Wireless Sensor Networks

13th Lecture

06.12.2006

CoNe Freiburg

University of Freiburg
Computer Networks and Telematics
Prof. Christian Schindelhauer

Christian Schindelhauer

schindel@informatik.uni-freiburg.de

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

Overview

- The time synchronization problem
- Protocols based on sender/receiver synchronization
- Protocols based on receiver/receiver synchronization
- Summary

Clocks in WSN nodes

- **Often, a *hardware clock* is present:**
  - Oscillator generates pulses at a fixed nominal frequency
  - A counter register is incremented after a fixed number of pulses
    - Only register content is available to software
    - Register change rate gives achievable time resolution
  - Node i's register value at real time t is $H_i(t)$
    - Convention: small letters (like t, t') denote real physical times, capital letters denote timestamps or anything else visible to nodes

- **A (node-local) software clock is usually derived as follows:**

$$L_i(t) = \theta_i \, H_i(t) + \phi_i$$

    - (not considering overruns of the counter-register)
  - θ_i is the (drift) rate, ϕ_i the phase shift
  - Time synchronization algorithms modify θ_i and ϕ_i, but not the counter register

Synchronization accuracy / agreement

- **External synchronization:**
  - synchronization with external real time scale like UTC
  - Nodes i=1, ..., n are accurate at time t within bound δ when

$$|L_i(t) - t| < \delta \text{ for all } i$$

    - Hence, at least one node must have access to the external time scale

- **Internal synchronization**
  - No external timescale, nodes must agree on common time
  - Nodes i=1, ..., n agree on time within bound δ when

$$|L_i(t) - L_j(t)| < \delta \text{ for all } i,j$$

Fundamental Building Blocks

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

- **Resynchronization event detection block:**
  - when to trigger a time synchronization round?
    - Periodically or after external event
- **Remote clock estimation block**
  - figuring out the other nodes clocks with the help of exchanging packets
- **Clock correction block**
  - compute adjustments for own local clock based on estimated clocks of other nodes
- **Synchronization mesh setup block**
  - figure out which node synchronizes with which other nodes

Constraints for Time Synchronization in WSNs

- **An algorithm should scale**
  - large networks of unreliable nodes
- **Diverse precision requirements**
  - from ms to tens of seconds
- **Use of extra hardware**
  - GPS receivers
  - Special radio receivers is mostly not an option
- **Low mobility**
- **Often no fixed upper bounds on packet delivery times**
  - due to MAC delays, buffering, ...
- **Negligible propagation delay between neighboring nodes**
- **Manual node configuration is not an option**

Post-facto synchronization

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

- **Basic idea:**
  - Do not nodes synchronized all the time
    - substantial energy costs due to need for frequent resynchronization
    - Especially true for networks which become active only rarely
  - When a node observes an external event at time t
    - it stores its local timestamp $L_i(t)$
    - achieves synchronization with neighbor node / sink node
    - and converts $L_i(t)$ accordingly
- **Can be implemented in different ways**

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

Overview

- The time synchronization problem
- Protocols based on sender/receiver synchronization
- Protocols based on receiver/receiver synchronization
- Summary

Protocols based on sender/receiver synchronization

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

- **Receiver synchronizes to the clock of a sender**

- **We have to consider two steps:**
  - Pair-wise synchronization:
    - how does a single receiver synchronize to a single sender?
  - Network-wide synchronization
    - how to figure out who synchronizes with whom to keep the whole network / parts of it synchronized?

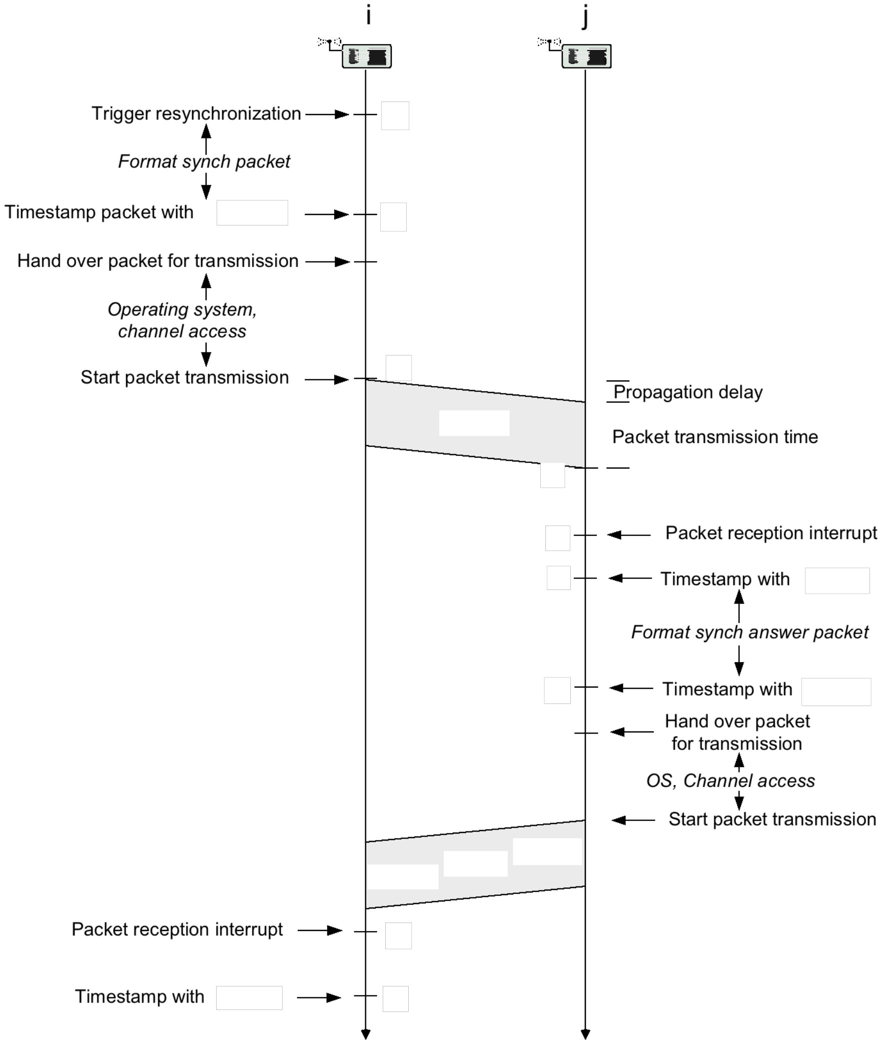
- **The classical NTP protocol [Mills, RFC 1305] belongs to this class**

