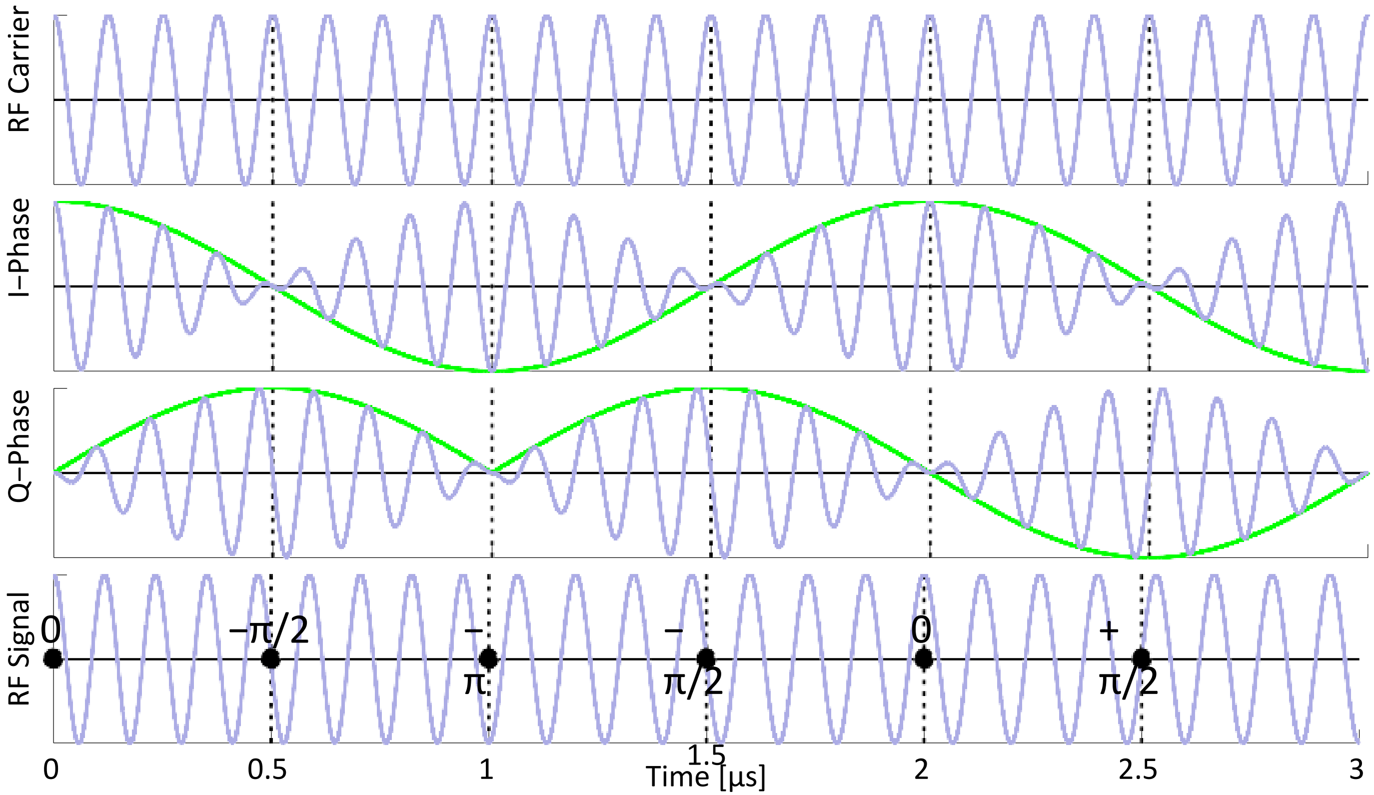
IEEE 802.15.4 Modulation Scheme

- IEEE 802.15.4: standard for 2,450 MHz wireless radios
- A 3-step process converts binary data to a baseband signal



- In-phase and quadrature-phase components of the baseband signal determine the phase of the transmitted RF signal

Half-Sine O-QPSK Modulation: Example

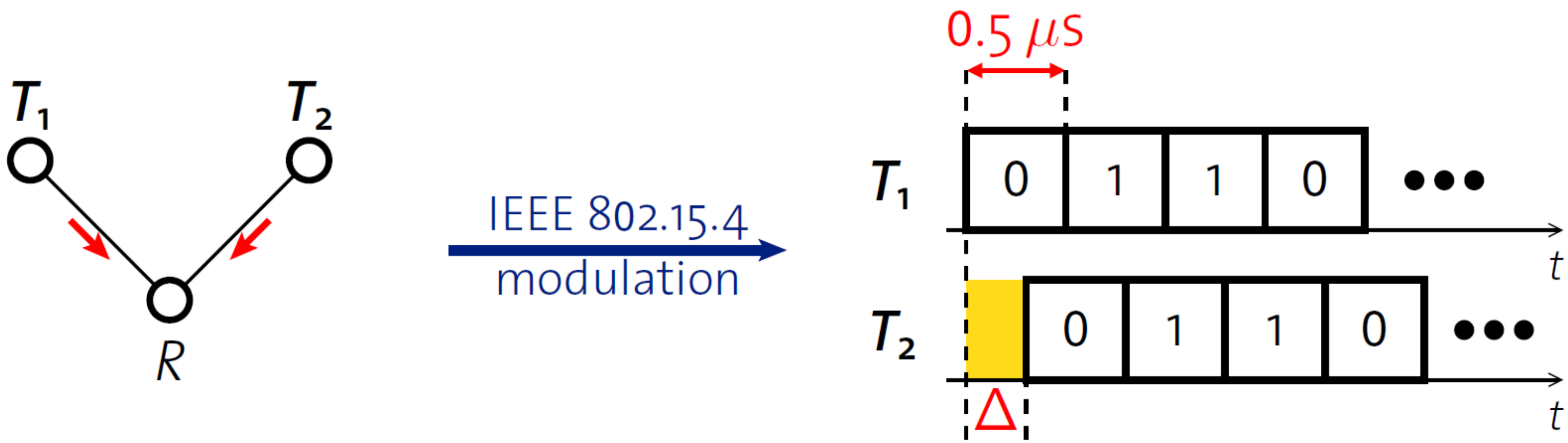


- Data rate: $1/T_C$ chip/s = 2 Mchip/s = 62.5 ksymbol/s = 250 kbps
- The information carried by each chip generates a complete phase change of the RF signal every 0.5 µs

Synchronous Transmissions

- Multiple nodes transmit **same packet** at **same time**

Ferrari, F. and Zimmerling, M. and Thiele, L. and Saukh, O. (2011). Efficient Network Flooding and Time Synchronization with Glossy. In *10th International Conference on Information Processing in Sensor Networks (IPSN 2011)* (pp. 73–84).

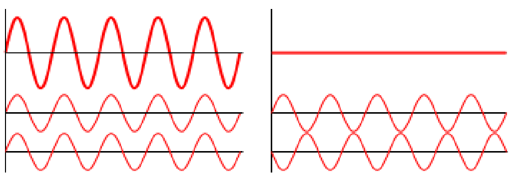


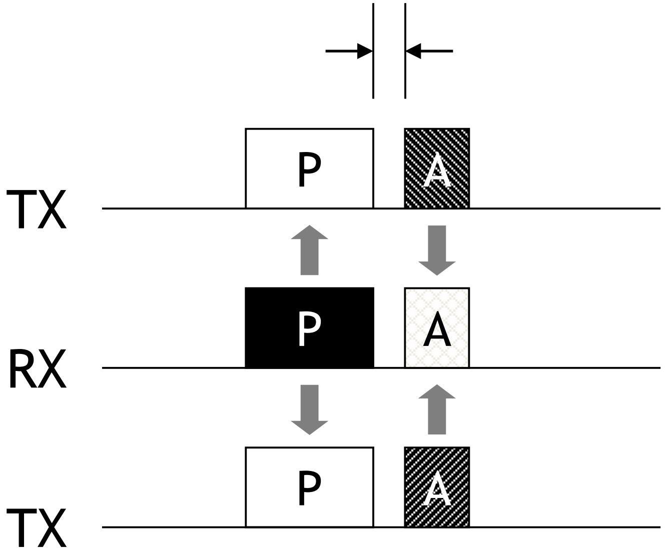
- R receives packet with high probability if $\Delta \leq 0.5$ μs

- Property exploited also in A-MAC [Dutta et al., SenSys ’10]

Synchronized Transmission with Backcast

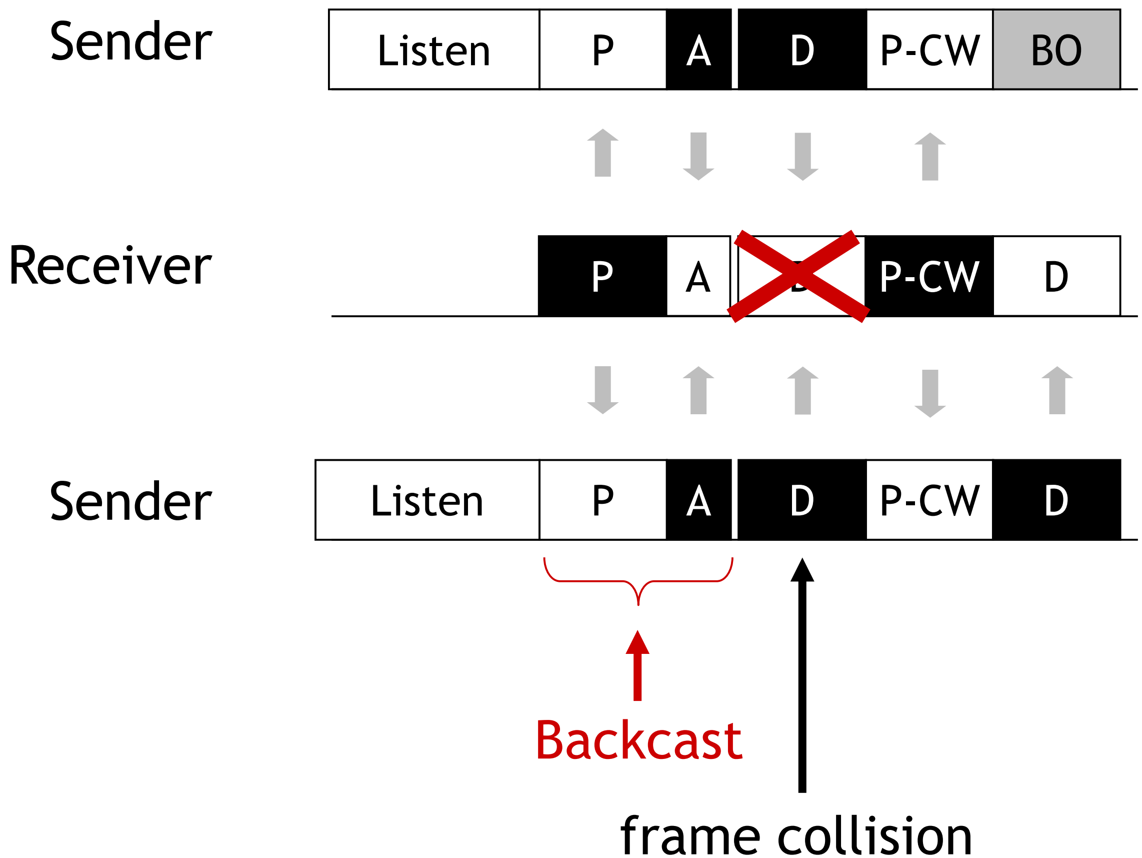
- A link-layer frame exchange in which:
  - A single radio PROBE frame transmission
  - Triggers zero or more *identical* ACK frames
  - Transmitted with tight timing tolerance
  - So there is minimum inter-symbol interference
  - And ACKs collide non-destructively at the receiver





P. Dutta, R. Musaloiu-E., I. Stoica, A. Terzis, "Wireless ACK Collisions Not Considered Harmful", HotNets-VII, October, 2008, Alberta, BC, Canada

A-MAC's Contention Mechanism



Networked Embedded Systems

ALL-TO-ALL NETWORK FLOODING: GLOSSY

Increasing Reliability: Glossy Floods

- Main objectives
  - Fast and reliable flooding of messages
  - Accurate global time synchronization
  - Hide complexity of multi-hop networks

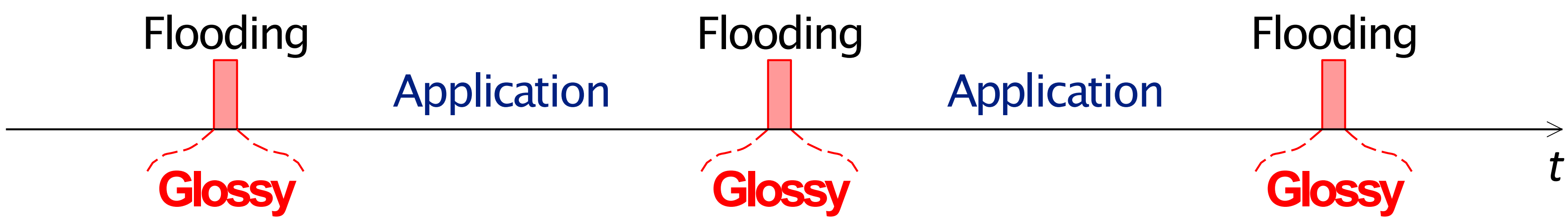
- Challenge in multi-hop wireless networks
  - Uncoordinated transmissions, packet loss, retransmission delays

- Glossy: Flooding architecture for wireless sensor networks
  - Fastest possible propagation, by design
  - Highly reliable (> 99.99 %)
  - Requires no network state information
  - Efficient also in dense networks
  - Time synchronization at no additional cost

Ferrari, F. and Zimmerling, M. and Thiele, L. and Saukh, O. (2011). Efficient Network Flooding and Time Synchronization with Glossy. In *10th International Conference on Information Processing in Sensor Networks (IPSN 2011)* (pp. 73–84).

Glossy: Key Techniques

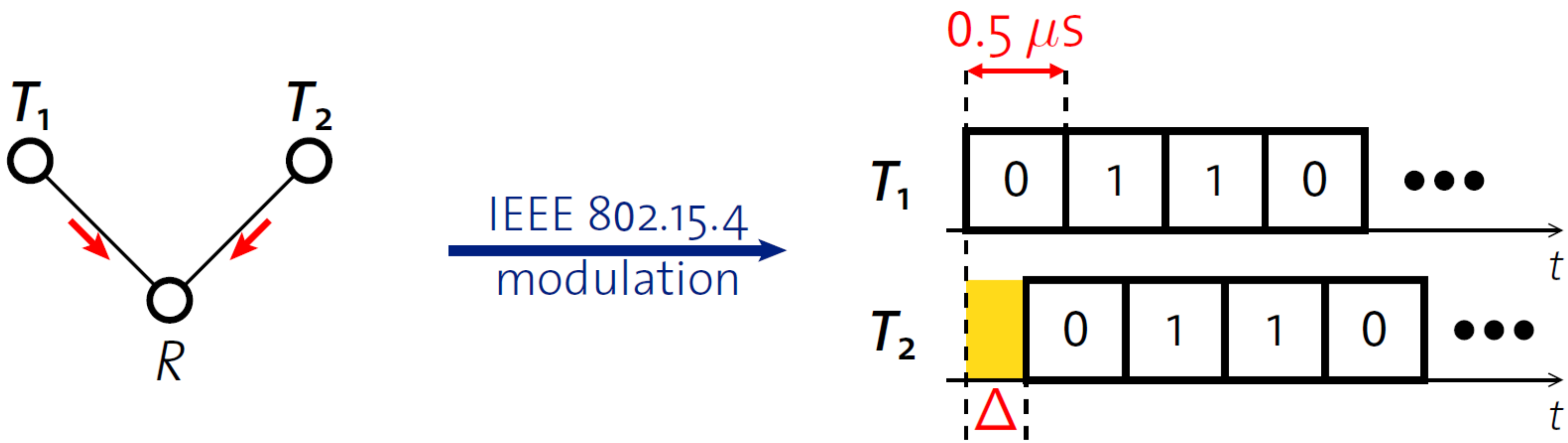
- **Temporally decouple** network flooding from application tasks



- Exploit **synchronous transmissions** for fast network flooding

Synchronous Transmissions

- Multiple nodes transmit **same packet** at **same time**



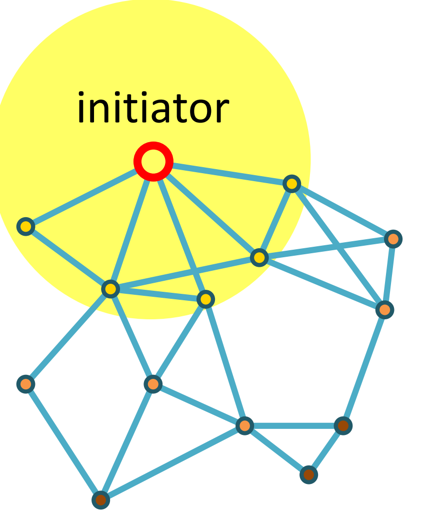
- R receives packet with high probability if Δ ≤ 0.5 µs
- Property exploited also in A-MAC [Dutta et al., SenSys '10]

Challenges for Efficient Flooding

How to relay packets **efficiently** and **reliably**?

- Avoid aggressive, uncoordinated broadcasts

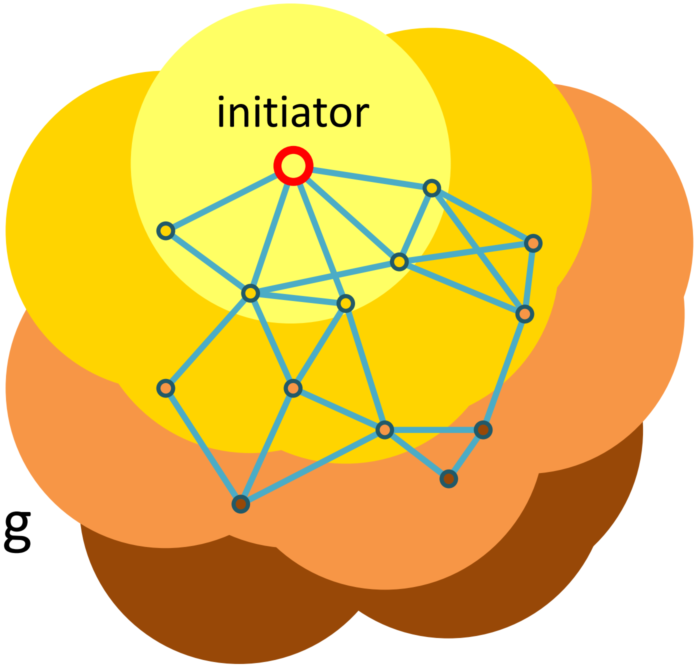
- Typical approach:
  Coordinate packet transmissions
  - CF [Zhu et al., NSDI 2010]
  - RBP [Stann et al., SenSys 2006]
  - Maintain topology-dependent state



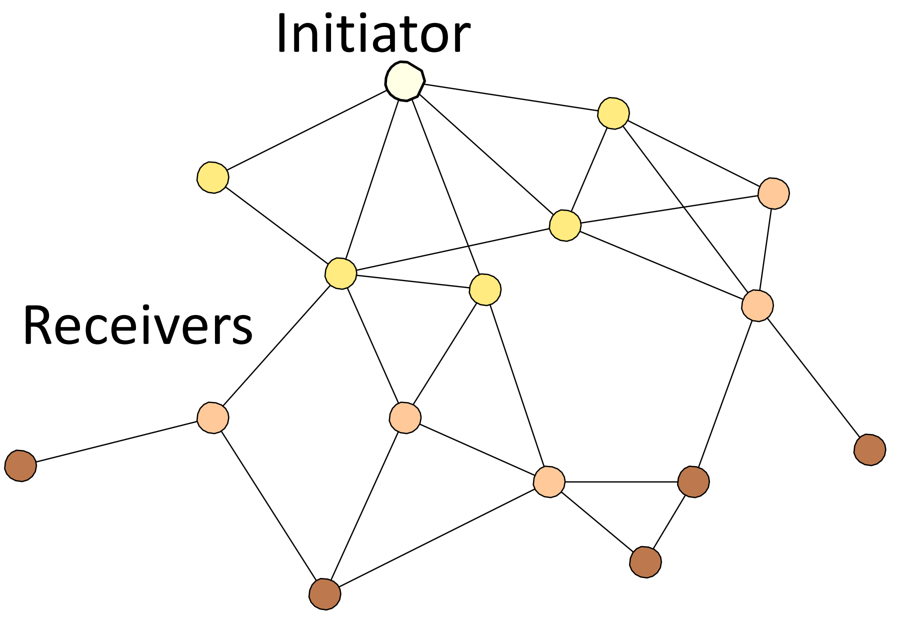
Glossy Flooding Architecture

- **All** receiving nodes relay packets **synchronously**
  - Simple, but radically different solution
  - No explicit routing
  - No topology-dependent state

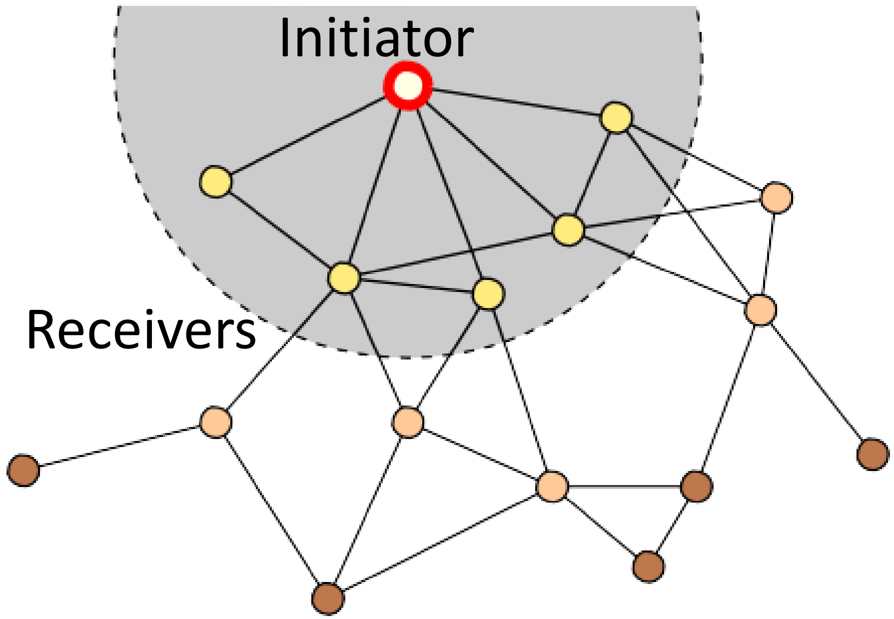
- Key Glossy mechanisms
  - Start execution at the same time
  - Compensate for hardware variations
  - Ensure deterministic execution timing



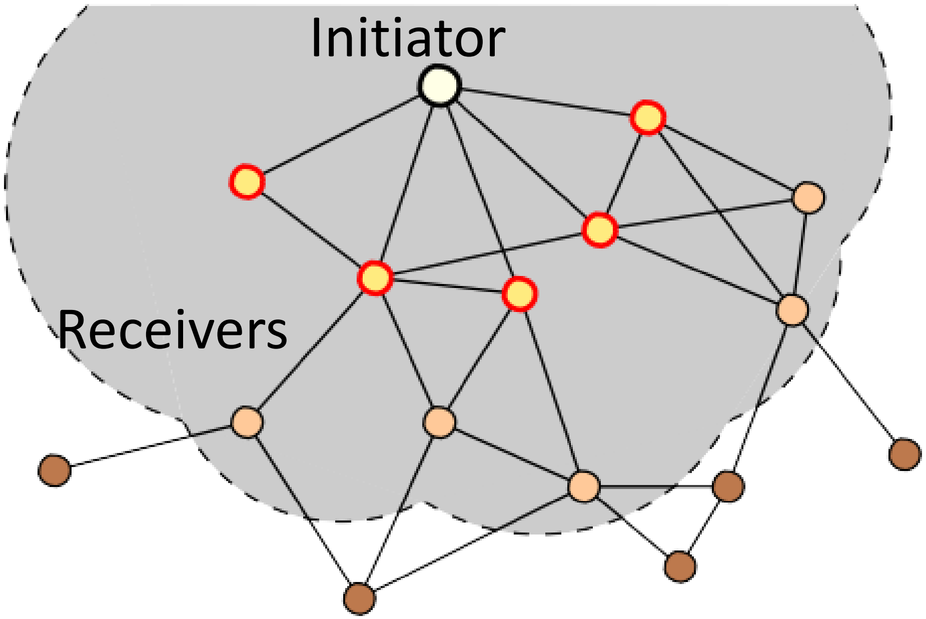
Glossy Example Flood Propagation



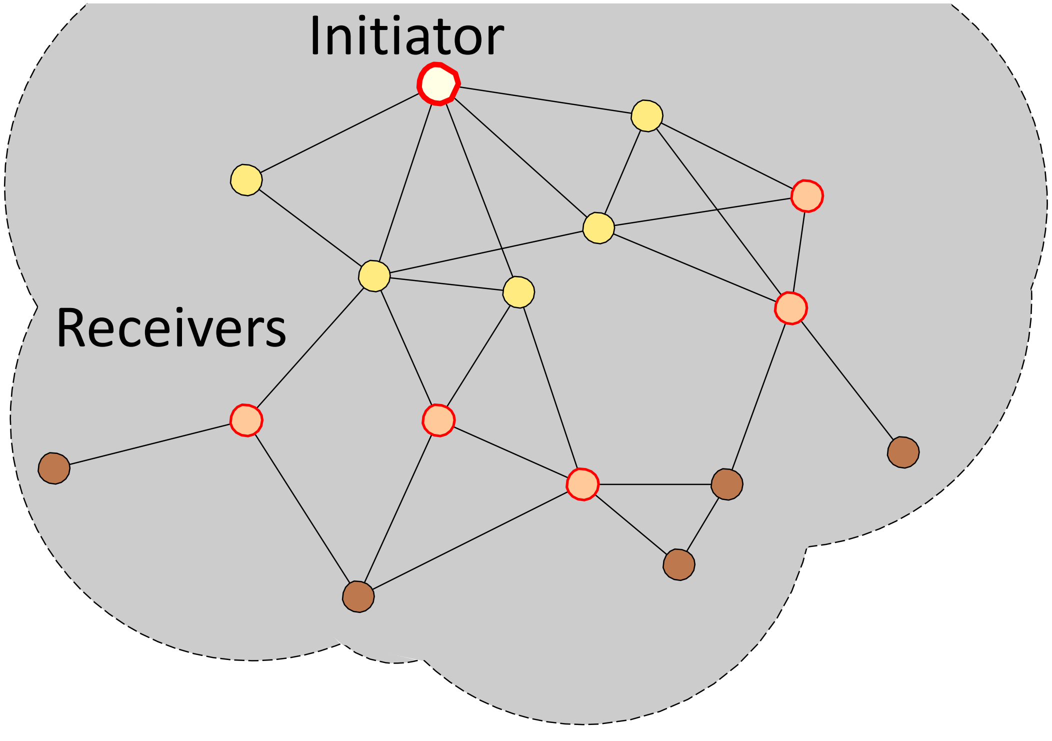
Glossy Example Flood Propagation



Glossy Example Flood Propagation



Glossy Example Flood Propagation

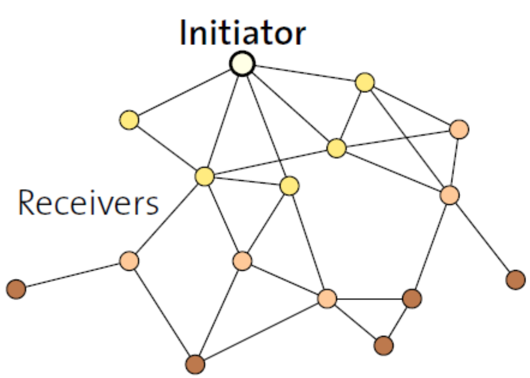


Glossy Fast Packet Propagation Details

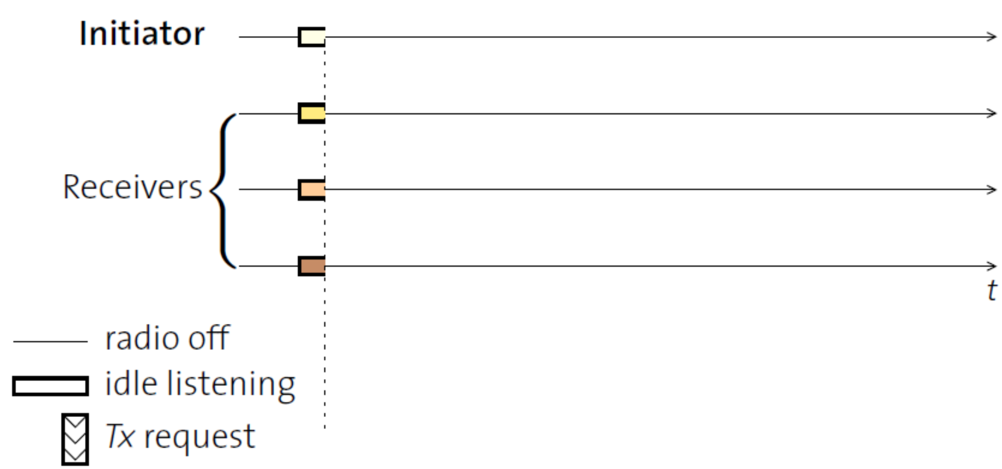
When Glossy starts:

- Turn on radio

Example



Timeline

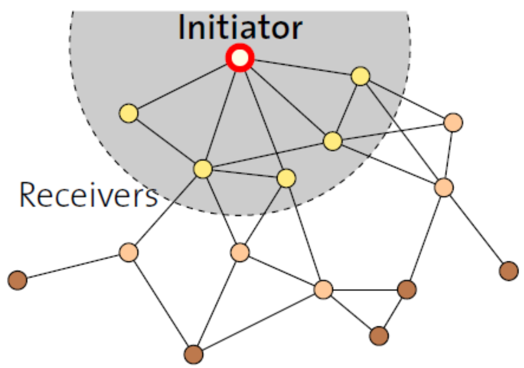


Glossy Fast Packet Propagation Details

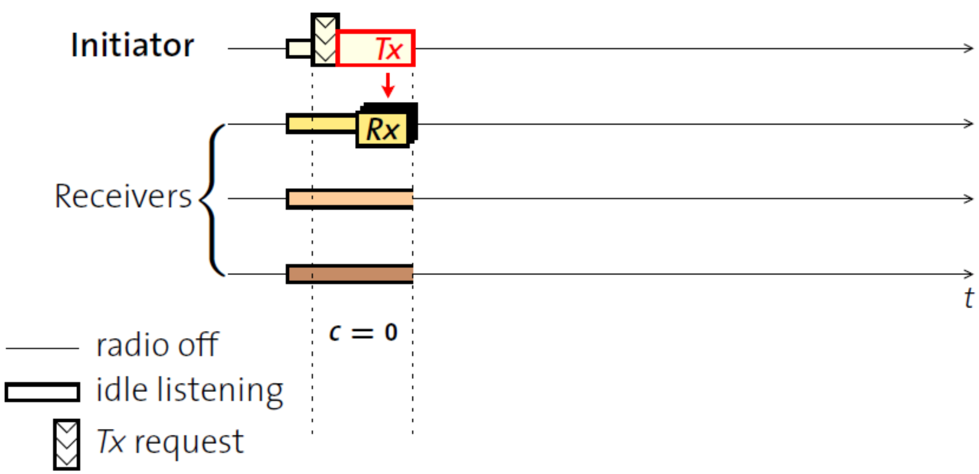
Initiator:

- Set relay counter $c = 0$
- Transmit packet

Example



Timeline

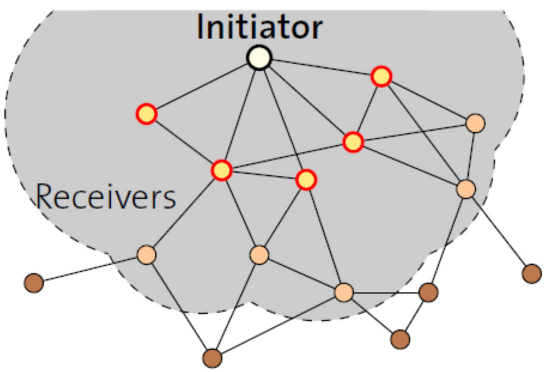


Glossy Fast Packet Propagation Details

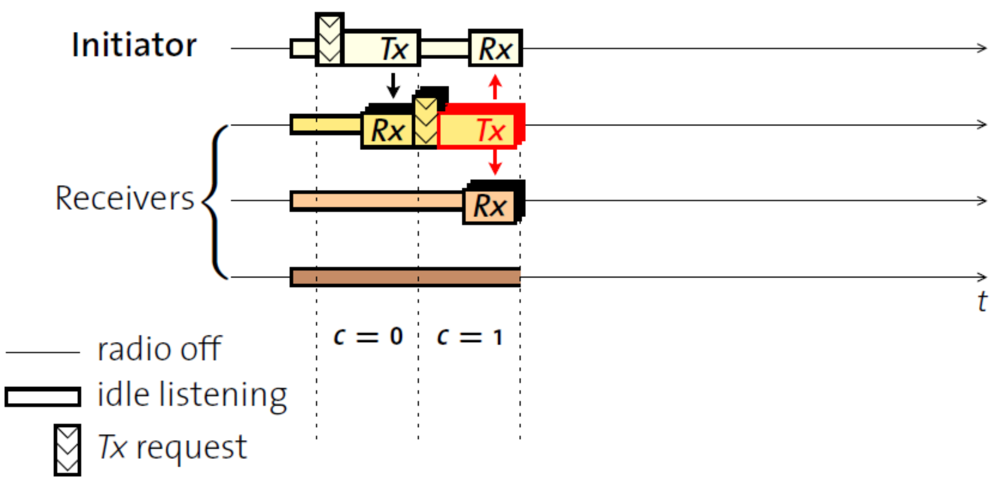
At packet reception:

- Increment relay counter *c*
- Transmit synchronously

Example



Timeline

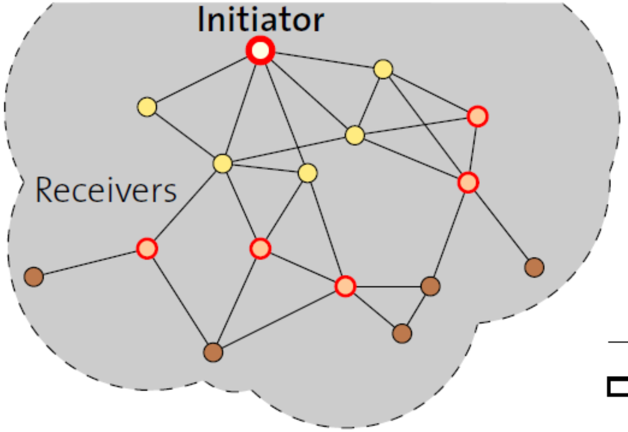


Glossy Fast Packet Propagation Details

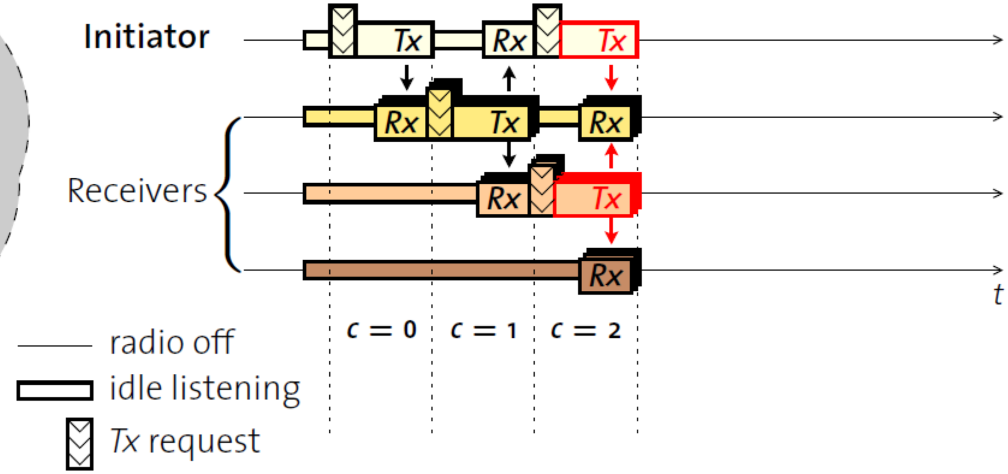
At packet reception:

- Increment relay counter *c*
- Transmit synchronously

Example



Timeline

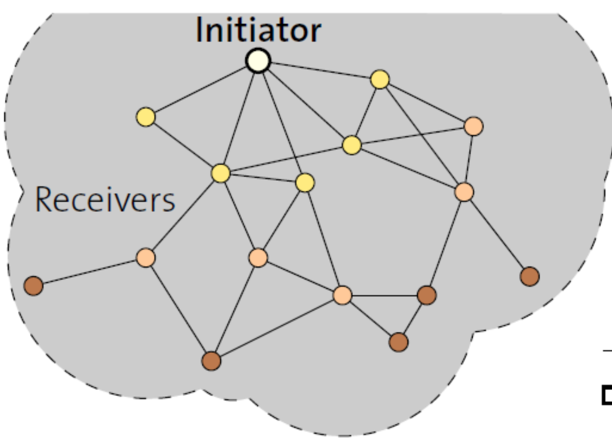


Glossy Fast Packet Propagation Details

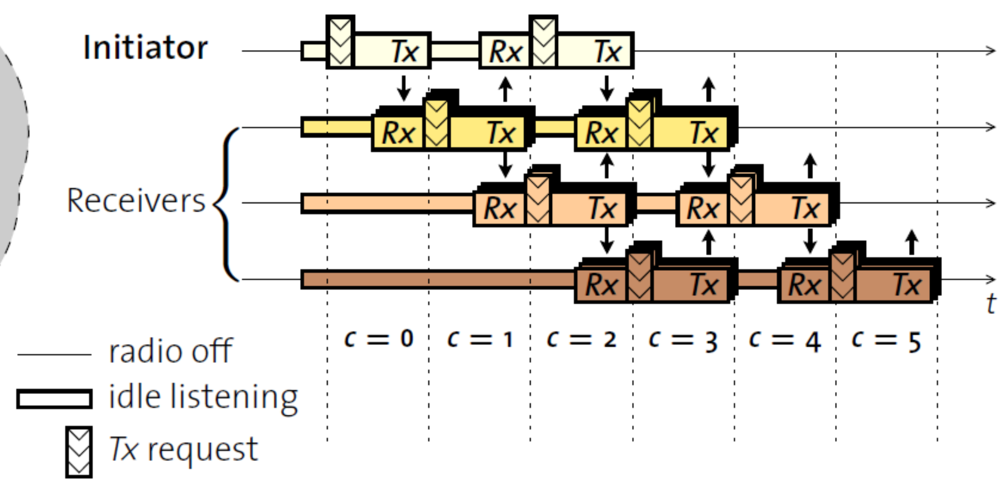
Stop and turn off radio when:

- Already transmitted *N* times

Example



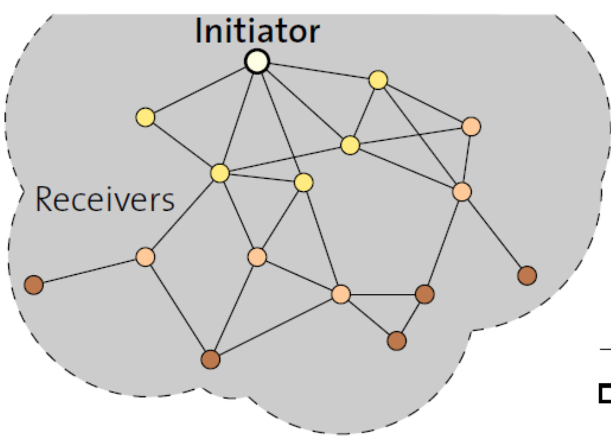
Timeline ($N = 2$)



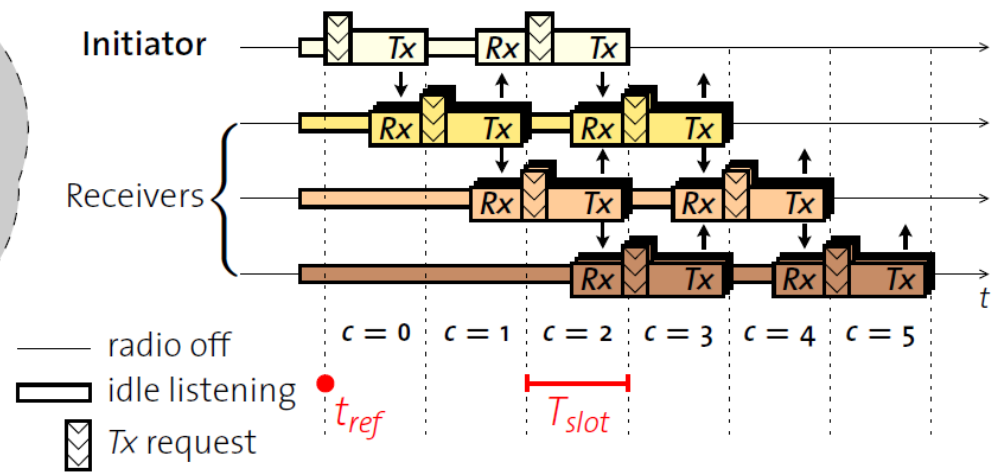
Glossy Fast Packet Propagation Details

- T_{slot} is constant by design
- Local estimates of T_{slot} / Received relay counter c } Reference time t_{ref}
- t_{ref} provides synchronized time

