Time Synchronization in Wireless Sensor Networks: Concepts and Research Trends

Djamel DJENOURI

CERIST Research Center, Algiers, Algeria

Outline

I Motivation & General Concepts

II Taxonomy of Protocols

III Estimators and Signal Processing Issues

Different delay distribution (Gaussian, Exponential, arbitrary), some lessons.

IV Implementation Issues

Challenges, impact of empirical parameters, lessons from existing implementations, etc.

V Some Proposed Solutions

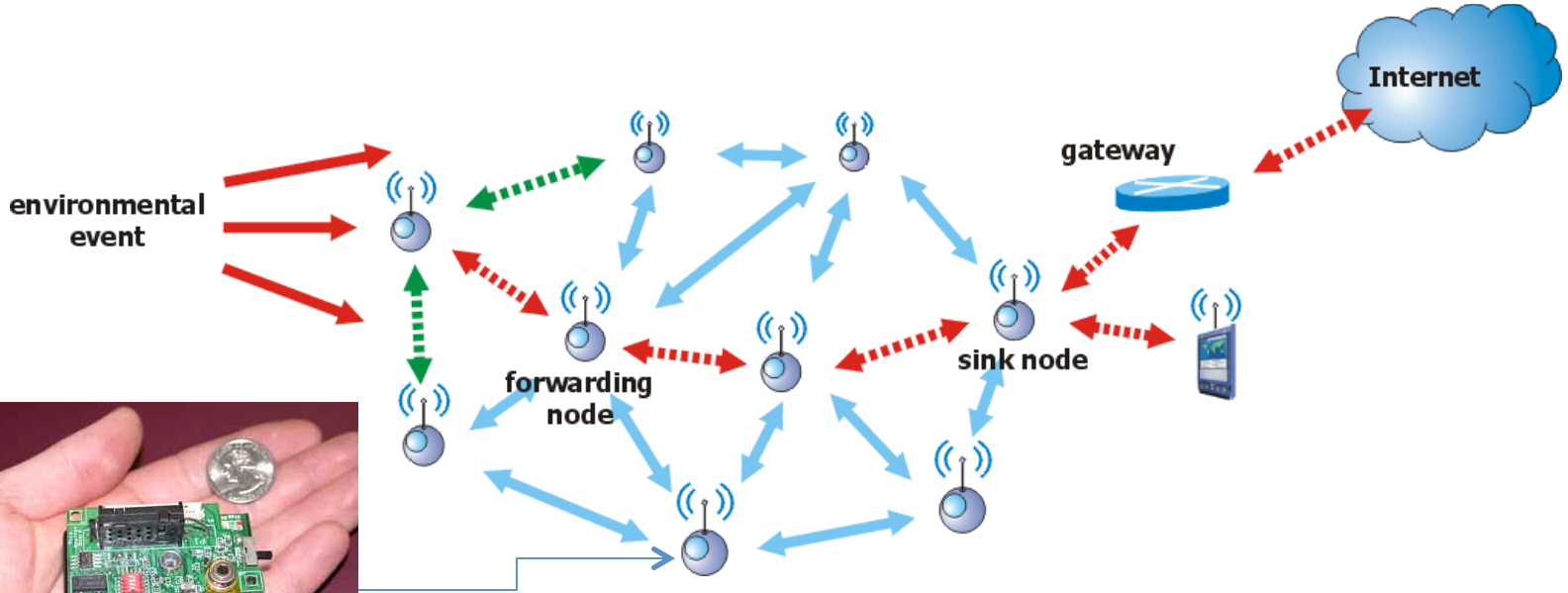
Distributed protocol, implementation, estimators for Gaussian and exponential delays, etc.

Motivation General Concepts

Motivation & General Concepts

WSN

- Set of sensor motes
- Mote: tiny computer, extended with sensing and wireless communication capabilities
- Many applications: agriculture, vehicular, environmental, medical, military and border surveillance, etc.



