Optimal Clock Synchronization in Networks

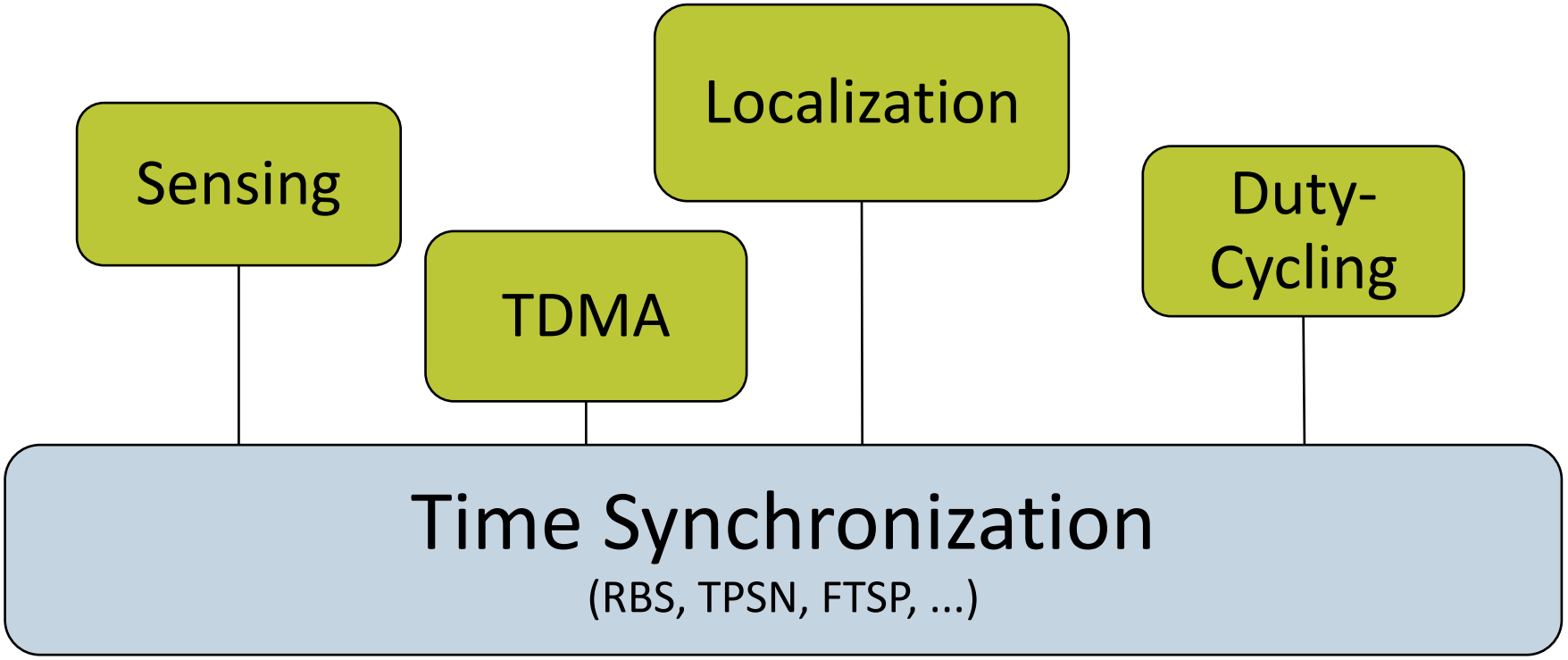
Christoph Lenzen
Philipp Sommer
Roger Wattenhofer

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing Group

Time in Sensor Networks

- Synchronized clocks are essential for many applications:

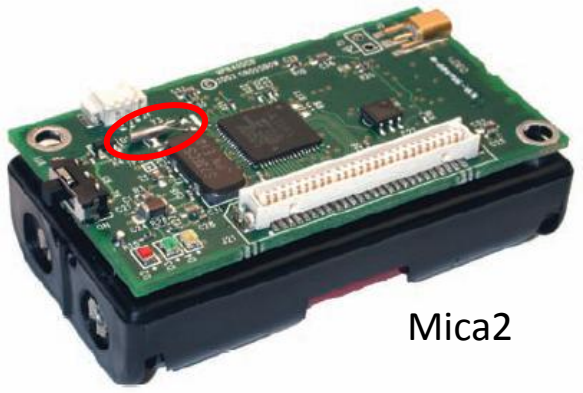


Hardware Clocks Experience Drift

- Hardware clock

  Counter register of the microcontroller

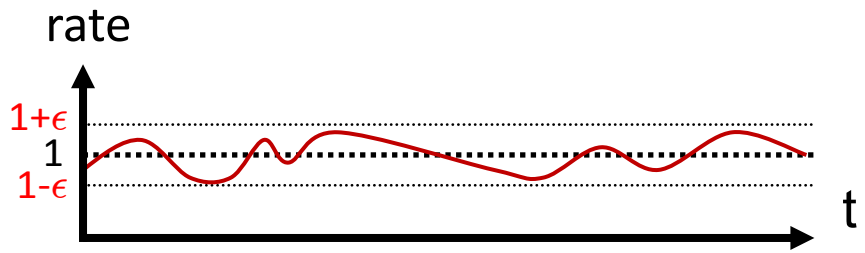
  Sourced by an external crystal (32kHz, 7.37 MHz)