Example

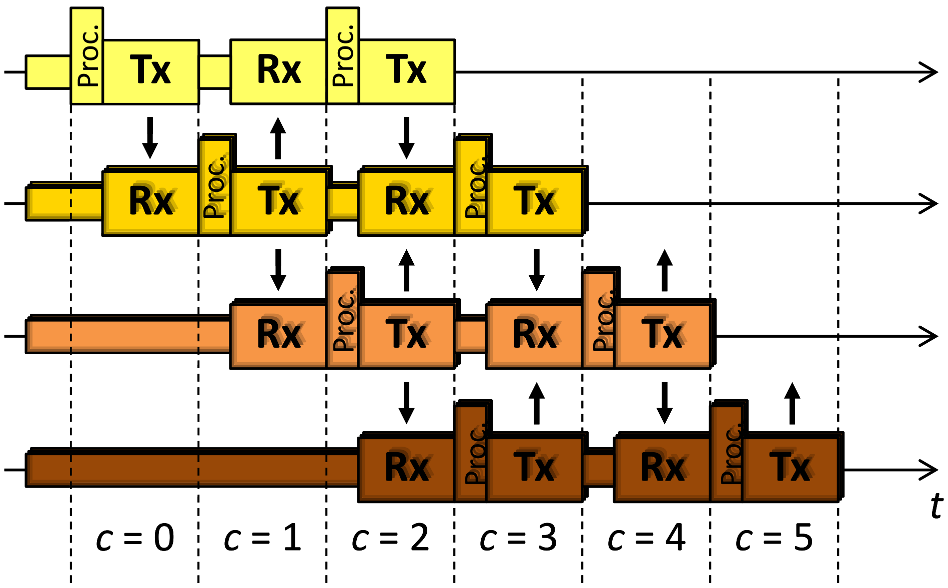


Timeline ($N = 2$)

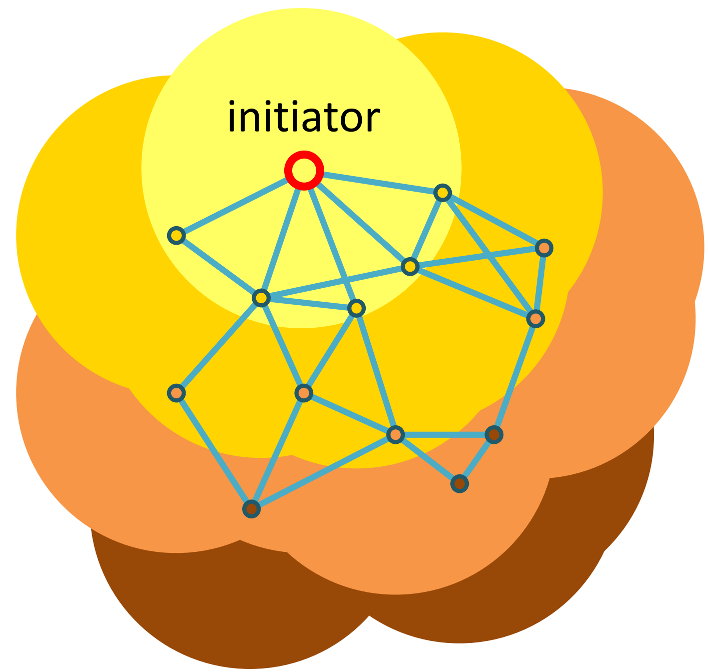


Propagation in Glossy

- A **relay counter** *c* is set to 0 at the first transmission
- A node increments *c* before relaying the packet

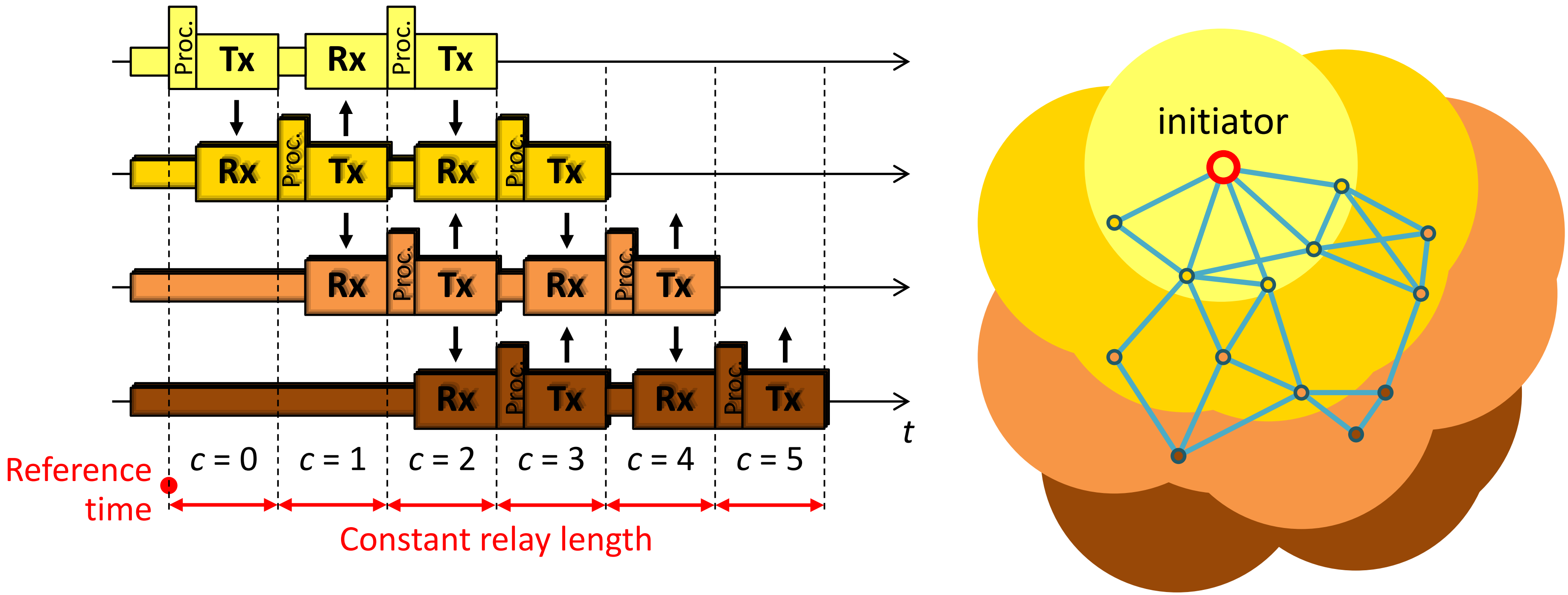


(In this example a node transmits at most twice)



Time synchronization in Glossy

- Estimate the **relay length** during propagation
- Compute a common **reference time**

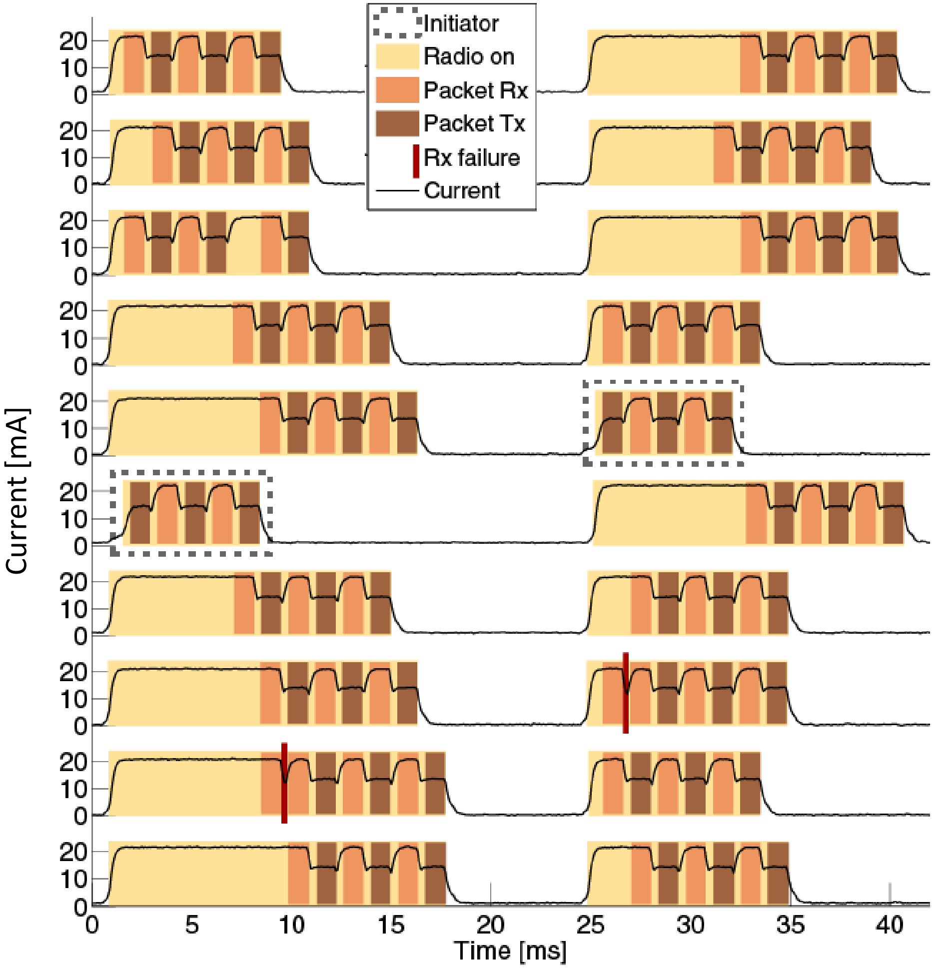


Glossy: Main Evaluation Findings

- **A few ms** to flood packets to hundreds of nodes
- Reliability **> 99.99 %** in most scenarios
- Synchronization error **< 1 μs** even after 8 hops

Evaluation of Glossy on FlockLab

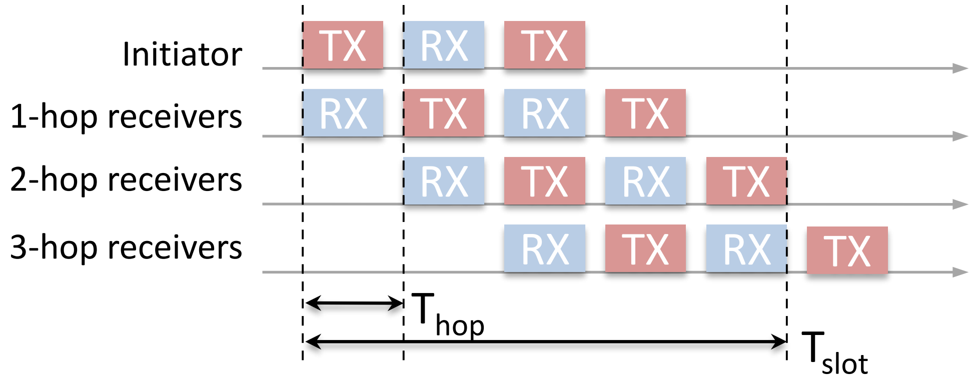
- Multi-modal monitoring at network scale
- Flooding protocol (Glossy)
  - Packet transmissions overlap
- Power
  - Find current consumption for each state
  - Expected behavior?
- Activity
  - Packet exchange

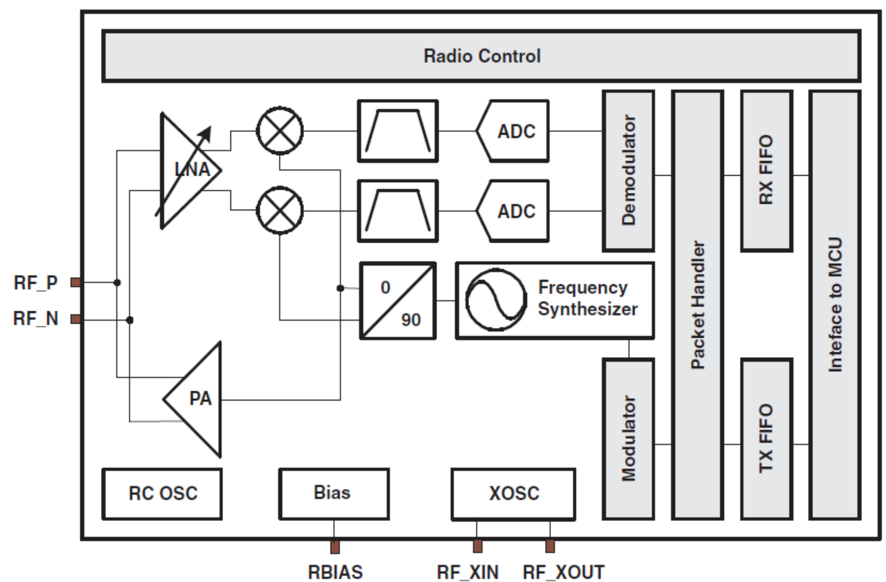


Critical Technique

MAC LAYER TIMESTAMPING

Details – MAC Layer Timestamping

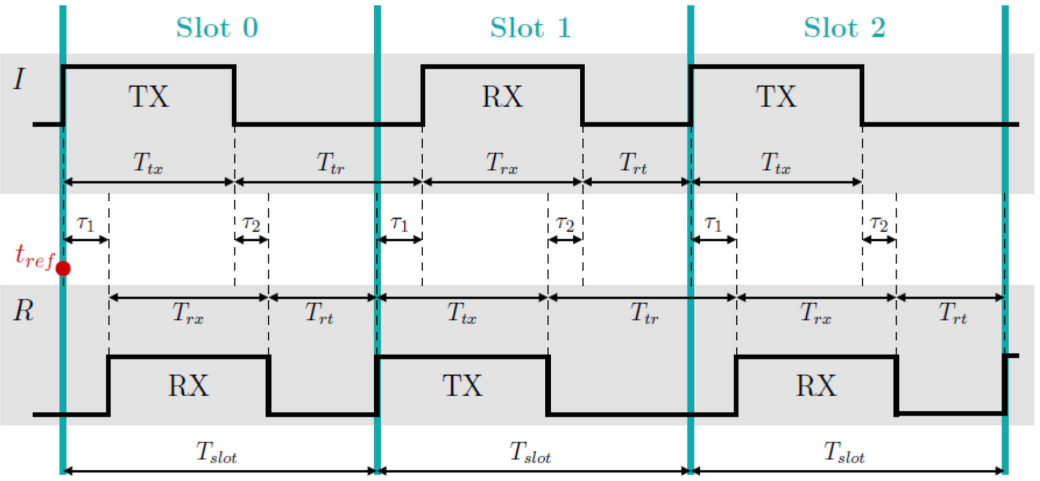




CC1101 radio module integrated into CC430

CC430 Implementation Details

- τ_1: time distance from when the sync word pin goes high at a transmitter (*i.e.*, beginning of the transmission) until when the sync word pin goes high at a receiver (*i.e.*, beginning of the reception).
- τ_2: time distance from when the sync word pin goes low at a transmitter (*i.e.*, end of the transmission) until when the sync word pin goes high at a receiver (*i.e.*, end of the reception).



$\tau_1 = 13.54\,\mu\text{s}$
$\tau_2 = 11.86\,\mu\text{s}$

$$\tau_1 - \tau_2 = T_{tx} - T_{rx}$$
$$\tau_1 + \tau_2 = T_{tr} - T_{rt}$$

$$T_{slot} = T_{rx} + T_{rt} + \tau_1 = T_{tx} + T_{tr} - \tau_1$$

Recent Research

EVALUATING CONCURRENT TRANSMISSIONS

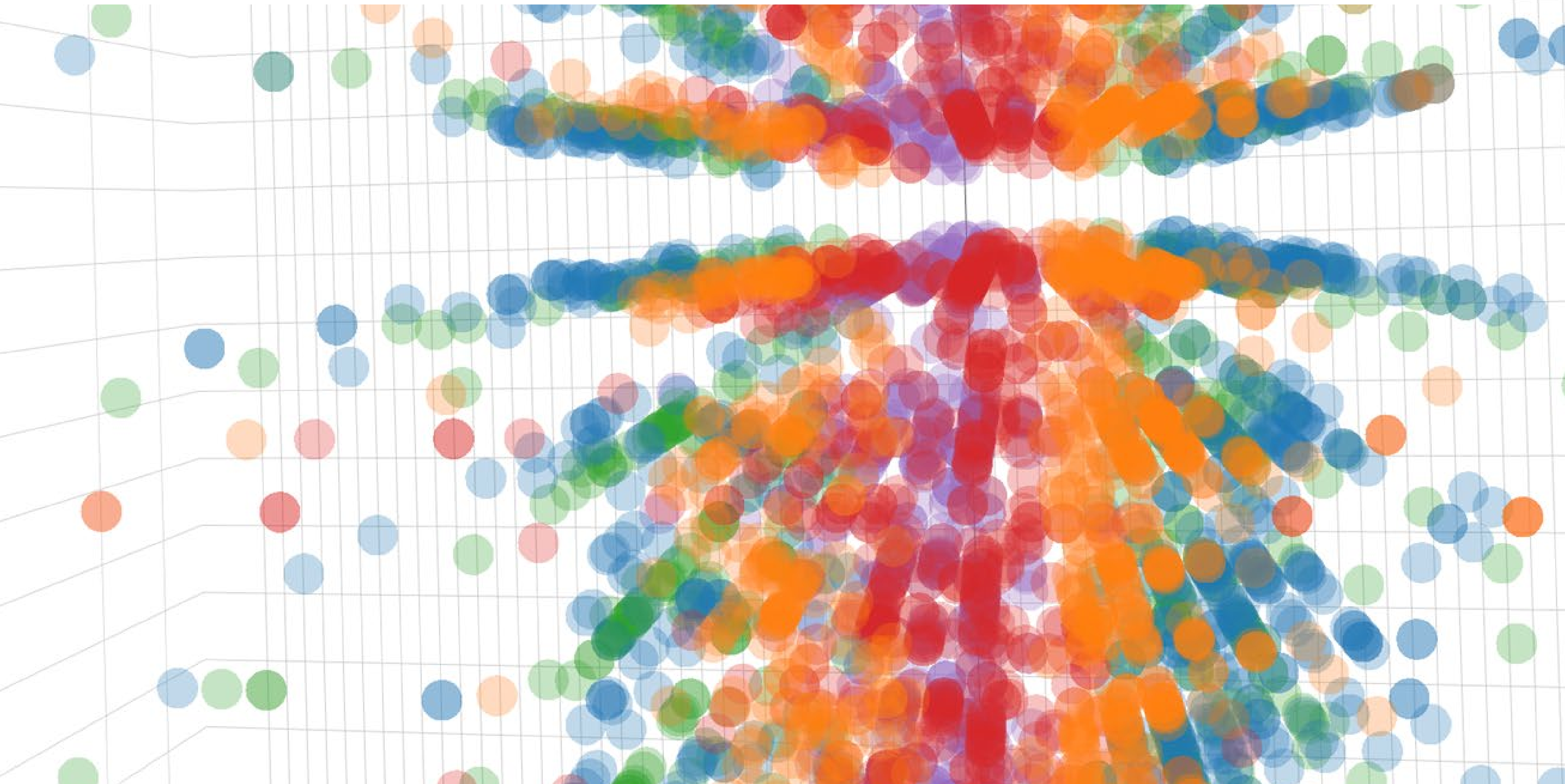
ETH zürich

Master Thesis

Truth be told: Benchmarking BLE and 15.4

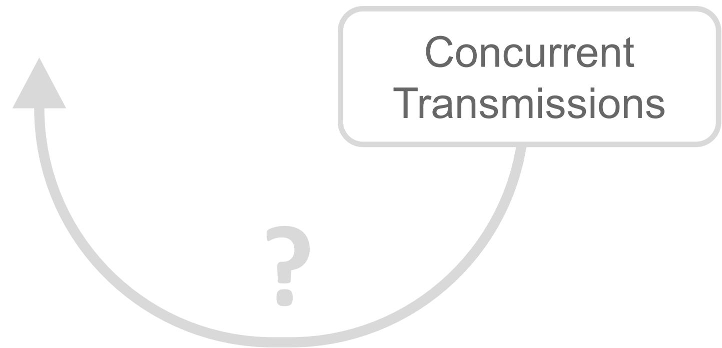
Anna-Brit Schaper

Supervisors: Romain Jacob | Reto Da Forno | Andreas Biri | Prof. Dr. Lothar Thiele



	BLE	IEEE 802.15.4
Used…	everywhere	in WSN, home automation
Made for…	single-hop point to point	multi-hop networking

	BLE	IEEE 802.15.4
Used…	everywhere	in WSN, home automation
Made for…	single-hop point to point	multi-hop networking



Our Objective:

To experimentally determine conditions for successful concurrent transmissions in IEEE 802.15.4 and BLE in a repeatable fashion using small, low-cost, COTS devices.

- IEEE 802.15.4
- BLE 1 Mbit
- BLE 2 Mbit
- BLE 500 Kbit
- BLE 125 Kbit

	IEEE 802.15.4	BLE 1 Mbit	BLE 2 Mbit	BLE 500 Kbit	BLE 125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs	0.25 µs	0.5 µs	0.5 µs	1 µs
	τ_s	$\tau_s/4$	τ_s	$\tau_s/2$	τ_s

ℹ the smaller the better for constructive interference

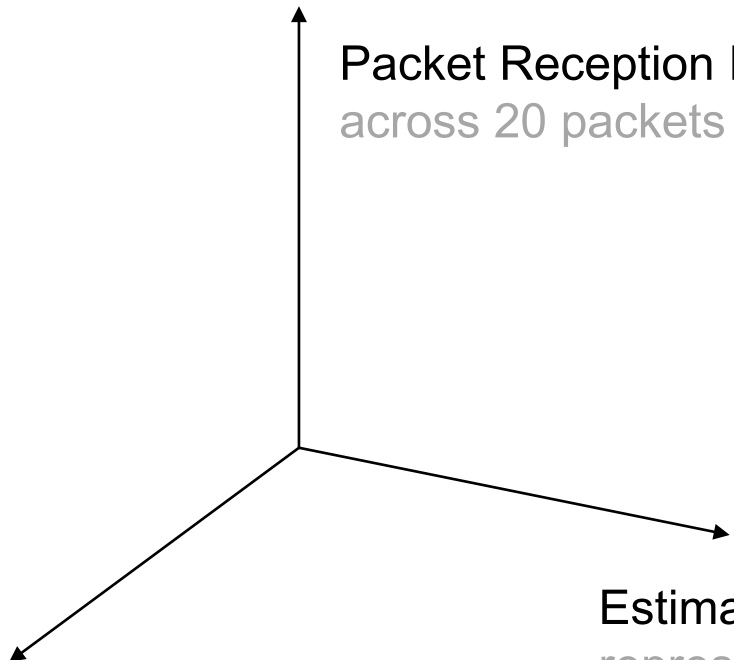
	IEEE 802.15.4	BLE 1 Mbit	BLE 2 Mbit	BLE 500 Kbit	BLE 125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs τ_s	0.25 µs $\tau_s/4$	0.5 µs τ_s	0.5 µs $\tau_s/2$	1 µs τ_s
Power Delta	3 dB	8 dB	8 dB	8 dB	8 dB

i the larger the better for the (power) capture effect

	IEEE 802.15.4	BLE			
		1 Mbit	2 Mbit	500 Kbit	125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs τ_s	0.25 µs $\tau_s/4$	0.5 µs τ_s	0.5 µs $\tau_s/2$	1 µs τ_s
Power Delta	3 dB	8 dB	8 dB	8 dB	8 dB

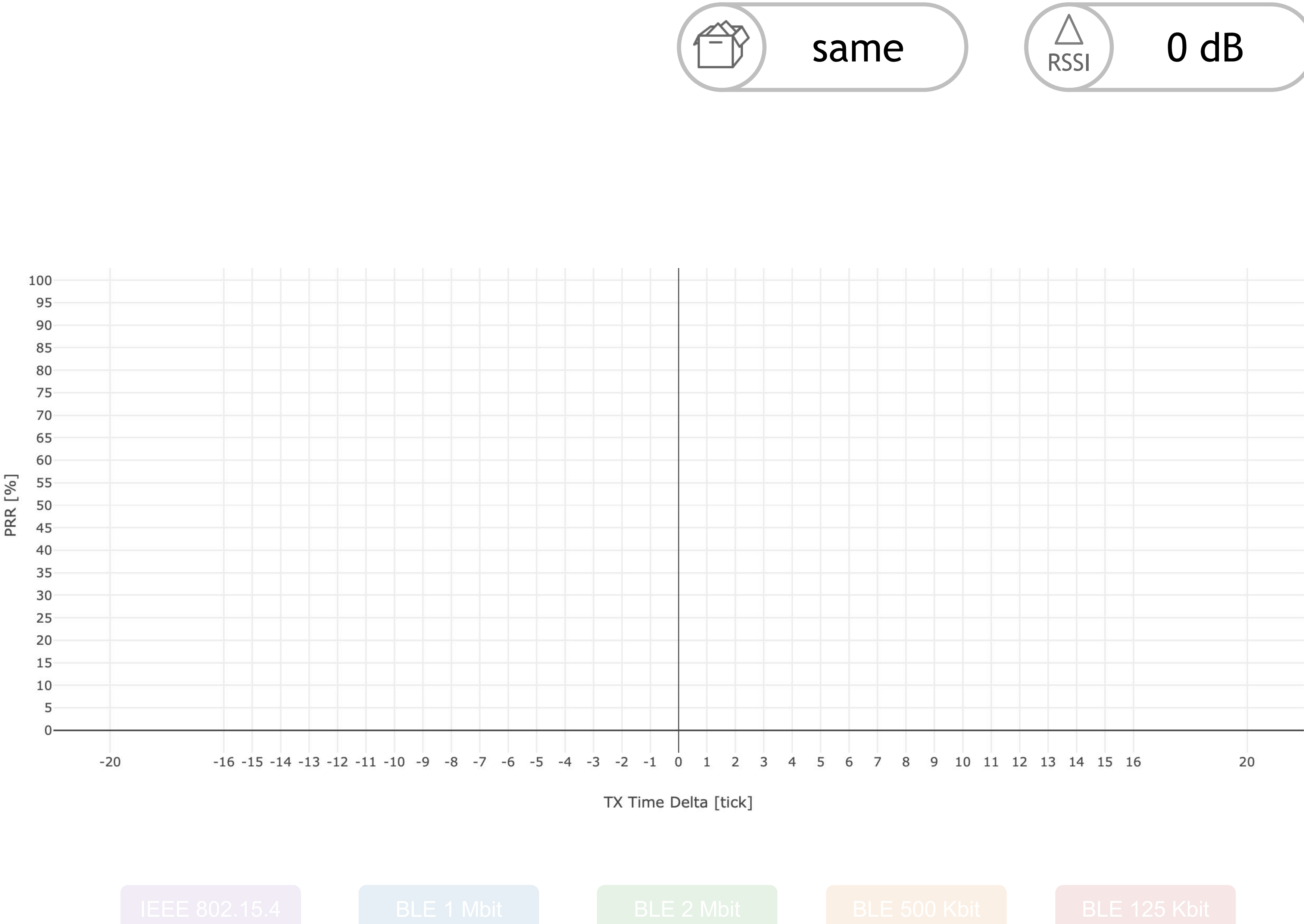
	IEEE 802.15.4	BLE			
		1 Mbit	2 Mbit	500 Kbit	125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs τ_s	0.25 µs $\tau_s/4$	[illegible]	0.5 µs $\tau_s/2$	1 µs τ_s
Power Delta	3 dB	8 dB	8 dB	8 dB	8 dB

Results for **same / different** packet contents in separate plots

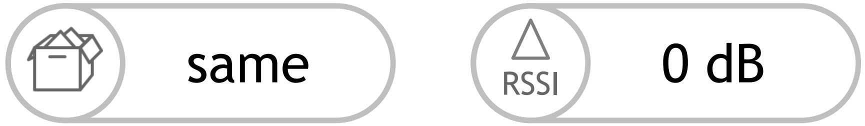


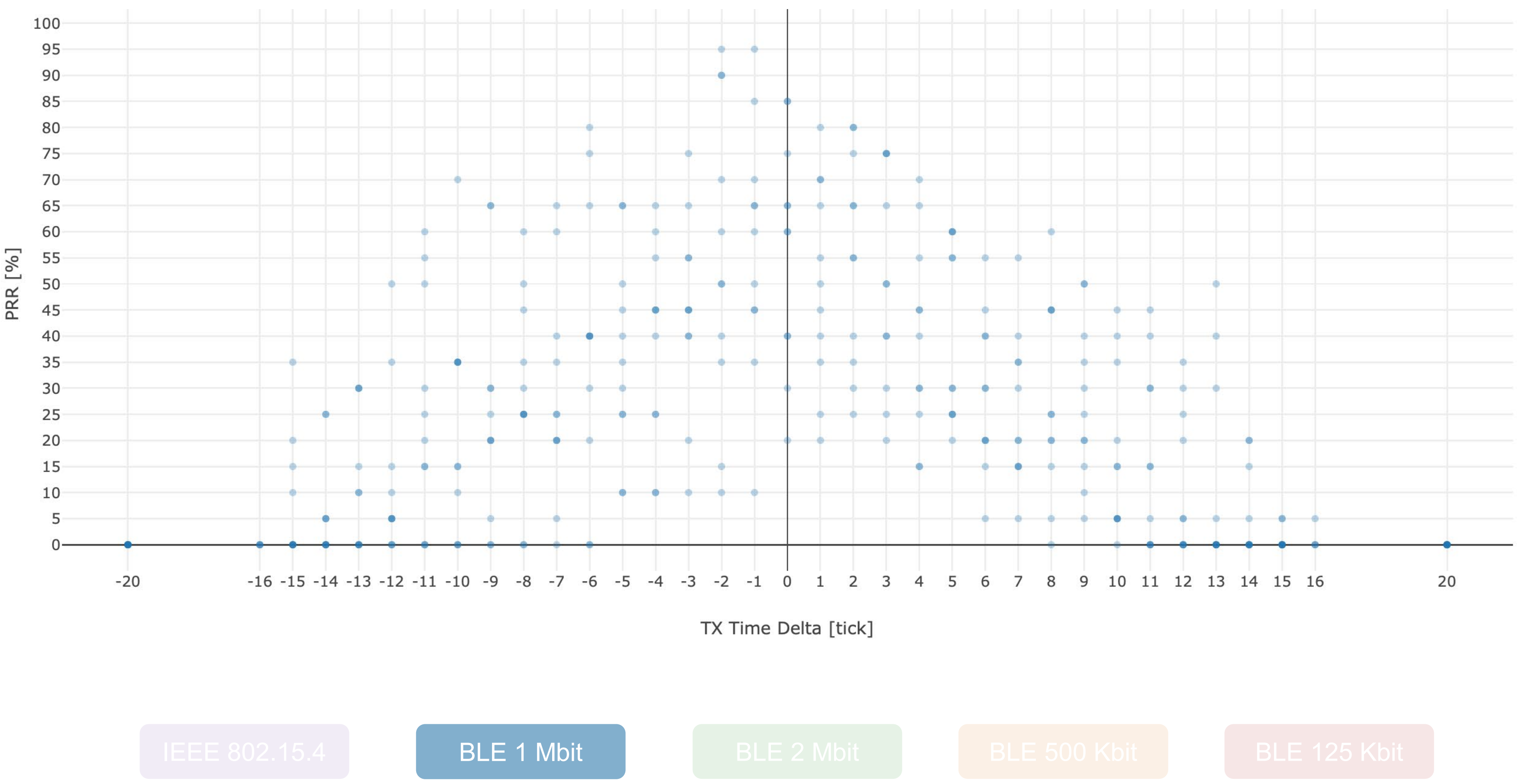
Colors represent modes:

IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit



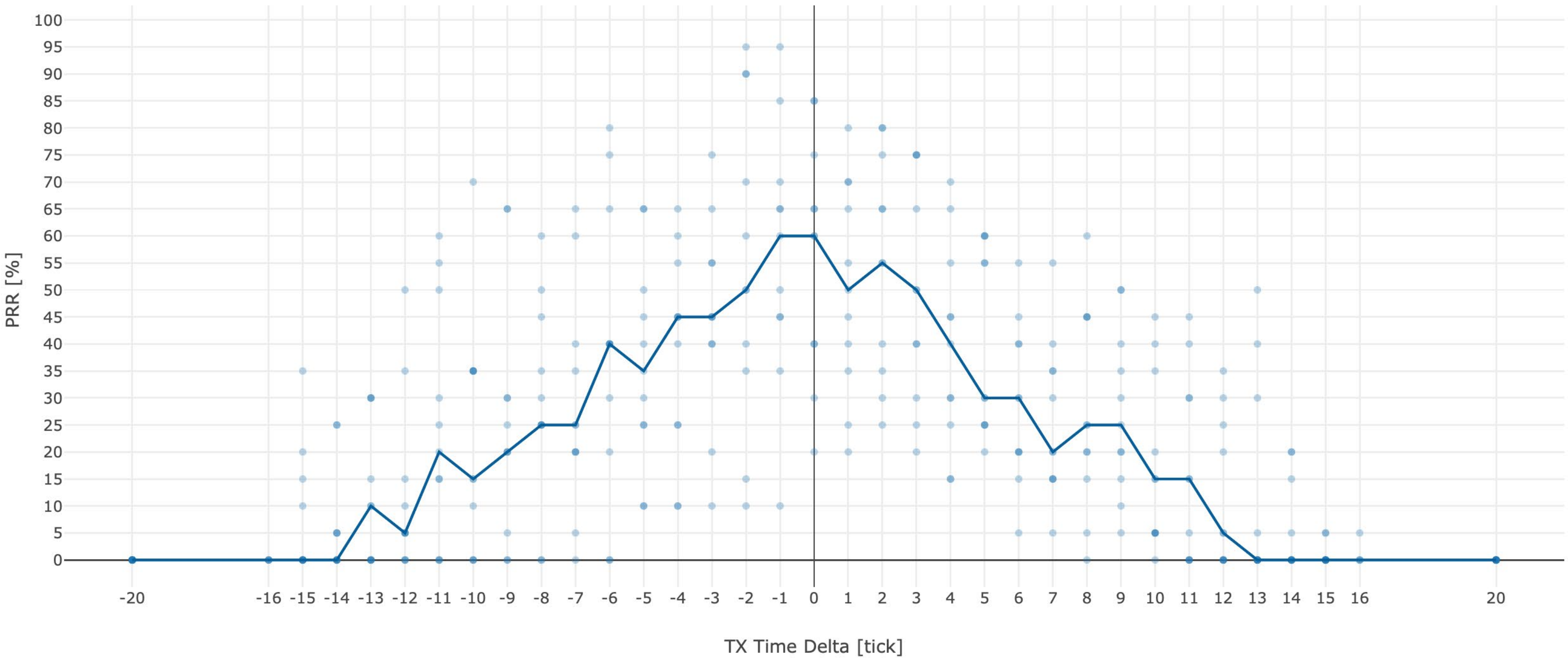
same
RSSI
0 dB
PRR [%]
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
-20 -16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 20
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit


same
RSSI
0 dB


PRR [%]
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
-20 -16 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 20
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit

same

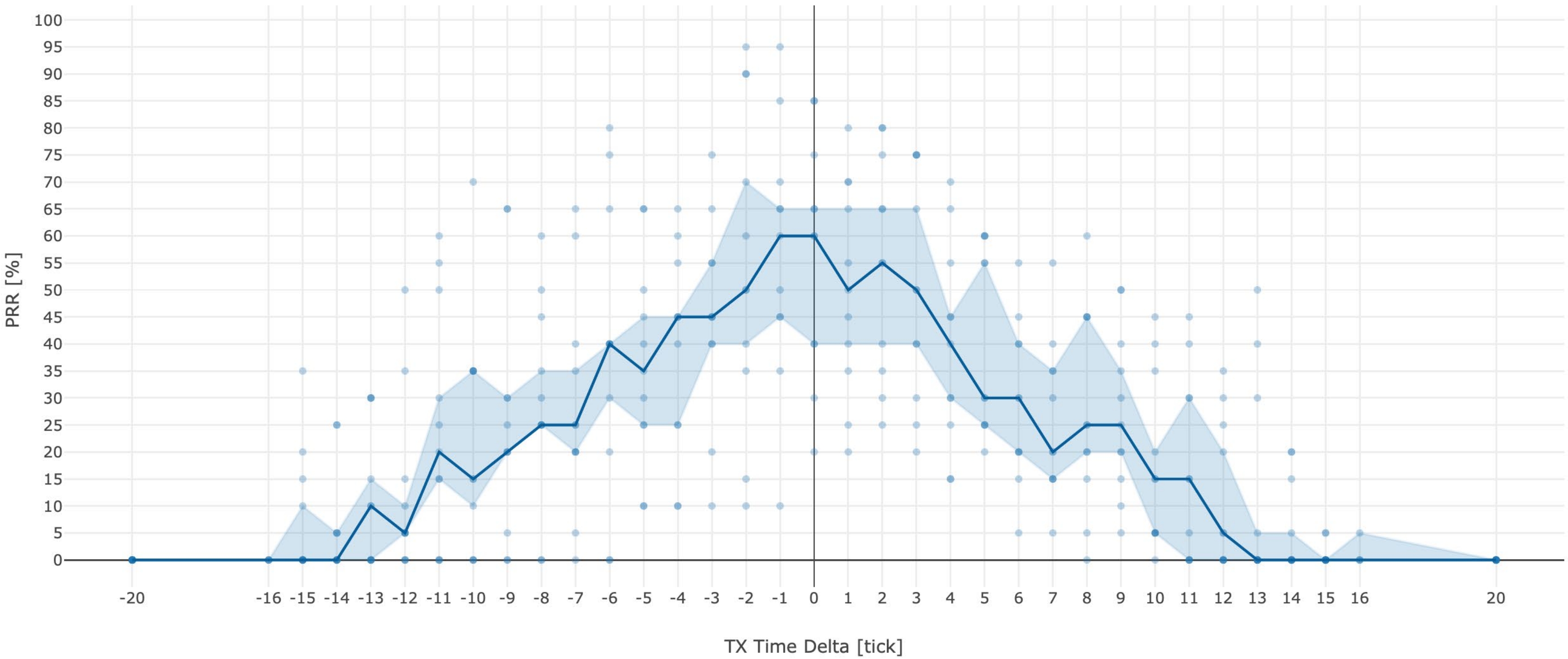
RSSI 0 dB



IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

same

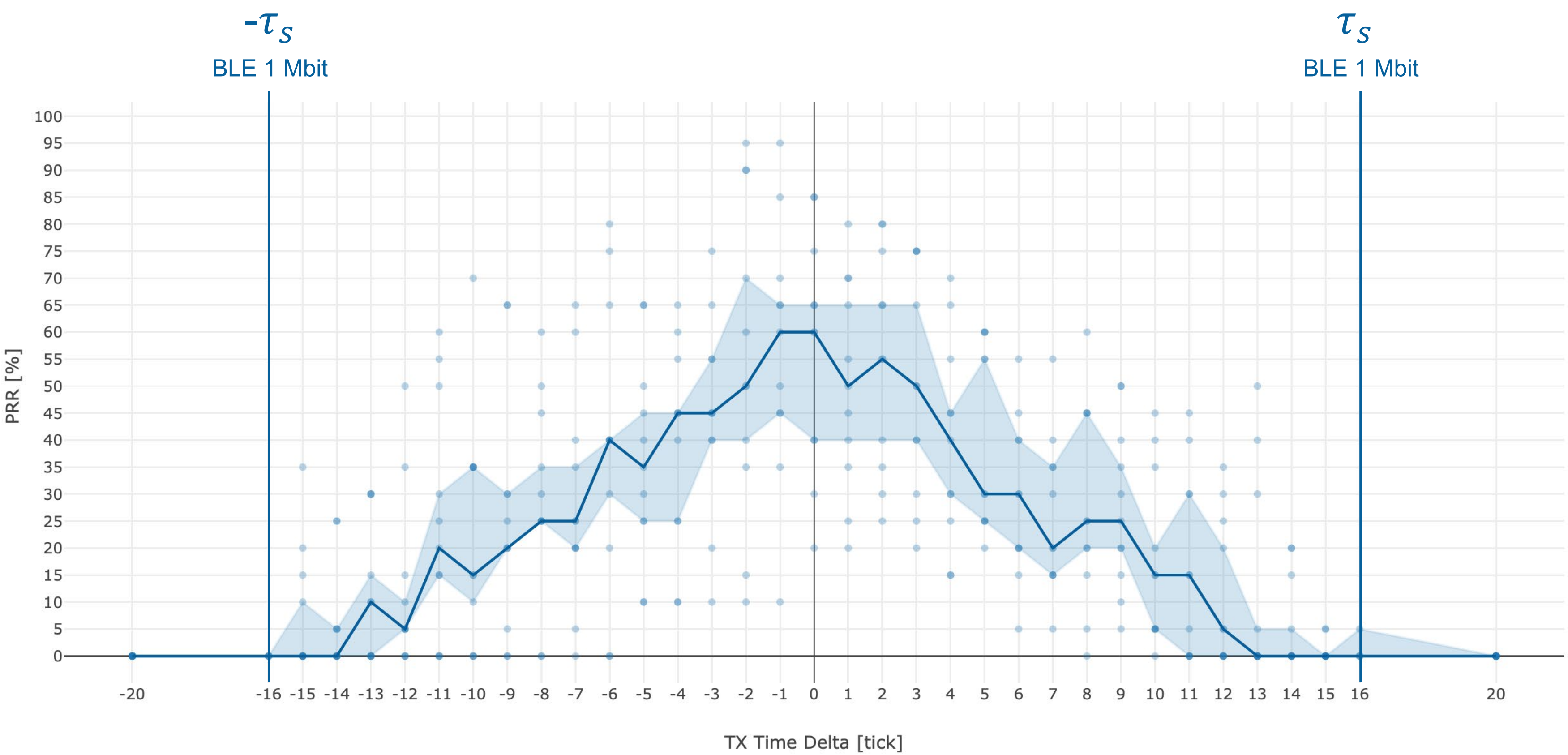
0 dB



IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

same

0 dB



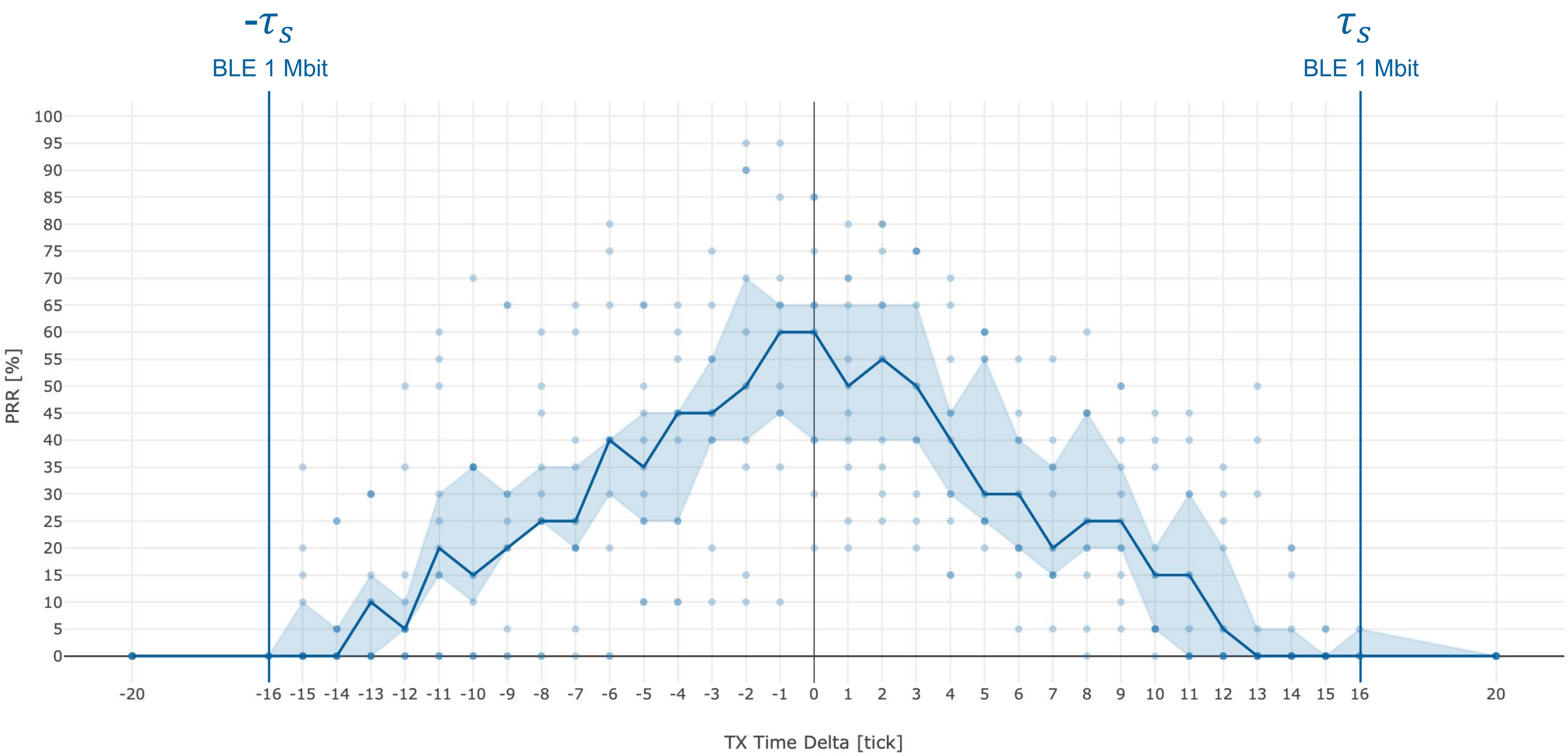
IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