LTS – Lightweight Time Synchronization

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

- **Overall goal**
  - synchronize the clocks of sensor nodes to one reference clock
  - e.g. equipped with GPS receiver
- **It allows to synchronize**
  - the whole network,
  - or parts of it
  - also supports post-facto synchronization
- **It considers only phase shifts**
  - does not try to correct different drift rates
- **Two components:**
  - pairwise synchronization: based on sender/receiver technique
  - networkwide synchronization: minimum spanning tree construction with reference node as root

LTS – Pairwise Synchronization



- **Assumptions:**
  - no drift
  - same hardware, same OS, same software
- **Goal: compute**

$$\Delta = L_i(t_1) - L_j(t_1)$$

- **Further assumptions**

$$\Delta = L_i(t_k) - L_i(t_k)$$

$$L_j(t_5) - L_j(t_1) = L_i(t_5) - L_i(t_1)$$

$$\approx$$

$$L_i(t_8) - L_i(t_6) = L_j(t_8) - L_j(t_6)$$

- **Solution:**

$$\Delta = \frac{L_i(t_8) - L_j(t_6)}{2} - \frac{L_j(t_5) - L_i(t_1)}{2}$$

LTS – Pairwise Synchronization Discussion

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

- **Sources of inaccuracies:**
  - MAC delay
  - interrupt latencies upon receiving packets
  - Delays between packet interrupts and time-stamping operation
  - Delay in operating system and protocol stack

- **Improvements:**
  - i timestamps its packet after the MAC delay, immediately before transmitting the first bit
    - removes the uncertainty due to i's operating system / protocol stack and the MAC delay
  - j timestamps received packets as early as possible
    - e.g. in the interrupt routine
    - removes the delay between packet interrupts and timestamping from the uncertainty
    - leaves only interrupt latencies

  ➔ hard to do when standard hardware is used

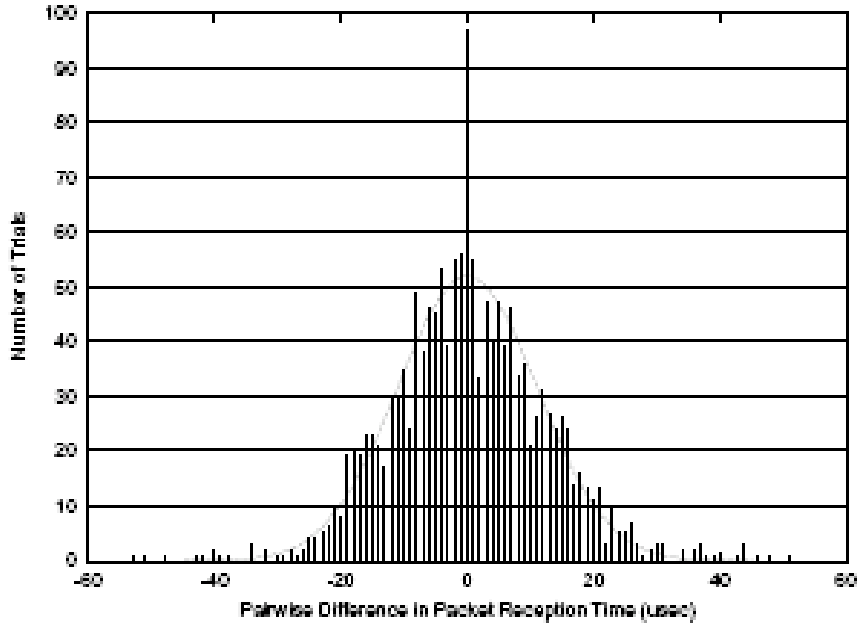
  ➔ easy when full source code of MAC and direct access to hardware are available

LTS – Pairwise Synchronization – Error Analysis

University of Freiburg
Institute of Computer Science
Computer Networks and Telematics
Prof. Christian Schindelhauer

- **Elson et al measured pairwise differences in timestamping times at a set of receivers when timestamping happens in the interrupt routine (Berkeley motes)**
- **Estimated distribution:**
  - normal random variable (rv) with zero mean/stddev of 11.1 μs
- **Additional assumption: uncertainty on the transmitter side has same distribution and is independent**
- **Hence:**
  - total uncertainty is a zero-mean normal rv with variance 4 σ^2
  - For a normal rv 99% of all outcomes have maximum distance of 2.3 σ to mean

  ➔ the maximum synchronization error is with 99% probability smaller than 2.3 * 2 * σ



Thank you

(and thanks go also to Andreas Willig for providing slides)

CoNe Freiburg

University of Freiburg
Computer Networks and Telematics
Prof. Christian Schindelhauer

Wireless Sensor Networks
Christian Schindelhauer
schindel@informatik.uni-freiburg.de

13th Lecture
06.12.2006