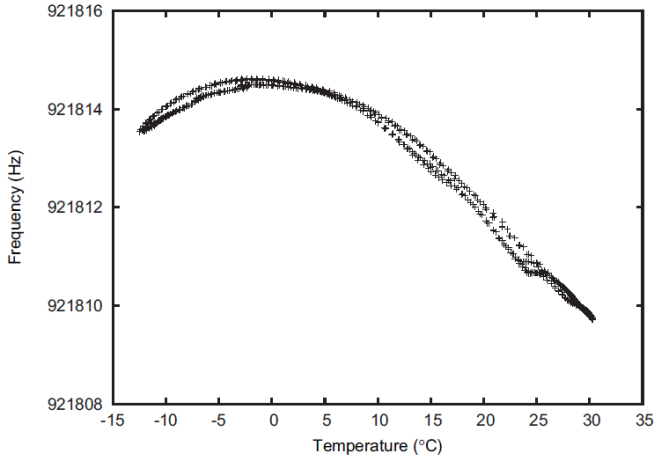


Mica2

- Clock drift

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

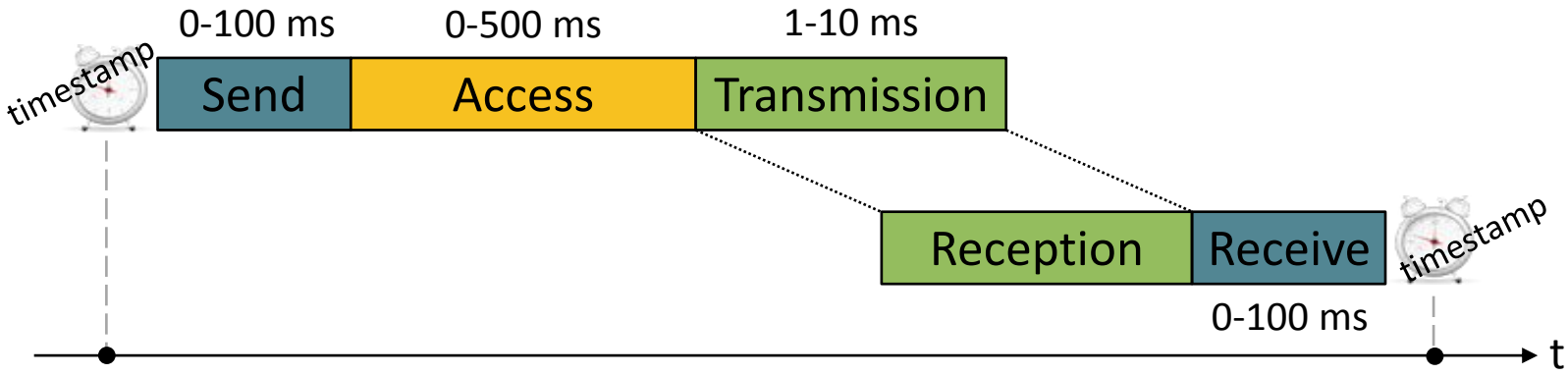




Messages Experience Jitter in the Delay

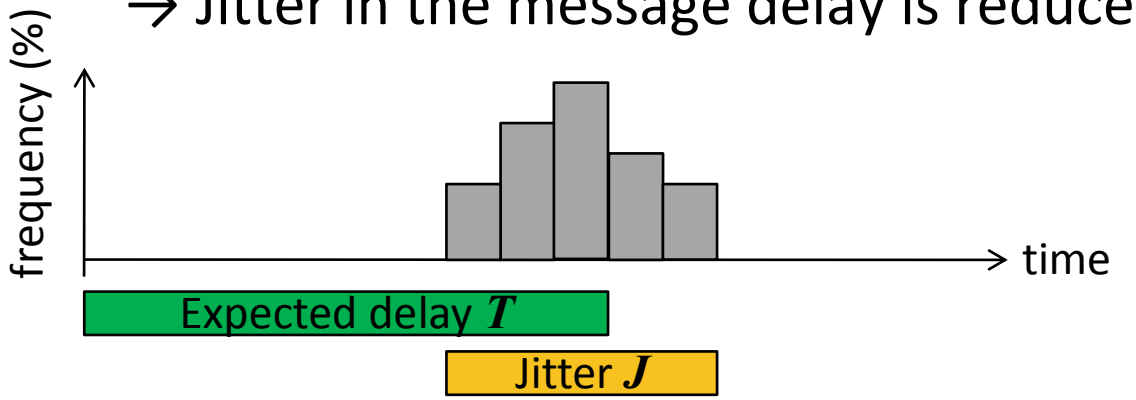
- Problem: Jitter in the message delay

  Various sources of errors (deterministic and non-deterministic)



- Solution: Timestamping packets at the MAC layer (Maróti et al.)

  → Jitter in the message delay is reduced to a few clock ticks

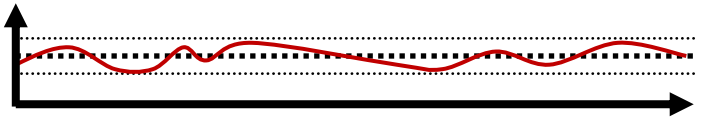


Summary: Clock Synchronization

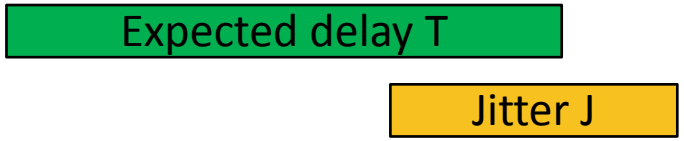
- Goal:

