Time Synchronization for Sensor Networks

Kay Römer
ETH Zürich
Switzerland

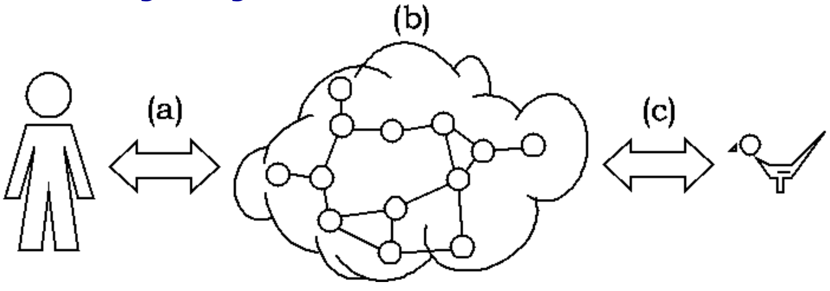
Outline

- Need for synchronization
- Why research needed?
- Variants of time sync
- Characteristics of sensor nodes
- One-shot synchronization
- Continuous synchronization
- Synchronization of multiple nodes
- Examples

1. Need for Synchronization

Kay Römer <roemer@inf.ethz.ch>

Why Synchronized Time?



- (a) Interaction user – sensornet
- (b) Coordination among sensor nodes
- (c) Interaction sensornet – real world

Kay Römer <roemer@inf.ethz.ch>

User - Sensornet

- Task specification
  - *When* shall be observed?
- Interpretation of sensor data
  - *When* did an event occur?
  - How *long* did it last?
  - How *fast* did an object move?

Kay Römer <roemer@inf.ethz.ch>

Sensornet – Real World

- Merging distributed observations
  - Multiple sensor nodes cooperate
  - Ex. velocity: How long did it take an object to move from one node to another?
- Separation of multiple events
  - Which observation belong to which event?
  - Time between observations!
- Consistent measurements
  - Multiple nodes must measure concurrently if real world changes fast!

Kay Römer <roemer@inf.ethz.ch>

Coordination of Sensor Nodes

- Wireless communication
  - Exclusive access to communication channel
- Energy efficiency
  - Radio duty cycling
  - Concurrent wakeup needed
- Message ordering
  - Processing of messages in the order of their generation
- Localization
  - Propagation time ~ distance

Precision?

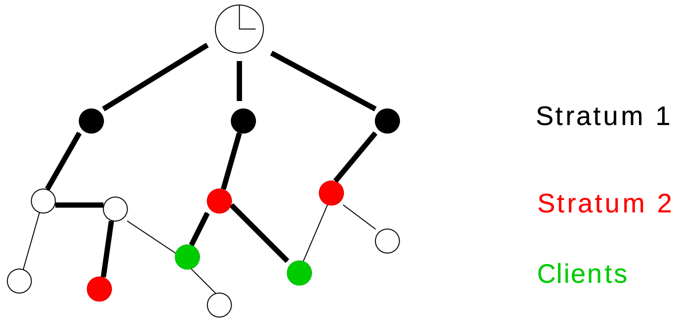
- Highly dependent on application
- From few micro seconds …
  - Communication
  - Localization
  - Data fusion
- … to minutes
  - Activation (10am to 1pm …)

2. Why Research needed?

Kay Römer <roemer@inf.ethz.ch>

Why Novel Approaches?

- Problems with Network Time Protocol (NTP)
  - Infrastructure
  - Energy
  - Network dynamics
  - Configuration



Kay Römer <roemer@inf.ethz.ch>

Infrastructure

- Stratum 1 servers access accurate global time
  - GPS
  - DCF77
- In sensornets often impossible or undesirable

Energy

- Always on: servers constantly listen for requests from clients
- All nodes always synchronized with best precision
- Scarce energy in sensornets
  - Communication dominant energy consumer
  - Even „listening“ is expensive

Network Dynamics

- NTP relies on (mostly) static network
  - Fixed paths between neighbors in overlay network
- Sensornets rather dynamic
  - Nodes added, die, mobile, temporary disconnects and partitions
  - Network paths rather unstable