Motivation & General Concepts

- In distributed systems where each machine has its own physical clock, time synchronization is of high importance.

- Clock synchronization is "*the process of ensuring that physically distributed processors have a common notion of time.*"

Message delays are unbounded in distributed systems

Motivation & General Concepts

Physical Clock in Computer Systems

$C(t)= \int \omega(t)d(t)+C(t_0)$

$\omega(t)$: the frequency of oscillator, $C(t_0)$ its initial value.

Most used model in computer networks

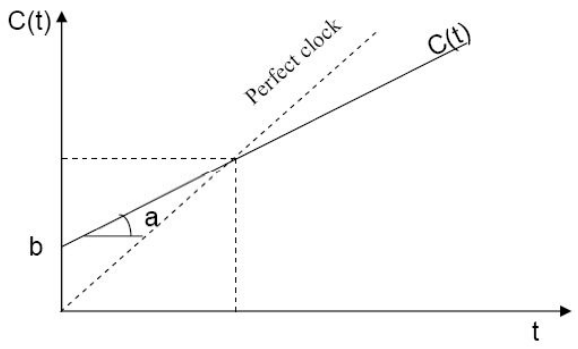
$C(t)=a*t+b$.

a: the clock skew

b: the clock offset

The skew RANDMLY deviates from the nominal rate depending on power supply, temperature, etc.

Clock DRIFT



Motivation & General Concepts

Forms of synchronization

1- Logical time: event ordering. Tell whether an event E1 has occurred before or after another event E2. [Lamp78]

2- Physical time: the use of physical (crystal) clocks

- Relative clock: nodes run their PHISICAL local clocks independently, but keep information about the relative skew and offset for conversion.
- Global "always on": all nodes maintain a clock synchronized (adjusted) to a reference clock in the network. Preserve a global timescale throughout the network.

[Lamp78] *L. Lamport, Time, clocks, and the ordering of events in a distributed system, Communications of the ACM 21 (7)(1978) 558-565.*

Motivation & General Concepts

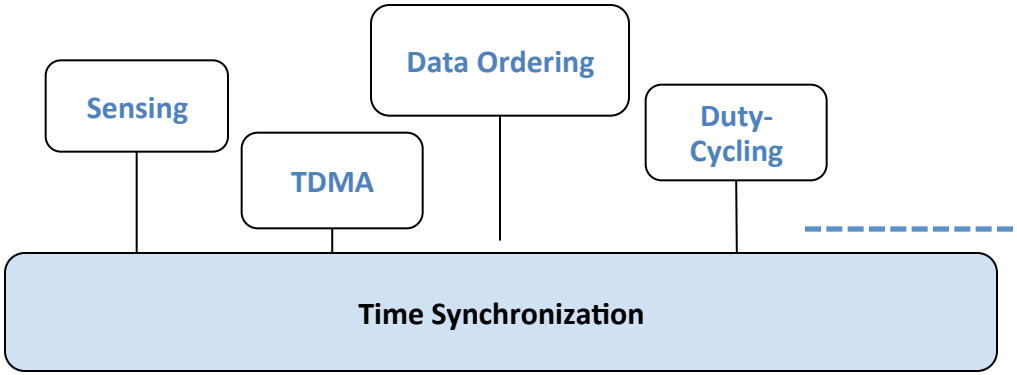
Why time synchronization in WSN ?

- Link to the physical world

→ Physical time synchronization

Motivation & General Concepts

- Many services and communication protocols of WSN require physical, sometimes fine-grained, time synchronization.
  - Coordination of wake-up and sleeping times (Duty Cycling)
  - TDMA & synchronous MAC protocols schedules
  - Ordering of collected sensor data/events
  - Co-operation of multiple sensor nodes.
  - ETC.



And applications

- Object tracking, vehicles, animals, etc.
- Critical, and real time applications.
- Etc, etc.