	IEEE 802.15.4	BLE			
		1 Mbit	2 Mbit	500 Kbit	125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs τ_s	0.25 µs $\tau_s/4$	0.5 µs τ_s	0.5 µs $\tau_s/2$	1 µs τ_s
Power Delta	5 dB	13 dB	11 dB	11 dB	9 dB

	IEEE 802.15.4	BLE 1 Mbit	BLE 2 Mbit	BLE 500 Kbit	BLE 125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs τ_s	1 µs τ_s	0.5 µs τ_s	0.5 µs $\tau_s/2$	1 µs τ_s
Power Delta	5 dB	13 dB	11 dB	11 dB	9 dB

same

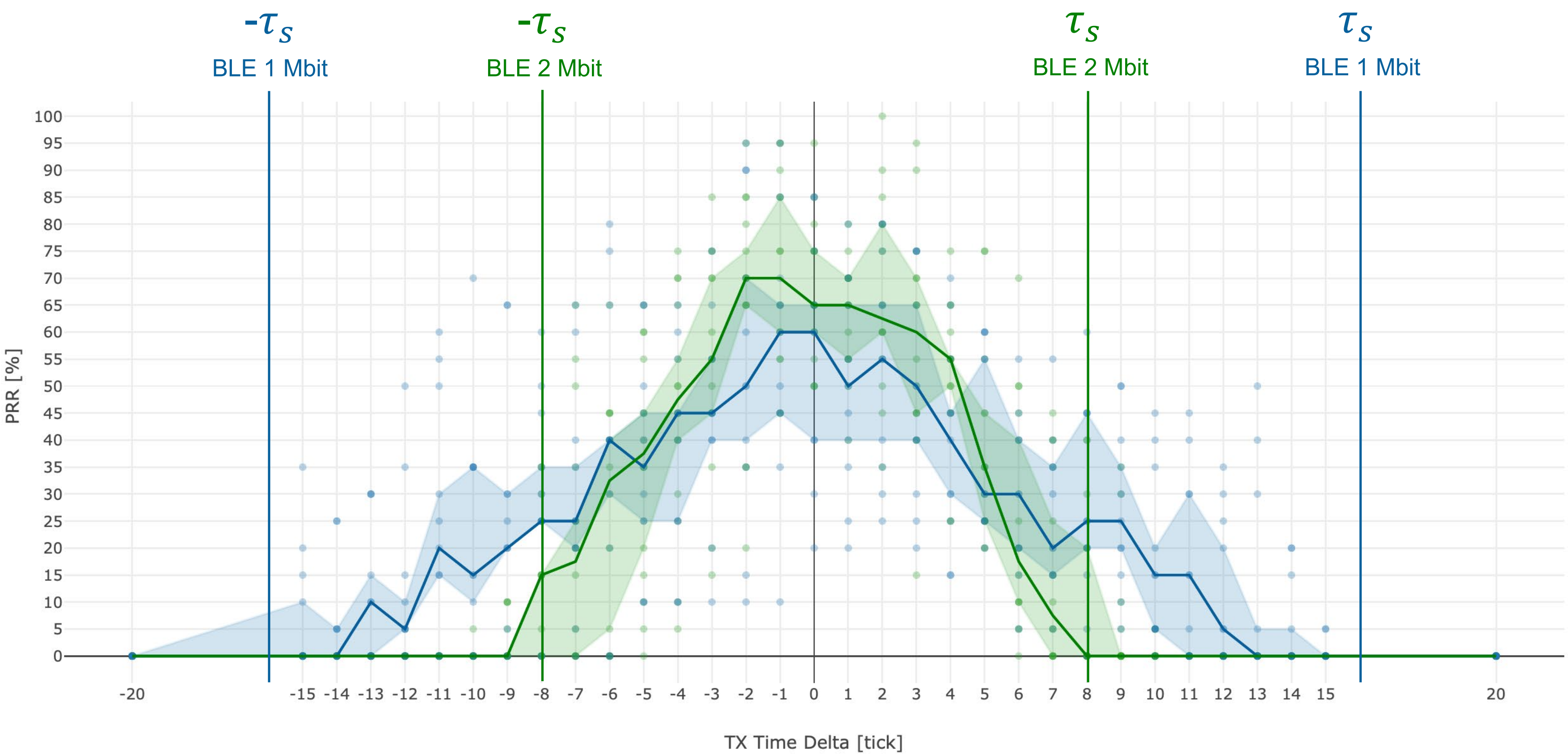
RSSI 0 dB



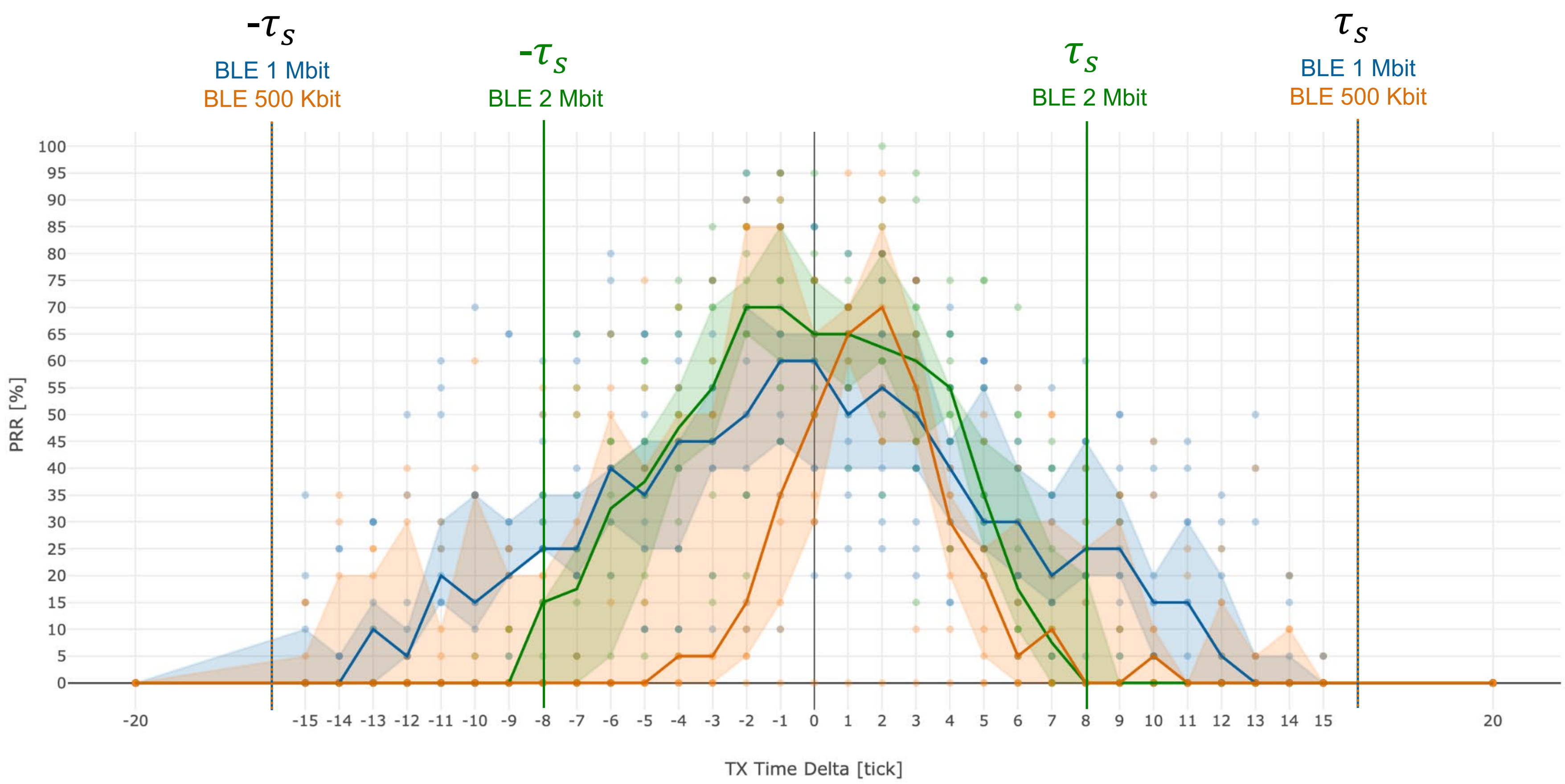
IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

same

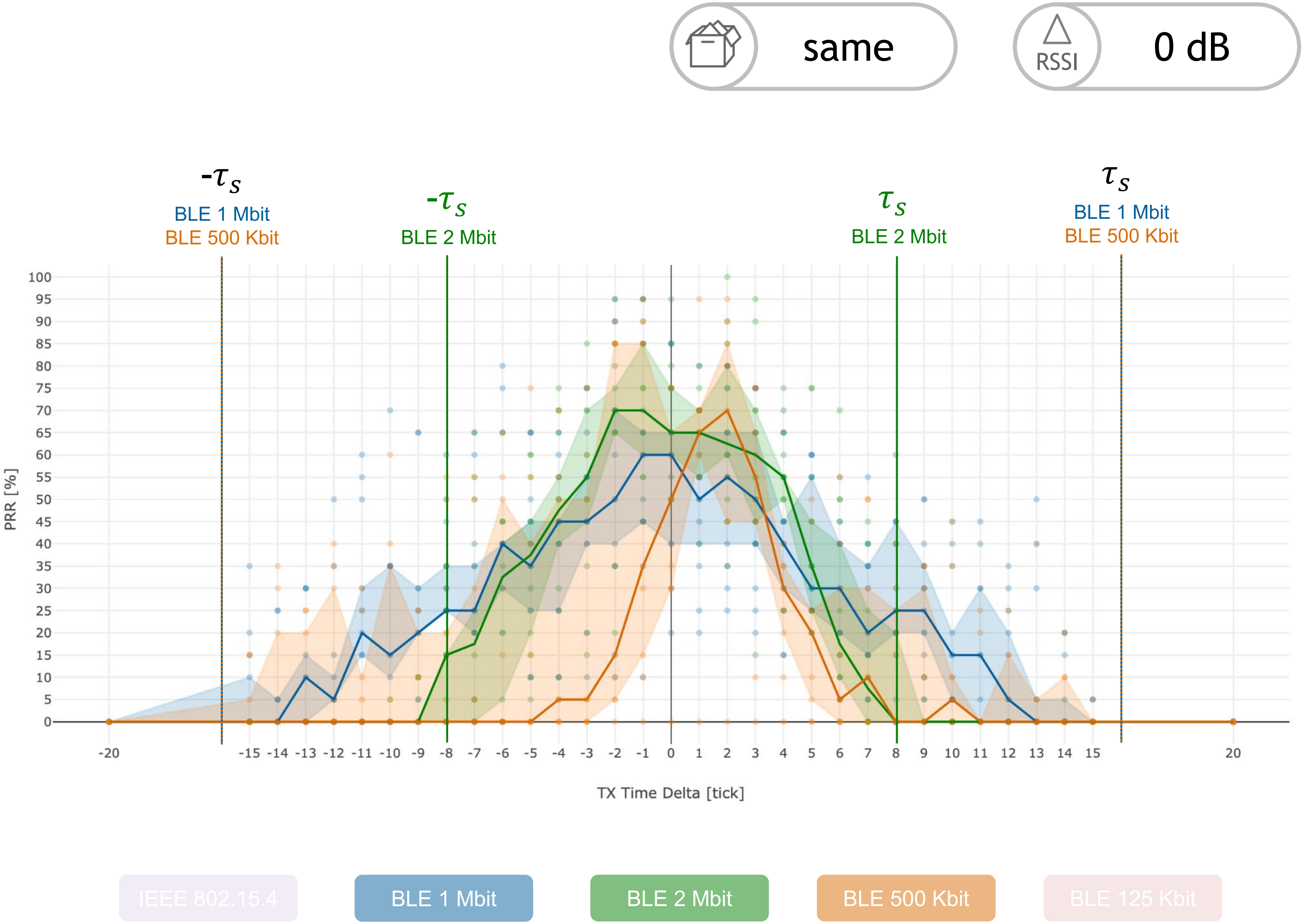
RSSI 0 dB



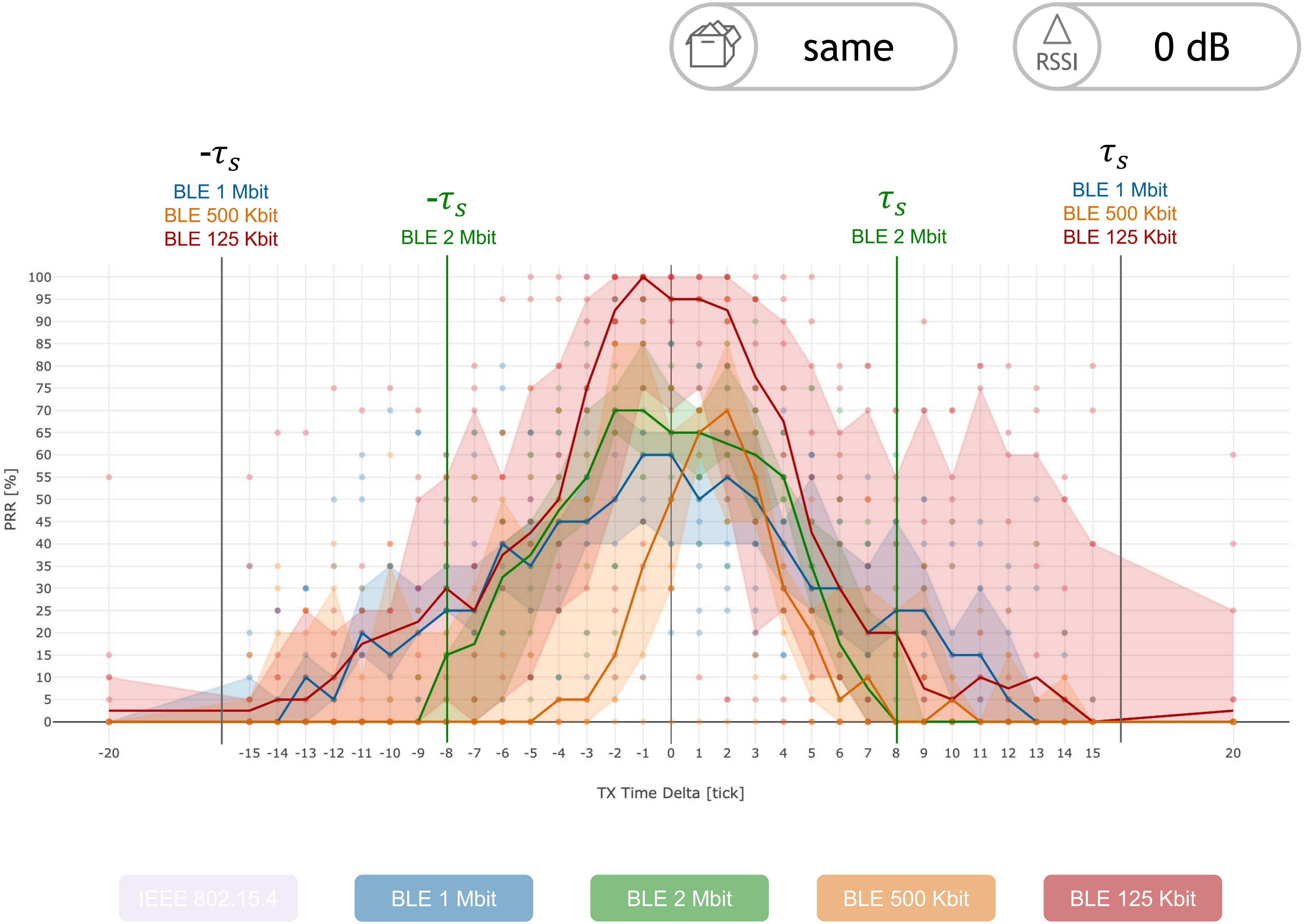
IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

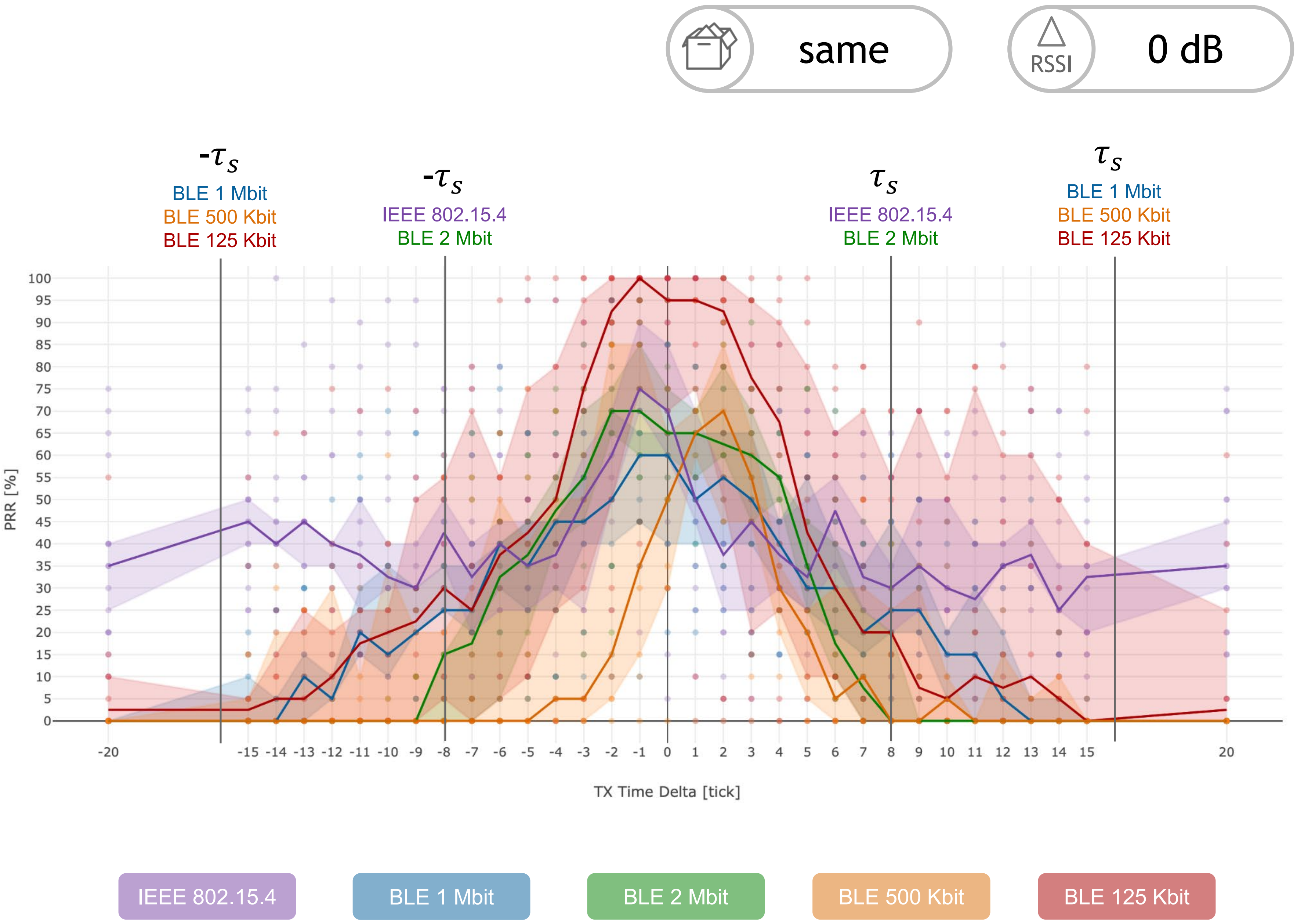


IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit



same
RSSI
0 dB
$-\tau_s$
BLE 1 Mbit
BLE 500 Kbit
$-\tau_s$
BLE 2 Mbit
τ_s
BLE 2 Mbit
τ_s
BLE 1 Mbit
BLE 500 Kbit
PRR [%]
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit

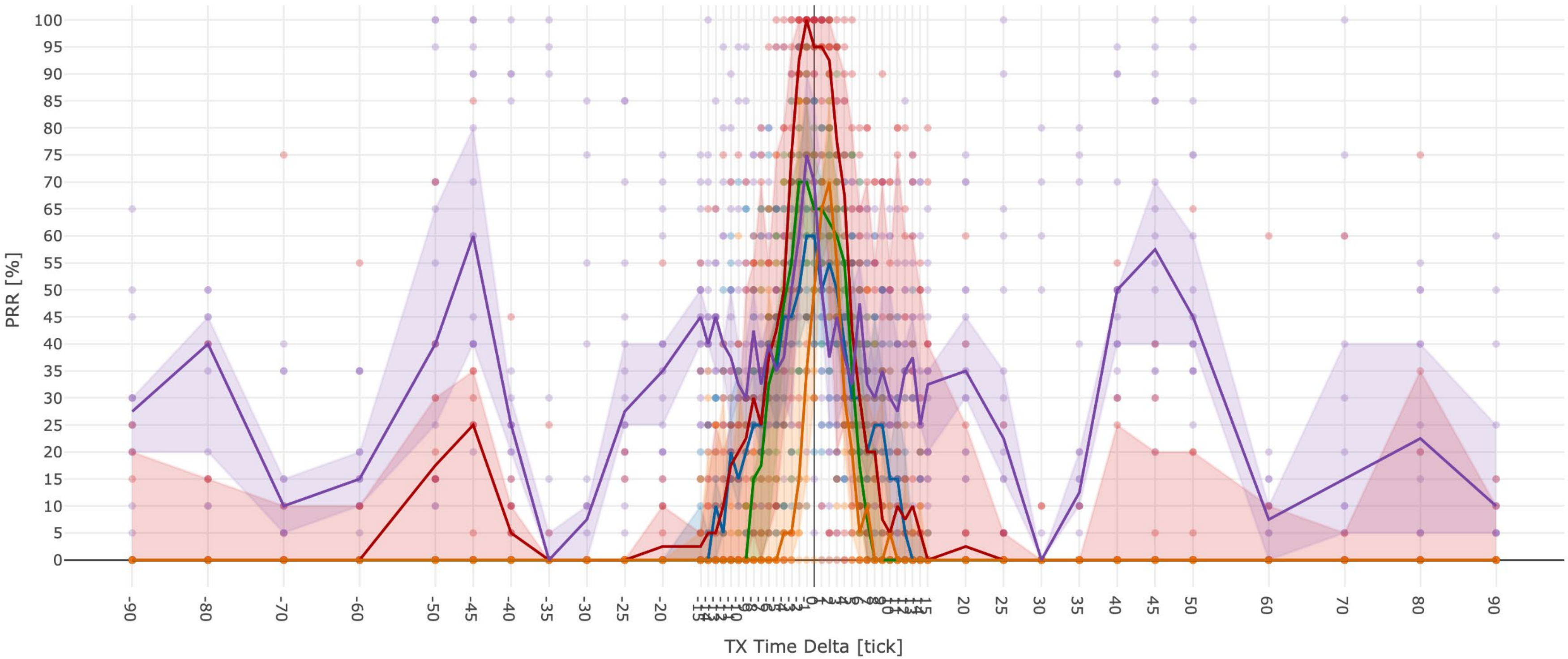

same
RSSI
0 dB
-τ_s
BLE 1 Mbit
BLE 500 Kbit
BLE 125 Kbit
-τ_s
BLE 2 Mbit
τ_s
BLE 2 Mbit
τ_s
BLE 1 Mbit
BLE 500 Kbit
BLE 125 Kbit
PRR [%]
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit



same
RSSI
0 dB
-τ_s
BLE 1 Mbit
BLE 500 Kbit
BLE 125 Kbit
-τ_s
IEEE 802.15.4
BLE 2 Mbit
τ_s
IEEE 802.15.4
BLE 2 Mbit
τ_s
BLE 1 Mbit
BLE 500 Kbit
BLE 125 Kbit
PRR [%]
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
-20 -15 -14 -13 -12 -11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 20
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit

same

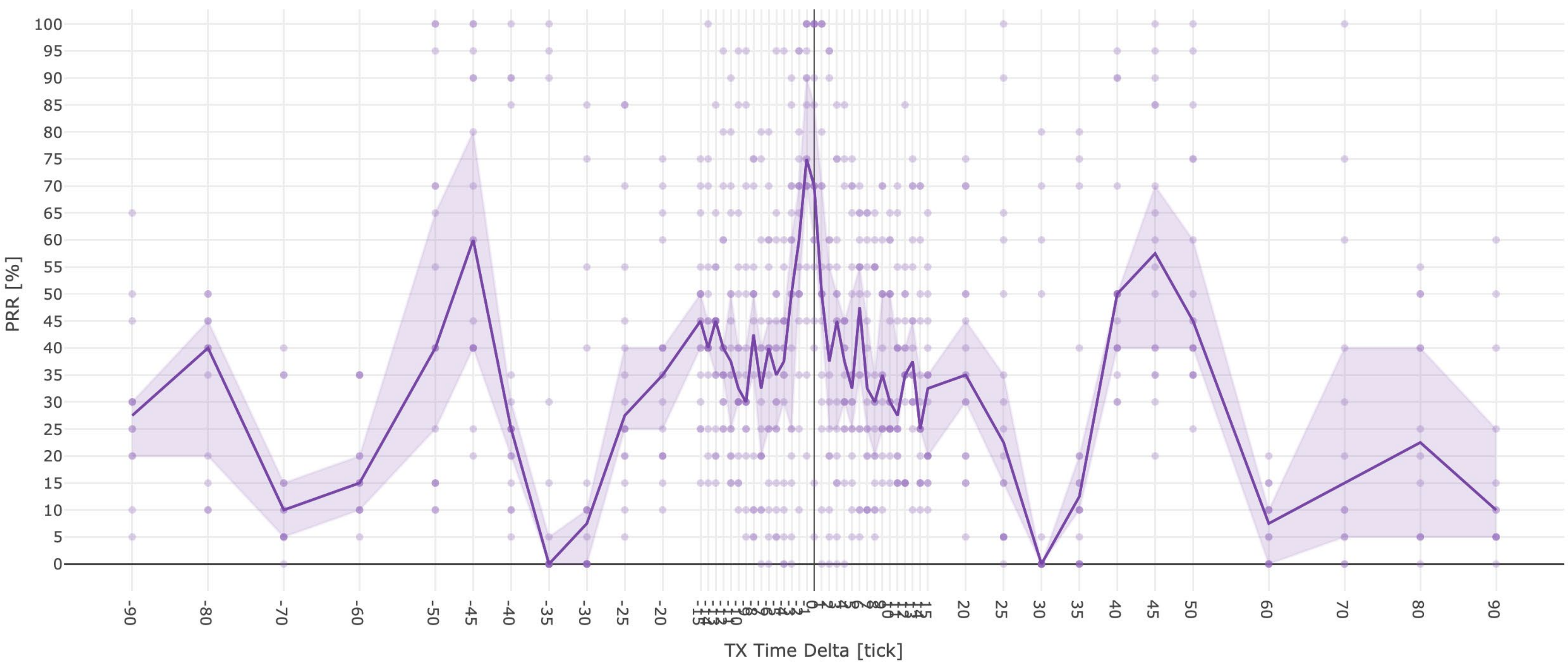
RSSI 0 dB



IEEE 802.15.4 BLE 1 Mbit BLE 2 Mbit BLE 500 Kbit BLE 125 Kbit

same

0 dB



IEEE 802.15.4

BLE 1 Mbit

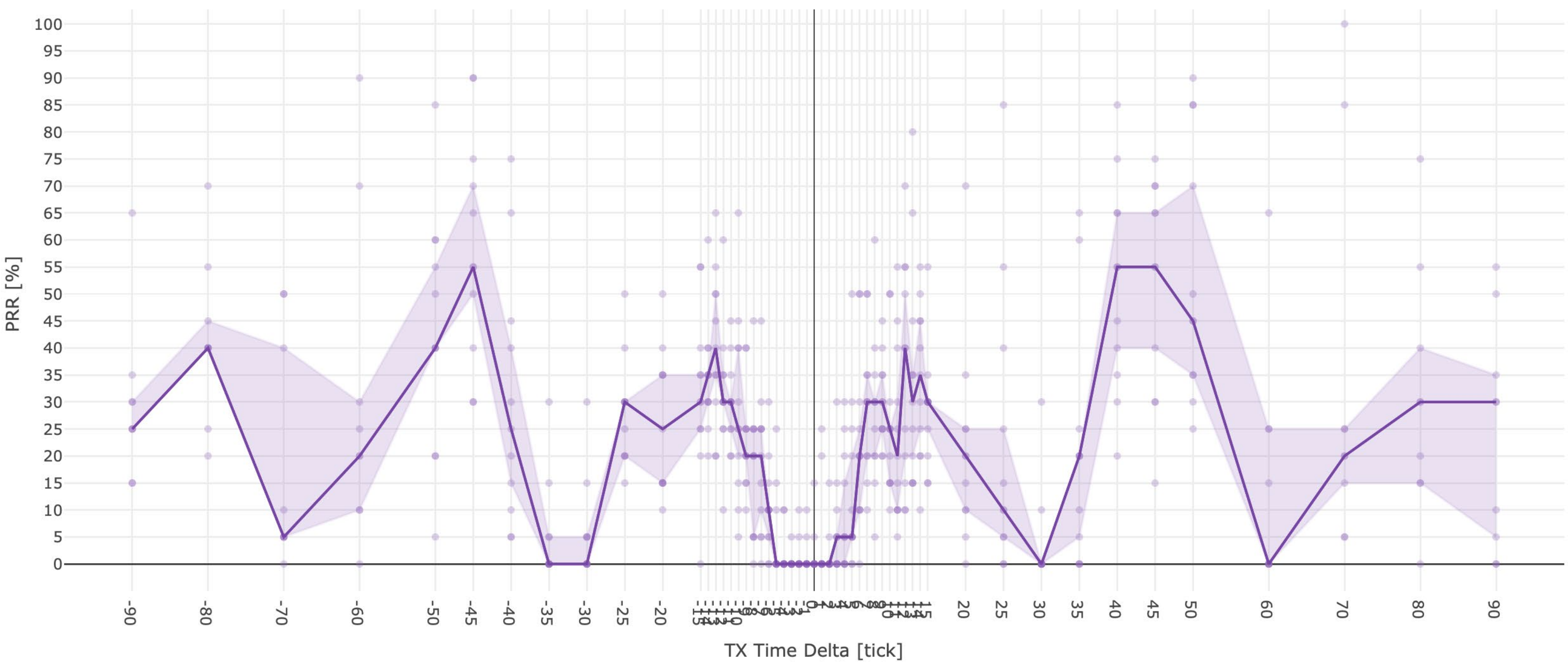
BLE 2 Mbit

BLE 500 Kbit

BLE 125 Kbit

different

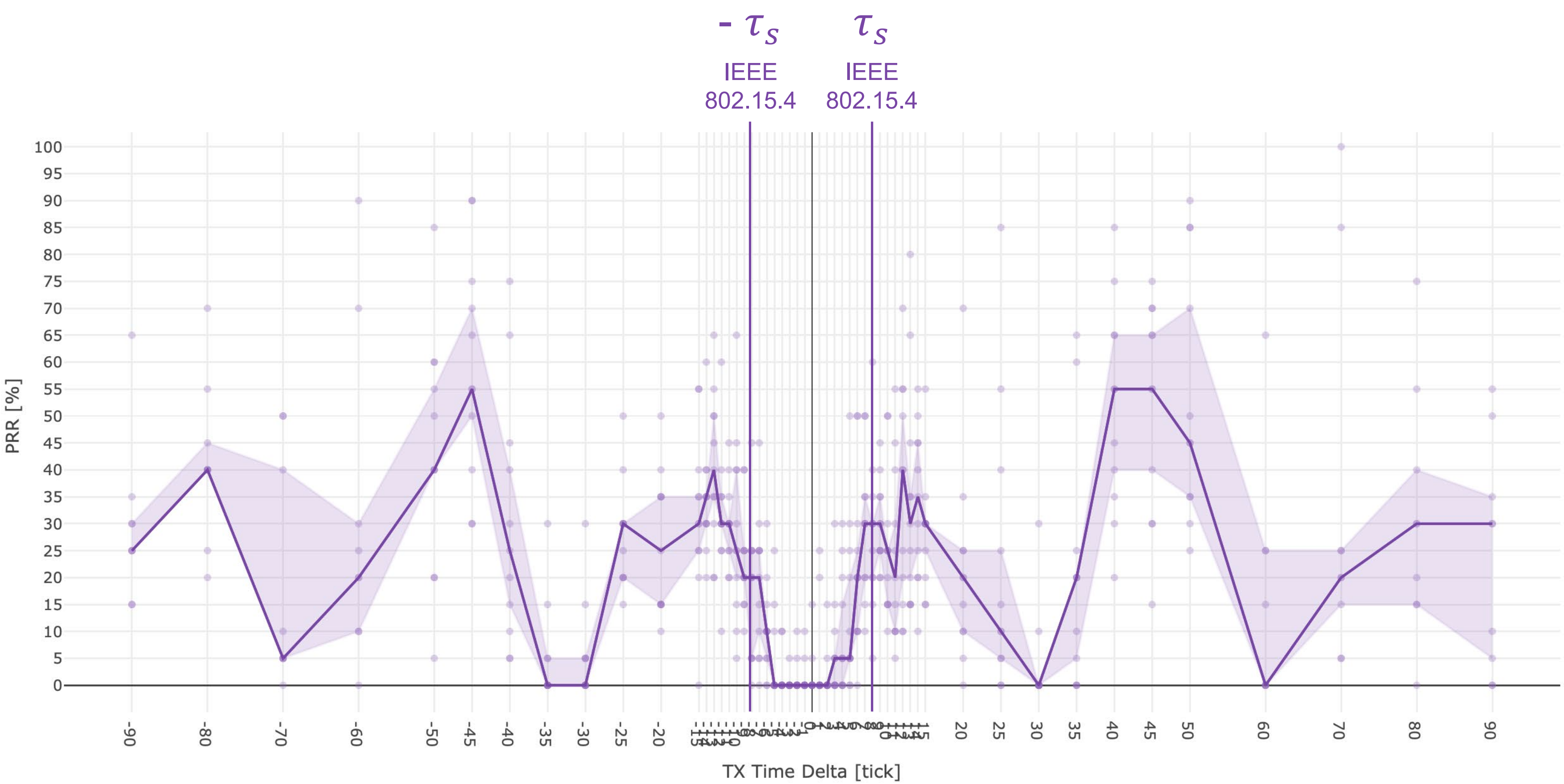
RSSI 0 dB



IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

different

RSSI 0 dB



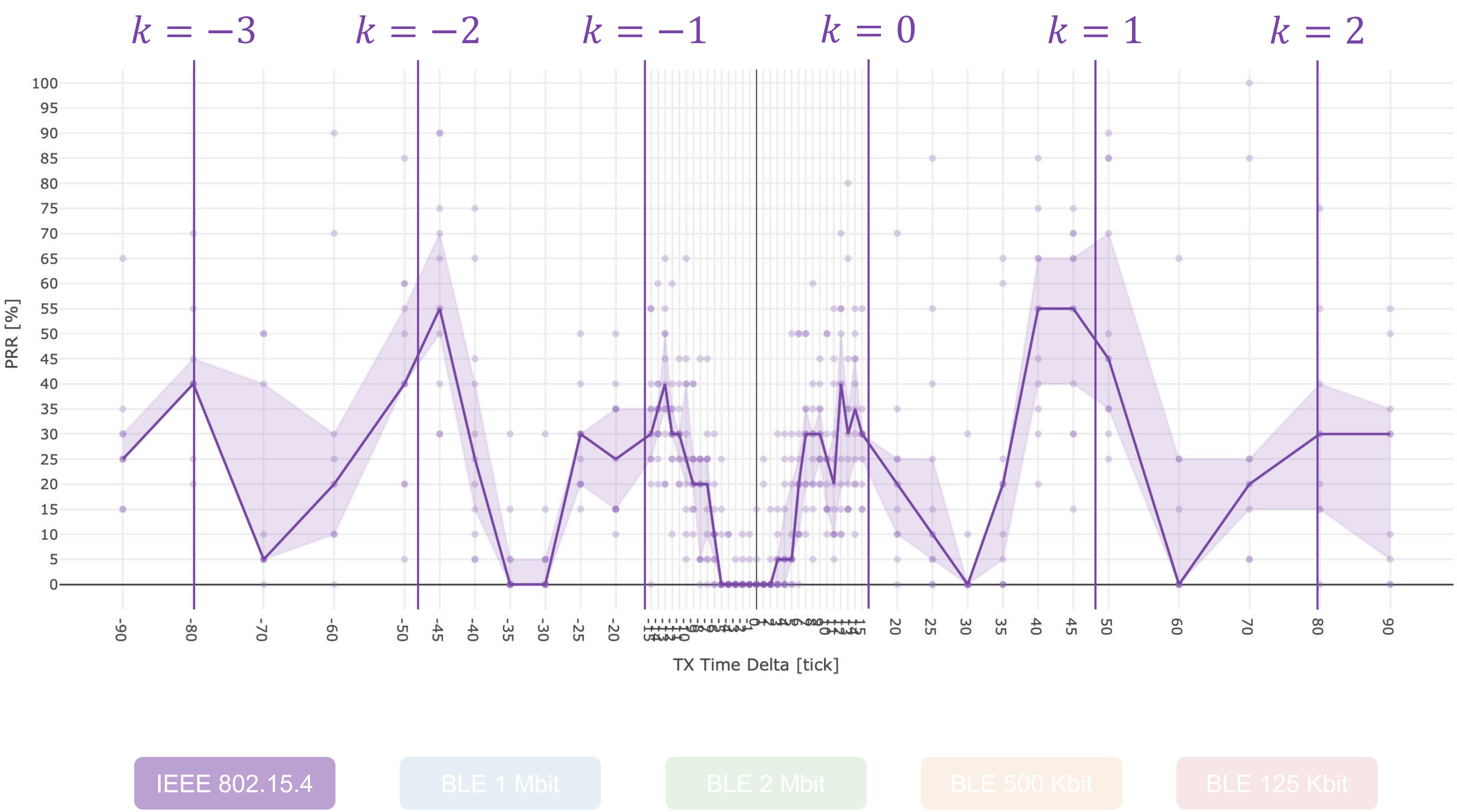
IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