Configuration

- Static configuration of overlay network
  - Who is my parent
  - Administrator responsible for rather small network
- Static configuration often infeasible in sensor networks
  - Configuration must be adapted for dynamic network topology
  - One admin would have to take care of thousands of nodes

3. Variants of Synchronization

Variants of Synchronization

- Internal vs. external
- Lifetime: always vs. on demand
- Scope: all vs. subsets
- Rate vs. offset
- Scale transformation vs. clock synchronization
- Instants vs. intervals

Internal vs. External

- Internal
  - Nodes agree on a common time, no matter which
- External
  - Common time is pre-specified
  - All nodes must synchronize to this time

Lifetime

- For how long most nodes be synchronized?
  - always
  - Only when needed by application
- Energy consumption!

Scope

- Which nodes must be synchronized?
  - All
  - Only those where needed
- Energy consumption!

Kay Römer <roemer@inf.ethz.ch>

Rate vs. Offset

- Rate synchronization
  - Clocks run at same rate, but may show different absolute time
  - Sufficient for measuring durations
- Offset synchronization
  - Clocks show same time when read concurrently
  - Necessary for measuring absolute time

Kay Römer <roemer@inf.ethz.ch>

Scale Xform vs. Clock sync

- Clock synchronization
  - Local node clocks are adjusted to common time
- Scale transformation
  - Clocks run unsynchronized, defining different time scales
  - Time stamps are transformed between scales

Kay Römer <roemer@inf.ethz.ch>

Instants vs. Intervals

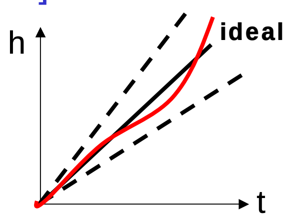
- Instants
  - Clock output: t = 5
  - Suggests correct time, but is almost always (slightly) incorrect
- Intervals
  - Clock output: $t \in [4, 5]$
  - Suggests imprecise time, but is correct

Kay Römer <roemer@inf.ethz.ch>

4. Characteristics of Sensor Nodes

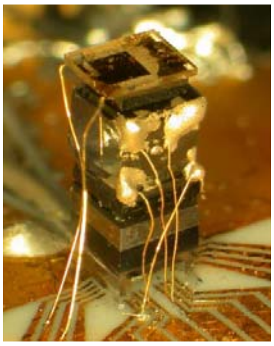
Hardware Clocks

- Count intervals of (ideally) fixed duration
- Typically based on quartz oscillators
- Imprecision due to clock drift
  - Measured as ppm (parts per million)
  - Deviation from real time
  - Typically ∈ [10ppm, 100ppm] in WSN
  - Ex. After 1h deviation of 0.04 – 0.4 s



Atomic Clocks?

- Smallest atomic clock (NIST, Boulder)
  - 5 mm^3
  - 30 mA, 2.5 V
  - Precision: $3.5x10^{-4}$ ppm
  - 1s deviation in 300 years
  - Better by factor 10^5!
- One-time factory synchronization sufficient



Communication

- Network messages used for synchronization
- Precision depends on variability of message delay
  - Send time (+)
    - Application sends message -> MAC tries to send
  - Access time (++), several milli seconds
    - Delay until communication channel free
  - Propagation time (--), tens of nano seconds
    - Propagation of radio waves
  - Receive time (-), bit duration, tens of micro seconds
    - Arrival of message -> Application is notified

Strategy 1: MAC Timestamping

- Integrate timestamping with the MAC layer
- Largely eliminates variability due to
  - Send time
  - Access time
  - Receive time

Strategy 2: Broadcast

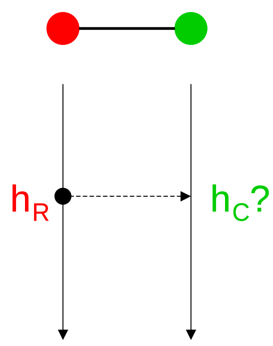
- Synchronize multiple receivers with one broadcast message
- All receivers see same
  - Send time
  - Access time