General Concept (NTP)

Network Time Synchronization (NTP)

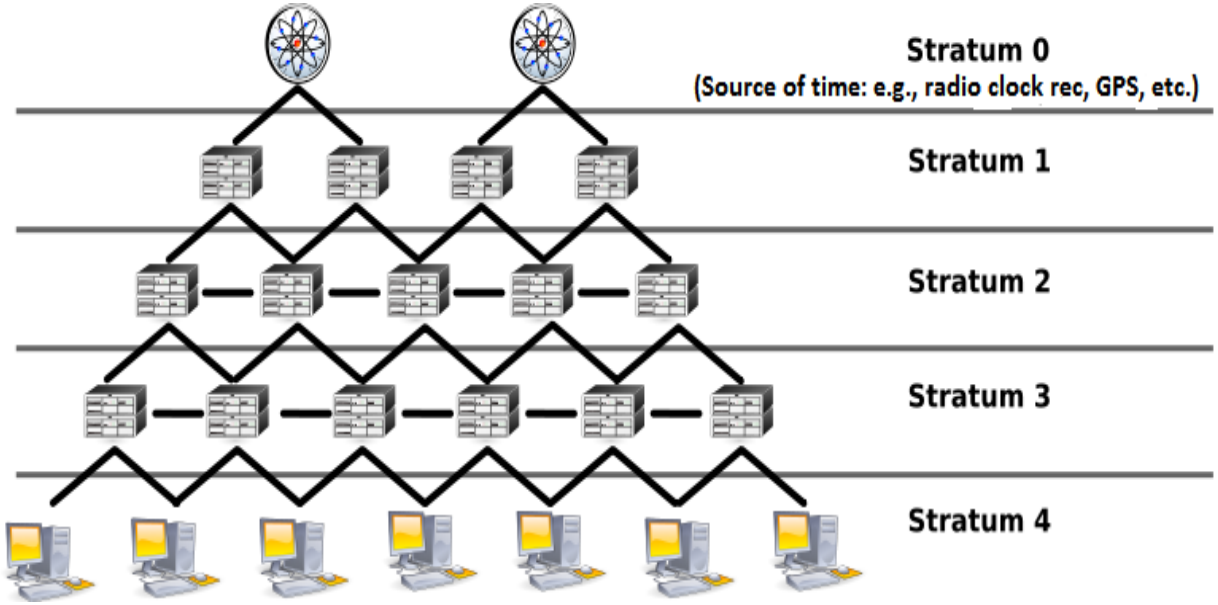
- The most used over the internet
- Developed in 1985 by David Mills at the University of Delaware, USA.
- Application layer protocol, default port number 123 using UDP
- Accuracy: approx. few tens of μs over LAN, approx. few ms over the Internet.
- Implementations: It runs as a daemon called *ntpd* under Unix or as a service under Windows

D. L. Mills , Internet time synchronization: the network time protocol, *IEEE Transactions on Communications (Volume:39 , Issue: 10), Oct 1991*

Network Time Synchronization (NTP)

Synchronization subnet

- Hierarchy: Primary servers (stratum 1) are connected directly to a time source, secondary servers are synchronized with primary servers.
- Each server (or client) can be configured to synchronize **to** one or more servers at upper stratum or at the same stratum.
- Allows nodes to synchronize to a global time and uses UTC (universal time unit).



Network Time Synchronization (NTP)

Three possible synchronization modes:

- Multicast (sometimes called broadcast),
- procedure-call (client),
- symmetric (peer)

Network Time Synchronization (NTP)

Multicast mode (introduced in 1992).

- Servers periodically multicasts their time to their clients in the network.
- Receivers set their clock assuming a small (estimated) delay.

- Problems:

i) Relatively low accuracy.

ii) Due to hardware limitations, this mode only works in IP multicast enabled networks.

Network Time Synchronization (NTP)

Procedure-call and symmetric modes

- Use two exchange of messages (to be explained).
- Higher accuracy.