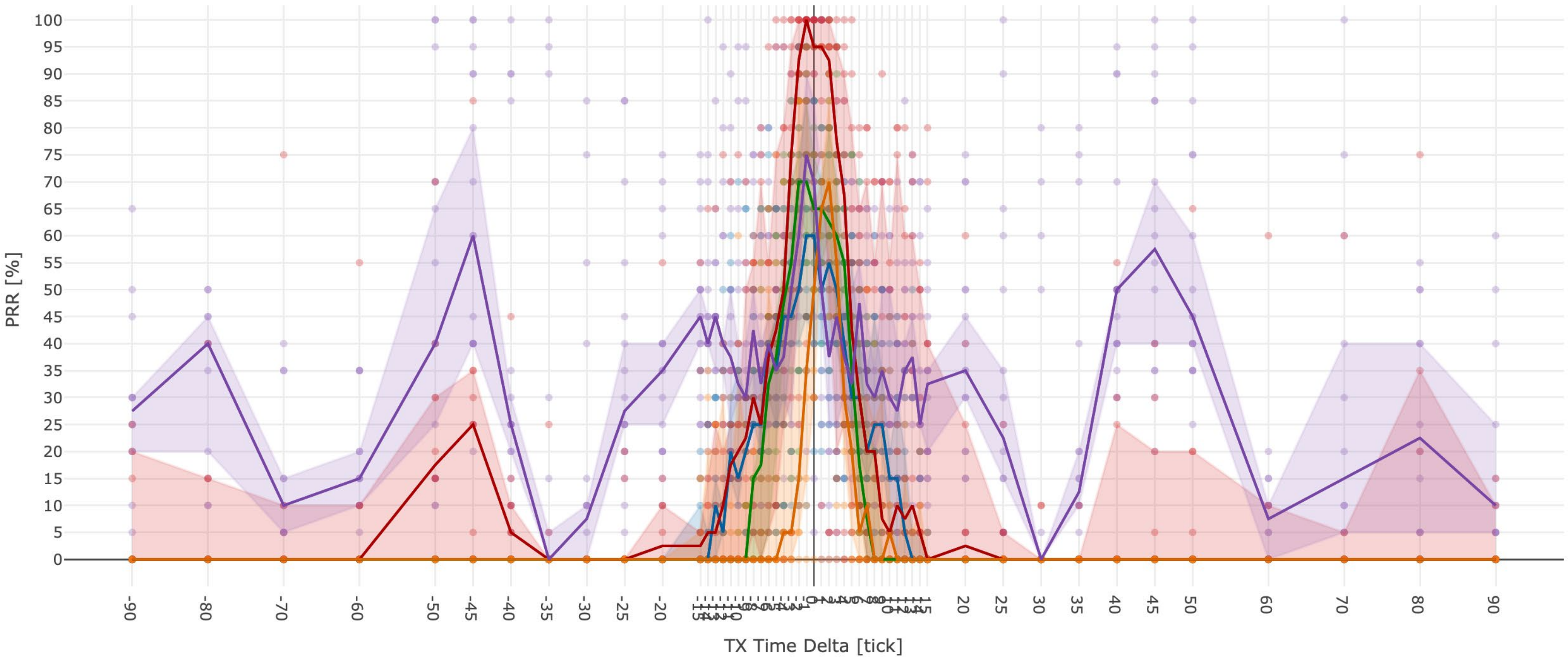
	IEEE 802.15.4	BLE 1 Mbit	BLE 2 Mbit	BLE 500 Kbit	BLE 125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs τ_s	1 µs τ_s	0.5 µs τ_s	0.25 µs $\tau_s/4$	1 µs τ_s
Power Delta	5 dB	13 dB	11 dB	11 dB	9 dB

DSSS causes peaks at $4k\tau_s + 2\tau_s$

different

0 dB

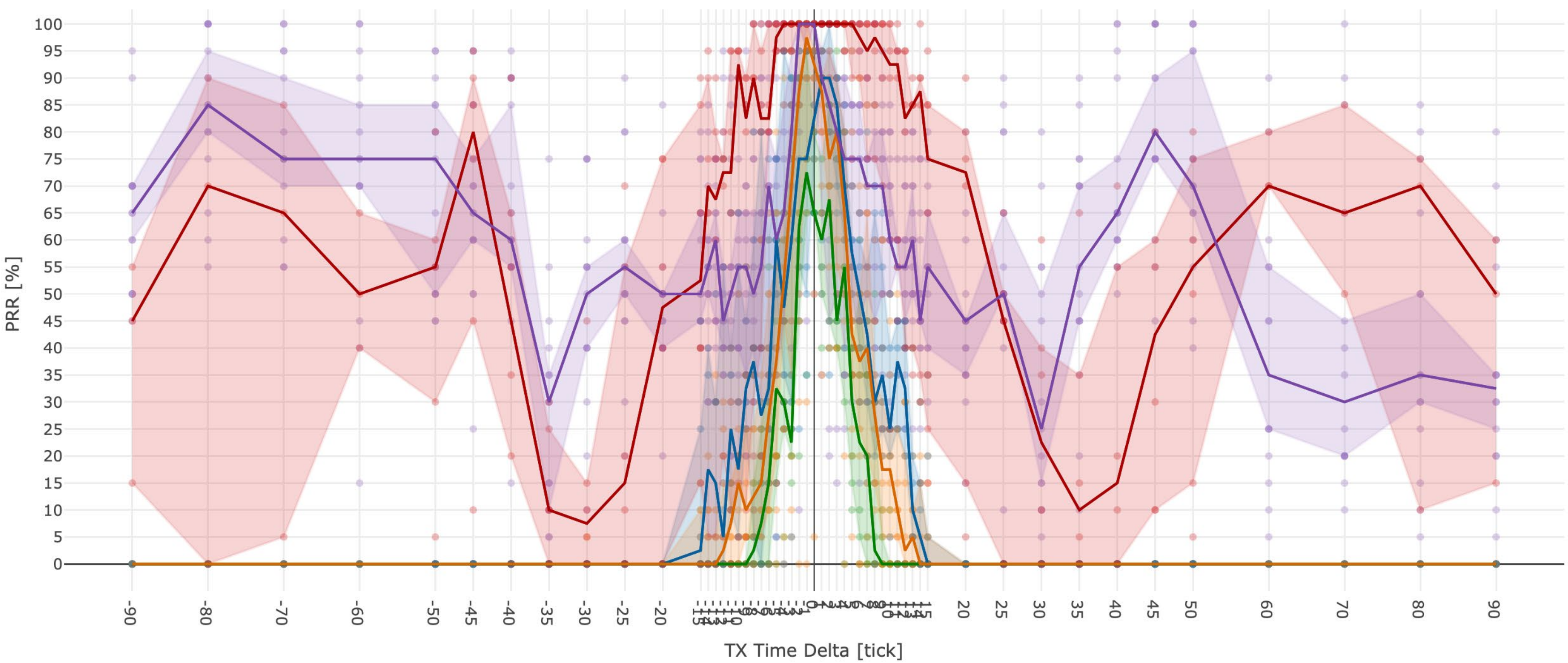




IEEE 802.15.4 BLE 1 Mbit BLE 2 Mbit BLE 500 Kbit BLE 125 Kbit

same

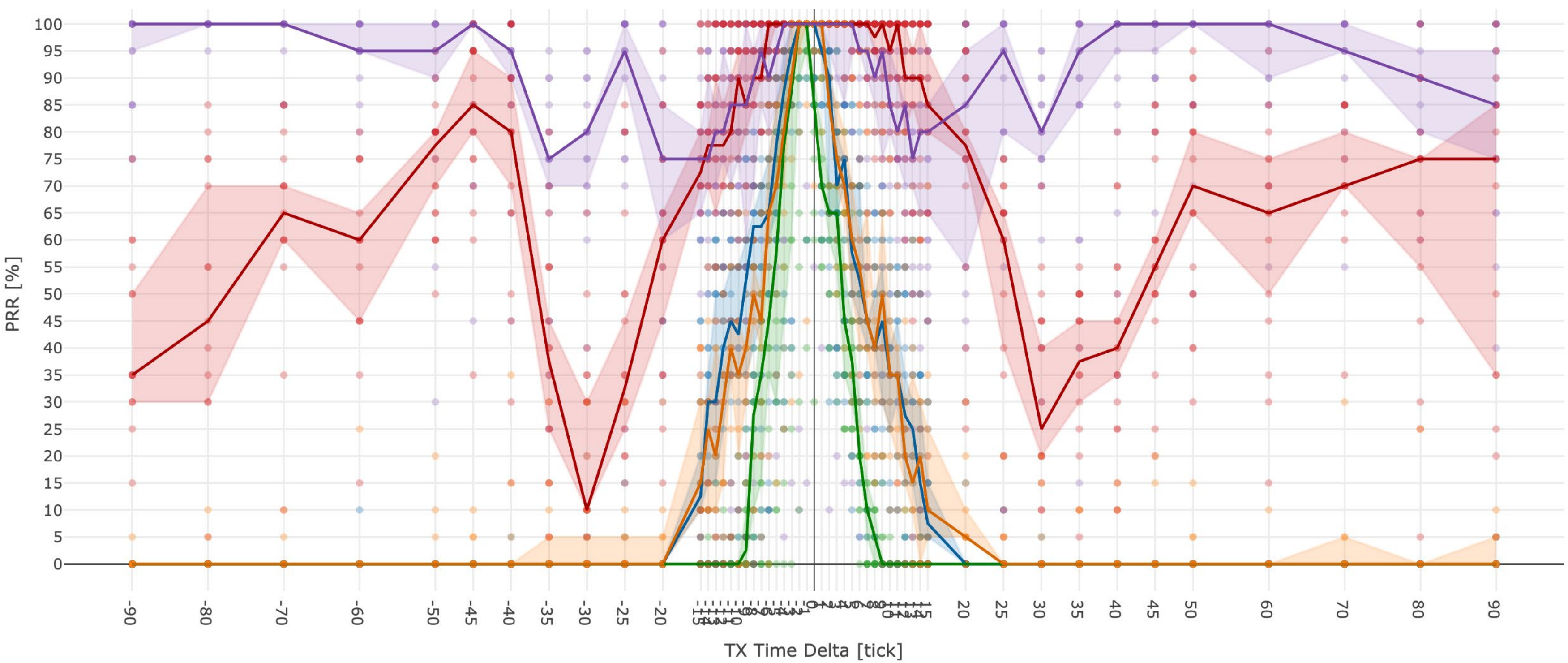
RSSI 1 dB



IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

same

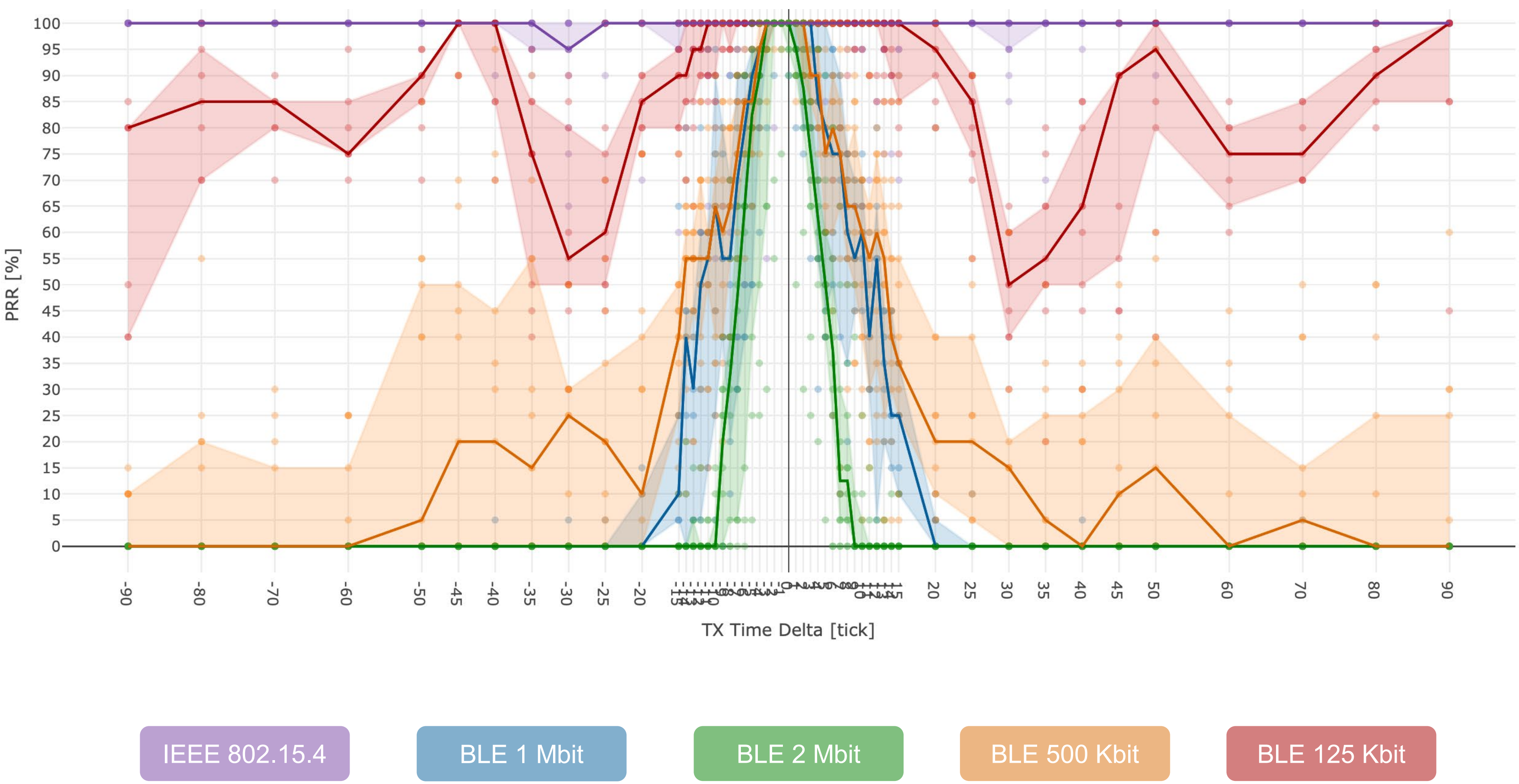
RSSI 2 dB



IEEE 802.15.4 BLE 1 Mbit BLE 2 Mbit BLE 500 Kbit BLE 125 Kbit

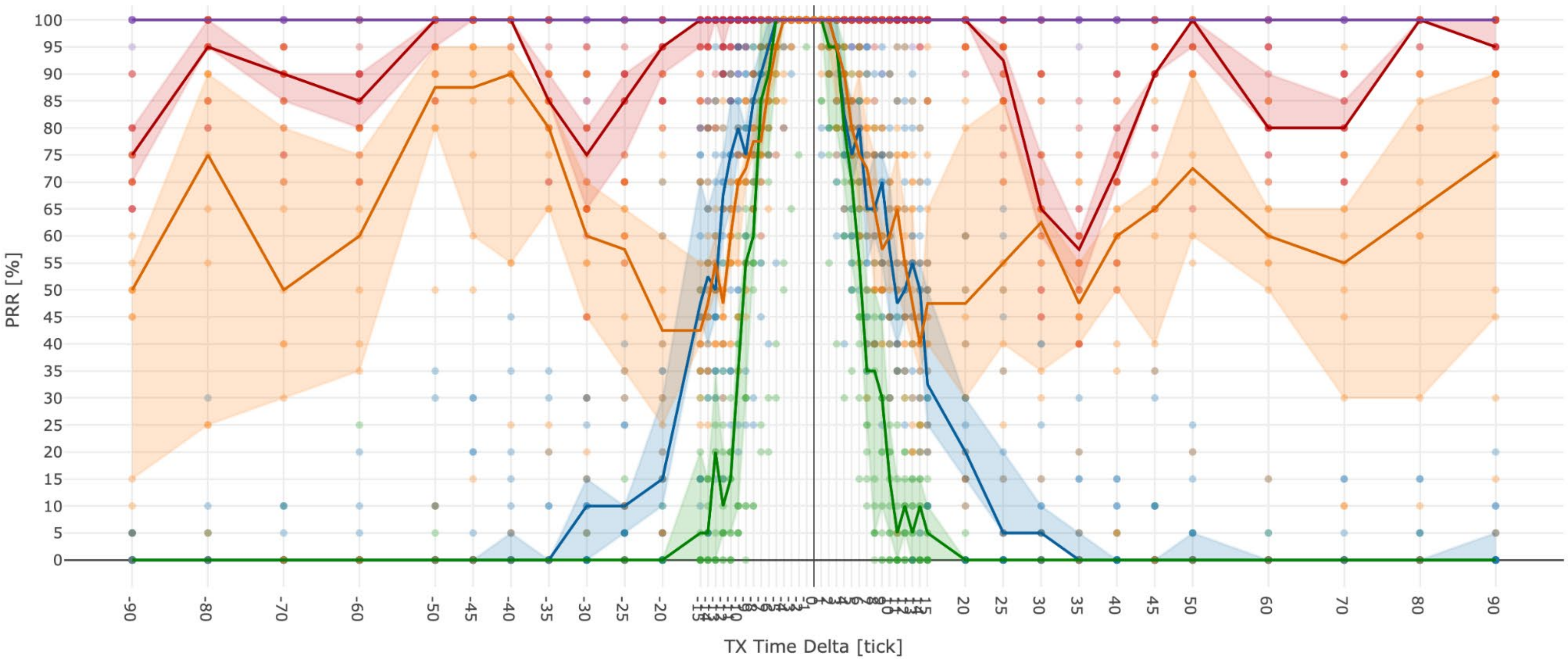
same

RSSI 3 dB



same

4 dB

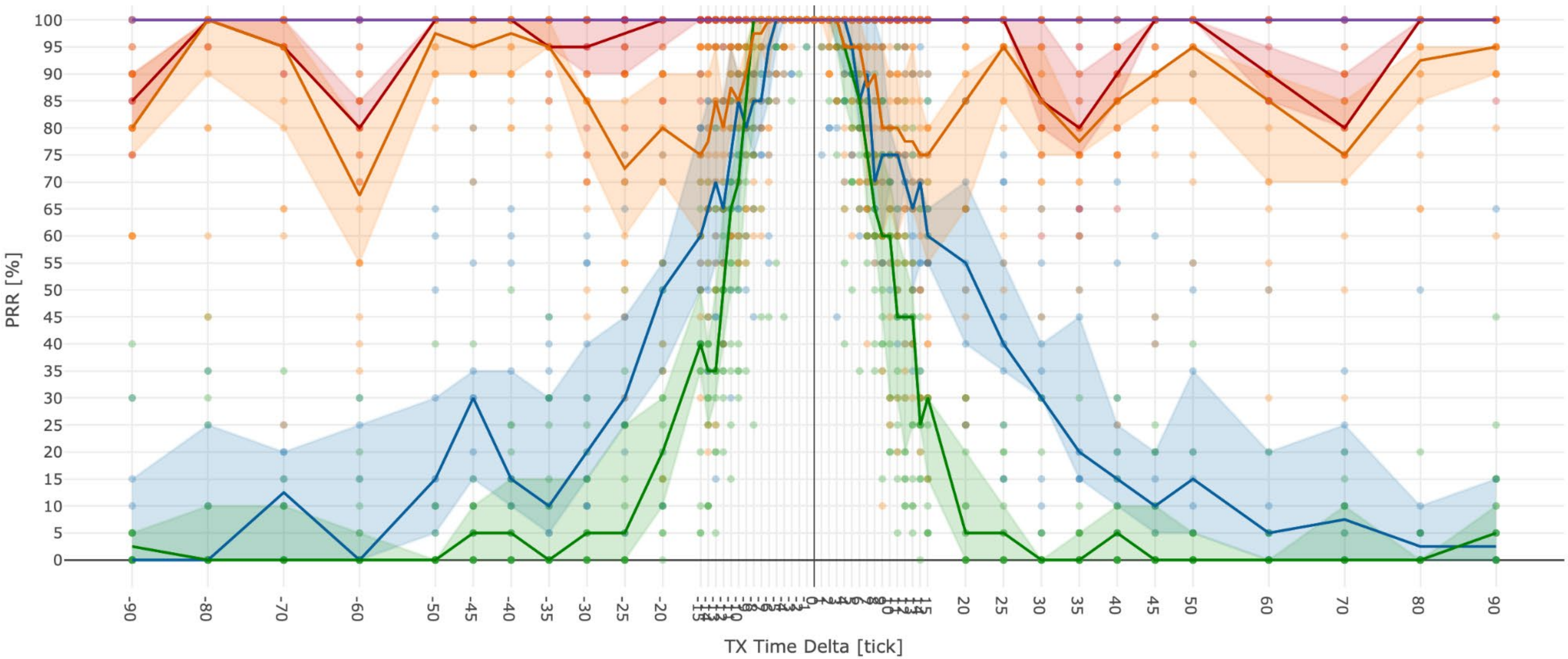


IEEE 802.15.4 | BLE 1 Mbit | BLE 2 Mbit | BLE 500 Kbit | BLE 125 Kbit

same

RSSI

5 dB



IEEE 802.15.4

BLE 1 Mbit

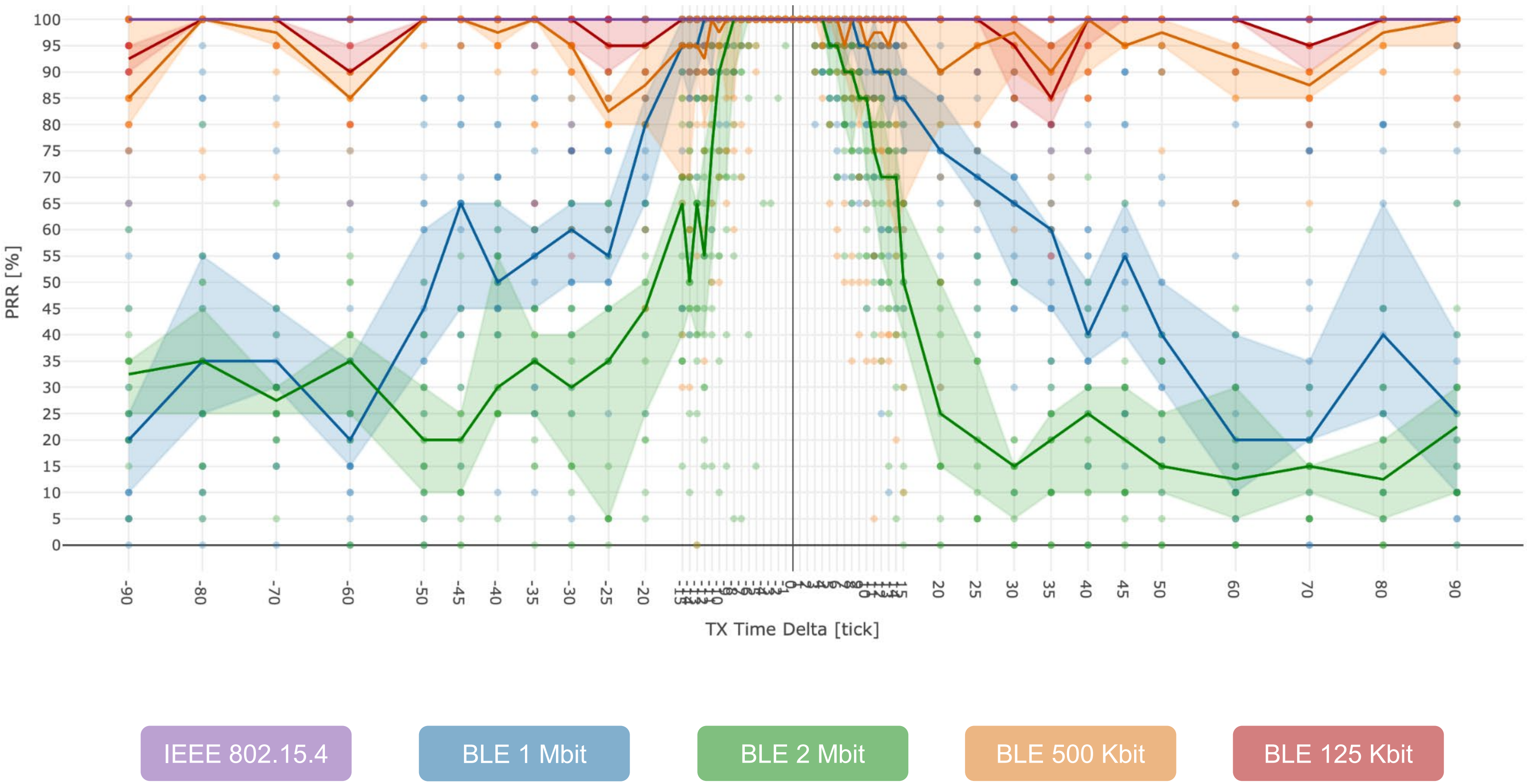
BLE 2 Mbit

BLE 500 Kbit

BLE 125 Kbit

same

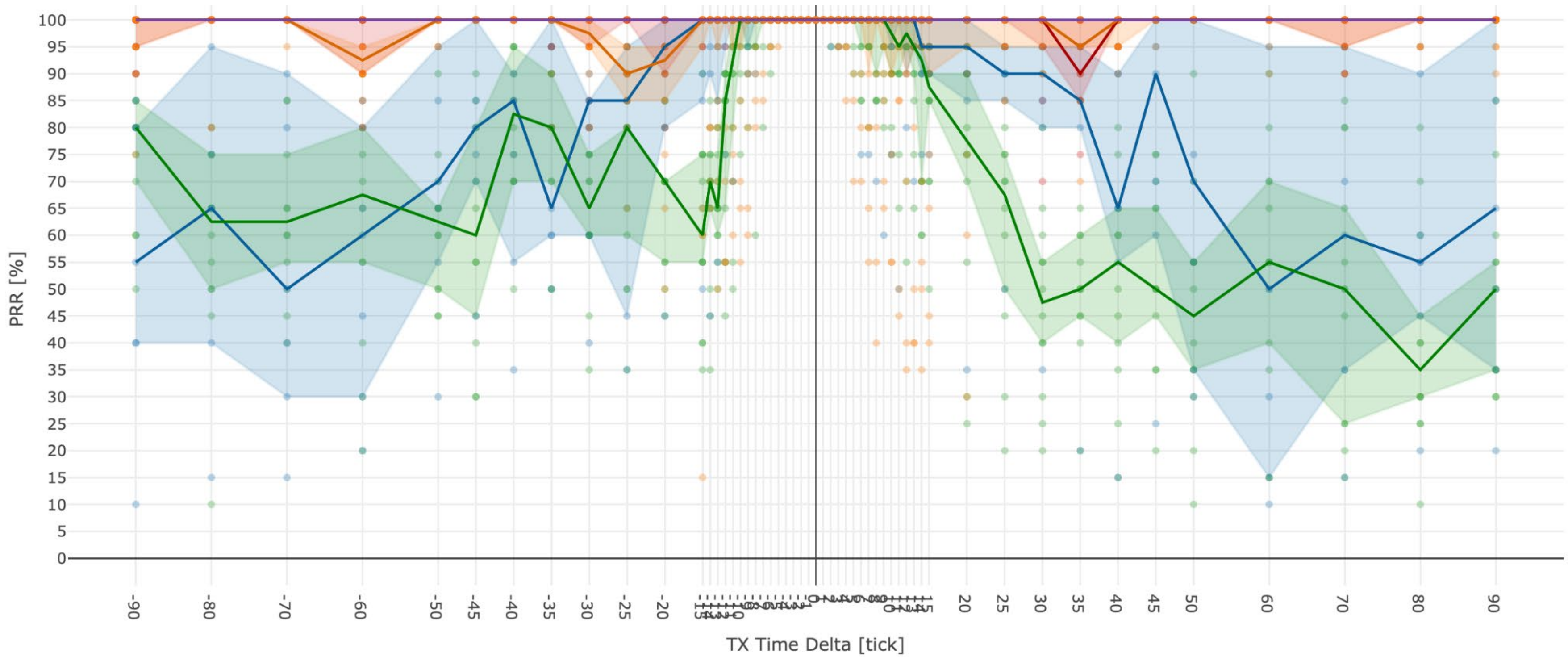
6 dB



RSSI
PRR [%]
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit

same

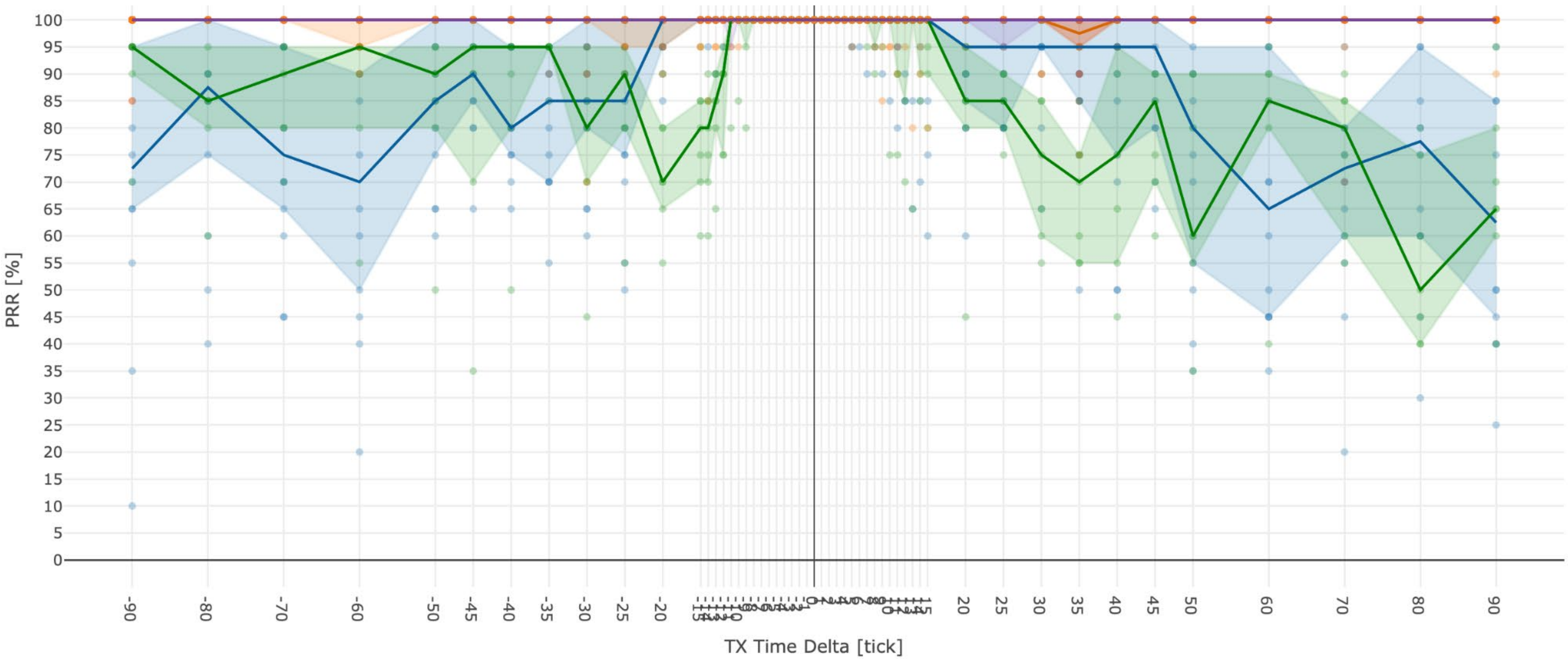
7 dB

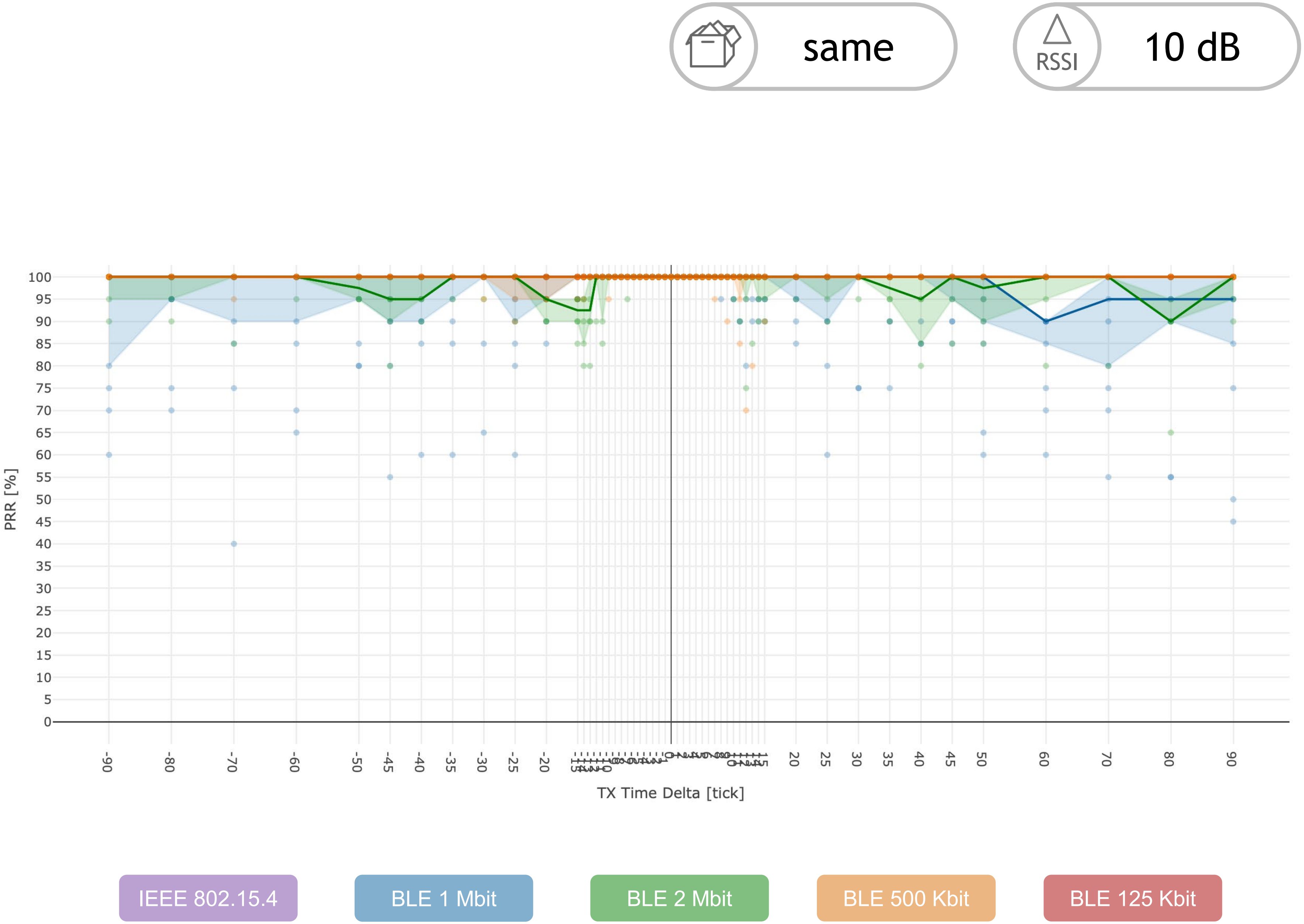


IEEE 802.15.4 BLE 1 Mbit BLE 2 Mbit BLE 500 Kbit BLE 125 Kbit

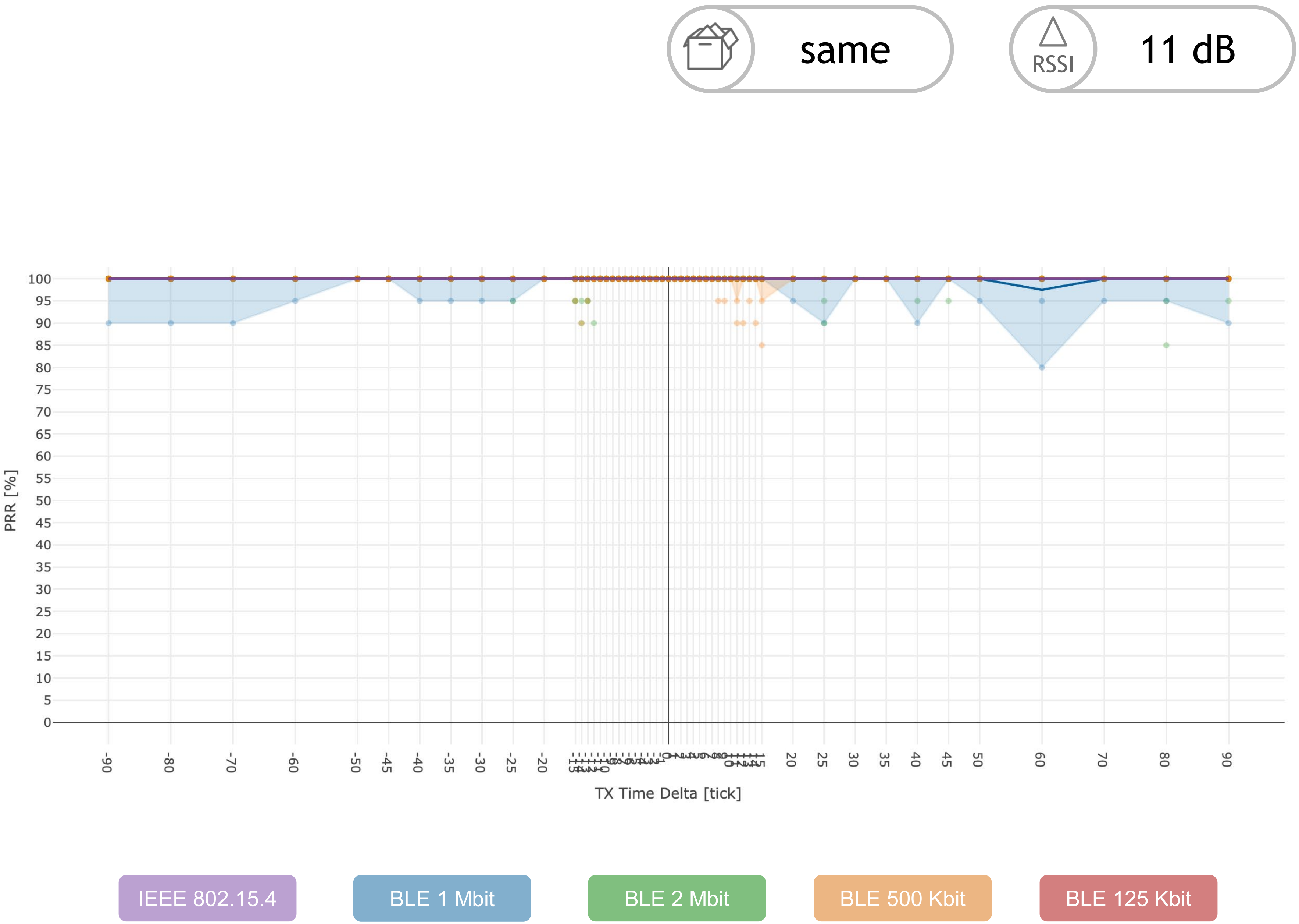
same

RSSI 8 dB

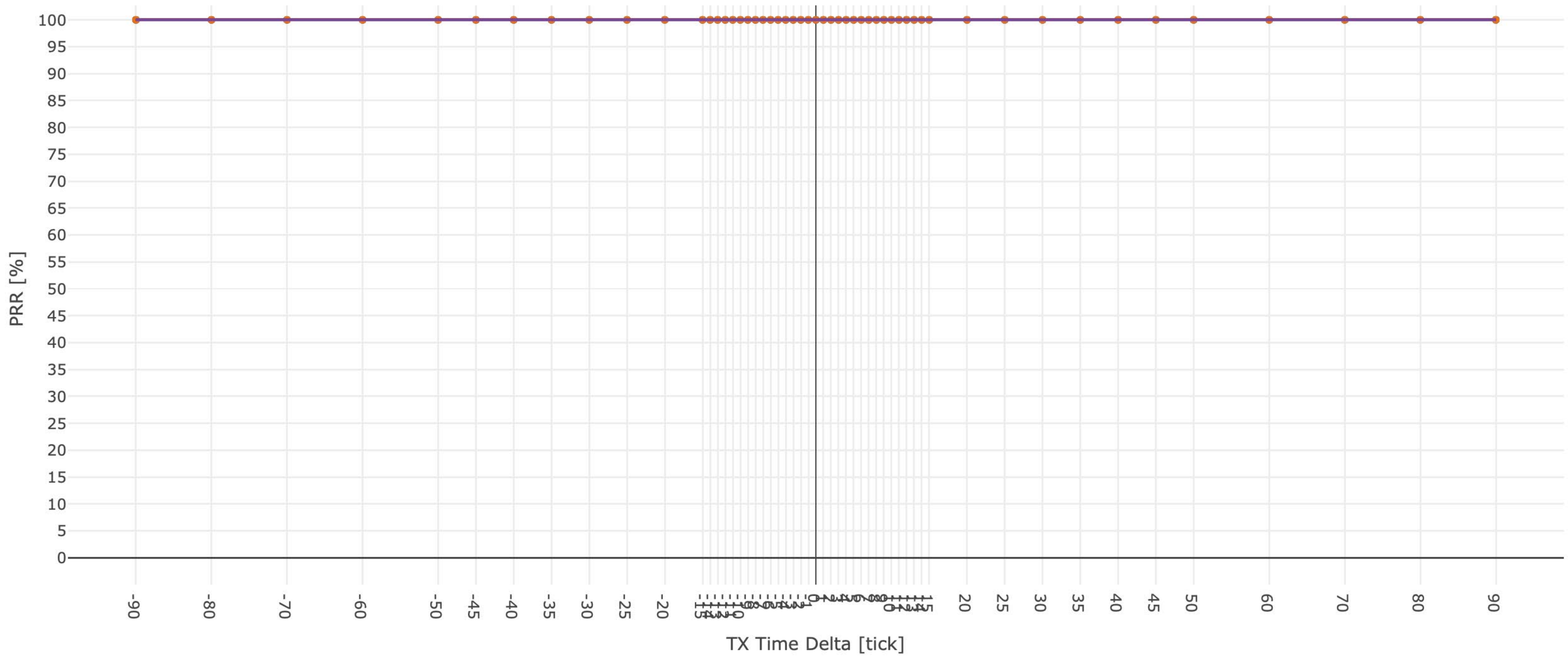




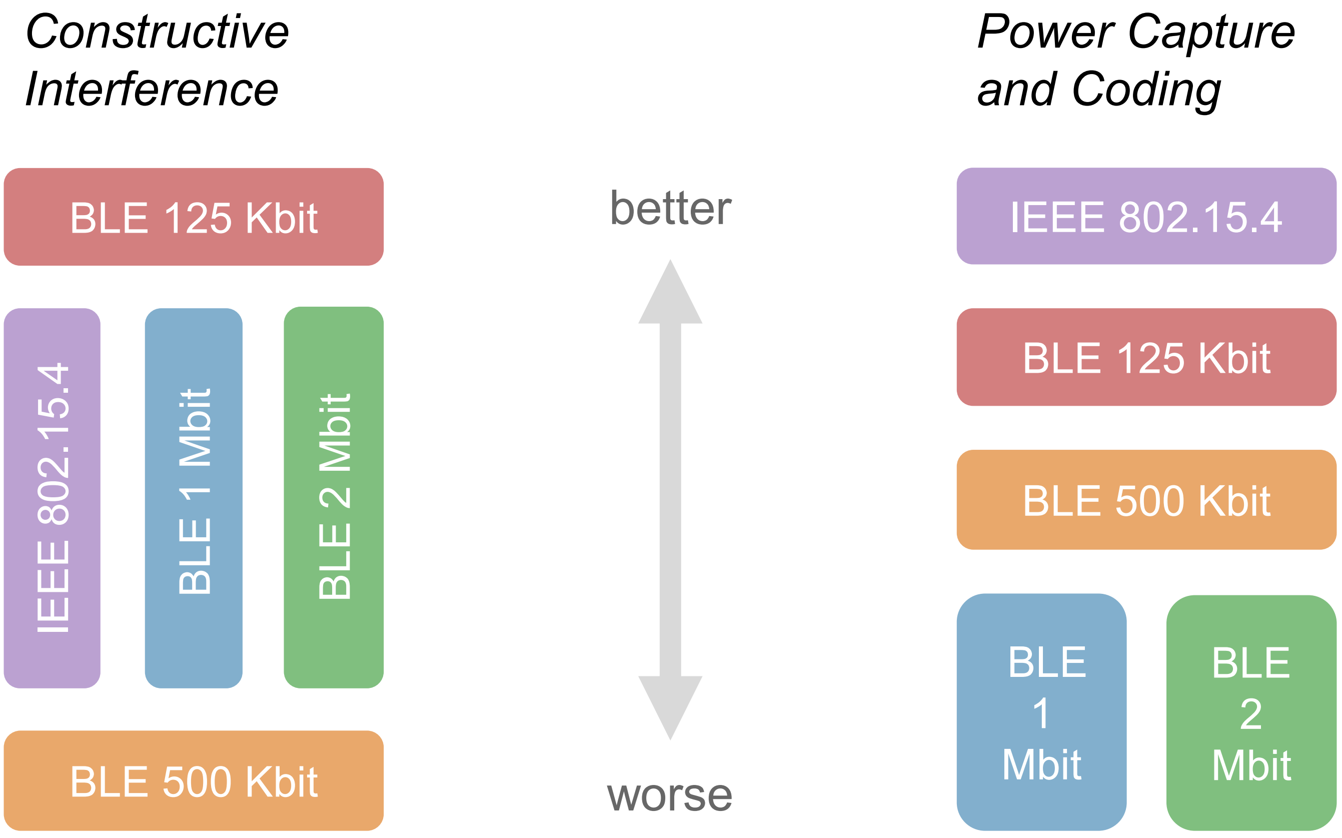
same
RSSI
10 dB
PRR [%]
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit


same
RSSI
11 dB
PRR [%]
TX Time Delta [tick]
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit


same
RSSI
18 dB


PRR [%]
100
95
90
85
80
75
70
65
60
55
50
45
40
35
30
25
20
15
10
5
0
-90
-80
-70
-60
-50
-45
-40
-35
-30
-25
-20
20
25
30
35
40
45
50
60
70
80
90
TX Time Delta [tick]

IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
BLE 125 Kbit


Constructive Interference
BLE 125 Kbit
IEEE 802.15.4
BLE 1 Mbit
BLE 2 Mbit
BLE 500 Kbit
better
worse
Power Capture and Coding
IEEE 802.15.4
BLE 125 Kbit
BLE 500 Kbit
BLE 1 Mbit
BLE 2 Mbit

	IEEE 802.15.4	BLE 1 Mbit	BLE 2 Mbit	BLE 500 Kbit	BLE 125 Kbit
Coding	DSSS	–	–	FEC S=2	FEC S=8
Time Delta	0.5 µs τ_s	1 µs τ_s	0.5 µs τ_s	0.25 µs $\tau_s/4$	1 µs τ_s
Power Delta	5 dB	13 dB	11 dB	11 dB	9 dB

Low-Power System Design

NETWORK TIME SYNCHRONIZATION BASICS

Clock Synchronization in Networks?