  Send time information (beacons) to synchronize clocks

- Problems:
  - Hardware clocks exhibit **drift**

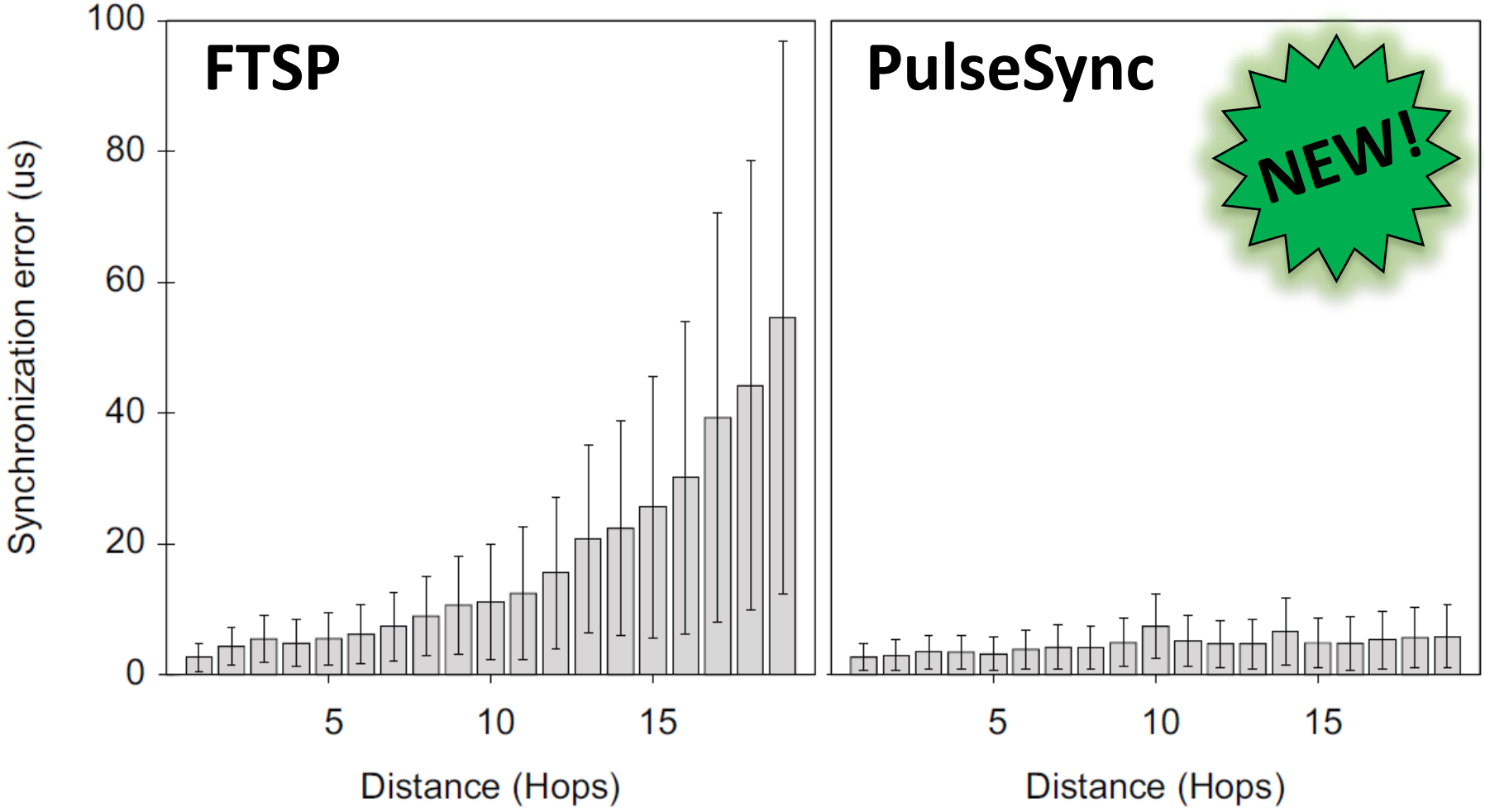


  - **Jitter** in the message delay



Preview: Experimental Results

- Sychnronization error vs. hop distance



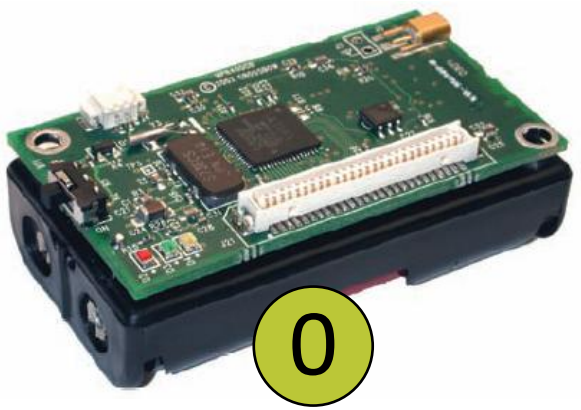
Outline

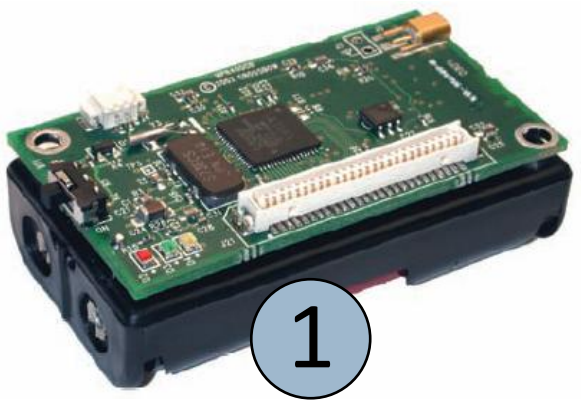
- Introduction
- Theory
- Practice

Synchronizing Nodes: Single-Hop

- How do we synchronize the clocks of **two** sensor nodes?

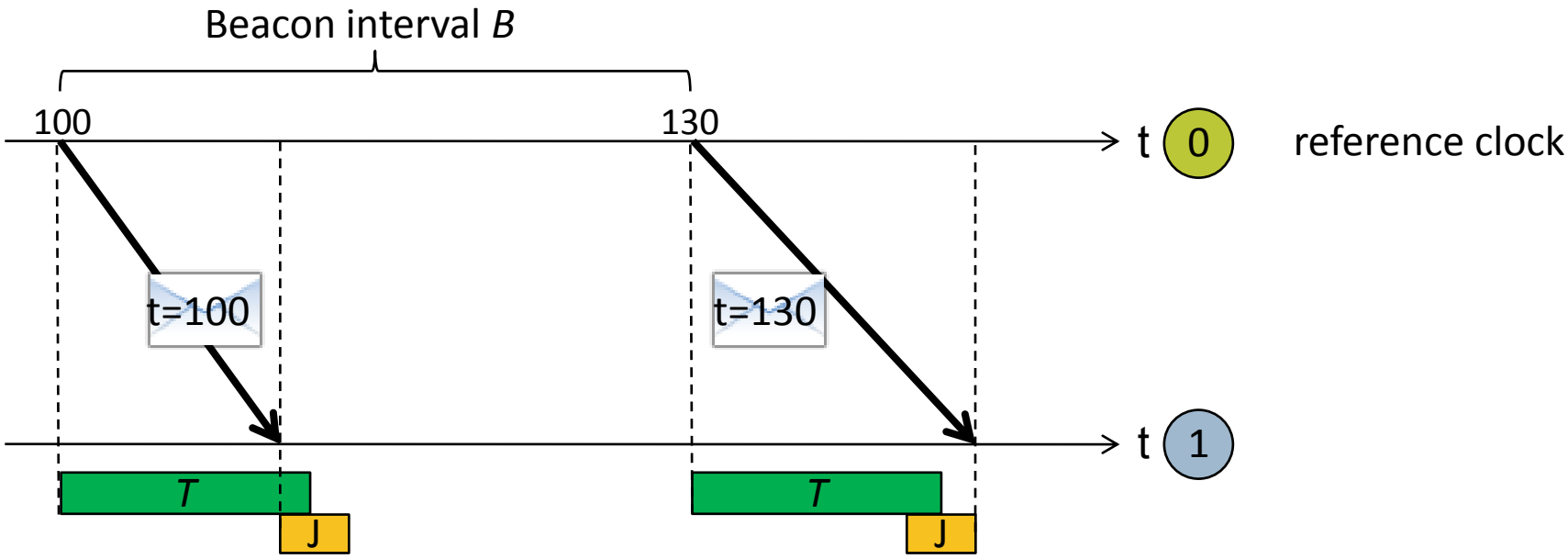
reference clock





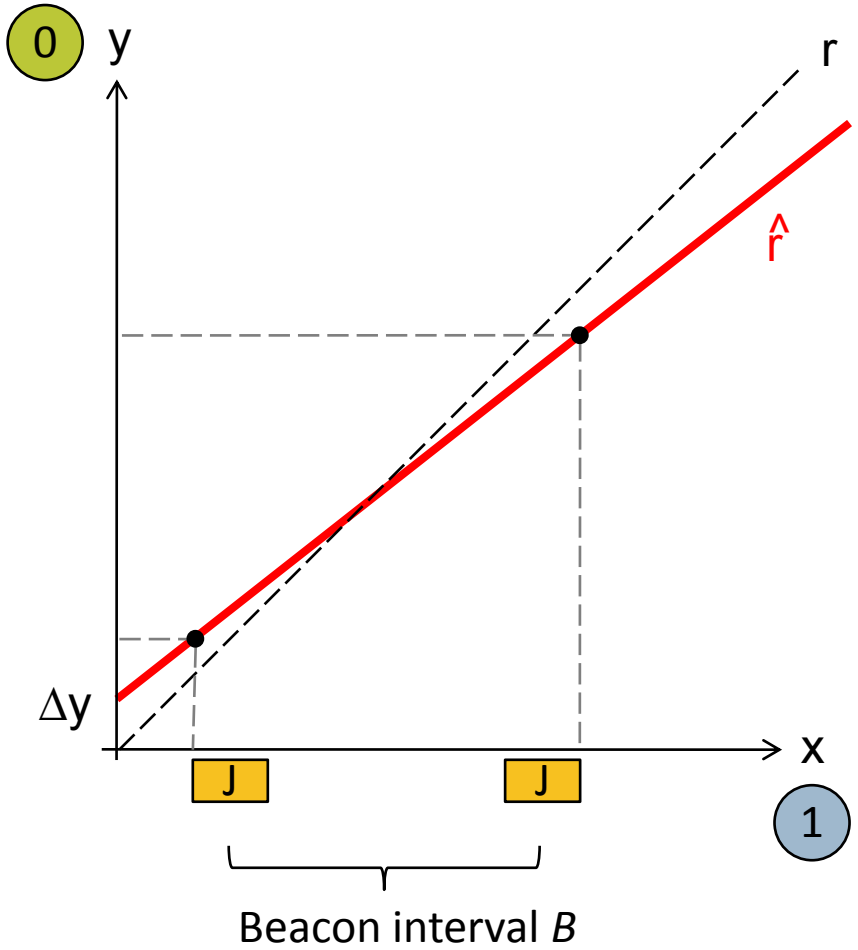
Synchronizing Nodes

- Sending periodic beacons to synchronize nodes



How accurately can we synchronize two Nodes?

