Principles of Wireless Sensor Networks

https://www.kth.se/social/course/EL2745/

Lecture 11

Time Synchronization

Carlo Fischione
Associate Professor of Sensor Networks
e-mail:carlofi@kth.se
http://www.ee.kth.se/∼carlofi/

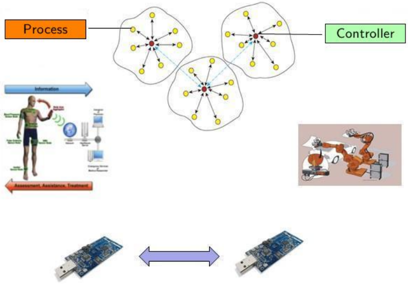
KTH Royal Institute of Technology
Stockholm, Sweden

October 9, 2014

Course content

- Part 1
  - Lec 1: Introduction to WSNs
  - Lec 2: Introduction to Programming WSNs
- Part 2
  - Lec 3: Wireless Channel
  - Lec 4: Physical Layer
  - Lec 5: Medium Access Control Layer
  - Lec 6: Routing
- Part 3
  - Lec 7: Distributed Detection
  - Lec 8: Static Distributed Estimation
  - Lec 9: Dynamic Distributed Estimation
  - Lec 10: Positioning and Localization
  - Lec 11: Time Synchronization
- Part 4
  - Lec 12: Wireless Sensor Network Control Systems 1
  - Lec 13: Wireless Sensor Network Control Systems 2
  - Lec 14: Summary and Project Presentations

Previous lecture



How to estimate the position of fixed and mobile nodes?

Today's learning goals

- Which measurements are used for synchronizing the nodes?
- What is the hardware clock?
- What is the software clock?
- How to synchronize pair of nodes?
- How to synchronize a network of nodes?

Outline

- Basics of time synchronization
- Synchronization protocols

Outline

- Basics of time synchronization
  - Hardware clock - Software clock
  - Message exchanges
- Synchronization protocols
  - Time synchronization protocol
    - Estimation based on LS
    - Estimation based on MMS
  - Distributed clock synchronization

Basics of time synchronization

Time synchronization is defined as the procedure for at least two nodes to have a common reference clock

A typical node possesses an oscillator of a specified frequency and a counter register, which is incremented in hardware after a certain number of oscillator pulses

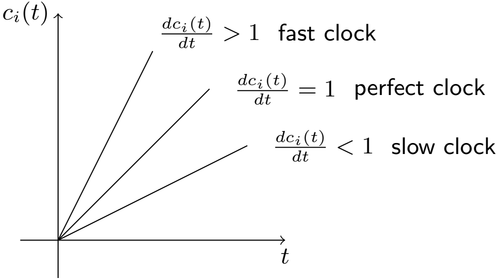
The value of the **hardware clock** of node i at real time t can be represented as $H_i(t)$

The time distance between two increments (ticks) determines the achievable **time resolution**

- Consider two nodes: node i, node j. Then,
    - **Clock offset** is defined as the difference between time at node i and time at node j
    - **Clock rate** is the frequency at which clock progresses
    - **Clock skew** represents the difference in the frequency of two clocks

Basics of time synchronization

Let us define $c_i\left(t\right)$ as the software time measured at node i at time t



Basics of time synchronization

Clock rate: $\frac{dc_i(t)}{dt}$

Defining ρ_i as the drift rate at node i,

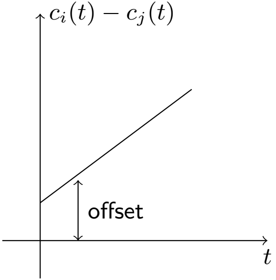
$$1 - \rho_i \leq \frac{dc_i(t)}{dt} \leq 1 + \rho_i$$

Two synchronized nodes, node i and node j, before being resynchronized can drift of at maximum $2\rho_{\max}$, that is

$$\frac{dc_i(t)}{dt} - \frac{dc_j(t)}{dt} \leq 2\rho_{\max}$$

where $2\rho_{\max} \cdot \tau_{\text{synch}} < \delta_{\max}$ and $\delta_{\max}$ a precision parameter that is equal to the maximum offset between two clocks

Basics of time synchronization



Offset is worsening as time goes by!

How can we model the offset between node i and node j ?

$$c_i(t) - c_j(t) = t_0 + \Delta f(t) \cdot t + \Delta\tau(t)$$

- t_0: constant offset
- $\Delta f(t)$: time dependent frequency offset
- $\Delta\tau(t)$: jitter due to noise

Frequency offset $\Delta f(t)$ is due to different rates among the clocks and environmental effects (e.g. temperature, humidity)

Basics of time synchronization

How do we synchronize the clocks?