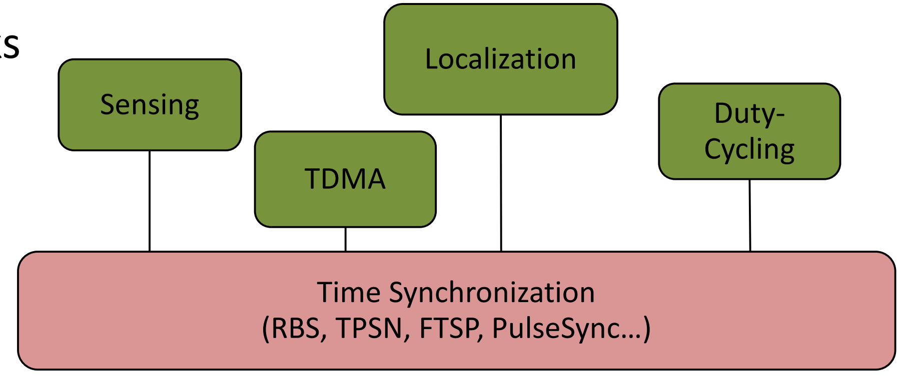
- *Time, Clocks, and the Ordering of Events in a Distributed System.* L. Lamport, Communications of the ACM, 1978.
- ***Internet Time Synchronization: The Network Time Protocol (NTP).* D. Mills, IEEE Transactions on Communications, 1991**
- *Reference Broadcast Synchronization (RBS).* J. Elson, L. Girod and D. Estrin, OSDI 2002
- *Timing-sync Protocol for Sensor Networks (TPSN).* S. Ganeriwal, R. Kumar and M. Srivastava, SenSys 2003
- ***Flooding Time Synchronization Protocol (FTSP).* M. Maróti, B. Kusy, G. Simon and Á. Lédeczi, SenSys 2004**
- and many more ...

Time in Sensor Networks

- Synchronizing time is essential for many applications
  - Coordination of wake-up and sleeping times (energy efficiency)
  - TDMA schedules
  - Ordering of collected sensor data/events
  - Co-operation of multiple sensor nodes
  - Estimation of position information (e.g. shooter detection)

- Goals of clock synchronization
  - Compensate *offset** between clocks
  - Compensate *drift** between clocks

*terms are explained on following slides



Properties of Clock Synchronization Algorithms

- External versus internal synchronization
  - External sync: Nodes synchronize with an external clock source (UTC)
  - Internal sync: Nodes synchronize to a common time
    - to a leader, to an averaged time, or to anything else
- One-shot versus continuous synchronization
  - Periodic synchronization required to compensate clock drift
- A-priori versus a-posteriori
  - A-posteriori clock synchronization triggered by an event
- Global versus local synchronization
- Accuracy versus convergence time, Byzantine nodes, …

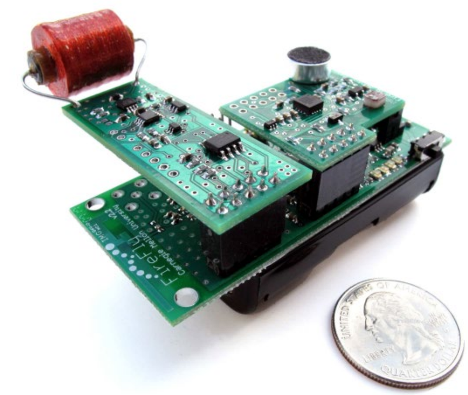
Global Clock Sources

- Radio Clock Signal
  - Clock signal from a reference source (atomic clock) is transmitted over a long wave radio signal
  - DCF77 station near Frankfurt, Germany transmits at 77.5 kHz with a transmission range of up to 2000 km
  - Accuracy limited by the distance to the sender, Frankfurt-Zurich is about 1ms.
  - Special antenna/receiver hardware required

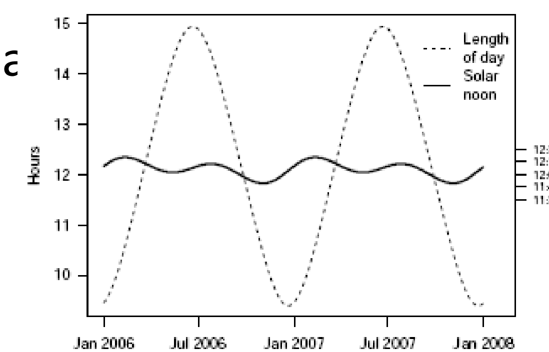
- Global Positioning System (GPS)
  - Satellites continuously transmit own position and time code
  - Line of sight between satellite and receiver required
  - Special antenna/receiver hardware required

Global Clock Sources (2)

- AC power lines
  - Use the magnetic field radiating from electric AC power lines
  - AC power line oscillations are extremely stable (10^{-8} ppm)
  - Power efficient, consumes only 58 µW
  - Single communication round required to correct phase offset after initialization



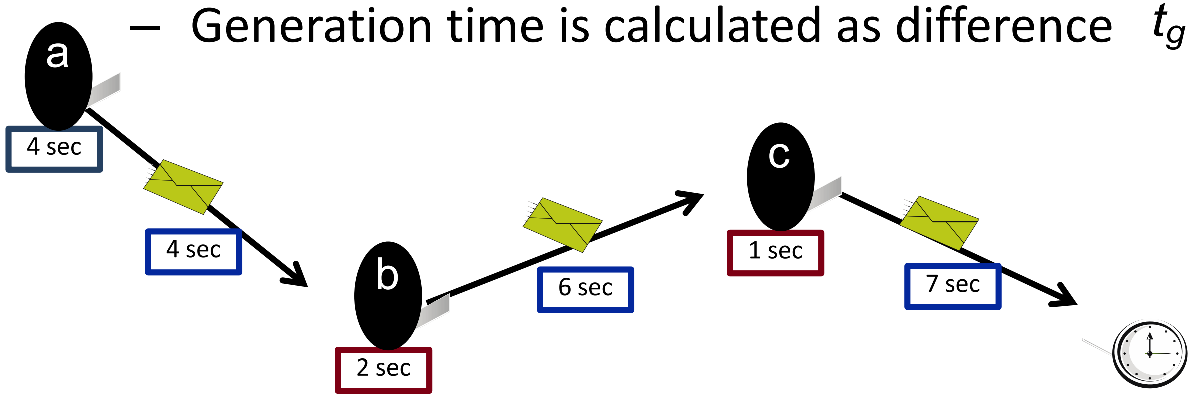
- Sensor Signals (Sunlight)
  - Using a light sensor to measure the length of a da
  - Offline algorithm for reconstructing global timestamps by correlating annual solar patterns (no communication required)



Global vs. Local Time Sync

- In cases where no network-wide time synchronization is available
  - Global time sync not available for network protocol control
  - Implications on data usage

- Solution: Elapsed time on arrival
  - Sensor nodes measure/accumulate packet sojourn time
  - Base station annotates packets with UTC timestamps
  - Generation time is calculated as difference $\tilde{t}_g = t_b - \tilde{t}_s$



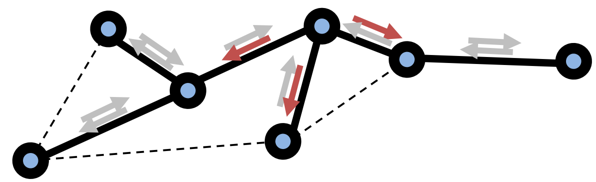
2011/04/14 10:03:31 – 7 sec
= 2011/04/14 10:03:24

Network Time Synchronization

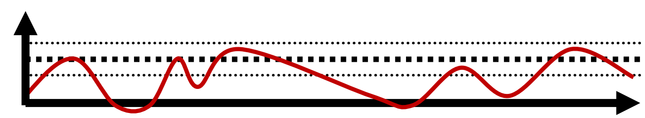
- Goal

  Send time information (beacons) across network to synchronize clocks

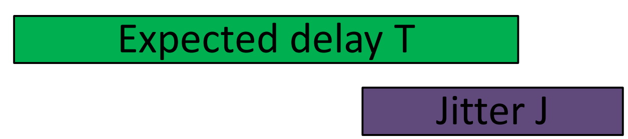
- Problems
  - Network ensemble **interactions**



  - Hardware clocks exhibit **drift**

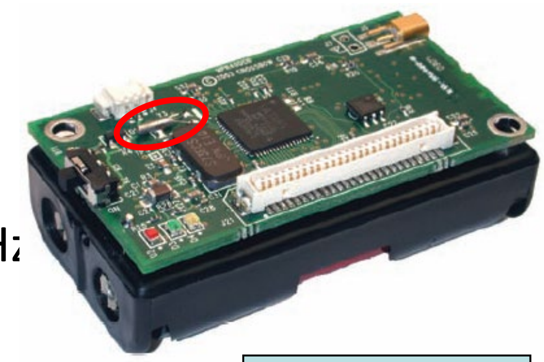


  - **Jitter** in the message delay



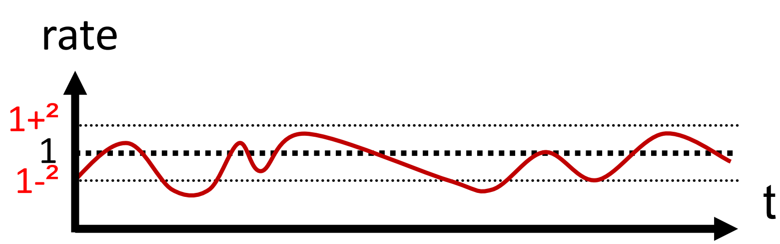
Hardware Clocks Experience Drift

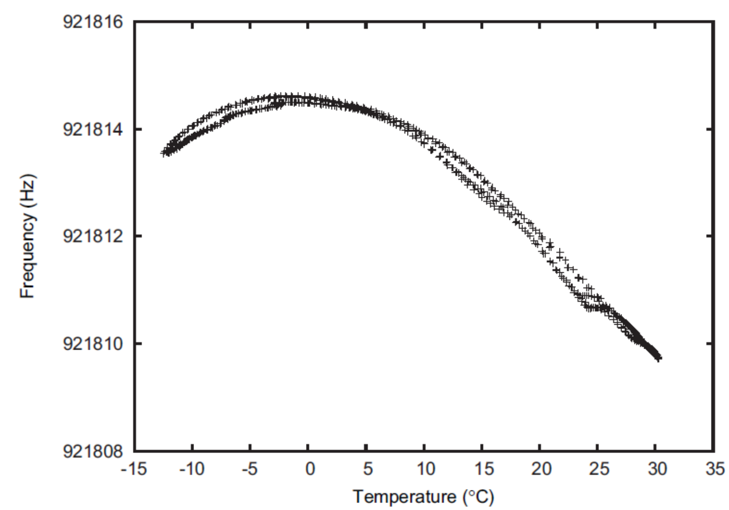
- Hardware clock
  - Counter register of the microcontroller
  - Sourced by an external crystal (32kHz, 7.37 MHz



- Clock drift
  - Random deviation from the nominal rate dependent on ambient temperature, power supply, etc. (30-100 ppm)

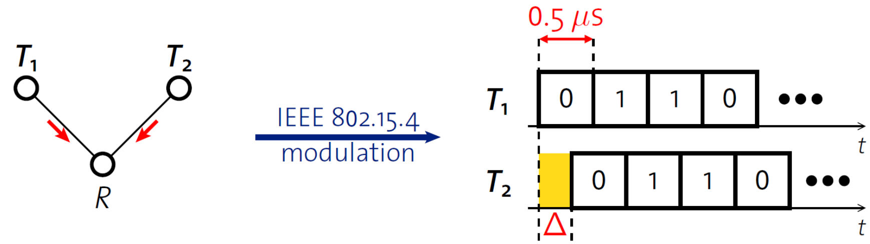
This is a drift of up to 50 μs per second or 0.18s per hour





Example Glossy and Timing

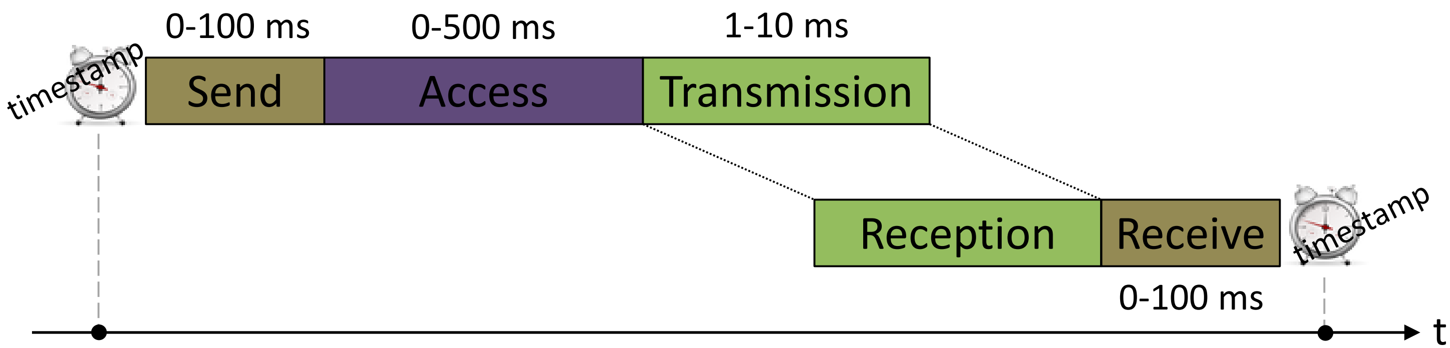
- Remember



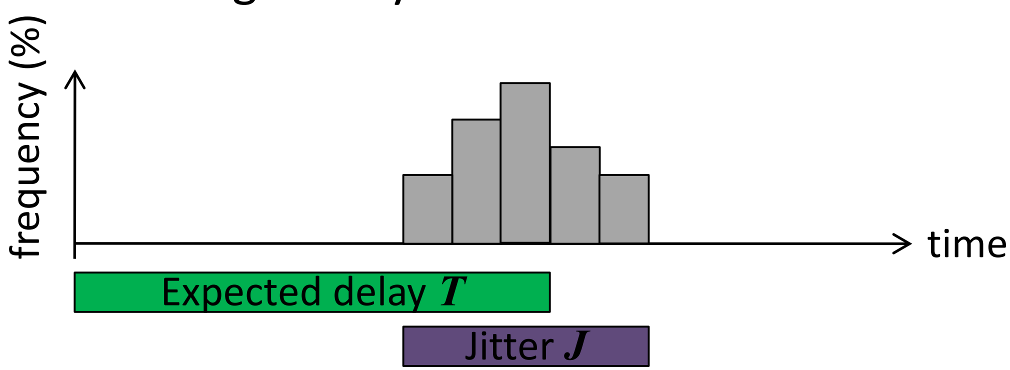
- R receives packet with high probability if Δ ≤ 0.5 µs
- 32.768 kHz @ 10/20/50ppm -> $\frac{1}{32768\ kHz} = 30.5\ \mu sec$
- +/-20 ppm results in 32.7673 to 32.7687 kHz
  $32768\ \pm 20\ ppm$ is x 0.999980 to x 1.000020
- 1 Month = 60x60*24*30 = 2.6 million seconds
  20 ppm crystal for wakeup results in error 1 min per month

Messages Delays Experience Jitter

- Problem: Jitter in the message delay
  - Various sources of errors (deterministic and non-deterministic)

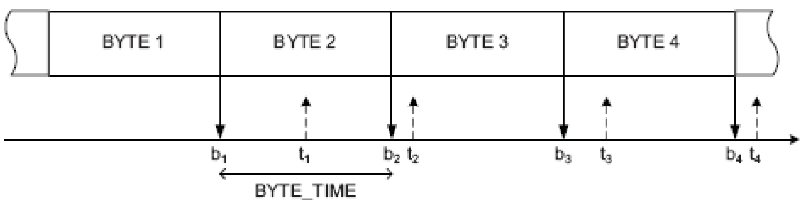


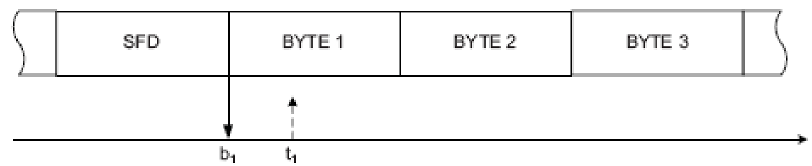
- Solution: Timestamping packets at the MAC layer
  - Jitter in the message delay is reduced to a few clock ticks



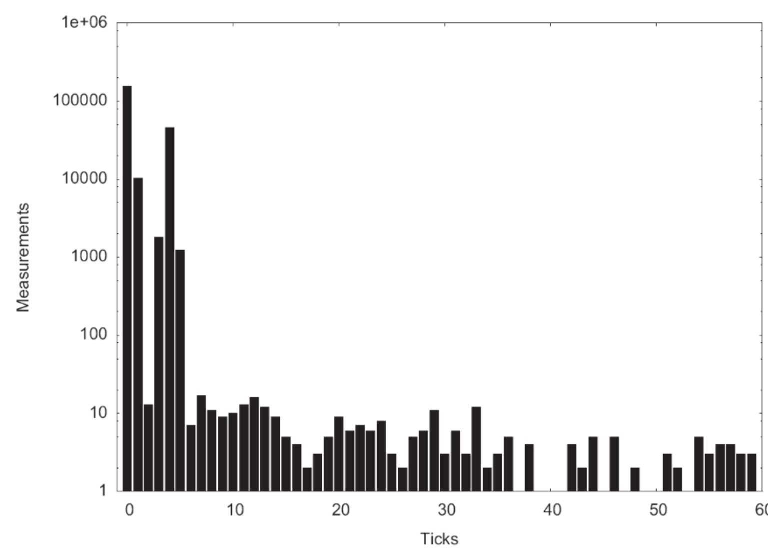
Jitter Practical Details

- Different radio chips use different paradigms
  - Left is a CC1000 radio chip which generates an interrupt with each byte
  - Right is a CC2420 radio chip that generates a single interrupt for the packet after the start frame delimiter is received



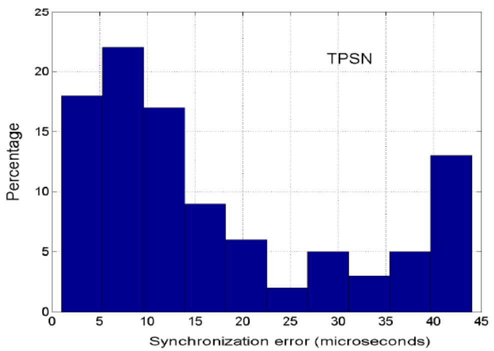


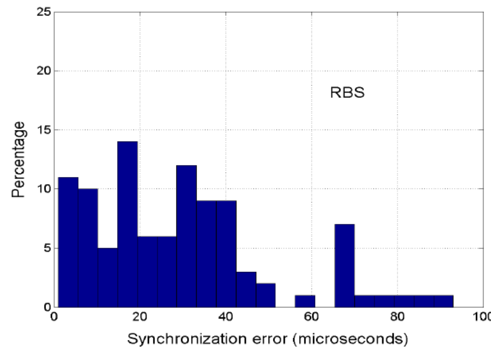
- Still there is quite some variance in transmission delay because of latencies in **interrupt handling**

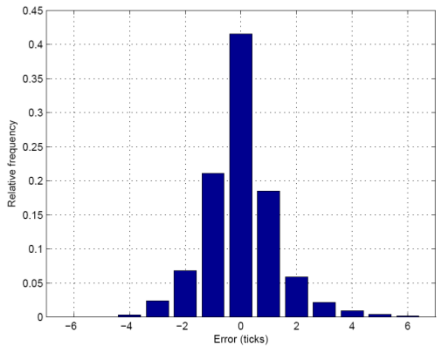


Symmetric Errors

- Many protocols don't even handle single-hop clock synchronization well. On the left figures we see the absolute synchronization errors of TPSN and RBS, respectively. The figure on the right presents a single-hop synchronization protocol minimizing systematic errors







- Even perfectly symmetric errors will sum up over multiple hops
  - In a chain of *n* nodes with a standard deviation σ on each hop, the expected error between head and tail of the chain is in the order of $cumulative\ error = \sigma\sqrt{n}$

Influence Factors

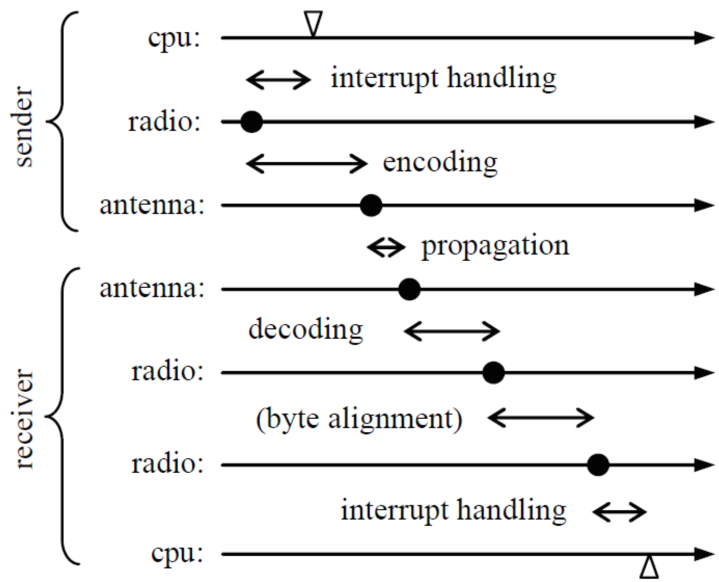


Table 1. The sources of delays in message transmissions

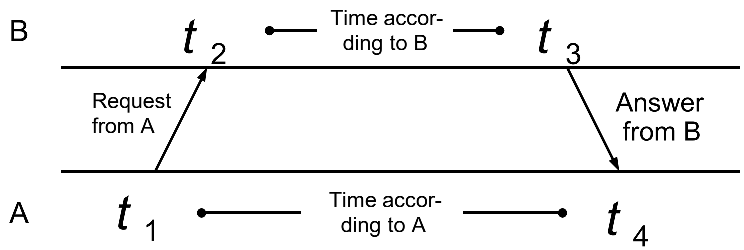
Time	Magnitude	Distribution
Send and Receive	0 – 100 ms	nondeterministic, depends on the processor load
Access	10 – 500 ms	nondeterministic, depends on the channel contention
Transmission / Reception	10 – 20 ms	deterministic, depends on message length
Propagation	< 1μs for distances up to 300 meters	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

Low-Power System Design

NETWORK TIME SYNCHRONIZATION ALGORITHMS

Sender/Receiver Synchronization

- Round-Trip Time (RTT) based synchronization



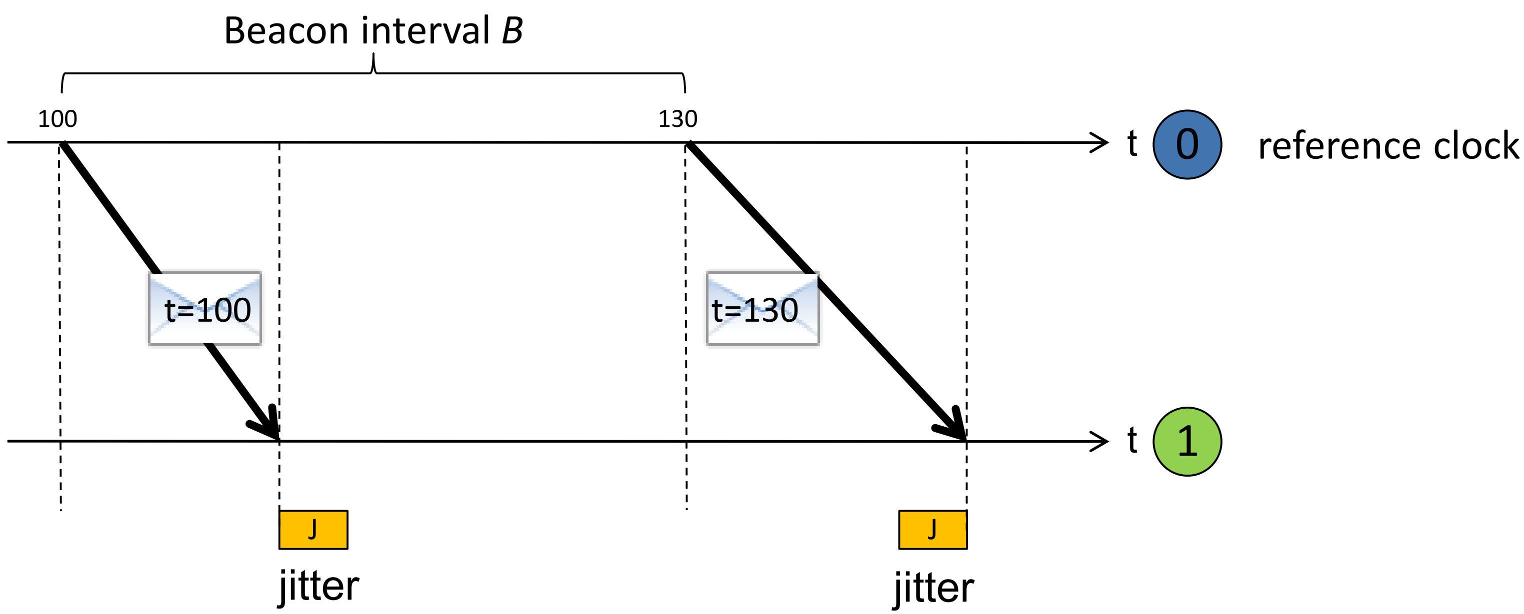
- Receiver synchronizes to the sender‘s clock
- Propagation delay δ and clock offset θ can be calculated

$$\delta = \frac{(t_4 - t_1) - (t_3 - t_2)}{2}$$

$$\theta = \frac{(t_2 - (t_1 + \delta)) - (t_4 - (t_3 + \delta))}{2} = \frac{(t_2 - t_1) + (t_3 - t_4)}{2}$$

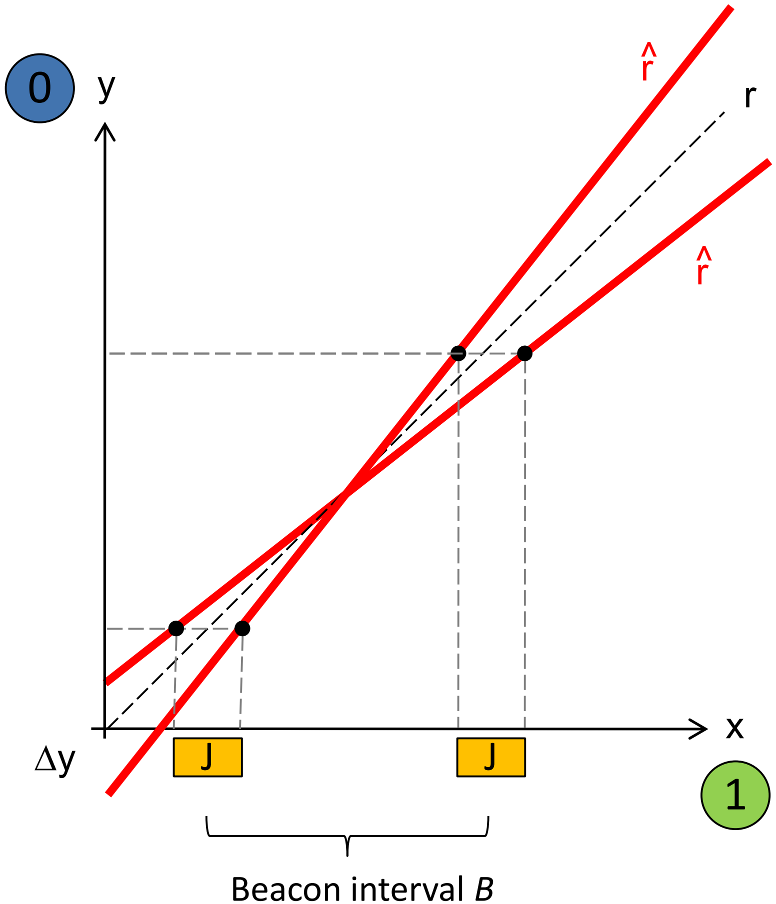
Synchronizing Nodes

- Sending periodic beacon messages to synchronize nodes
- Payload contains local time information

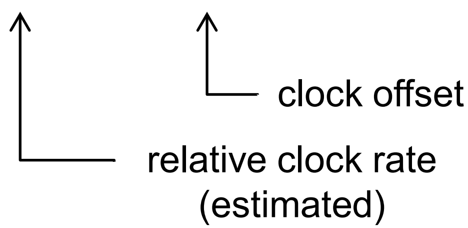


How Accurately Can We Synchronize?

- Message delay jitter affects clock synchronization quality

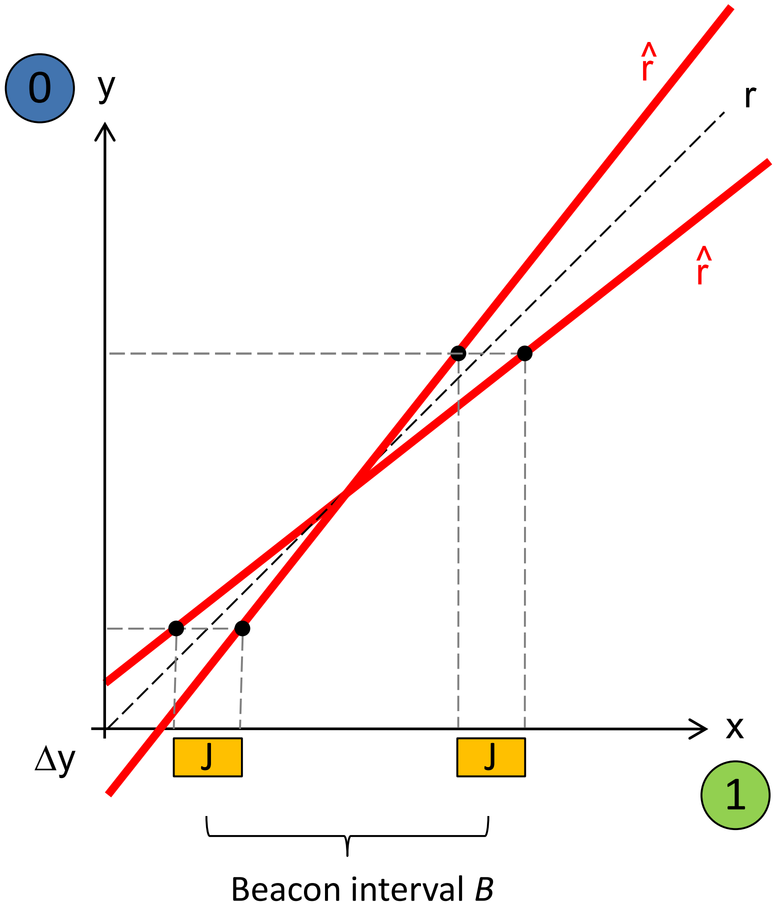


$$y(x) = \hat{r} \cdot x + \Delta y$$



Clock Skew between two Nodes

- Lower Bound on the clock skew between two neighbors



Error in the rate estimation:

- Jitter in the message delay
- Beacon interval
- Number of beacons k

$$|\hat{r} - r| \sim \frac{J}{Bk\sqrt{k}}$$

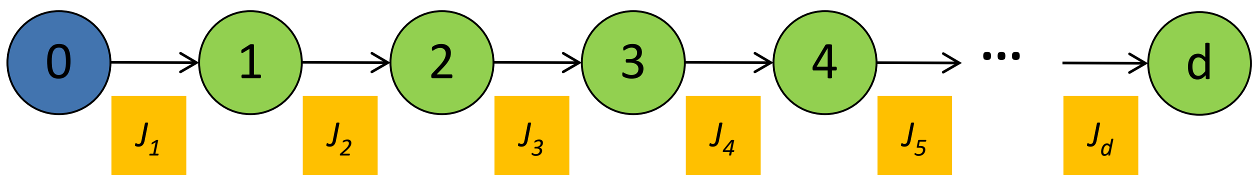
Synchronization error:

$$|\hat{y} - y| \sim \frac{J}{\sqrt{k}}$$

Multi-hop Clock Skew

- Nodes forward their current estimate of the reference clock

  Each synchronization beacon is affected by a **random jitter *J***



- Sum of the jitter grows with the square-root of the distance

$$stddev(J_1 + J_2 + J_3 + J_4 + J_5 + \ldots Jd) = \sqrt{d} \times stddev(J)$$

Single-hop:

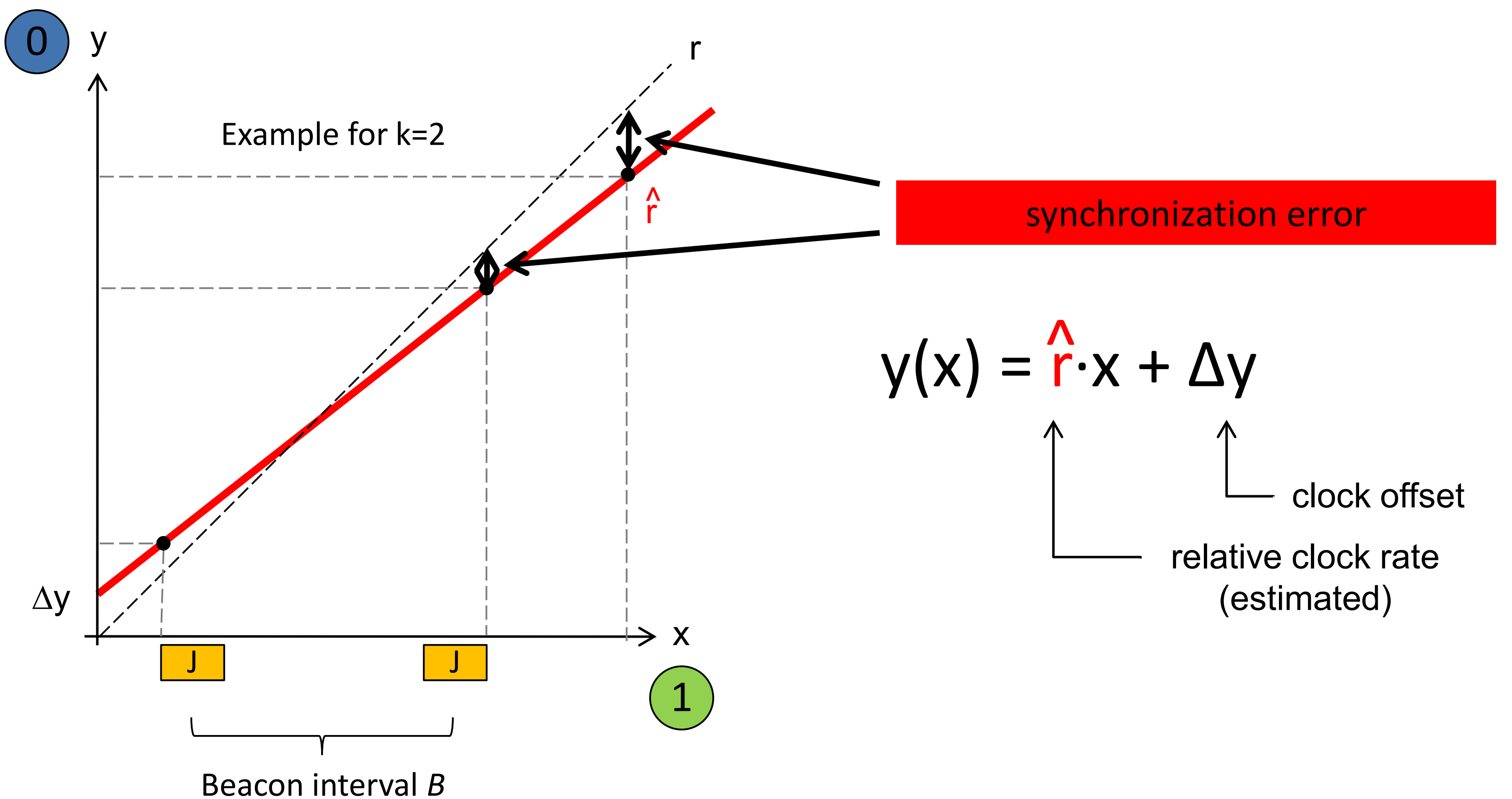
$$|\hat{y} - y| \sim \frac{J}{\sqrt{k}}$$