- Message delay jitter affects clock synchronization quality



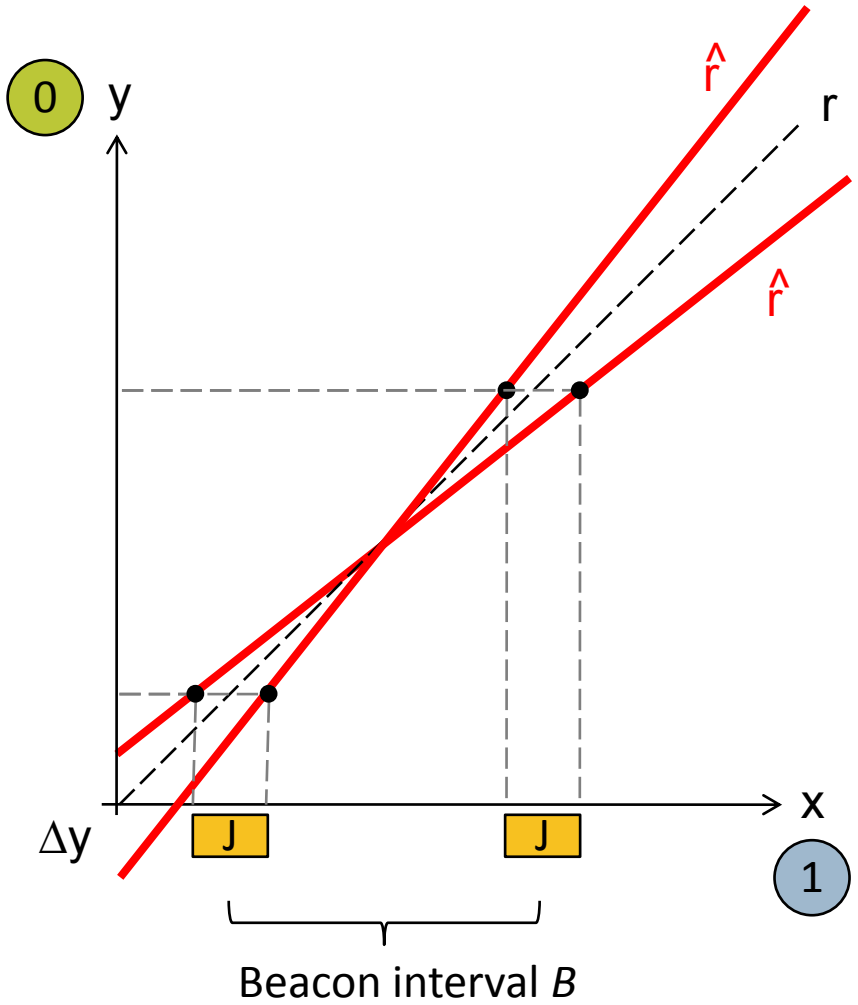
$$y(x) = \hat{r} \cdot x + \Delta y$$

clock offset

relative clock rate (estimated)

How accurately can we synchronize two Nodes?

- Message delay jitter affects clock synchronization quality



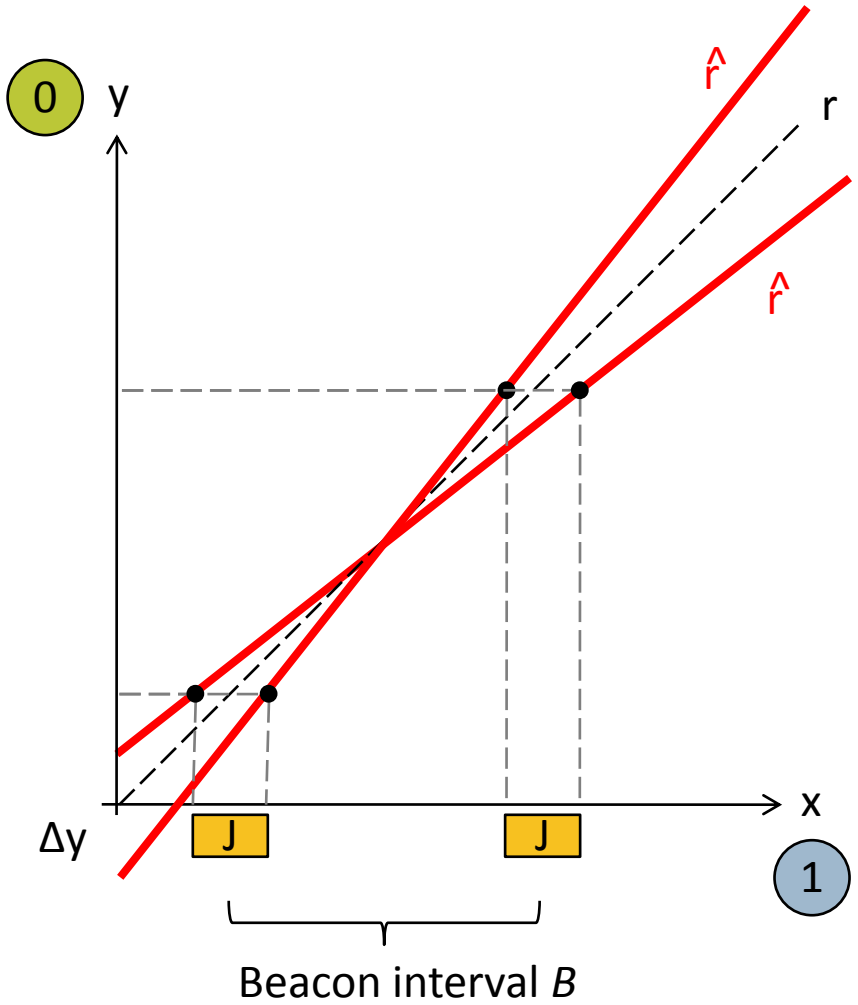
$$y(x) = \hat{r} \cdot x + \Delta y$$

clock offset

relative clock rate (estimated)

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:

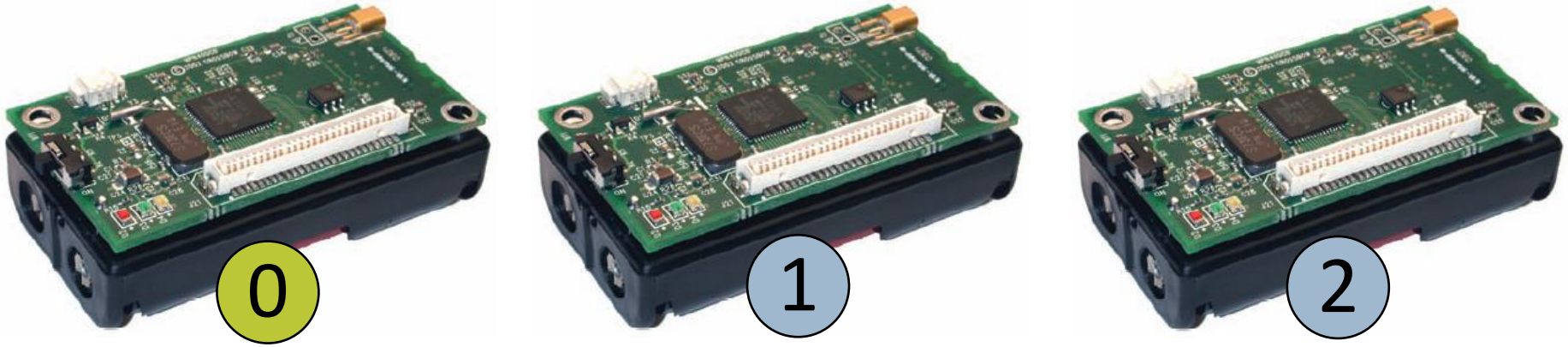
$$|\hat{y} - y| \sim \frac{J}{\sqrt{k}}$$

(complete proof is in the paper)

Synchronizing Nodes: Multi-hop

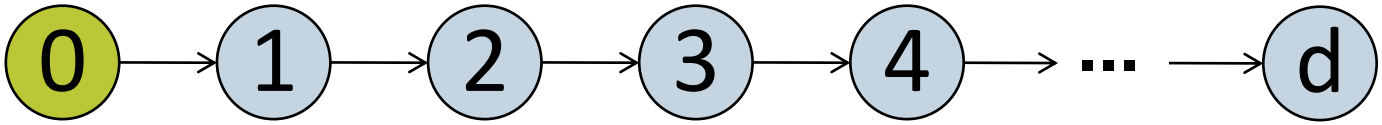
- How do we synchronize the clocks of **multiple** sensor nodes?

reference clock



Now we have a network of nodes!