Procedure-call

- One server accepts requests from clients (client initiated).
- Each node is configured to a set of servers (sources), using configuration files.

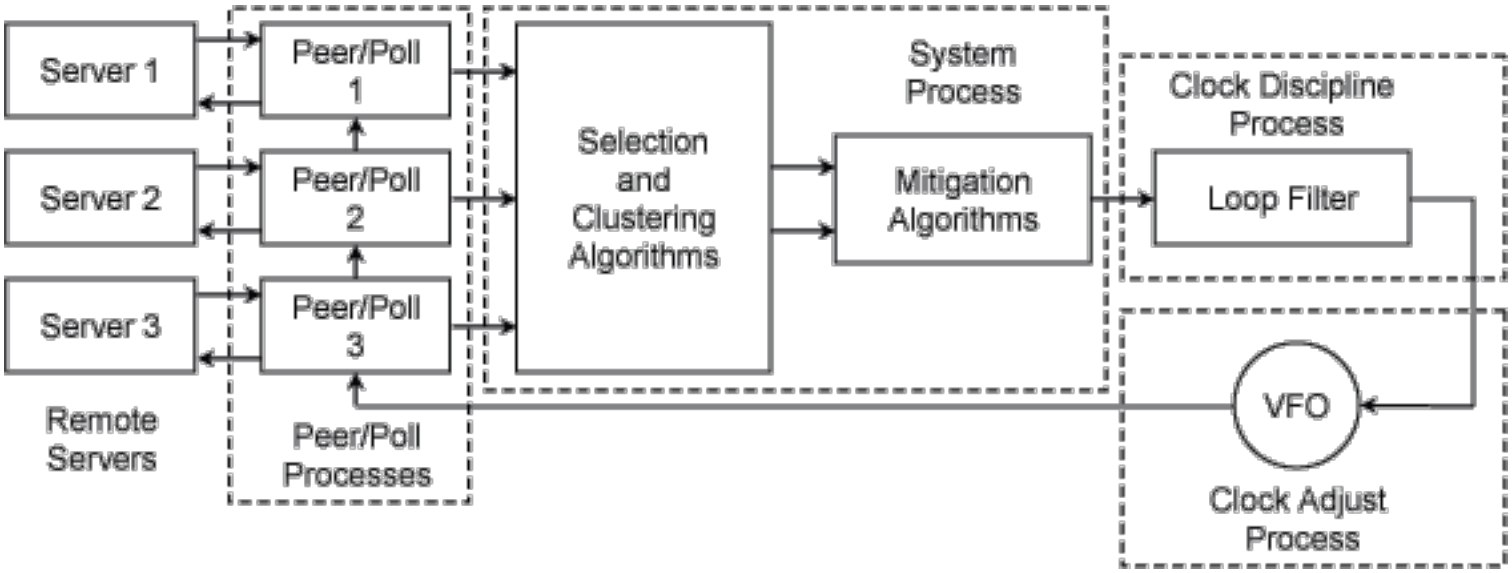
Symmetric (peer)

- Any can initiate and becomes a client.