Multi-hop:

$$|\hat{y} - y| \sim \frac{J\sqrt{d}}{\sqrt{k}}$$

Error Mitigation: Linear Regression

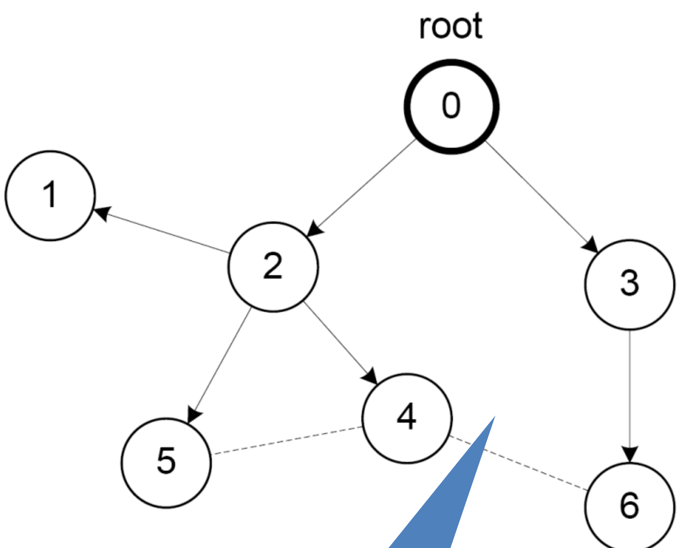
- FTSP uses linear regression to compensate for clock drift
  - Jitter is amplified before it is sent to the next hop



Clock Synchronization Algorithms

Tree-like Algorithms

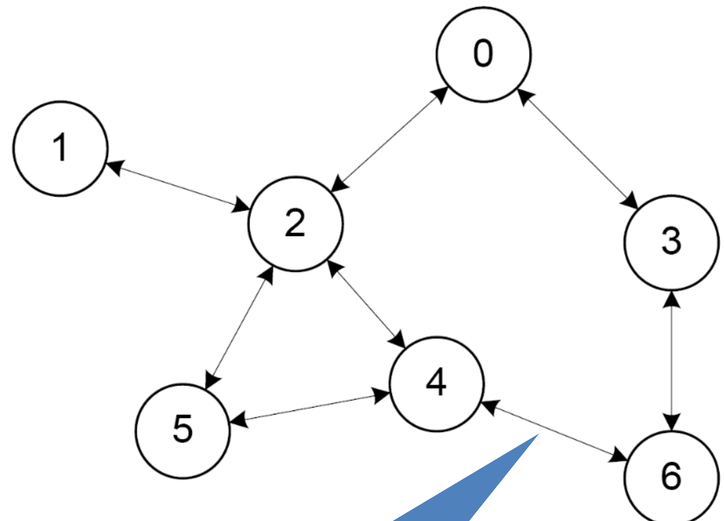
e.g. FTSP



Bad local skew

Distributed Algorithms

e.g. GTSP

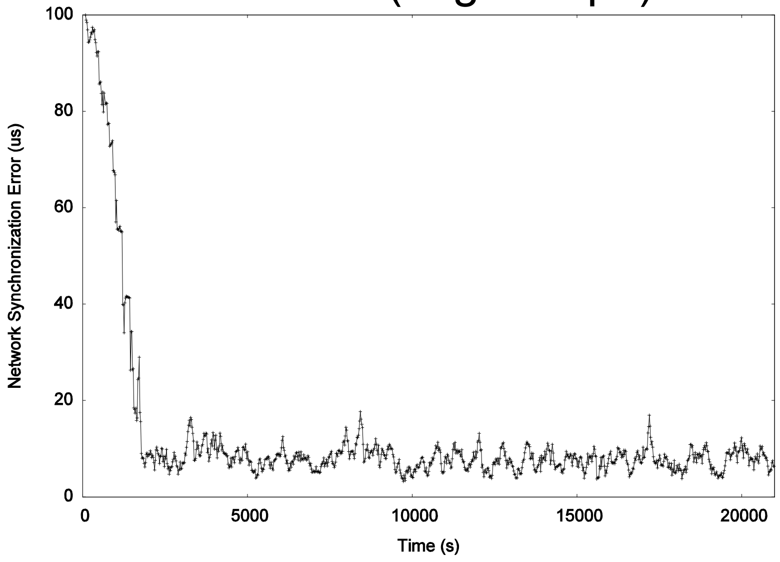


All nodes consistently average errors to *all* neighbors

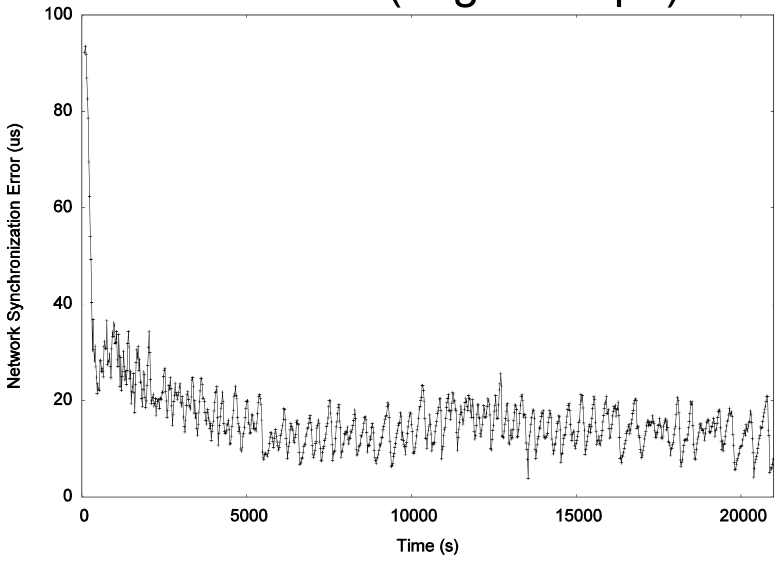
FTSP vs. GTSP: Global Skew

- Network synchronization error (global skew)
  - Pair-wise synchronization error between any two nodes in the network

FTSP (avg: 7.7 µs)



GTSP (avg: 14.0 µs)

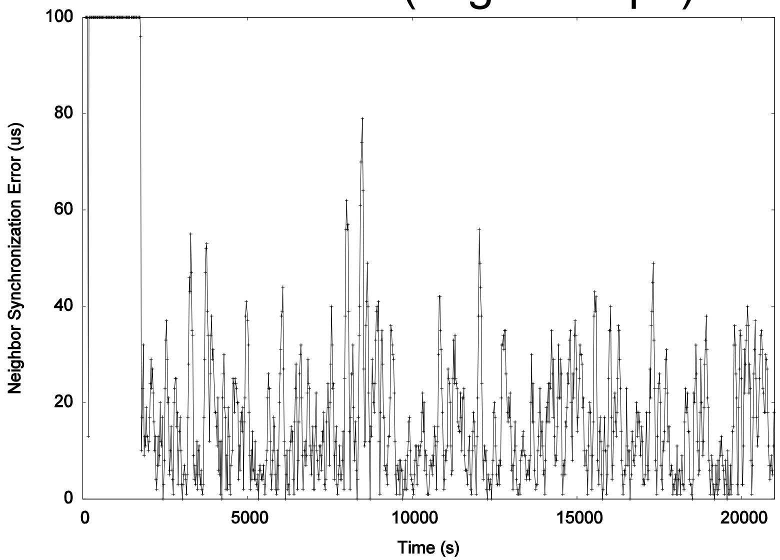


FTSP vs. GTSP: Local Skew

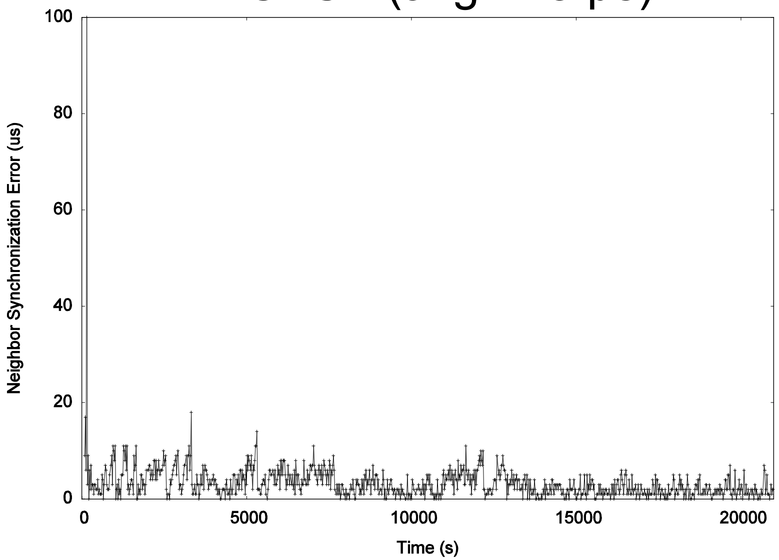
- Neighbor Synchronization error (local skew)
  - Pair-wise synchronization error between neighboring nodes

- Synchronization error between two direct neighbors:

FTSP (avg: 15.0 μs)



GTSP (avg: 2.8 μs)

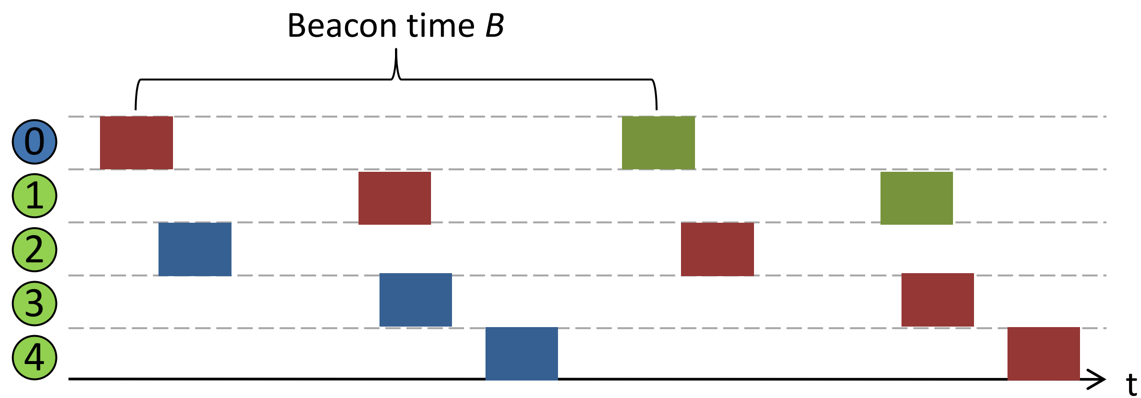


The PulseSync Protocol

- Send fast synchronization pulses through the network
  - Speed-up the initialization phase
  - Faster adaptation to changes in temperature or network topology

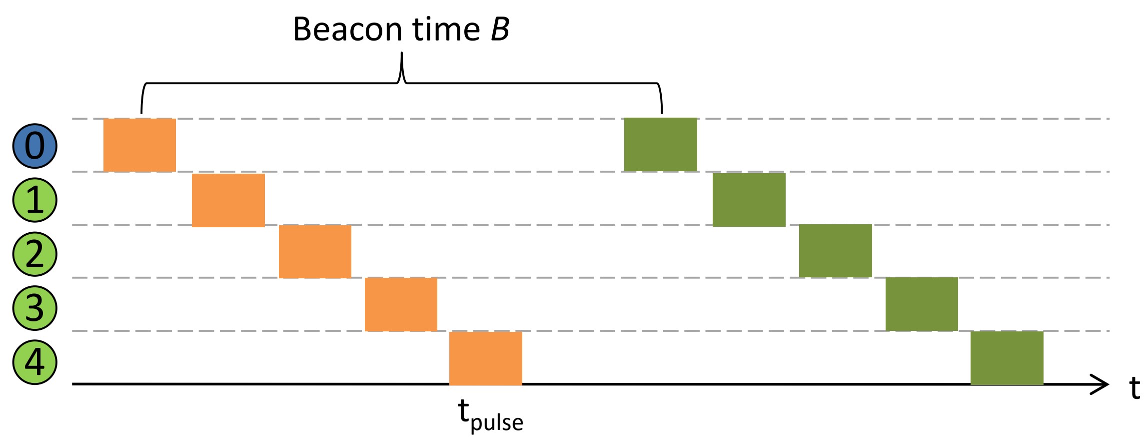
FTSP

Expected time $= D \cdot B/2$



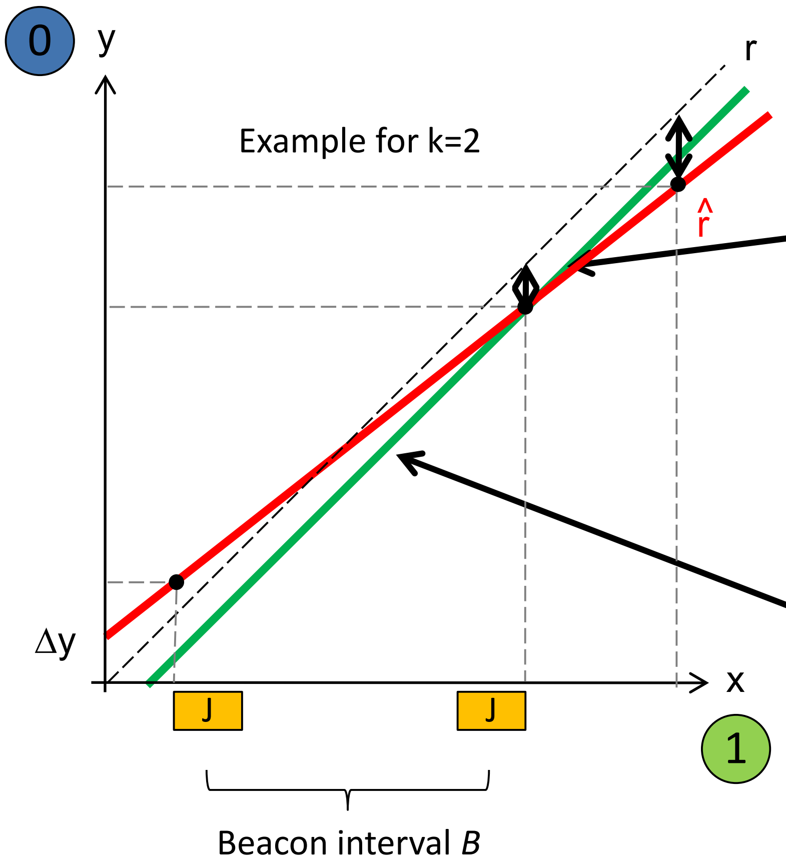
PulseSync

Expected time $= D \cdot t_{pulse}$



The PulseSync Protocol (2)

- Remove self-amplification of synchronization error
  - Fast flooding cannot completely eliminate amplification



synchronization error

$$y(x) = \hat{r} \cdot x + \Delta y$$

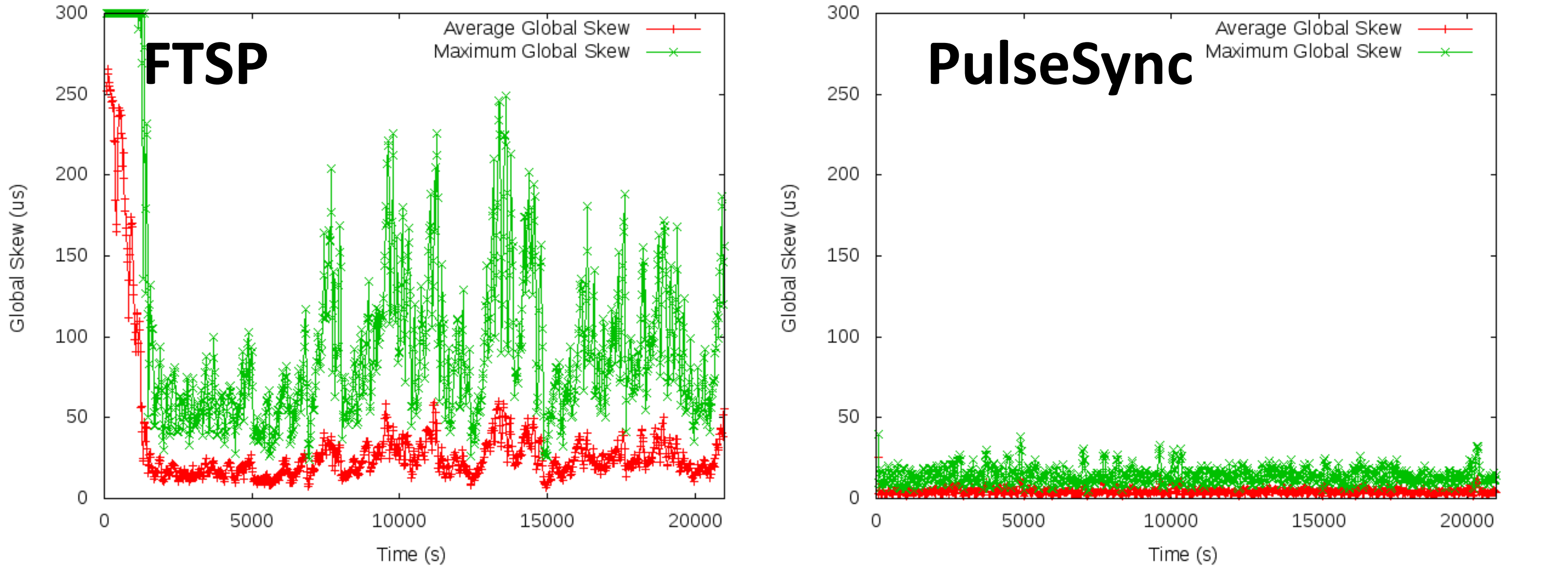
clock offset

relative clock rate (estimated)

The green line is calculated using k measurement points that are statistically *independent* of the red line.

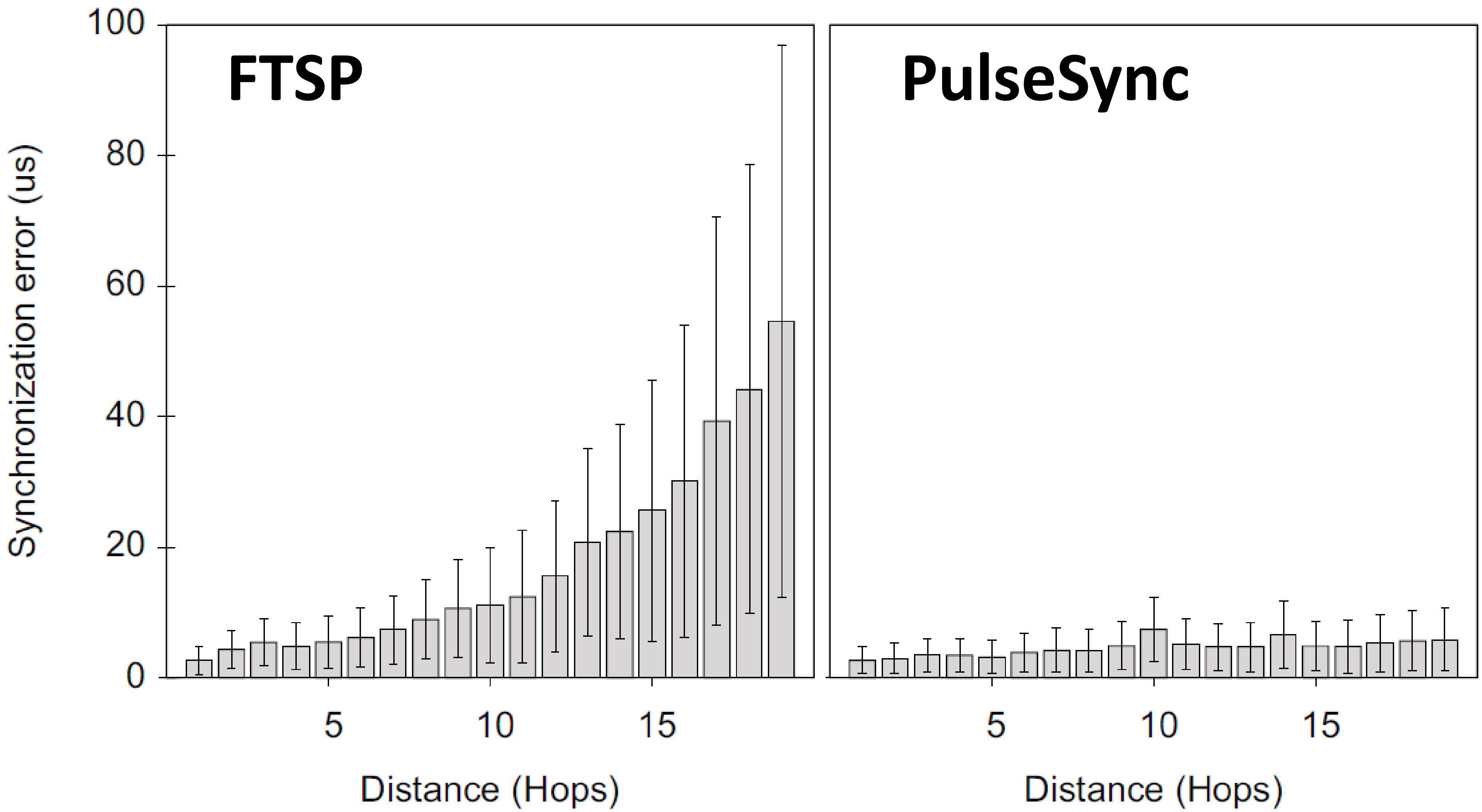
FTSP vs. PulseSync

- Global Clock Skew
  - Maximum synchronization error between any two nodes

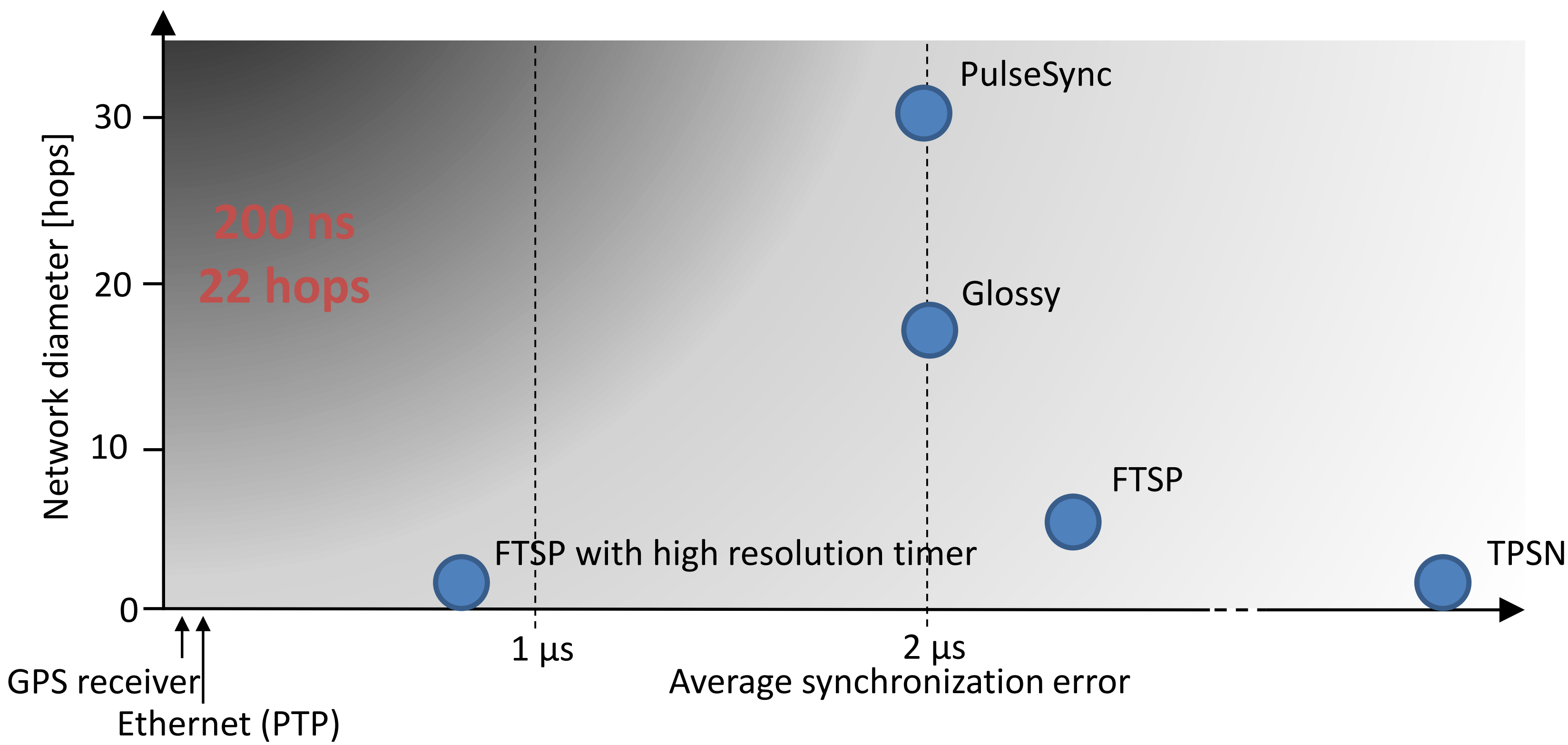


Synchronization Error	FTSP	PulseSync
Average (t>2000s)	23.96 µs	4.44 µs
Maximum (t>2000s)	249 µs	38 µs

FTSP vs. PulseSync



Wireless Multi-hop Time Synchronization



Low-Power System Design

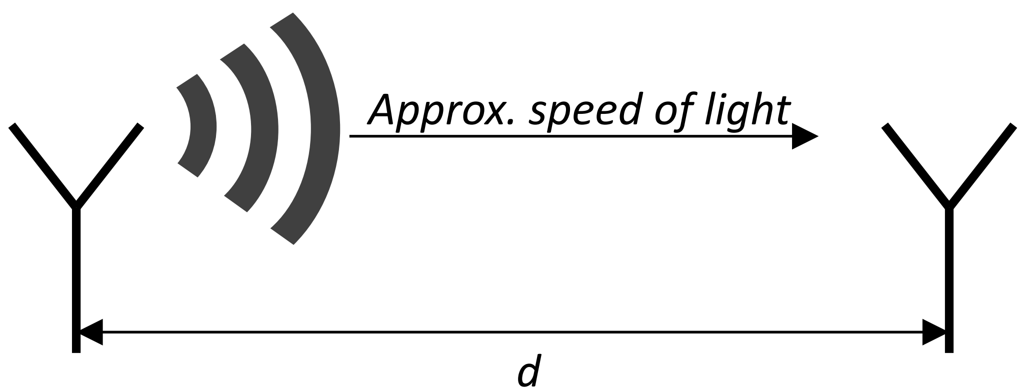
INCORPORATING TIME-OF-FLIGHT

Is Time-of-flight Really Negligible?

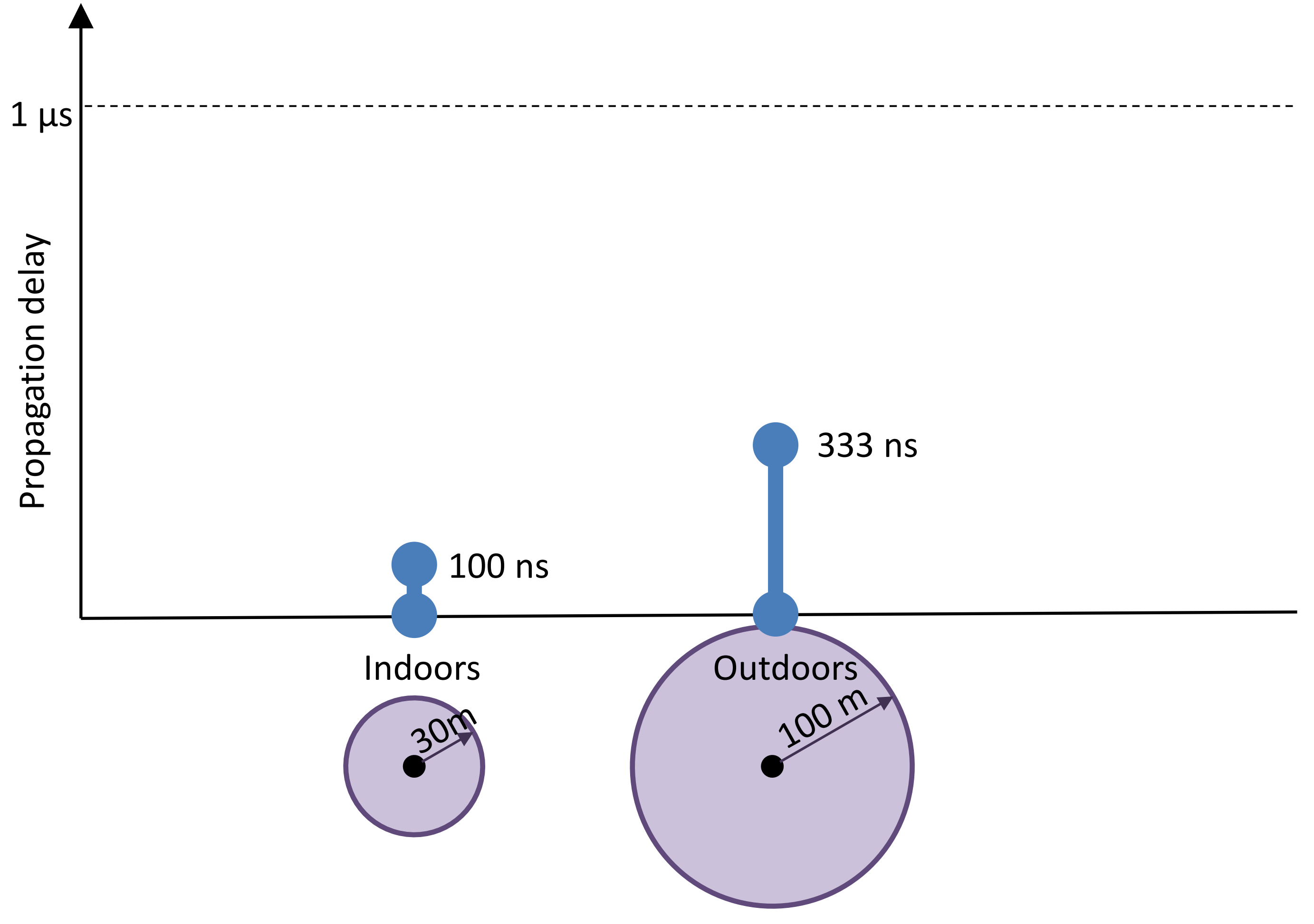
"The absolute value of this delay is negligible as compared to other sources of packet latency." [TPSN 2003]

"… it does not and cannot compensate for the propagation delay. This is not a major limitation of the approach in typical WSN…" [FTSP 2004]

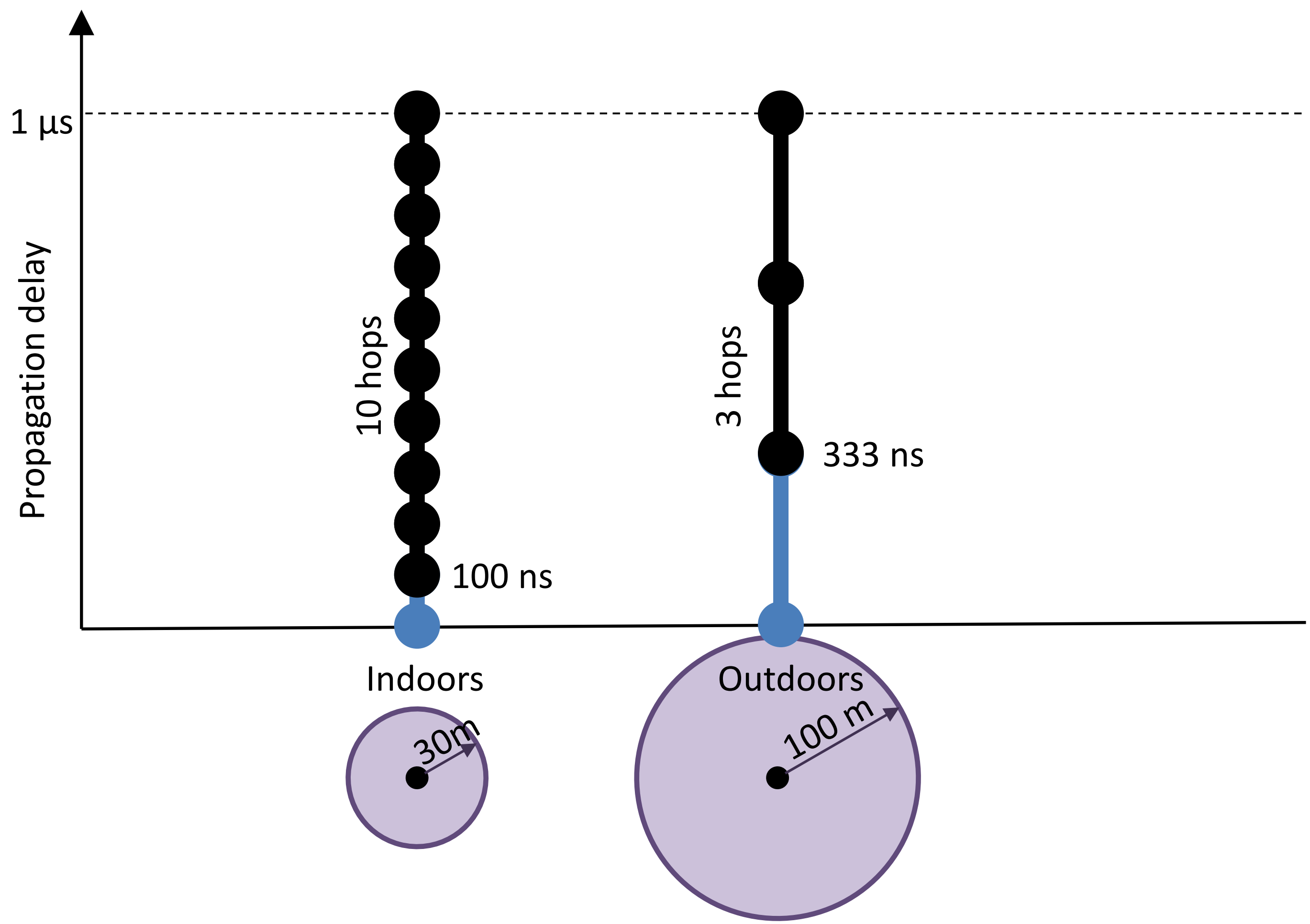
"… over short distances (less than 300 meters) its duration is negligible (less than one microsecond)." [RATS 2006]



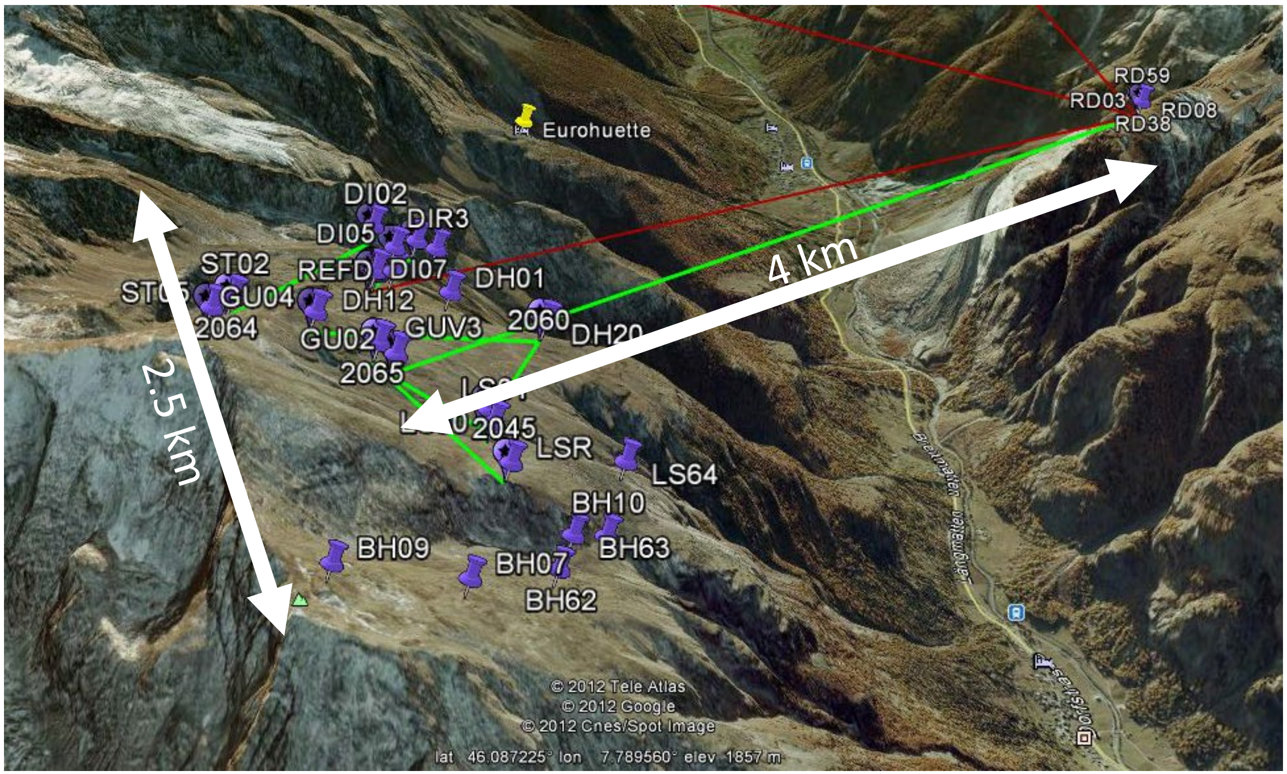
Time-of-flight Matters



Time-of-flight Matters



Outdoor Distances Might Be Long



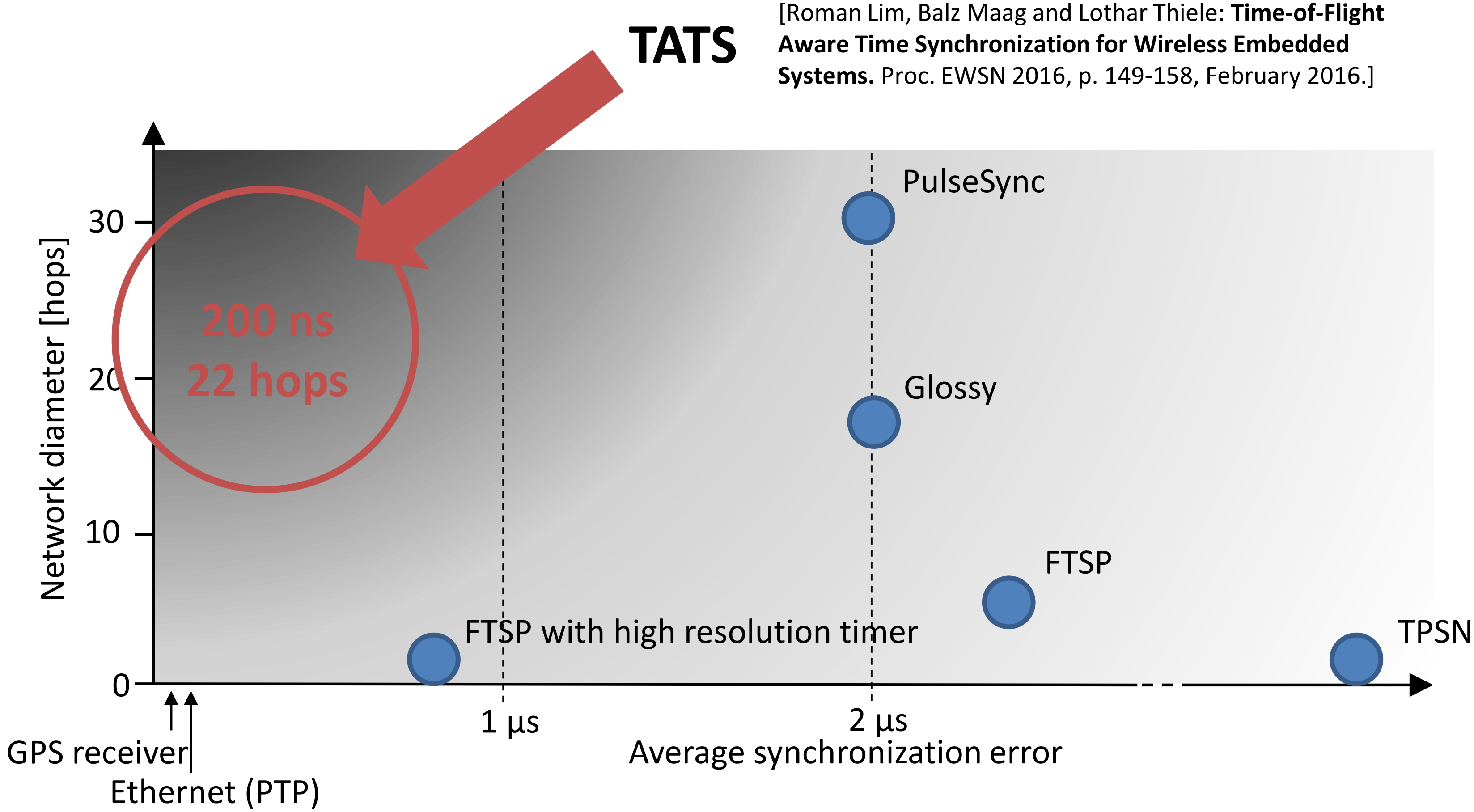
Deployment of the PermaSense Project [1] in the Swiss Alps

Propagation delay 13.33 µs

[1] www.permasense.ch

Wireless Multi-hop Time Synchronization

[Roman Lim, Balz Maag and Lothar Thiele: **Time-of-Flight Aware Time Synchronization for Wireless Embedded Systems.** Proc. EWSN 2016, p. 149-158, February 2016.]