- How does the network diameter affect synchronization errors?



- Examples for sensor networks with high diameter

  Bridge, road or pipeline monitoring

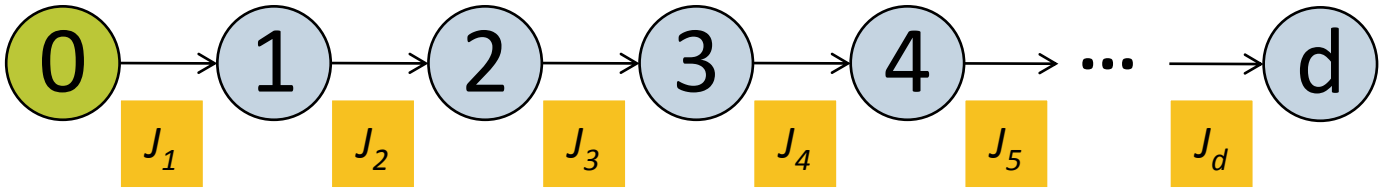


  Deployment at Golden Gate Bridge with 46 hops
  (Kim et al., IPSN'07)

Multi-hop Clock Synchronization

- Nodes forward their current estimate of the reference clock

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

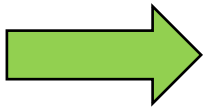


- Sum of the jitter grows with the square-root of the distance

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

Single-hop:

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



Multi-hop:

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

(proof is in the paper)

Outline

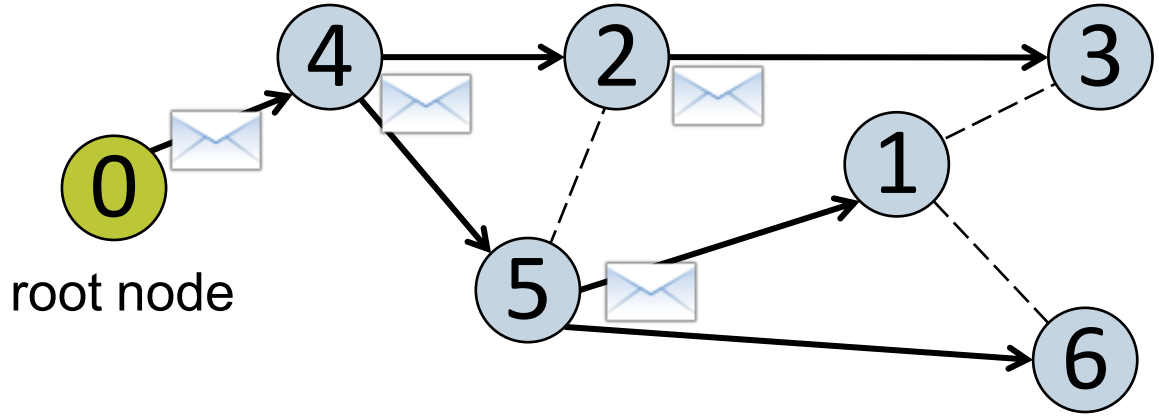
- Introduction
- Theory
- Practice

Clock Synchronization in Practice

- Flooding Time Synchronization Protocol (FTSP)
  - Nodes synchronize to a root (leader) node
  - Leader-election phase (by smallest id)
  - Periodic synchronization beacons (unaligned)
  - Linear-regression table to correct clock drift

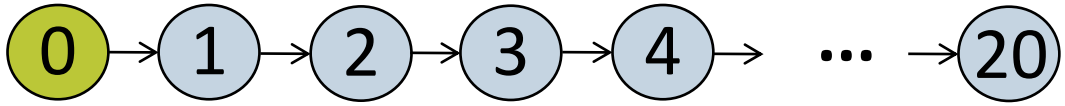
The Flooding Time Synchronization Protocol

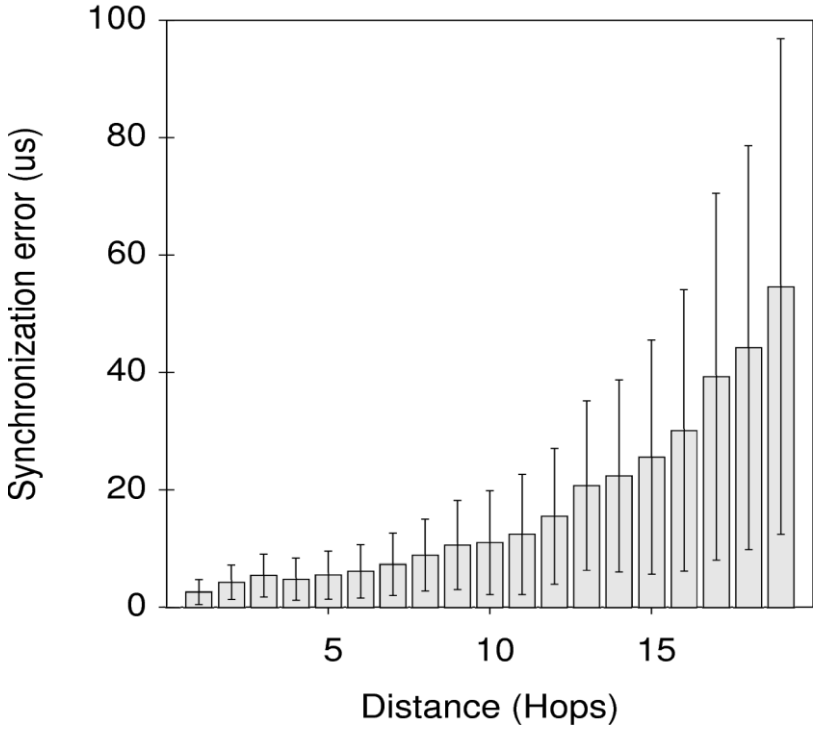
Maroti et al. (SenSys'04)



Testbed Experiments (FTSP)