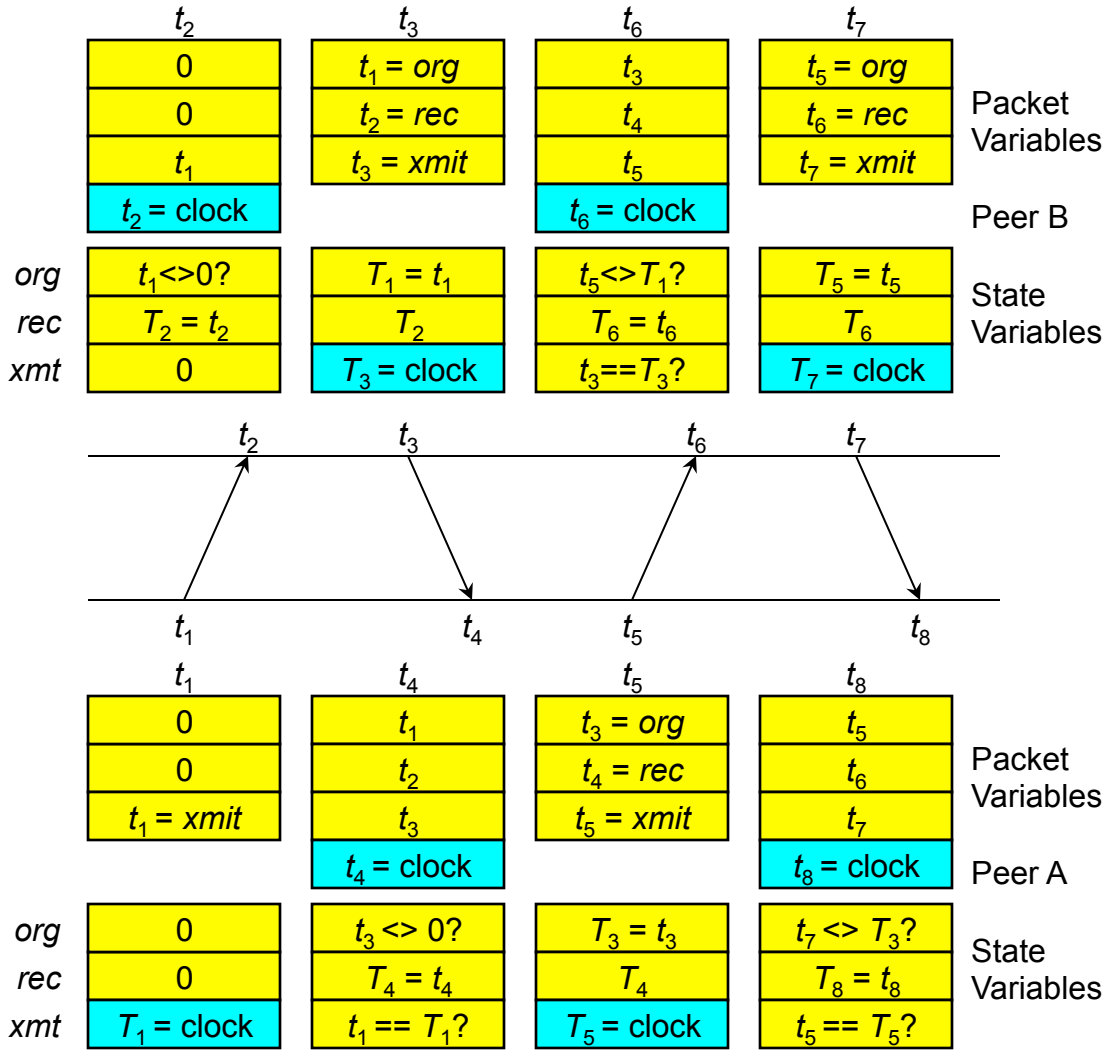
Network Time Synchronization (NTP)

Process Decomposition

- Peer process runs when a packet is received.
- Poll process sends packets at intervals ranging from 8 s to 36 hours
- Peer/Poll processes consist in the running of the on-wire algorithm that provides updates (offset and delay).



On-wire Protocol (client-server and peer modes)



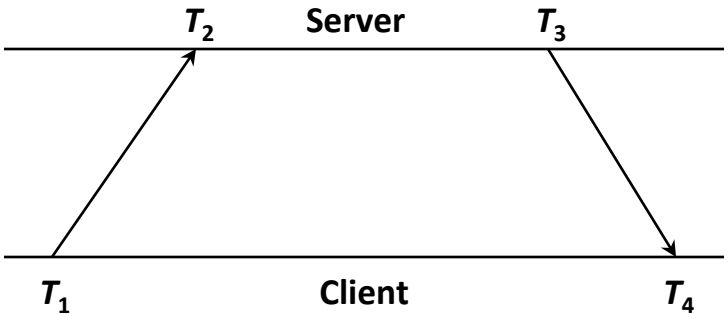
State Variables	
Name	Description
org	originate timestamp
rec	receive timestamp
xmt	transmit timestamp