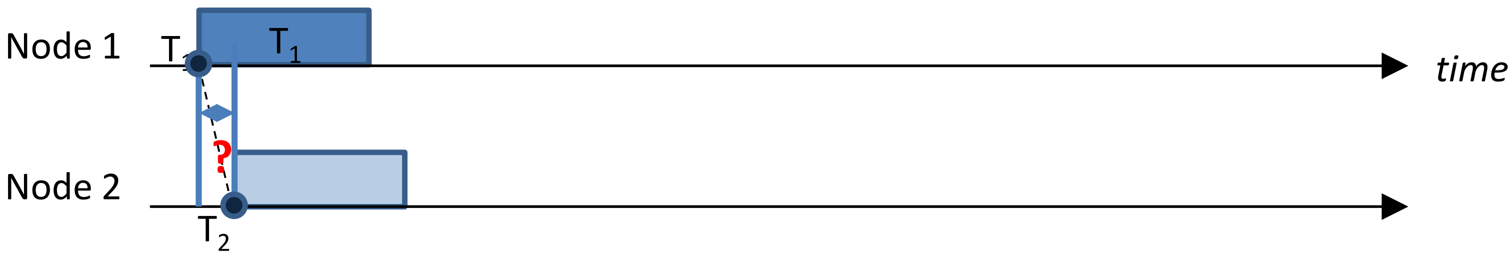


Ingredients for Accurate Synchronization

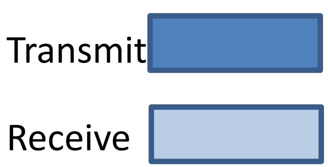
	TPSN	FTSP	PulseSync	Glossy	TATS *
MAC-layer timestamping	✓	✓	✓	✓	✓
Linear regression for offset and clock rate estimation		✓	✓		✓
Two-way delay measurements	✓				✓
Fast flooding			✓	✓	✓

* **T**ime-of-flight **A**ware **T**ime **S**ynchronization

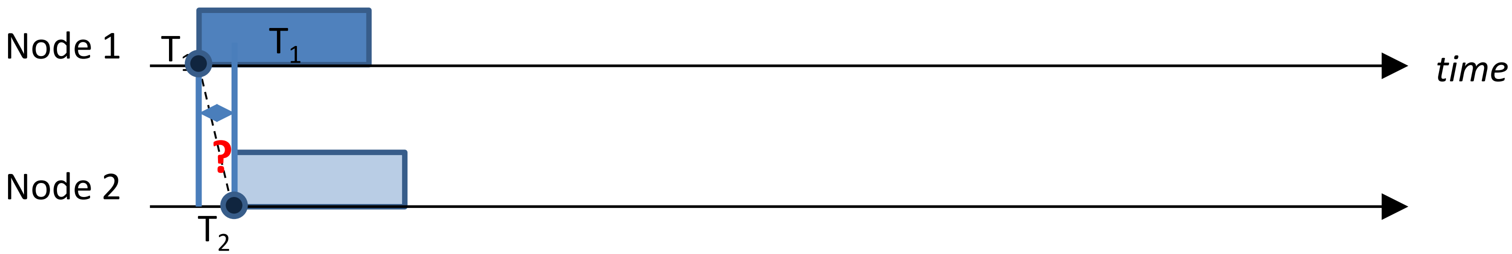
Synchronizing a Pair of Nodes



Time$_1$ (T$_2$) = T$_1$ + **?**

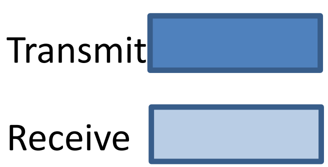


Synchronizing a Pair of Nodes

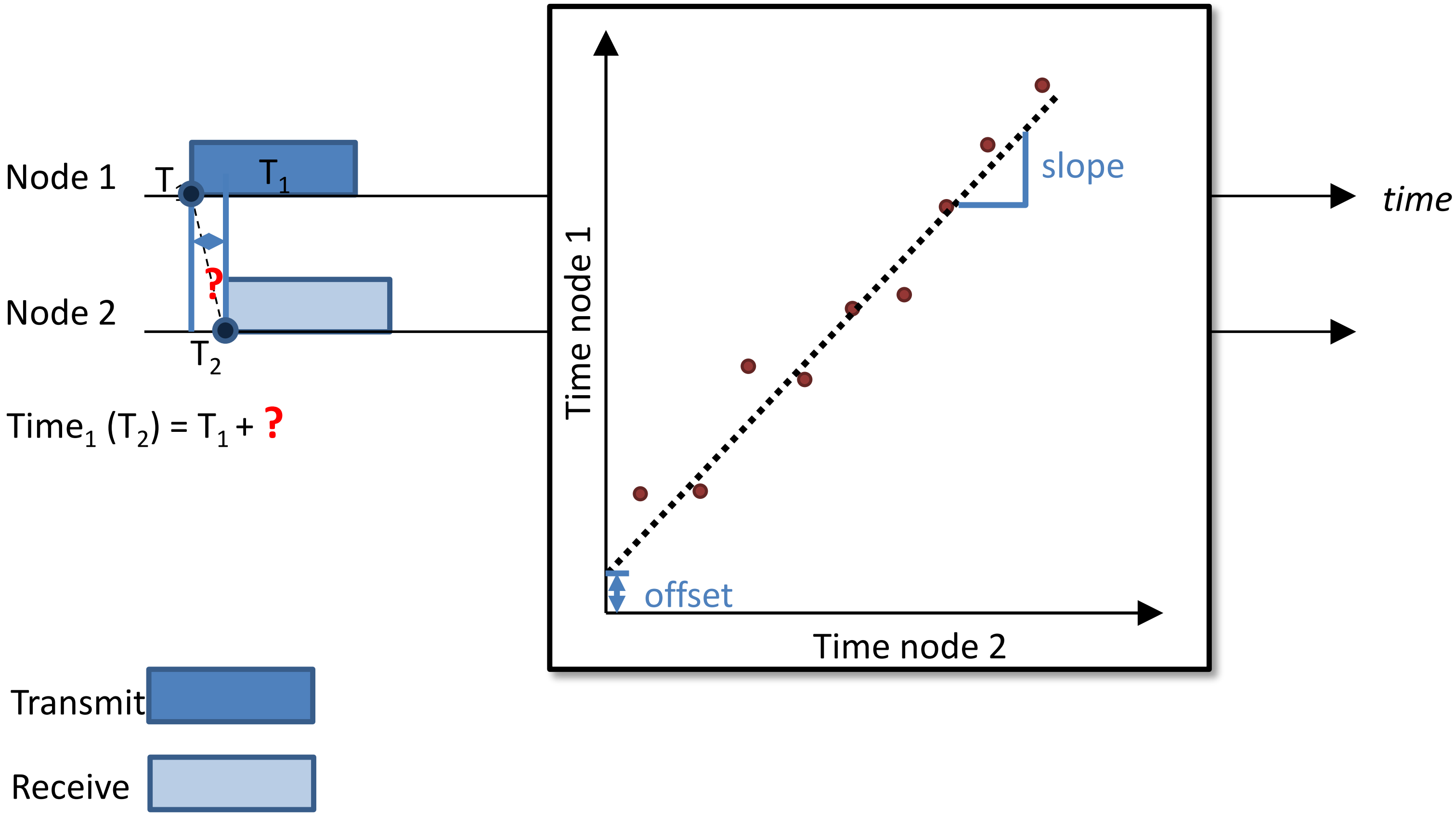


$Time_1(T_2) = T_1 + ?$

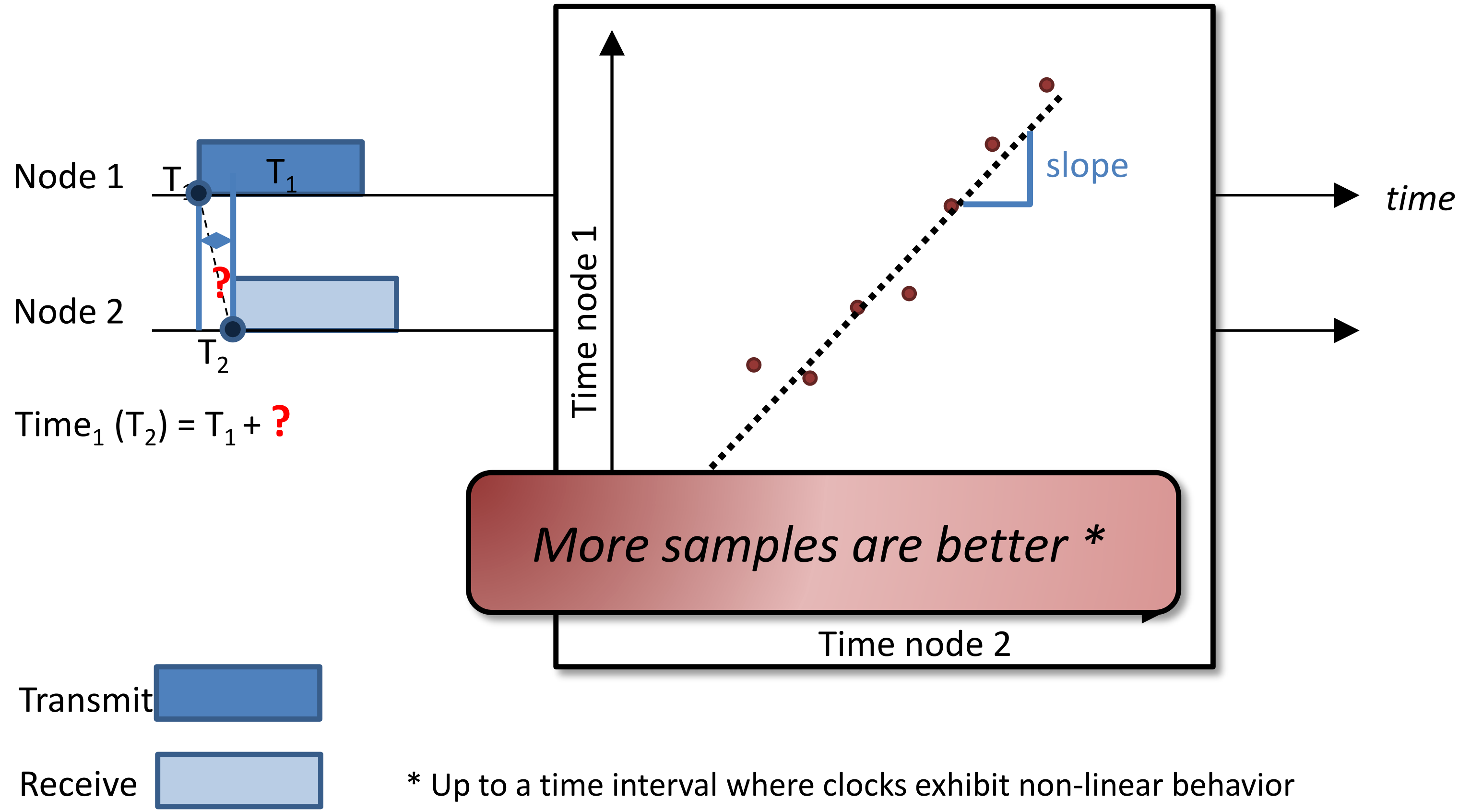
We need accurate packet timestamps with sufficient time resolution



Synchronizing a Pair of Nodes



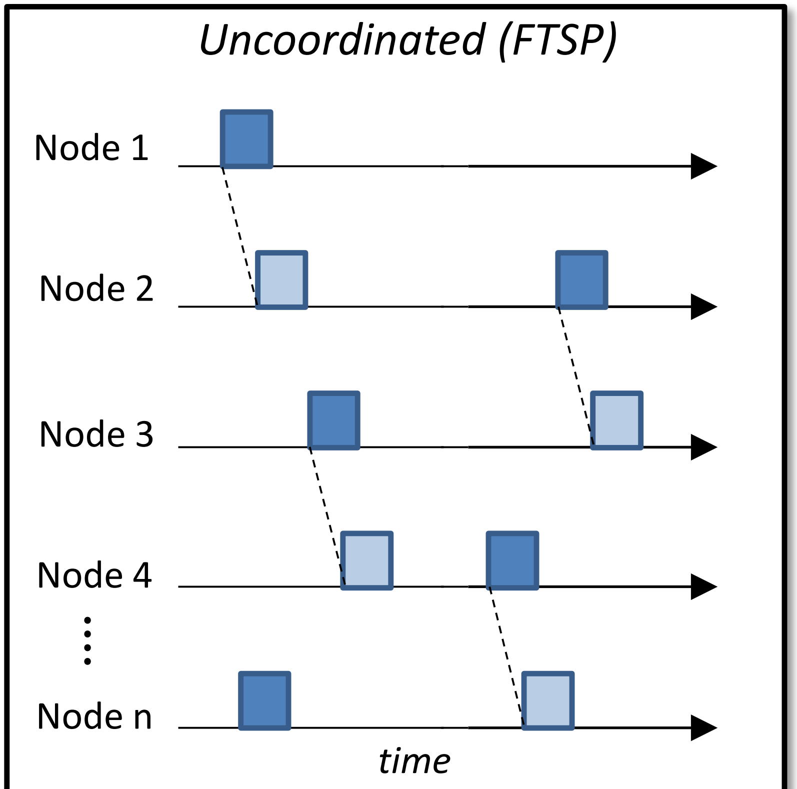
Synchronizing a Pair of Nodes

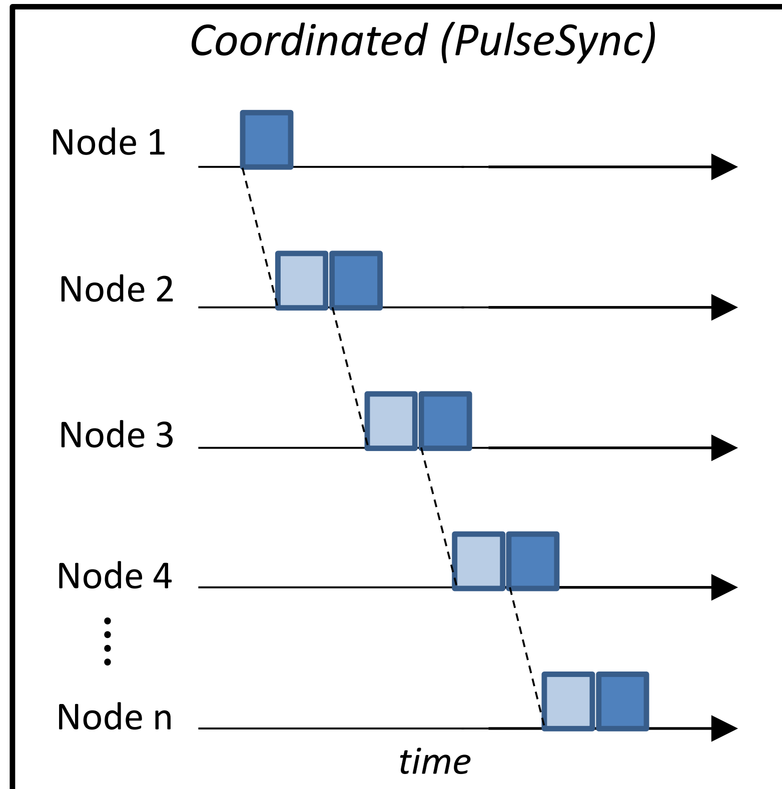


$Time_1(T_2) = T_1 +$?

* Up to a time interval where clocks exhibit non-linear behavior

Synchronization Based on Network Flooding

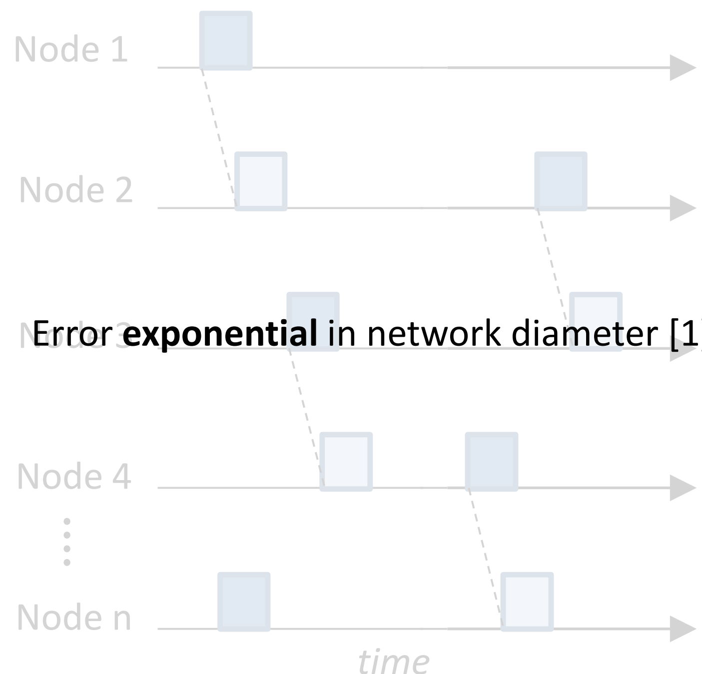




ETH zürich

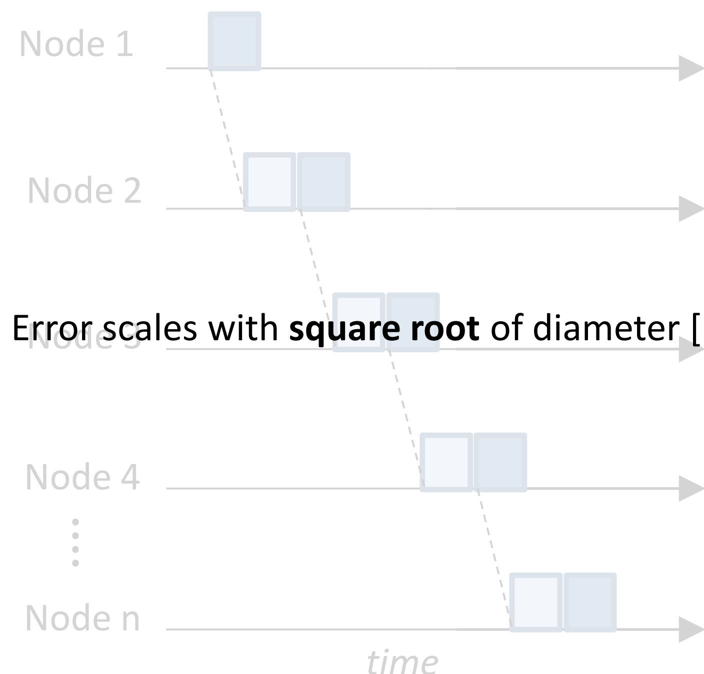
Synchronization Based on Network Flooding

Uncoordinated (FTSP)



Error **exponential** in network diameter [1]

Coordinated (PulseSync)

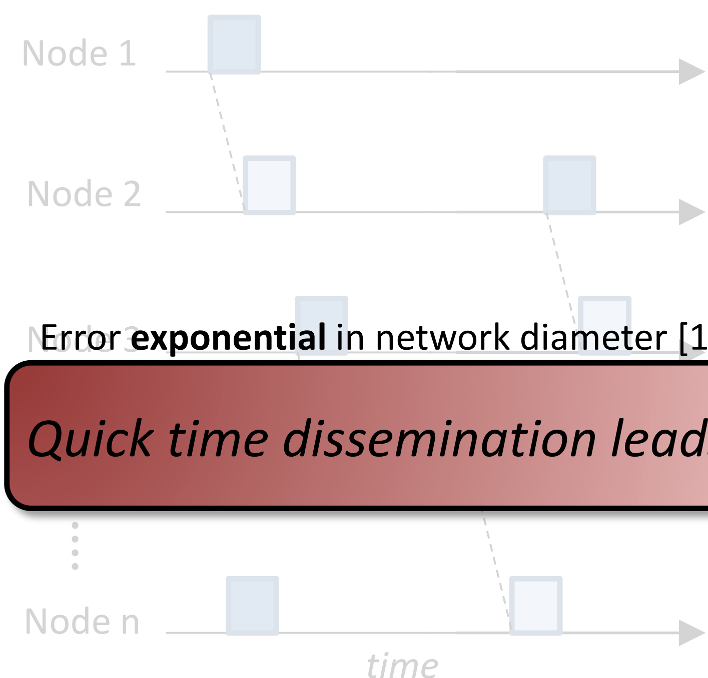


Error scales with **square root** of diameter [1]

[1] C. Lenzen et al, Optimal clock synchronization in networks, SenSys 2009

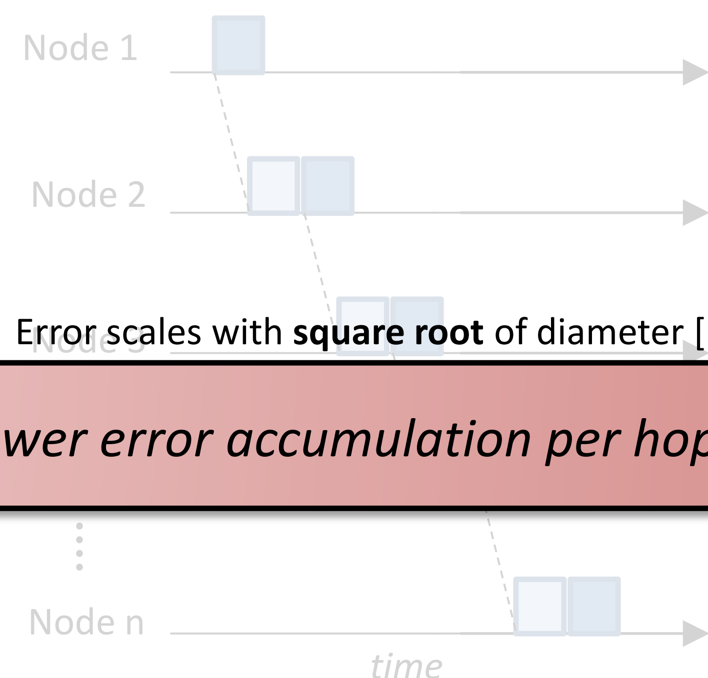
Synchronization Based on Network Flooding

Uncoordinated (FTSP)



Error **exponential** in network diameter [1]

Coordinated (PulseSync)



Error scales with **square root** of diameter [1]

Quick time dissemination leads to lower error accumulation per hop

[1] C. Lenzen et al, Optimal clock synchronization in networks, SenSys 2009

TATS MAC Layer Timestamping

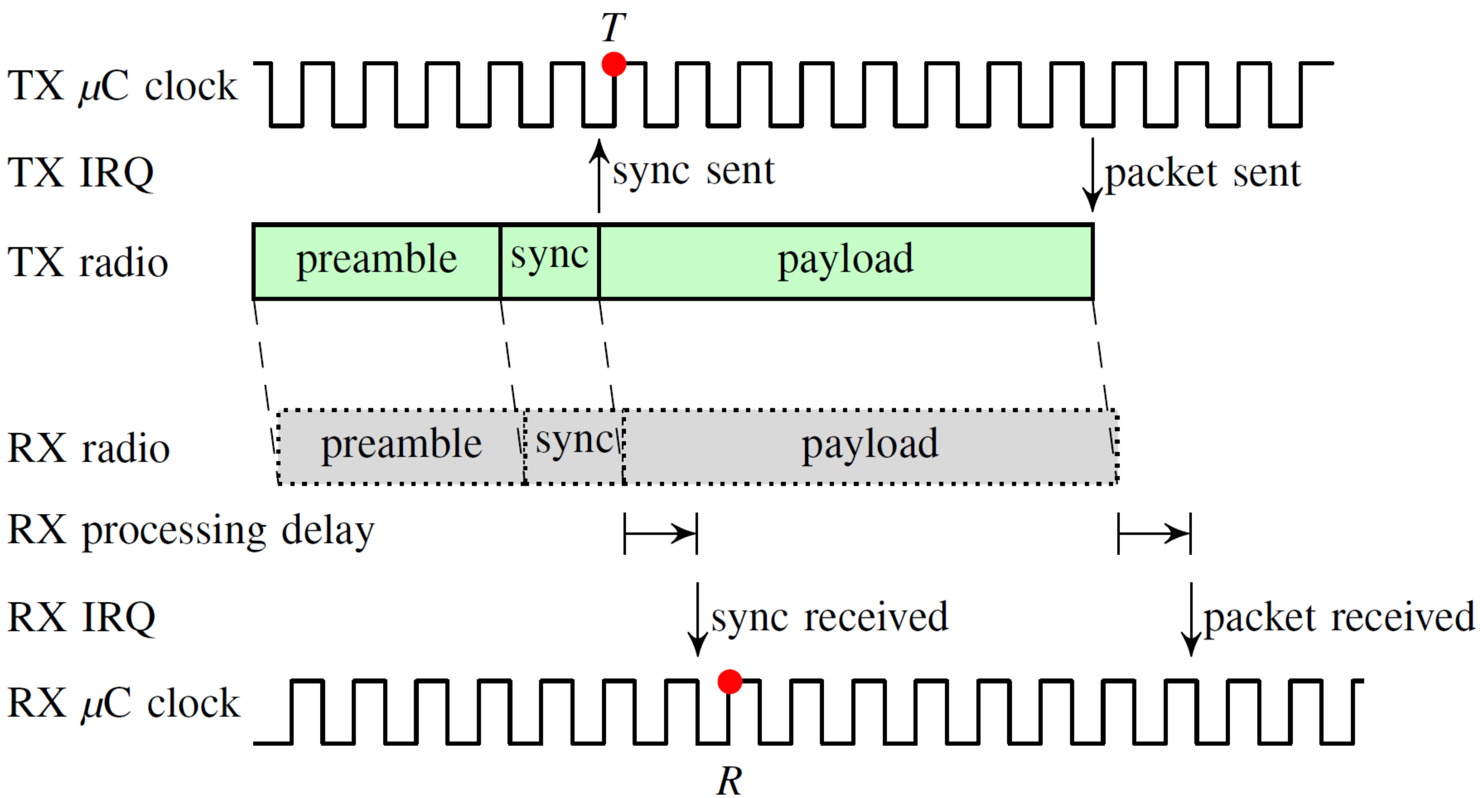
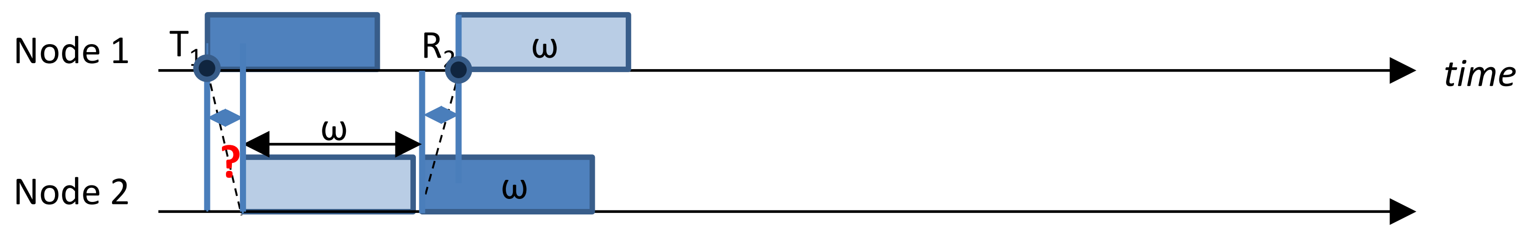
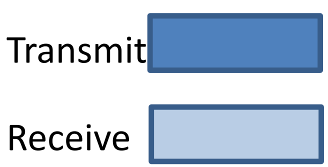


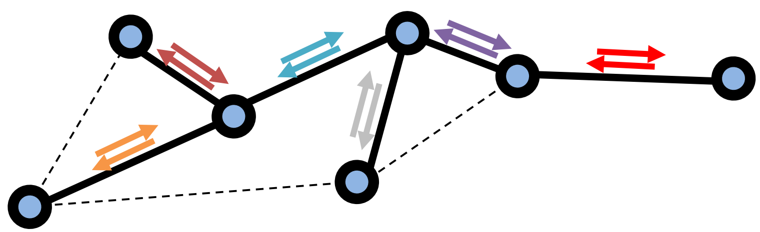
Figure 5. Timestamps for one message transmission. *Timestamps T and R are inaccurate due to asynchronous clocks and uncertainties introduced with radio modulation.*

Propagation Delay Measurement



Propagation delay: $(R_2 - T_1 - \omega) / 2$

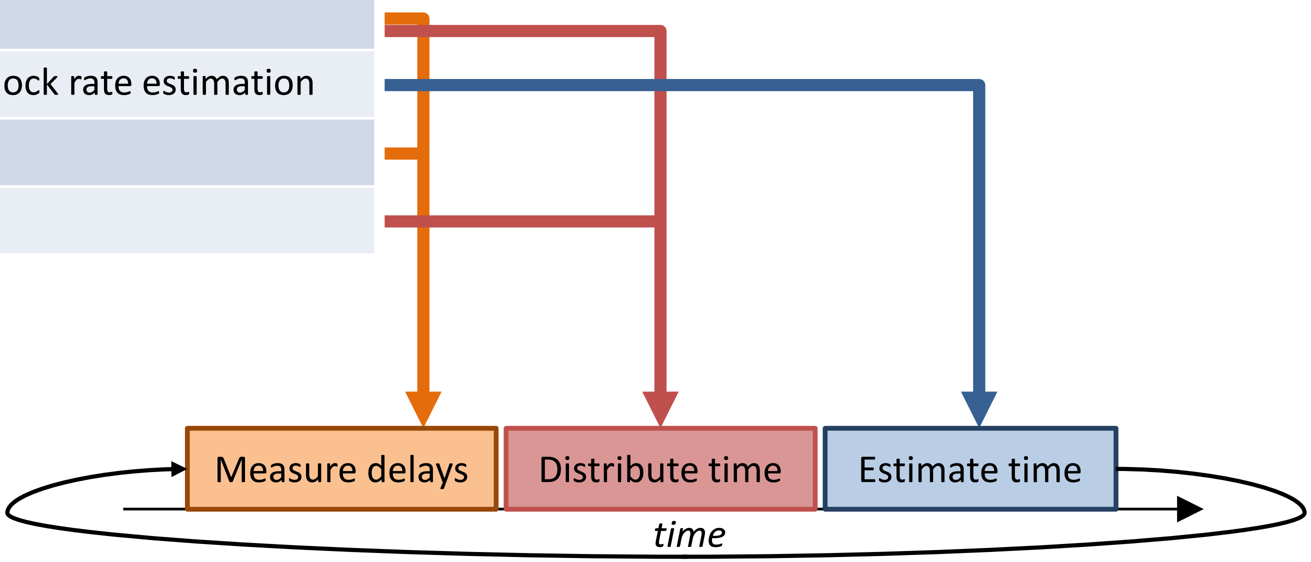




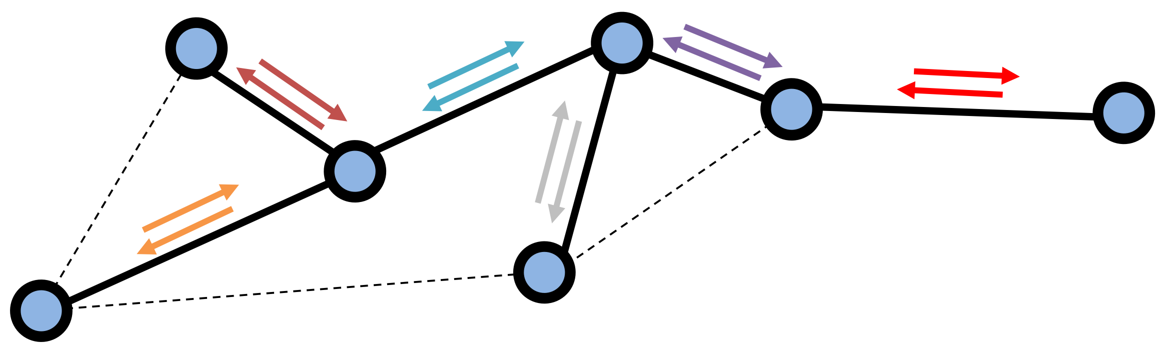
Two-way delay measurement on each link

Putting it Together

- MAC-layer timestamping
- Linear regression for offset and clock rate estimation
- Two-way delay measurements
- Fast flooding

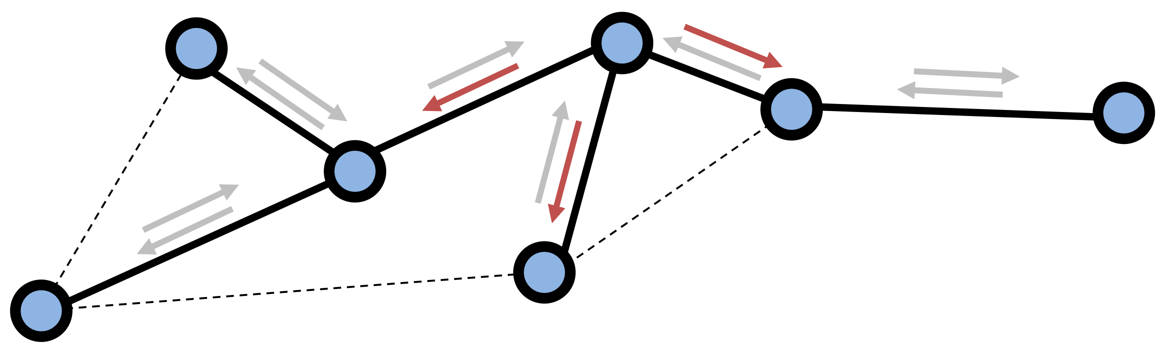


Can we measure delays using only ***one packet per node***?



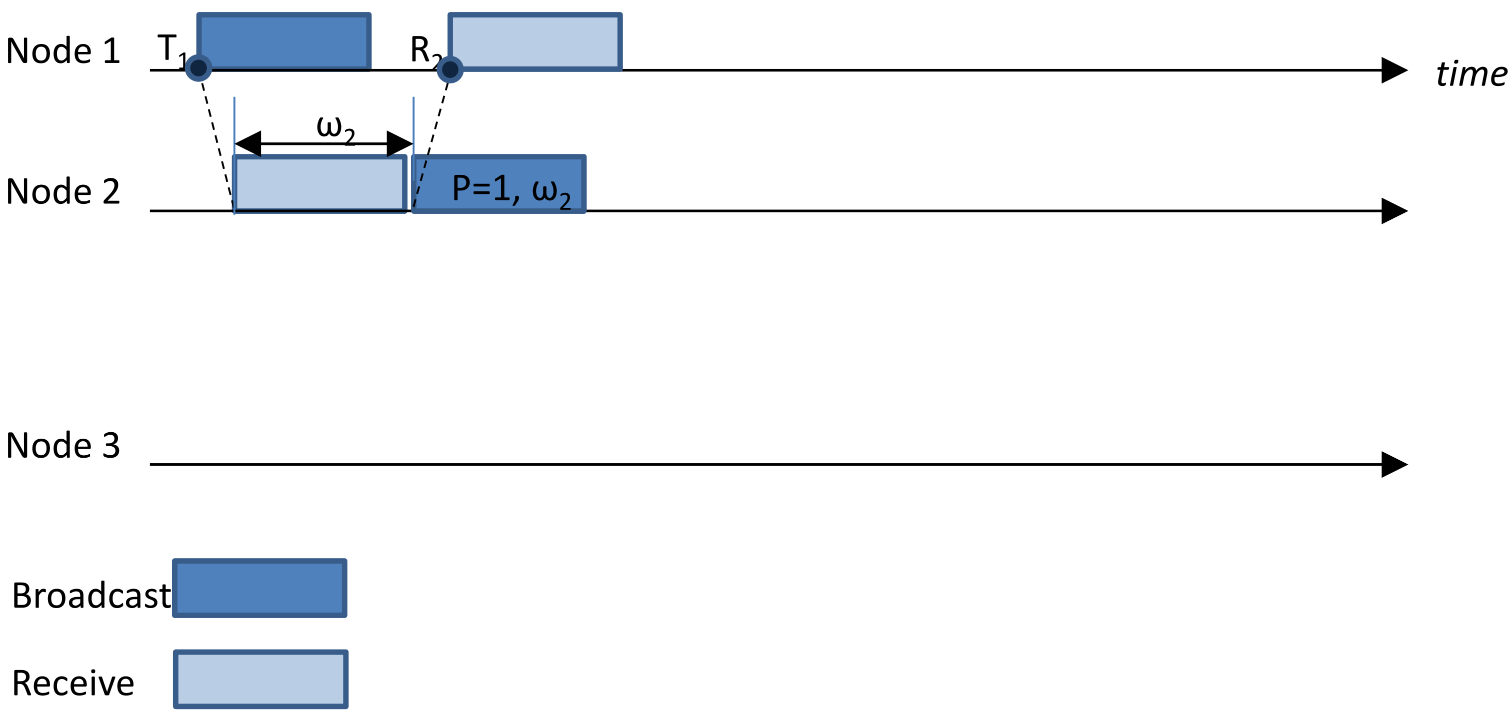
Less transmissions
Fits into existing flooding communication schemes
No need for explicit tree topology creation

Can we measure delays using only ***one packet per node***?