- Measurement results from testbed with 20 Mica2 nodes



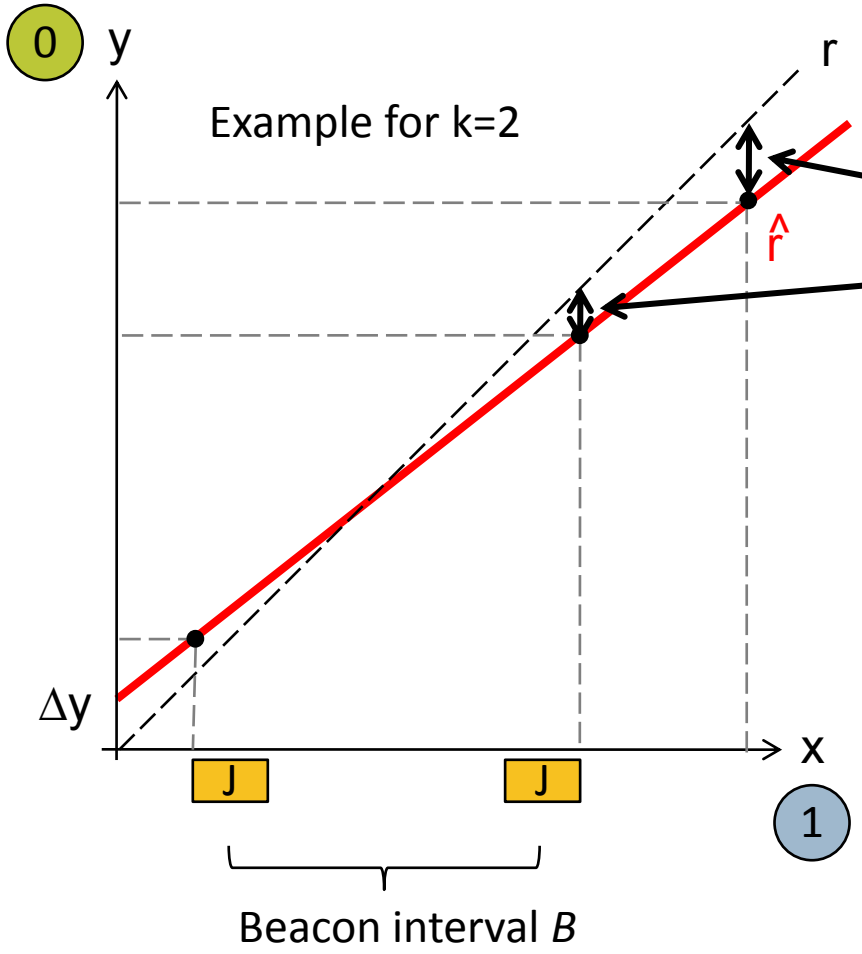


- Synchronization error grows **exponentially**
- Nodes far away from the root failed to synchronize with their parent node

Linear Regression (FTSP)

- FTSP uses linear regression to compensate for clock drift

  Jitter is amplified before it is sent to the next hop



synchronization error

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

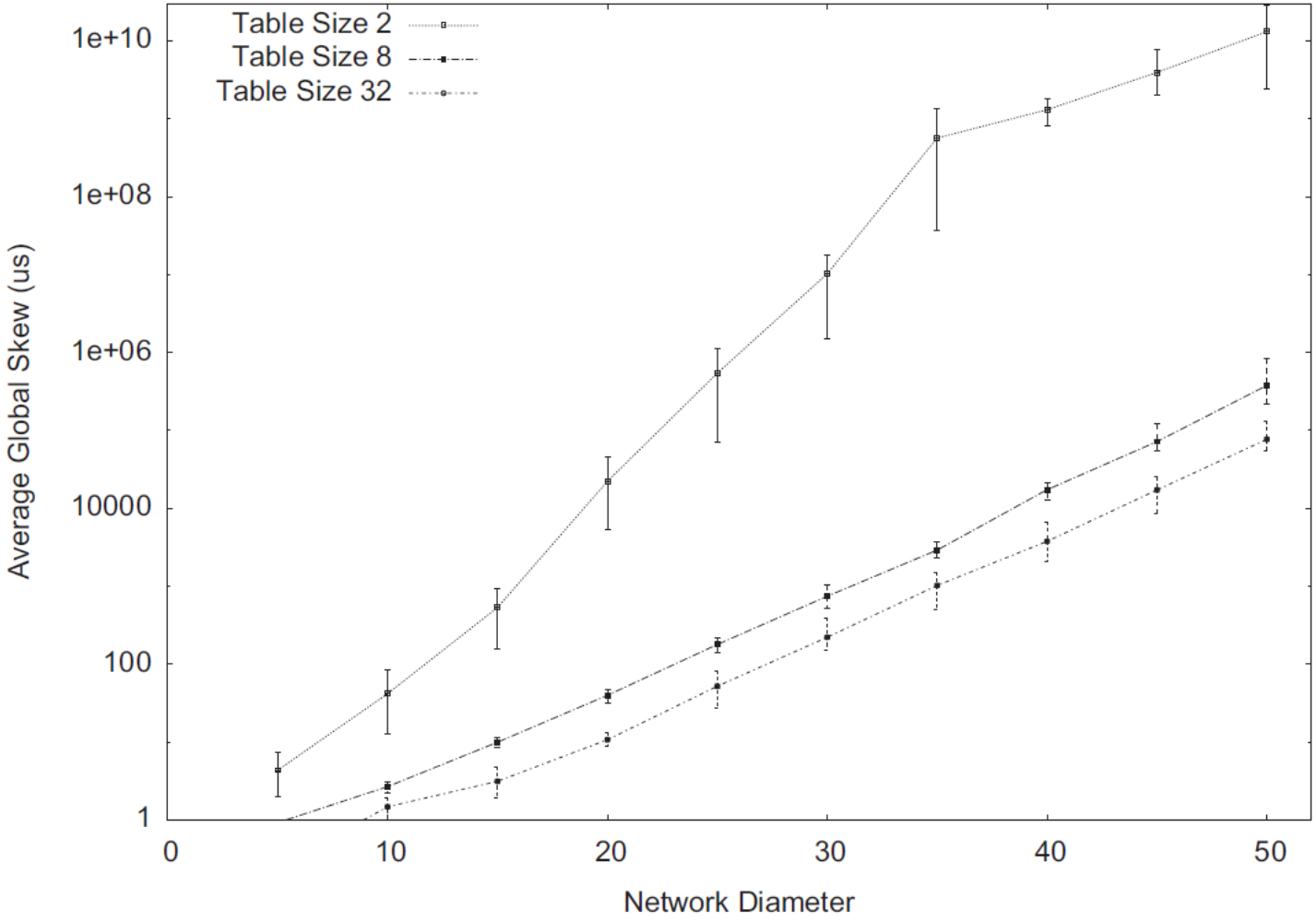
clock offset

relative clock rate (estimated)

Linear Regression (FTSP)

- Simulation of FTSP with regression tables of different sizes (k = 2, 8, 32)

Log Scale!