Packet Header Variables	
Name	Description
t_n	originate timestamp
t_{n+1}	receive timestamp
t_{n+2}	transmit timestamp
t_{n+3}	destination timestamp

$t_7 <> T_3$? *org* Duplicate Test

$t_5 == T_5$? *xmt* Bogus Test

NTP: Clock filter algorithm



$$\theta = \frac{1}{2}[(T_2 - T_1) + (T_3 - T_4)]$$

$$\delta = (T_4 - T_1) - (T_3 - T_2)$$

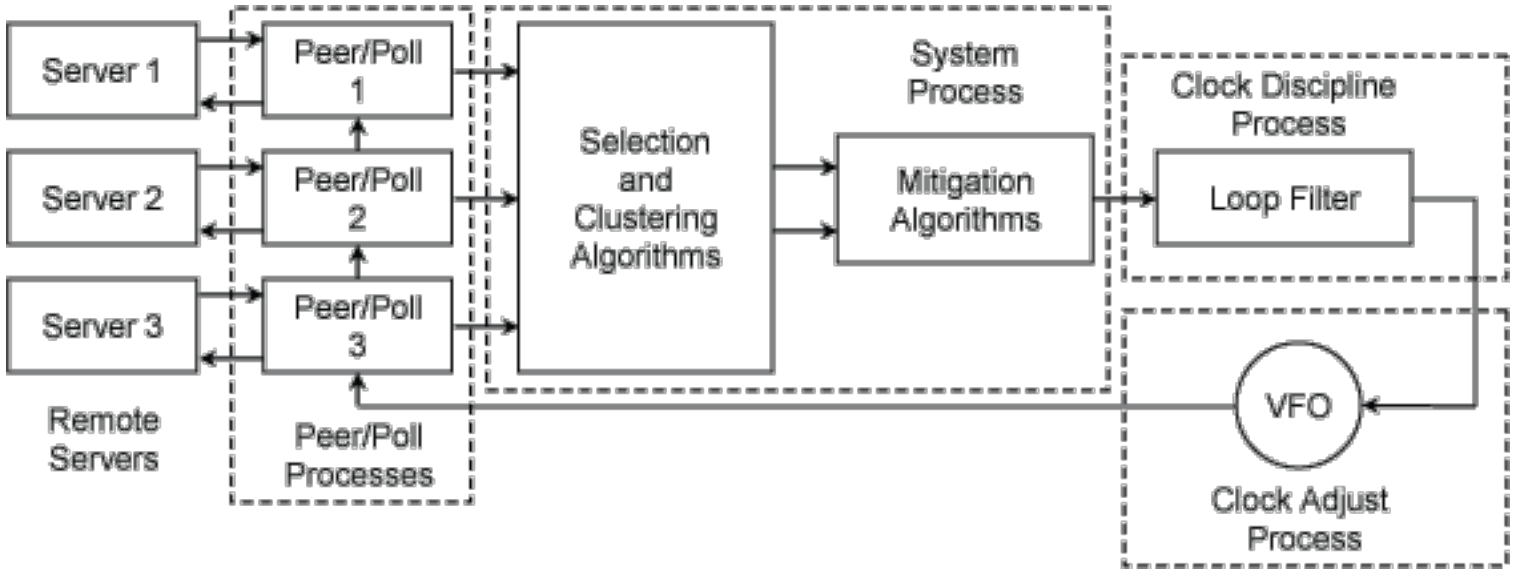
- θ: Offset, δ: round trip delay.
- The most accurate offset ,θ_0 is measured at the lowest delay δ_0.
- In NTP: δ_0 is estimated as the minimum of the last **eight** delay measurements and (δ_0 , θ_0) becomes the peer update.