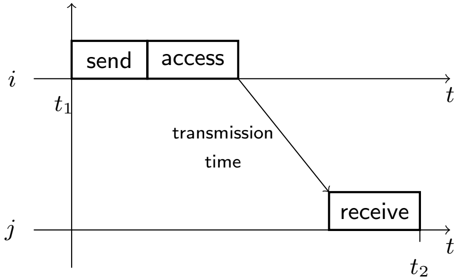
Problem: Non-determinism of communication delay

Outline

- Basics of time synchronization
  - Hardware clock - Software clock
  - Message exchanges
- Synchronization protocols
  - Time synchronization protocol
    - Estimation based on LS
    - Estimation based on MMS
  - Distributed clock synchronization

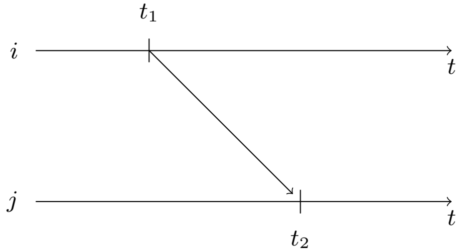
Message exchanges

Consider two nodes: node i and node j.
A message is sent to synchronize i with j



Message exchanges

One way message exchange



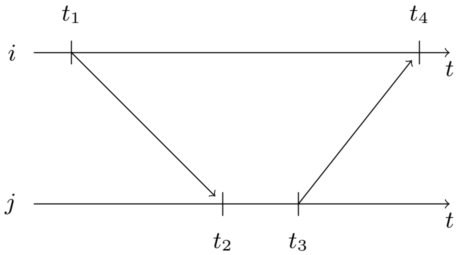
$$c_i\left(t_1\right) = t_1 + n_1$$
$$c_j\left(t_2\right) = t_2 = t_1 + D + \delta + n_2$$

propagation delay (D), offset (δ)

Node j makes an estimate of the offset and adjusts its clock if $D \simeq 0$ (negligible) and $n_1 \approx n_2$

Message exchanges

Two way message exchange (in case D is non negligible)



$t_2 = t_1 + D + \delta + n_2$

$t_4 = t_3 + D - \delta + n_4$

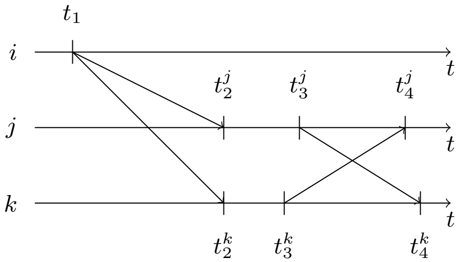
t_1, t_2, t_3, t_4 measured $\Rightarrow$ solve for D and δ , assuming that $n_2 \approx n_4$

Therefore,

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

Message exchanges

Receiver-receiver synchronization



$t_4^k = t_3^j + \delta_{jk} + n_4^k$

$t_3^j = t_2^j + \Delta_j = t_1 + \delta_{ij} + \Delta_j + n_3^j$

$t_4^j = t_3^k + \delta_{kj} + n_4^j$

$t_3^k = t_2^k + \Delta_k = t_1 + \delta_{ik} + \Delta_k + n_3^k$

where Δ_j and Δ_k are known and δ_{ik}, δ_{kj}, δ_{jk}, δ_{kj} unknown

Outline

- Basics of time synchronization
  - Hardware clock - Software clock
  - Message exchanges
- Synchronization protocols
  - Time synchronization protocol
    - Estimation based on LS
    - Estimation based on MMS
  - Distributed clock synchronization

Time synchronization protocol

Consider two nodes, node i and node j, with different drifts and offsets

Let us define $c_i(t)$ as the clock of node i and $c_j(t)$ as the clock of node j

Assume also,

$$c_j(t) = a_0 + a_1 \cdot c_i(t) + n_{ij}$$

For synchronization, measurements are required in order to determine a_0 and a_1

Time synchronization protocol

$$c_j(t) = a_0 + a_1 \cdot c_i(t) + n_{ij}$$

STEP 1

$c_i(t_0) \triangleq x_0$ (measured)

$c_j(t_0) = a_0 + a_1 \cdot c_i(t_0) + n_{ij,0} = a_0 + a_1 \cdot x_0 + n_{ij,0} \triangleq y_0 + n_{ij,0}$ (measured)

STEP 2

$c_i(t_1) \triangleq x_1$

$c_j(t_1) = a_0 + a_1 \cdot x_1 + n_{ij,1} \triangleq y_1 + n_{ij,1}$

$\vdots$

STEP n

$c_i(t_{n-1}) \triangleq x_{n-1}$

$c_j(t_{n-1}) = a_0 + a_1 \cdot x_{n-1} + n_{ij,n-1} \triangleq y_{n-1} + n_{ij,n-1}$

Time synchronization protocol

Putting all the measurements together, we end up in the following system of equations

$$A \cdot X + N = Y$$

where

$$A = \begin{bmatrix} x_0 & 1 \\ x_1 & 1 \\ \vdots & \vdots \\ x_{n-1} & 1 \end{bmatrix} \qquad X = \begin{bmatrix} a_1 \\ a_0 \end{bmatrix} \qquad N = \begin{bmatrix} n_{ij,0} \\ n_{ij,1} \\ \vdots \\ n_{ij,n-1} \end{bmatrix} \qquad Y = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix}$$