5. One-Shot Synchronization of One Node

Kay Römer <roemer@inf.ethz.ch>

Synchronization of One Node

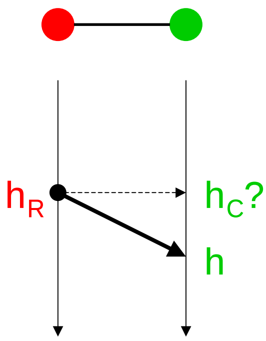
- R: Reference
- C: Client
- Synchronization problem:
  - When R's clocks shows h_R, which time h_C shows C's clock?
- Client could then correct ist clock by $h_R - h_C$



Kay Römer <roemer@inf.ethz.ch>

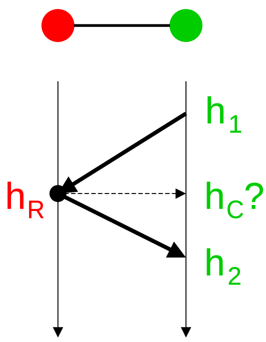
Unidirectional

- Reference sends message containing h_R
- Client: $h_C := h$
- Any number of clients with one broadcast message
- Upper bound for h_C



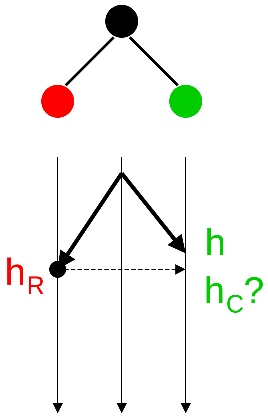
Round Trip

- Client sends message, reference answers with h_R
- Client: $h_C := (h2 - h1)/2$
- Precise when constant delay
- Linear overhead with multiple clients
- Upper and lower bound on h_C



Reference Broadcast

- Beacon broadcasts message to reference and client
- Client: $h_C := h$
- Precise, broadcast arrives almost concurrently
- Another message exchange needed to send h_R to client
- Can be done with constant overhead for any number of clients (how?)



6. Continuous Synchronization

Continuous Sync

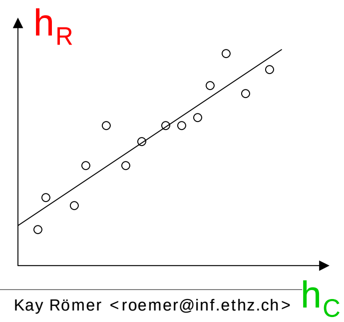
- Precision of one-shot synchronization degrades quickly due to drift
- To maintain synchronization:
  - Repeat one-shot sync
  - Adjust rate of the clocks

Repeated One-Shot Sync

- Overall precision depends on frequency of one-shot synchronization
- Frequency limited by duration of synchronization
- Drawback: large overhead and energy consumption

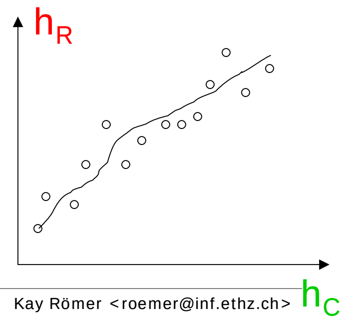
Linear Regression

- How much faster/slower does client's clock run compared to reference?
- Perform multiple one-shot measurements and fit line
- Slope equals average rate difference
  - slope 1 = same rate
- Drawbacks
  - Memory overhead
  - Outliers have quadratic impact

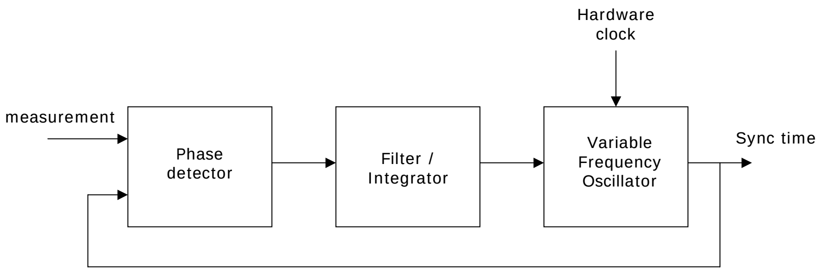


Phase-Locked Loops

- Running estimation of sync time in client
- Regular one-shot measurements
  - If new measurement is larger than estimate, then reduce clock rate
  - If new measurement is smaller than estimate, then increase clock rate
- Drawbacks
  - Long convergence time