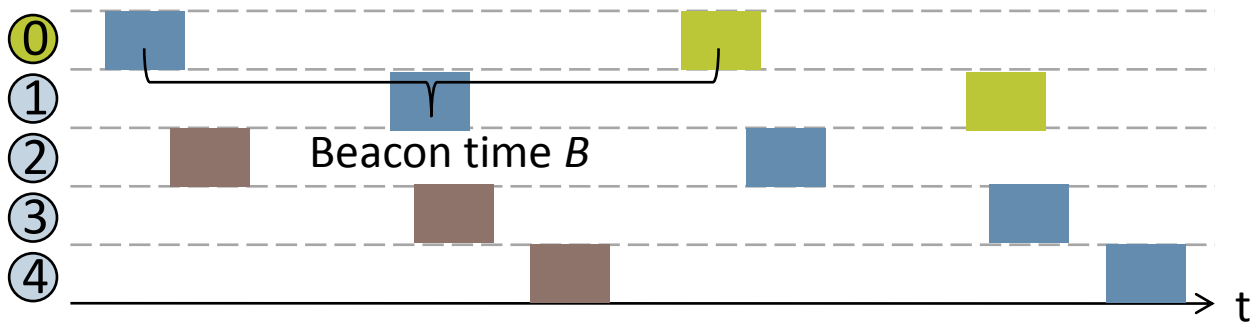


The PulseSync Protocol

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

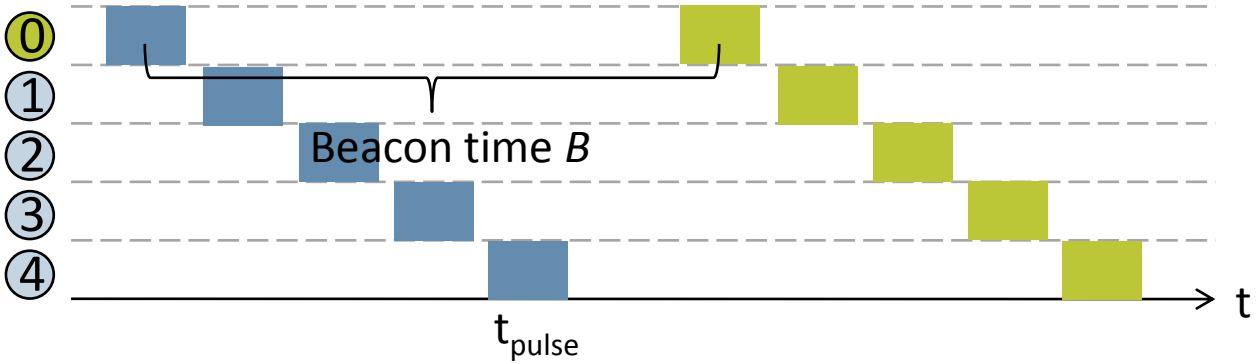
FTSP

Expected time
= D·B/2



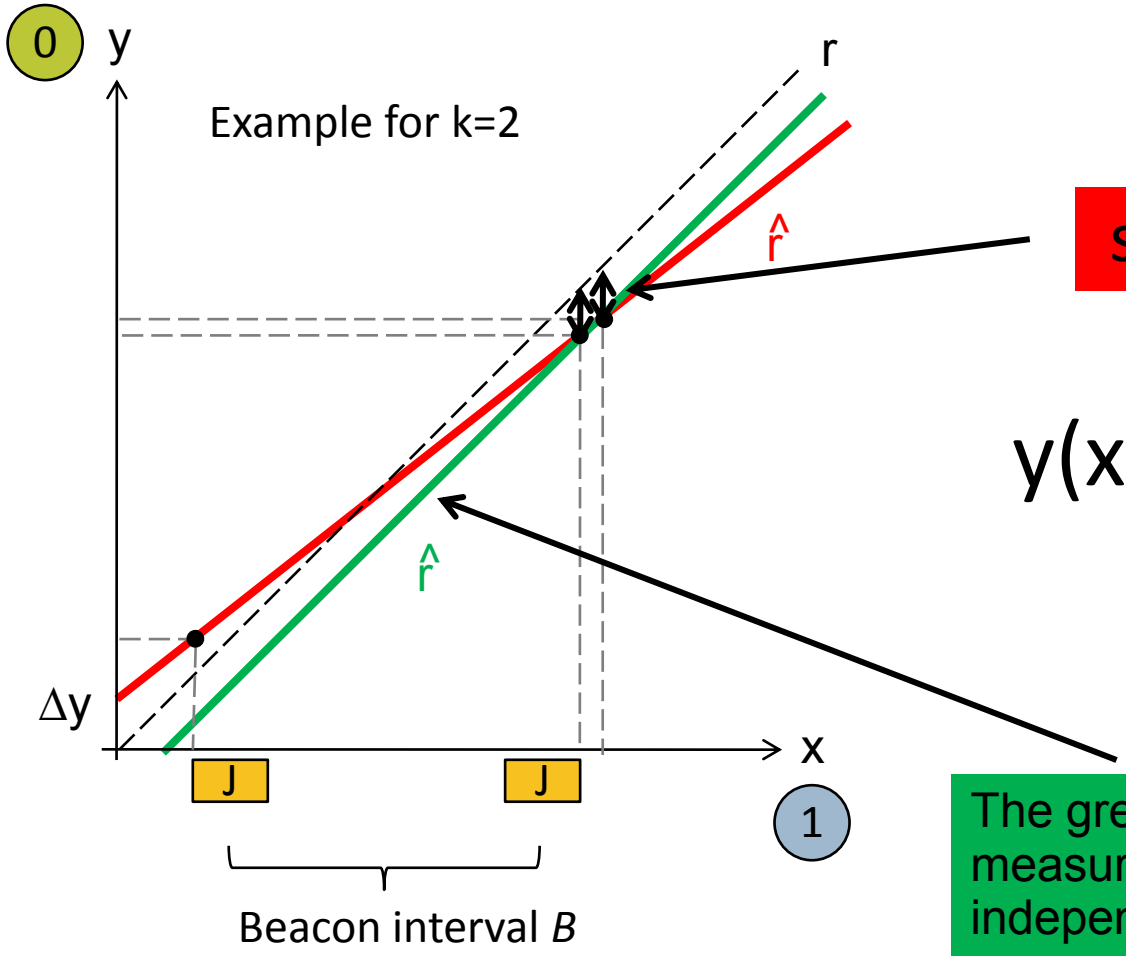
PulseSync

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

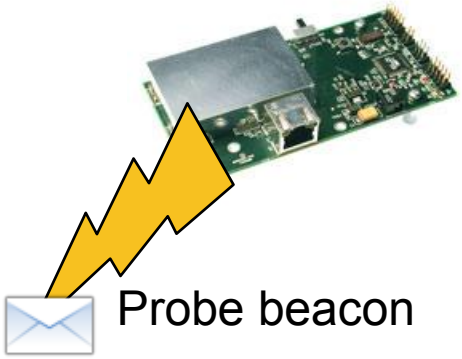
relative clock rate

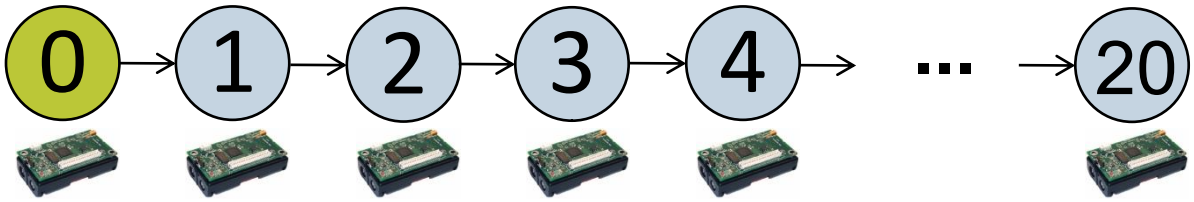
The green line is calculated using k measurement points that are statistically independent of the red line (see paper).

Evaluation

- Testbed setup
  - 20 Crossbow Mica2 sensor nodes
  - PulseSync implemented in TinyOS 2.1
  - FTSP from TinyOS 2.1