Estimation based on LS

LS estimator of X

$$\widehat{X} = L \cdot Y$$

where

$$L = \left(A^T \cdot A\right)^{-1} A^T$$

$$A^T \cdot A = \begin{bmatrix} \sum_{i=0}^{n-1} {x_i}^2 & \sum_{i=0}^{n-1} x_i \\ \sum_{i=0}^{n-1} x_i & n \end{bmatrix} \qquad A^T \cdot Y = \begin{bmatrix} \sum_{i=0}^{n-1} x_i y_i \\ \sum_{i=0}^{n-1} y_i \end{bmatrix}$$

Estimation based on MMS

MMSE estimator of X

The MMSE estimator of X given that $Y = y$ is

$$P^{-1}\hat{X} = AR_N^{-1}y$$

with error covariance

$$P^{-1} = R_X^{-1} + A^T R_N^{-1} A$$

where R_N the covariance of zero mean Gaussian noise N

In this specific case, $\hat{X} = PAR_N^{-1}y$ where

$$P^{-1} = \begin{bmatrix} \sum_{k=0}^{n-1} \frac{x_k^2}{\sigma^2_{n_{ij,k}}} & \sum_{k=0}^{n-1} \frac{x_k}{\sigma^2_{n_{ij,k}}} \\ \sum_{k=0}^{n-1} \frac{x_k}{\sigma^2_{n_{ij,k}}} & \sum_{k=0}^{n-1} \frac{1}{\sigma^2_{n_{ij,k}}} \end{bmatrix}$$

and

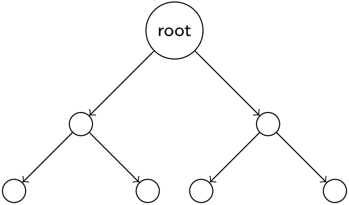
$$R_N^{-1} = \begin{bmatrix} \frac{1}{\sigma^2_{n_{ij,0}}} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sigma^2_{n_{ij,1}}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \end{bmatrix}$$

Outline

- Basics of time synchronization
  - Hardware clock - Software clock
  - Message exchanges
- Synchronization protocols
  - Time synchronization protocol
    - Estimation based on LS
    - Estimation based on MMS
  - Distributed clock synchronization

Distributed clock synchronization

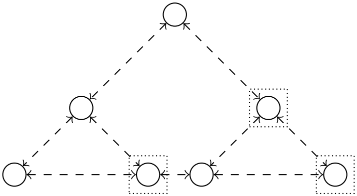
Previous part of lecture



- Nodes synchronize with the root
- Synchronization tends to deteriorate as we move away from the root

Distributed clock synchronization

Now, we would like to have a real common synchronization across the network



No root for synchronization - only P2P

Distributed clock synchronization

- Hardware clock of node i: $H_i(t) = \int_{t_0}^{t} h_i(\tau)\, d\tau + \phi_0(t_0)$
- Hardware clock rate of time τ: $h_i(\tau)$ with $1 - \rho_{\max} \leq h_i(t) \leq 1 + \rho_{\max}$ where ρ is the hardware clock drift
- Hardware clock offset at time t_0: $\phi_0(t_0)$
- Software clock of node i at time t: $c_i(t) = \int_{t_0}^{t} h_i(\tau)\, l_i(\tau)\, d\tau + \theta_i(t_0)$

Distributed clock synchronization

$$c_i(t) = \int_{t_0}^{t} h_i(\tau)\, l_i(\tau)\, d\tau + \theta_i(t_0)$$

where $l_i(\tau)$ the relative logical clock rate which can be properly designed to achieve synchronization and $\theta_i(t_0)$ a clock offset

We can then define the absolute logical clock rate of node i at time t as

$$x_i(t) \triangleq h_i(t) \cdot l_i(t)$$

If nodes have the term $h_i(t) \cdot l_i(t)$ similar, they are synchronized. How to do so?

Distributed clock synchronization

Compute iteratively for all nodes

$$x_i(k+1) \triangleq \frac{\sum_{j \in N_i(k)} x_j(k) + x_i(k)}{|N_i| + 1}$$

where

- $N_i(k)$ the set of neighbors of node i at time k
- $|N_i|$ the cardinality of neighbor nodes

Distributed clock synchronization

Set $X(k) = \begin{bmatrix} x_1(k) \\ \vdots \\ x_N(k) \end{bmatrix}$

$$X(k+1) = A(k) \cdot X(k)$$

where

$$A(k) = [a_{i,j}(k)] = \begin{cases} \frac{1}{|N_i|+1} & \text{if } i,j \text{ are connected,} \\ 0 & \text{otherwise.} \end{cases}$$

Distributed clock synchronization

Properties of matrix $A(k)$

- Matrix dimensions depend on the topology. They change everytime the neighbors of node i change
- Row stochasticity $$A(k) \cdot \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$$

Distributed clock synchronization

Proposition

Suppose the graph connected in the long run.

Then, the $\lim_{k \to \infty} X(k) = x_{ss} \cdot \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}$

The limit converges to a common value $\Rightarrow$ synchronization is achieved

Summary

- We have studied the basic of synchronization for sensor networks
- Synchronizing the nodes consists in applying estimation techniques

Next lecture

- The fourth and last part of the course starts: control over WSNs