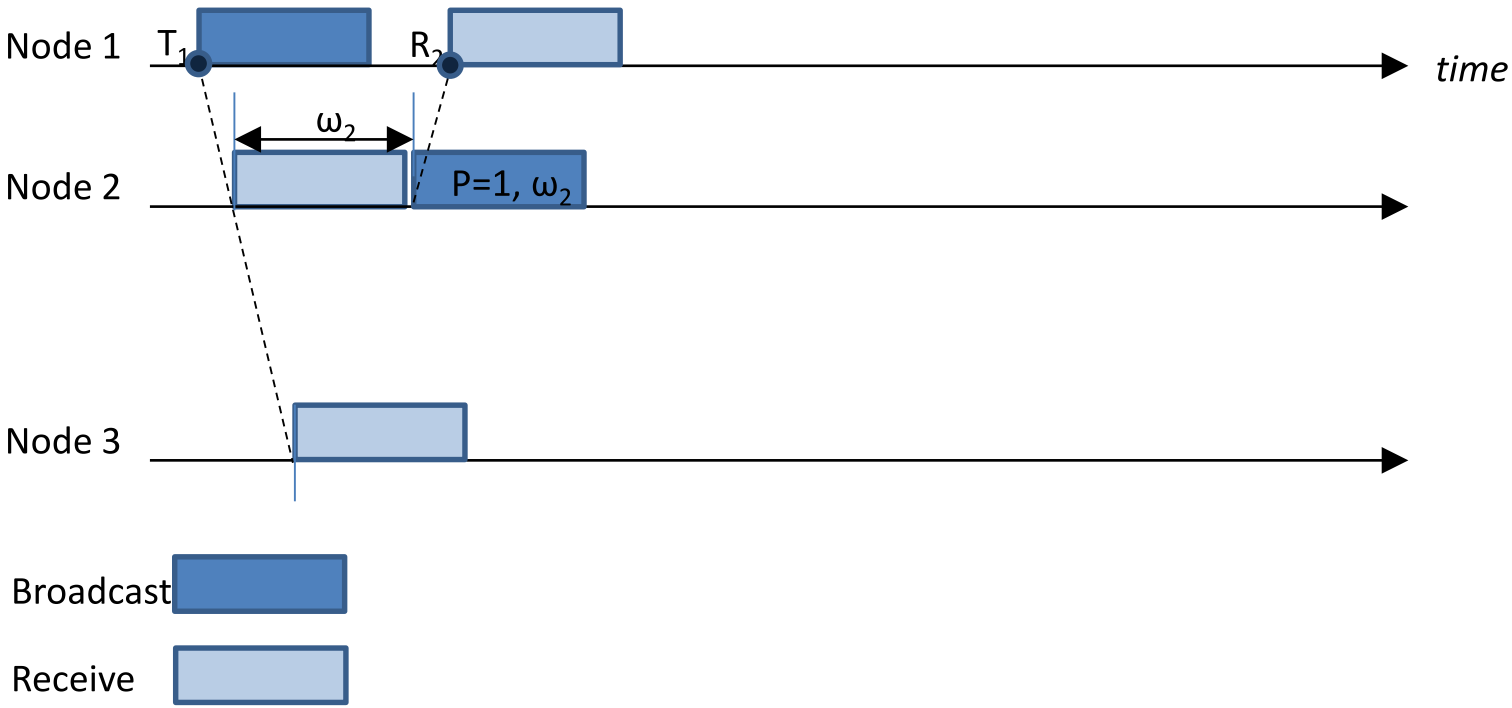


Less transmissions
Fits into existing flooding communication schemes
No need for explicit tree topology creation

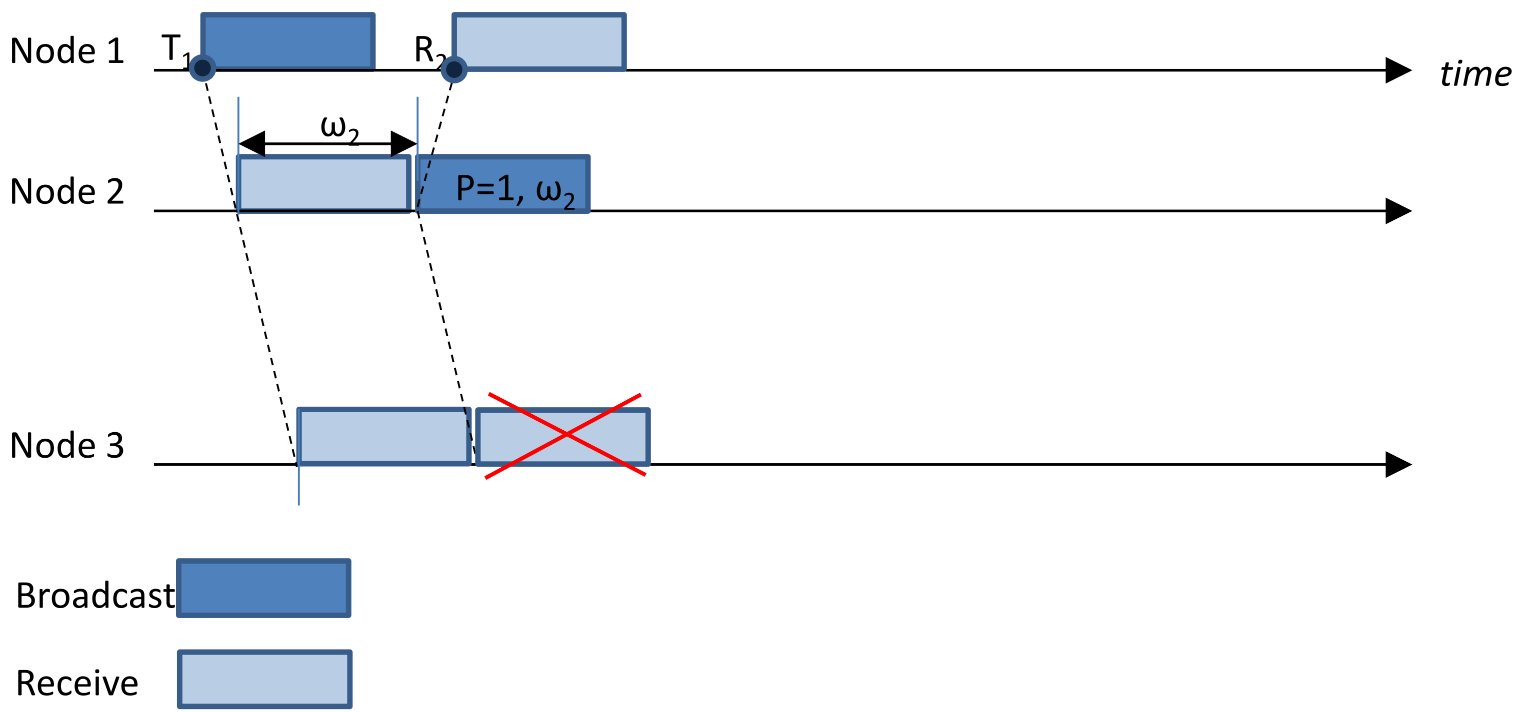
Propagation Delay Measurements in Floods



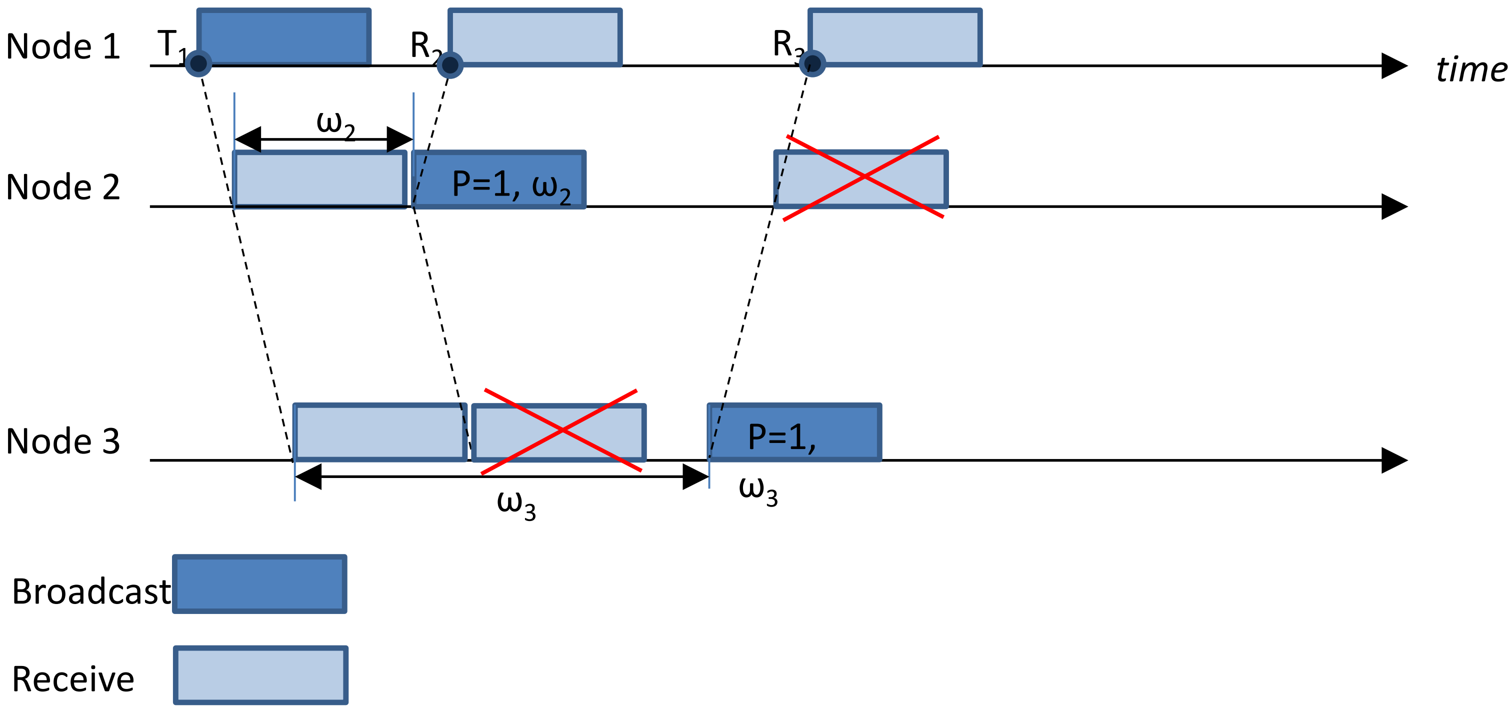
Propagation Delay Measurements in Floods



Propagation Delay Measurements in Floods

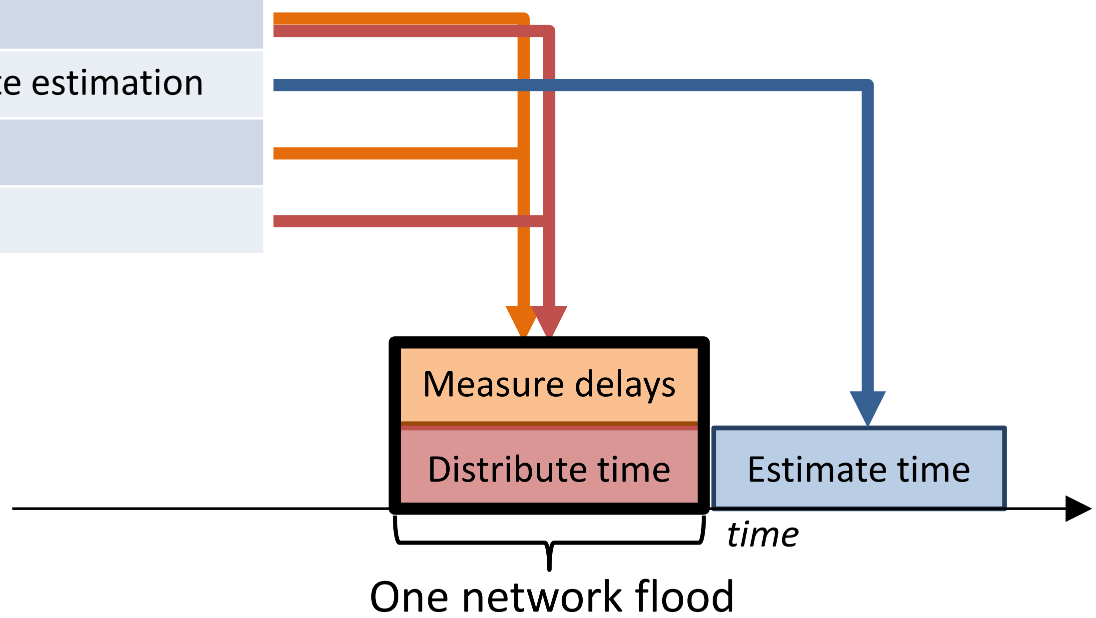


Propagation Delay Measurements in Floods

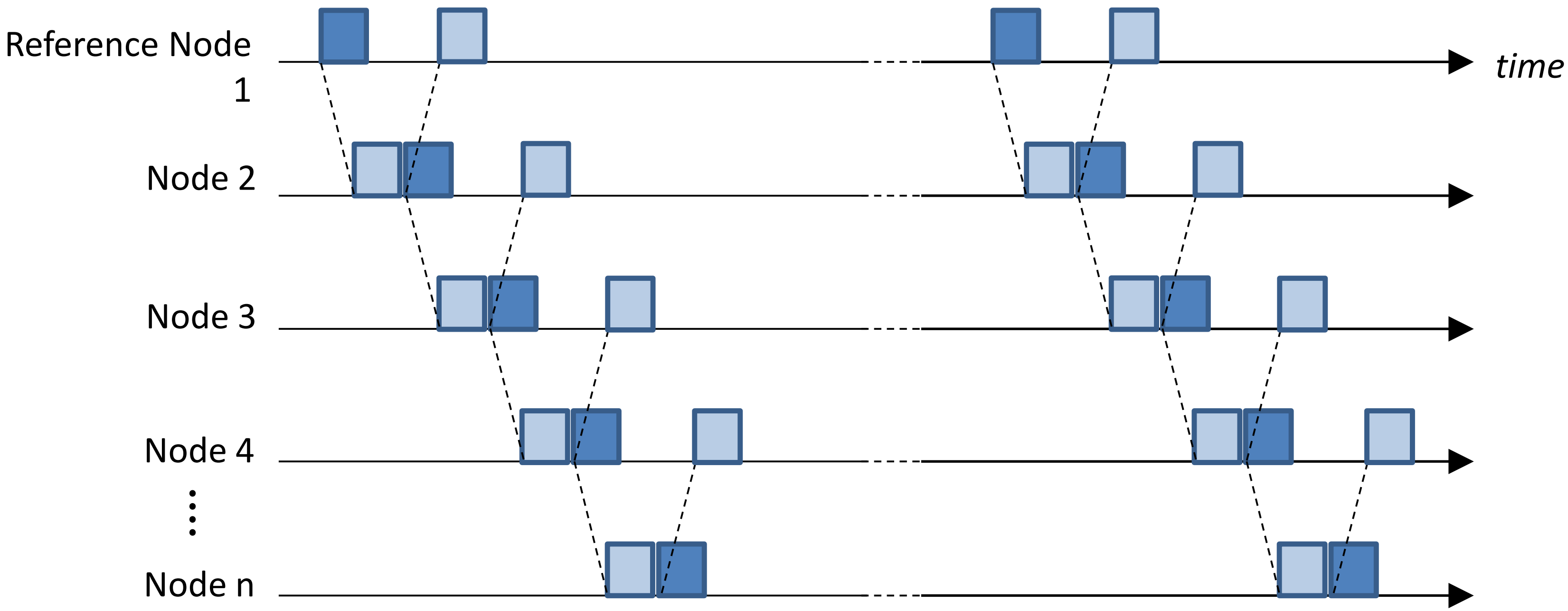


Putting it Together

- MAC-layer timestamping
- Linear regression for offset and clock rate estimation
- Two-way delay measurements
- Fast flooding

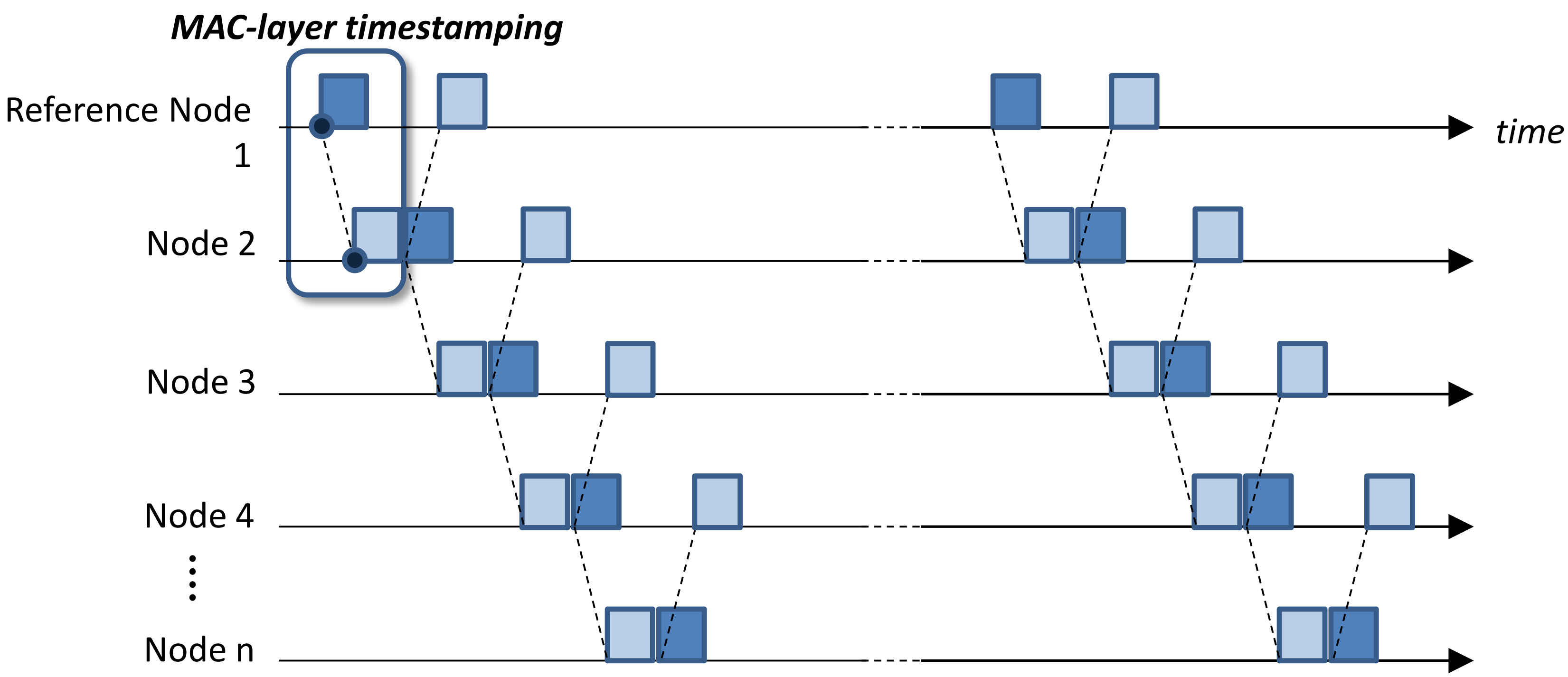


TATS in a Nutshell

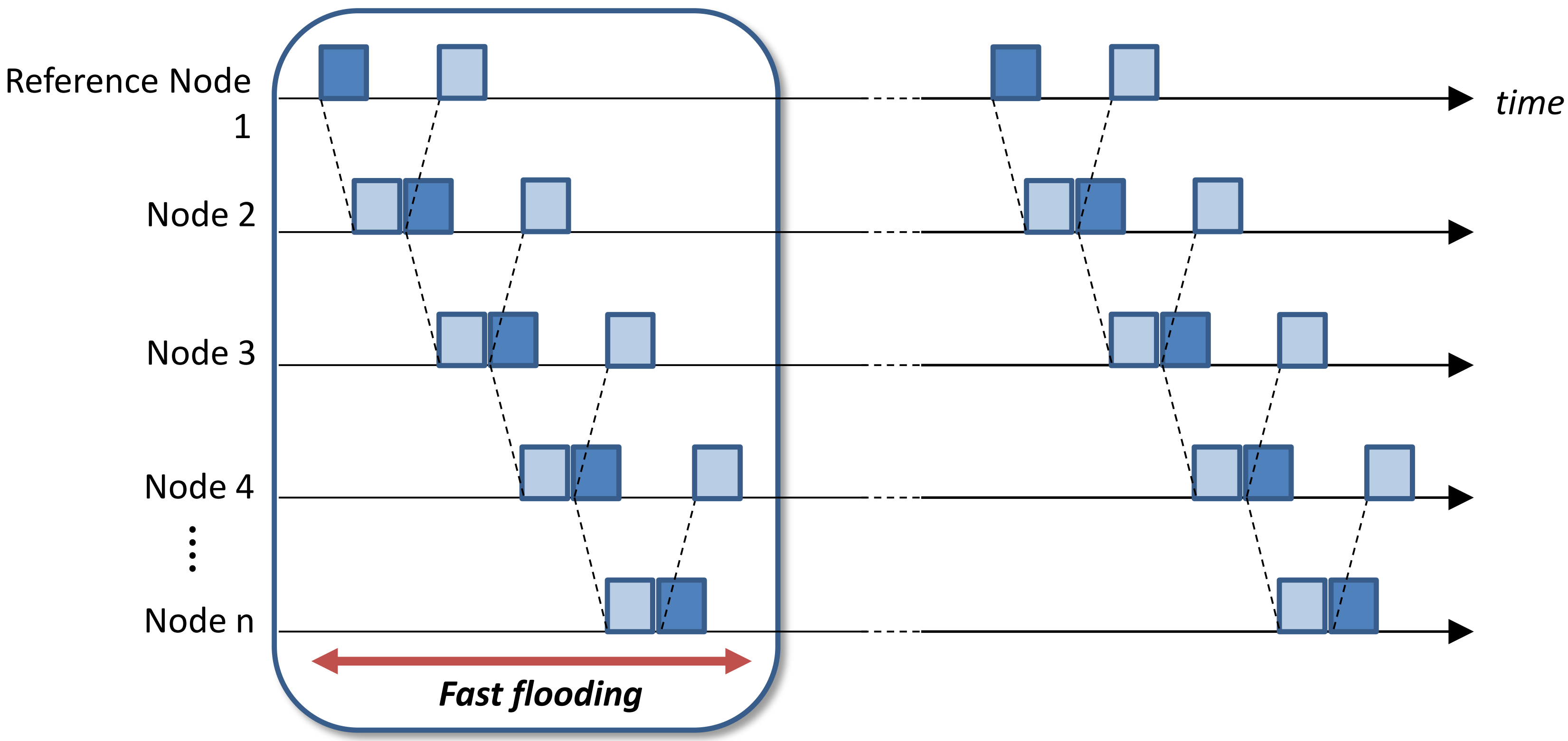


One broadcast packet per round and node, same as in FTSP and PulseSync

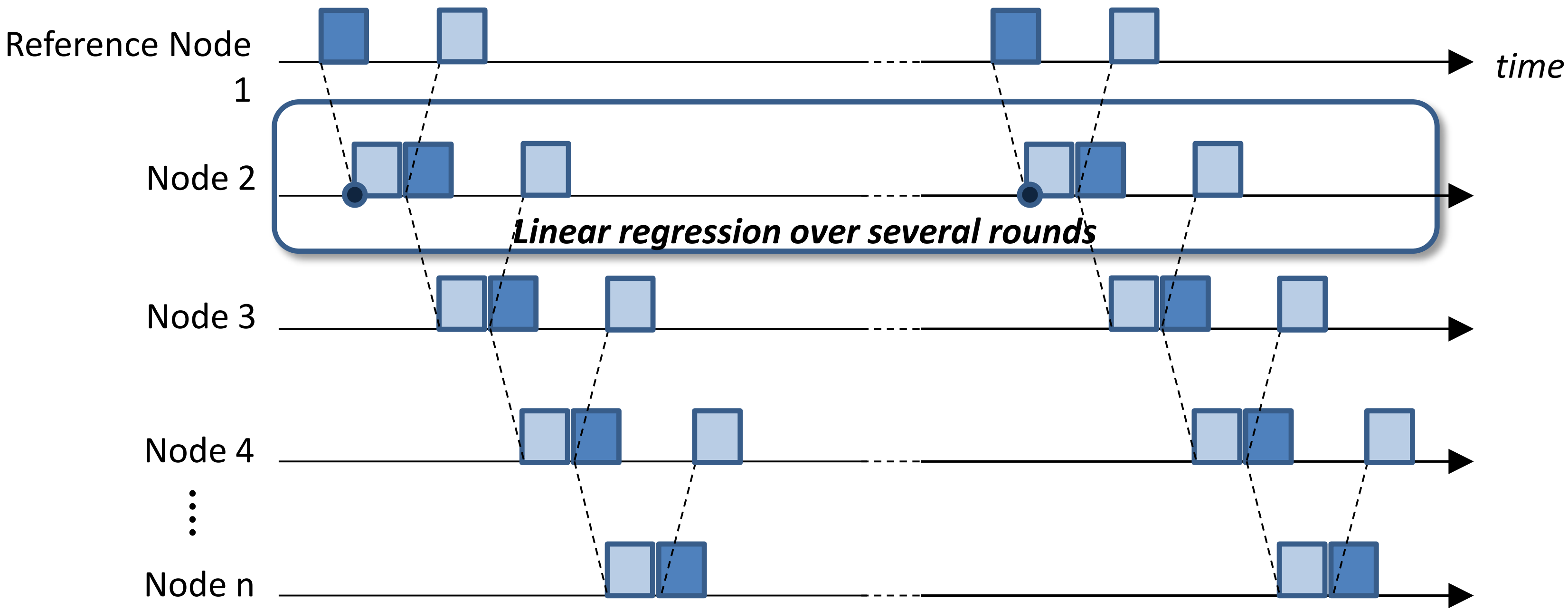
TATS in a Nutshell



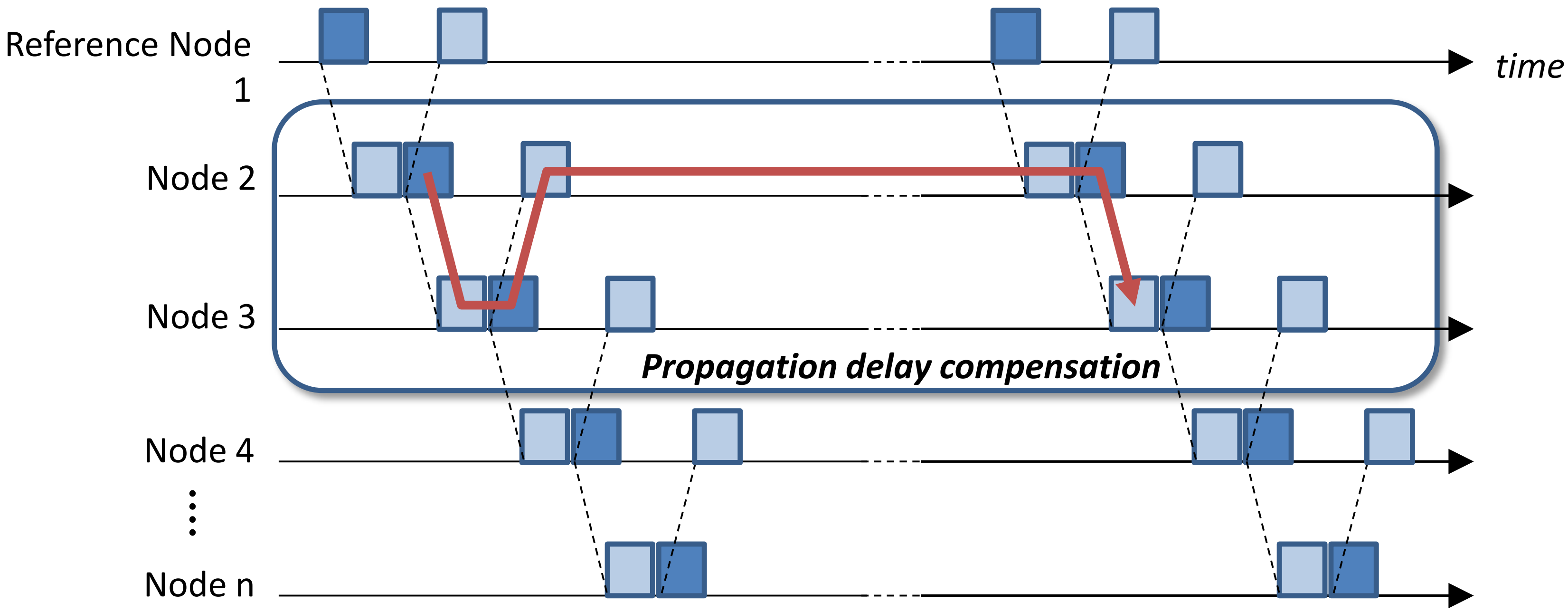
TATS in a Nutshell



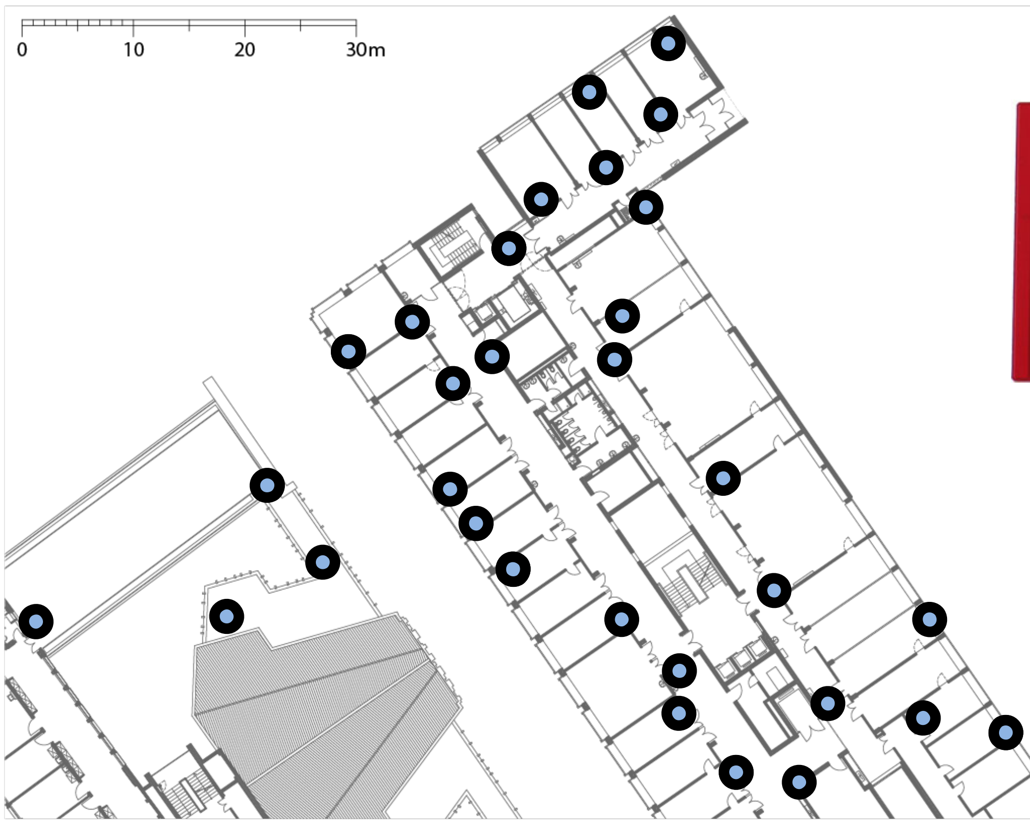
TATS in a Nutshell

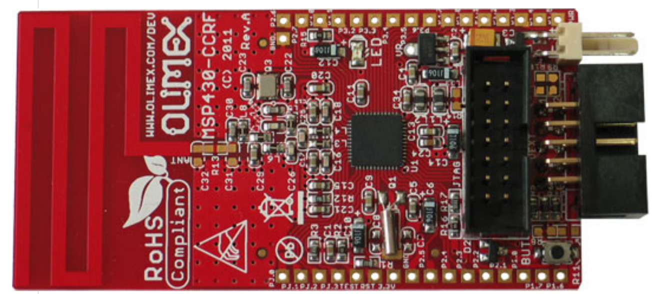


TATS in a Nutshell



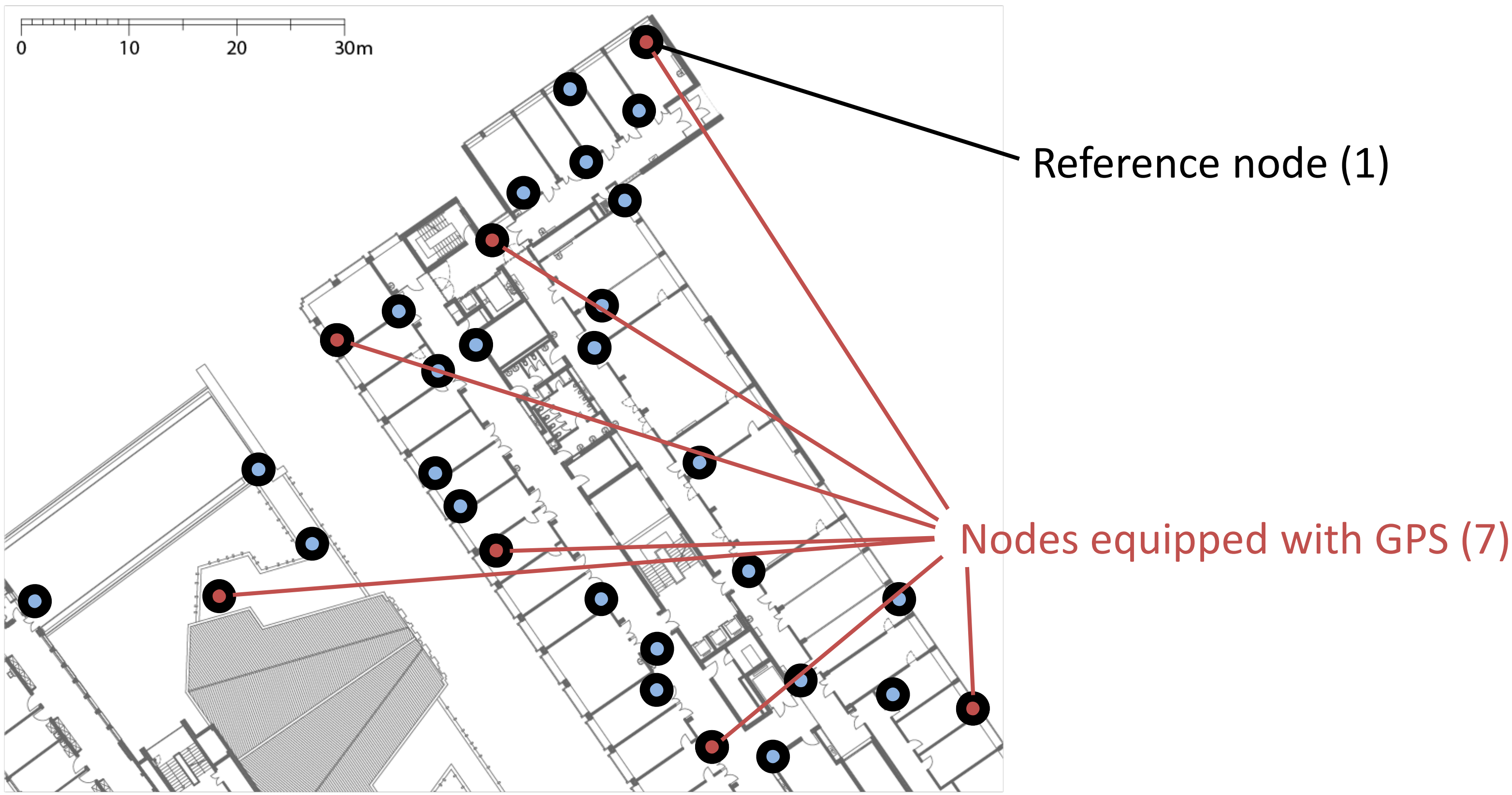
Experimental Evaluation on FlockLab



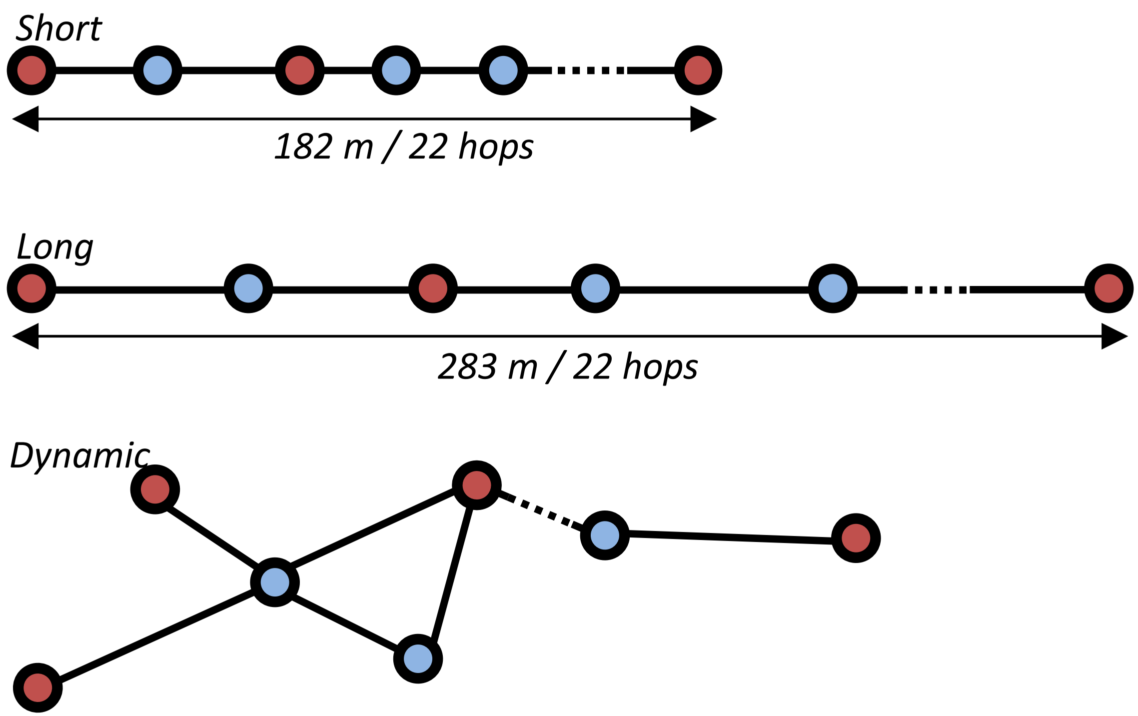


CC430 SoC, MSP430 + sub-1GHz radio
Use radio clock as system clock (13 MHz)

Comparison to PulseSync and Glossy

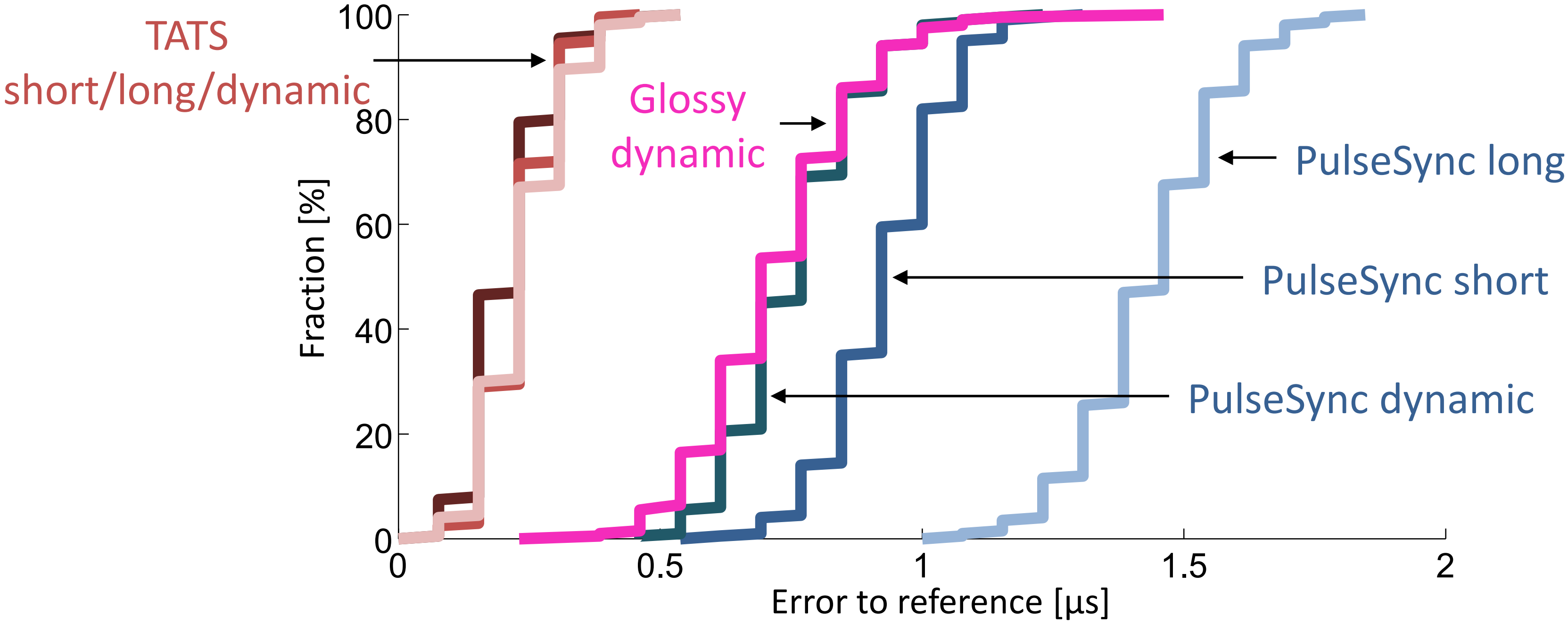


Comparison to PulseSync and Glossy

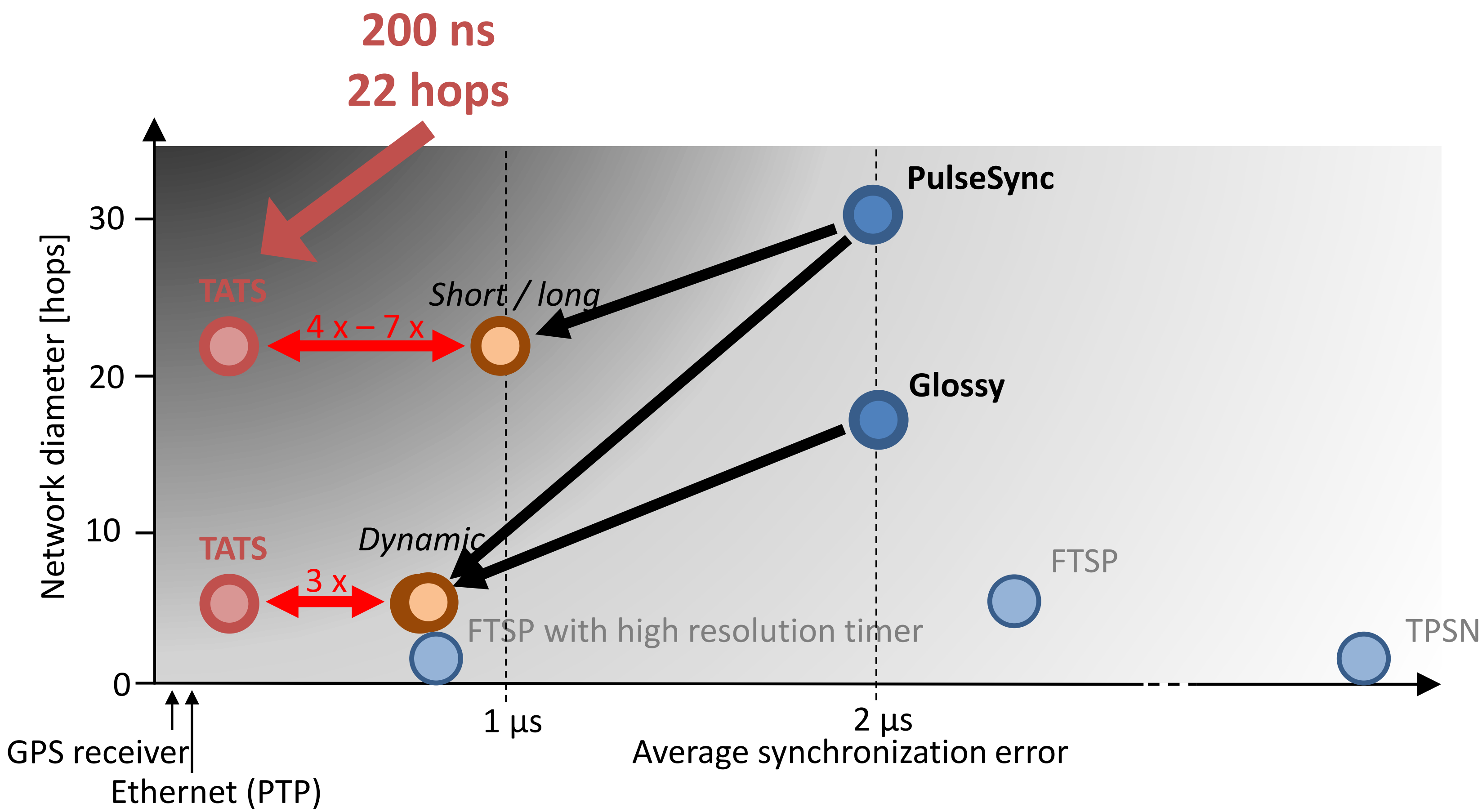


Parameters
- 1 s synchronization interval
- Regression over 80 samples
- Duration: 1 h

Head-to-head Comparison



Head-to-head Comparison



Today's Hot Researcher & Paper

- Adam Dunkels
  - Former researcher at SICS
  - Founder/CEO Thingsquare

- Pionieered IP on embedded devices
- Author/creator of
  - uIP (micro-IP) and lwIP TCP/IP protocol stacks
  - Protothreads
  - Contiki operating system
- MIT Technology Review TR35 (2009)

- "Interconnecting Smart Objects with IP - the Next Internet", co-authored with JP Vasseur and a foreword by Vint Cerf.

Recap of Today

- System-level design requires to think across all layers
- (Temporal) Co-ordination helps a lot to
- Often “surprising” discoveries (constructive interference) can be leveraged

- State-of-the-Art protocols allow very low-power communication at little energy cost