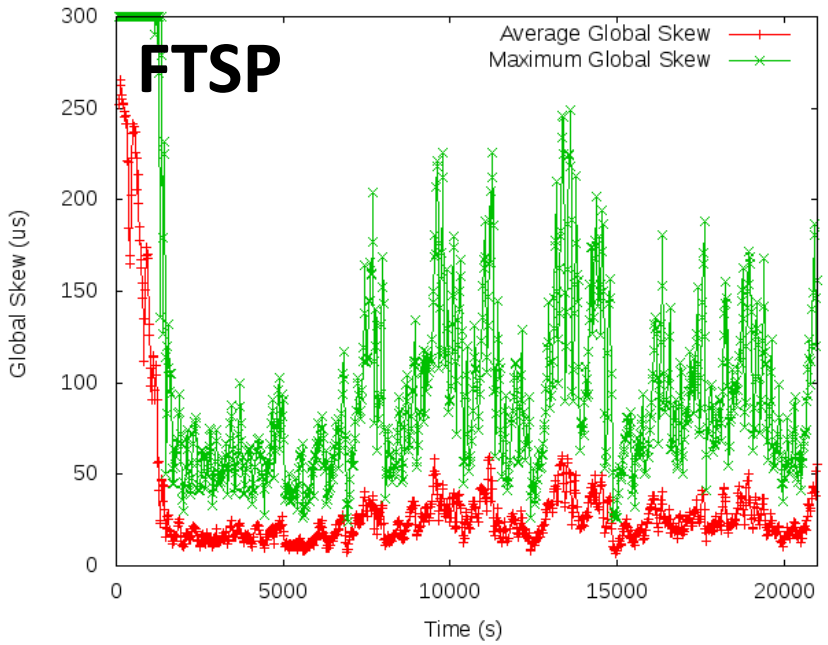
- Network topology
  - Single-hop setup, basestation
  - Virtual network topology (white-list)
  - Acknowledgments for time sync beacons

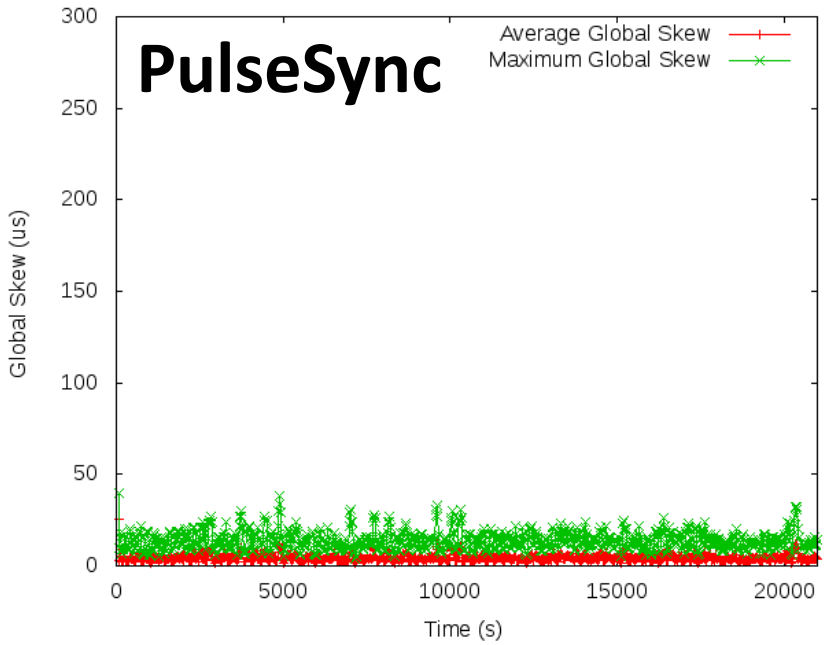




Experimental Results

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

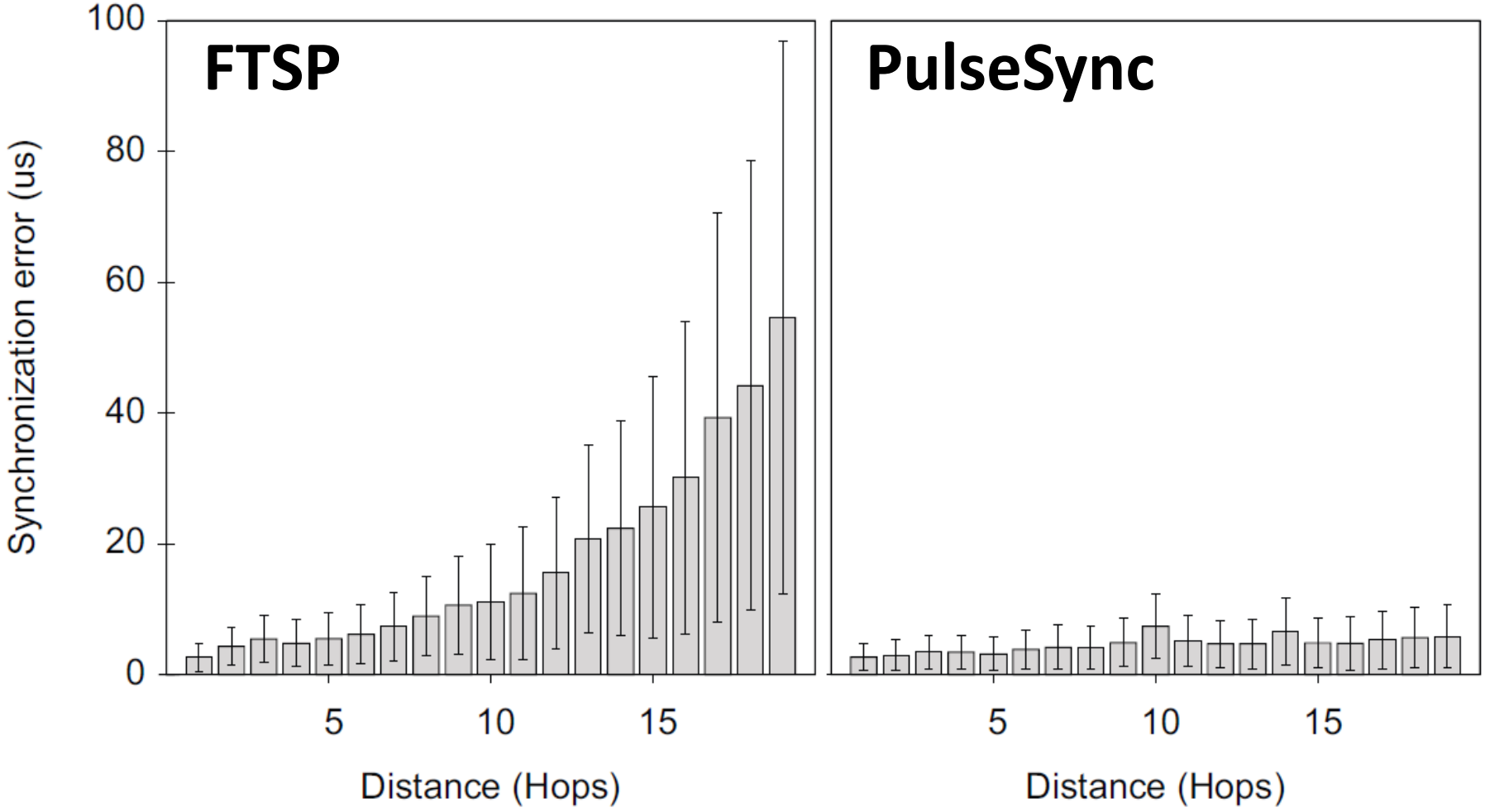




Synchronization Error	FTSP	PulseSync
Average (t>2000s)	23.96 µs	4.44 µs
Maximum (t>2000s)	249 µs	38 µs

Experimental Results (2)

- Sychnronization Error vs. distance from root node



Outlook

- Extension to more general network topologies
- Schedule synchronization beacons without collisions
  - Time information has to propagate quickly through the network
  - Avoid loss of synchronization pulses due to collisions

This is known as **wireless broadcasting**, a well-studied problem (in theory)

Conclusions

- Theoretical insights into clock synchronization
  - Lower bound on the global clock skew

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

- PulseSync: a novel clock synchronization algorithm
  - Flooding sync pulses at high speed through the network

  - Matches the lower bound on the global skew (shown in the paper)

- Testbed experiments on a 20-node line topology
  - Prototype implementation of PulseSync
  - PulseSync outperforms FTSP for this setting