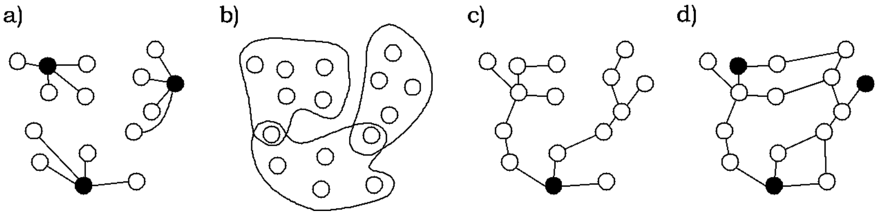


Phase-Locked Loops II



7. Multiple Clients

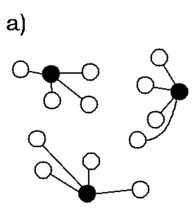
Multiple Clients



- (a) Out-of-band
- (b) Clustering
- (c) Tree
- (d) Unstructured

Out-of-Band

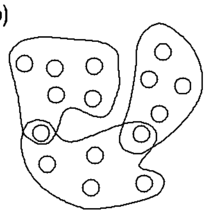
- Every client has at least one reference neighbor
- All references are synchronized out-of-band (GPS, …)
  - Infrastructure!



Clustering

- Network is clustered
  - One cluster head (may act as reference)
  - Gateways (interconnect clusters)
  - Slaves (everything else)
- Overhead for construction and maintenance
- Each cluster performs internal synchronization
- Gateways translate between clusters

b)

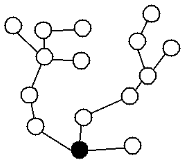


Kay Römer <roemer@inf.ethz.ch>

Tree

- Single reference root of a spanning tree
  - Overhead for construction and maintenance
- Children synchronize to its parent in the tree
  - Long paths may lead to inaccuracy
  - Neighbor nodes may be poorly synchronized

c)

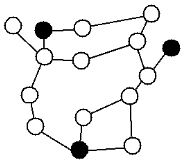


Kay Römer <roemer@inf.ethz.ch>

Unstructured

- No overlay topology at all
  - No overhead for maintenance
- Nodes use arbitrary paths for synchronization with reference nodes
  - Piggybacking!
  - Performance depends on network traffic

d)



 Kay Römer <roemer@inf.ethz.ch>

8. Examples

 Kay Römer <roemer@inf.ethz.ch>

Some Concrete Algorithms

- RBS: Reference Broadcast Synchronization
  - Jeremy Elson, OSDI 2002
- TPSN: Timing Sync Protocol for Sensor Networks
  - Saurabh Ganeriwal, Sensys 2003
- TSS: Time Stamp Synchronization
  - Kay Römer, MobiHoc 2001

RBS

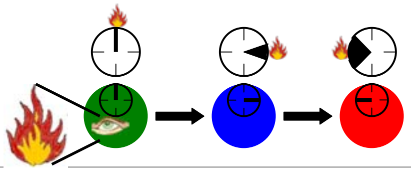
- Clustering
- Cluster head acts as beacon for reference broadcast
- Each slave computes offset and rate difference to all other slaves
  - Multiple broadcasts
  - Linear regression
- Gateway nodes transform timestamps between clusters
- Precision per hop on Berkeley motes approx. 11 us

TPSN

- Single master, tree-based
- Child synchronizes with parent using round-trip measurements
  - MAC layer timestamping
- Precision per hop on Berkeley motes approx. 17 us

TSS

- Clocks not synchronized
- Transformation of time stamps along message path
- Round trip measurements between neighbors
- Intervals
- Precision approx. 1 ms



Conclusions

- Old problem, new requirements, new platform characteristics, new solutions
  - Problems: scarce energy, no infrastructure, self-configuration, networks dynamics
  - „Enemies" of synchronization (drift, variable delays) especially distinctive

Further Reading

- K. Römer, P. Blum, L. Meier: „Time Synchronization and Calibration in Wireless Sensor Networks", in I. Stojmenovic (Editor), „Wireless Sensor Networks", Wiley, Sept. 2005
  - Preprint available on www.inf.ethz.ch/~roemer