Network Time Synchronization (NTP)

Process Decomposition

-System process runs when a new peer process update is received: Selection and clustering algorithms allows to filter the updates and keep only significant one (survivor).

- Clock Discipline Algorithms **gradually** adjust the system clock time and frequency.



Network Time Synchronization (NTP)

Why does not work in WSN ?

-Jeremy Elson, Kay Römer: Wireless sensor networks: a new regime for time synchronization. Computer Communication Review 33(1): 149-154 (2003)

Why does not work in WSN ?

Energy Limitation Related Constraints

NTP assumes:

- Listening to the network has no cost, and no efforts is made to predict when the packets arrive.
- CPU is always on.

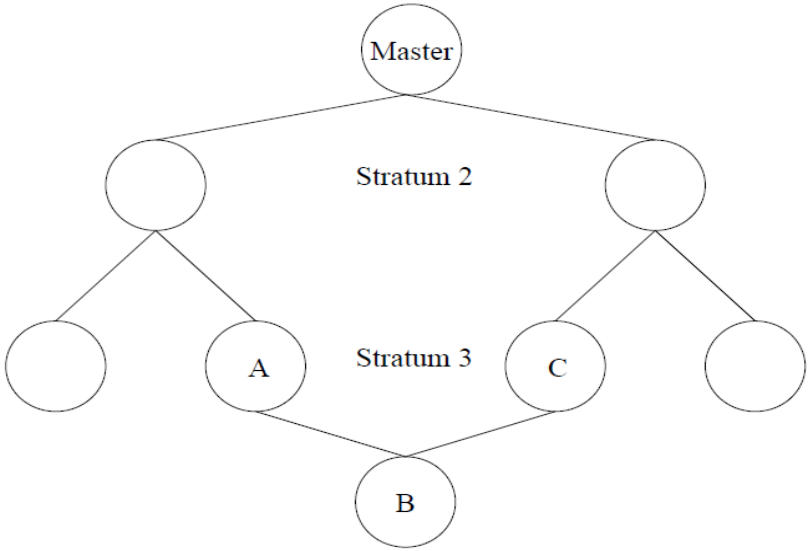
Why does not work in WSN ?

Infrastructurless Related Constraints

- WSN: large-diameter networks (several hops) without an external infrastructure.
- NTP uses a hierarchy rooted at a single node.

OK for few hops (internet), but not WSN. Nodes located far away from the source are poorly synchronized.

Why does not work in WSN ?



- B gets synchronized either via A or C. Any choice will potentially cause large error with respect to synchronization towards the opposite node. In many applications and services local synchronization may have more importance.

Why does not work in WSN ?

Dynamic topology

- Internet: despite transient link failures, the topology remains relatively stable for months. Manual configuration for nodes peering works.
- Doesn't work for WSN: high dynamicity, the need of unattended operation in many applications.

Motivation

Why time synchronization in **WSN** has been a research trend?

- In addition to the reasons making traditional protocols (e.g., NTP) inappropriate for WSN

Particular Challenges:

- HIGH delay variability: Wireless
- Hardware Limitation: sensor motes

Protocol Taxonomy

Protocols Taxonomy

B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: A survey," Ad Hoc Networks, (3) 3, 2005.

- Single-hop (local) vs. Multi-hop (global).
- Relative synchronization (Untethered clocks) vs. time update.

Relative synchronization easier to implement; no time freeze/ backward problem +

can convert local time to universal time if some of the nodes synchronized with is tuned to such a time.

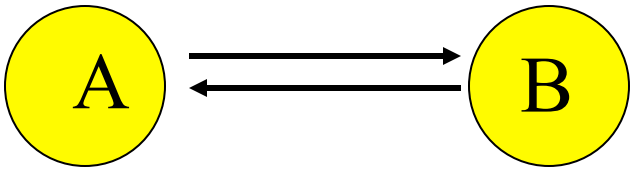
- High layer vs. MAC layer timestamping

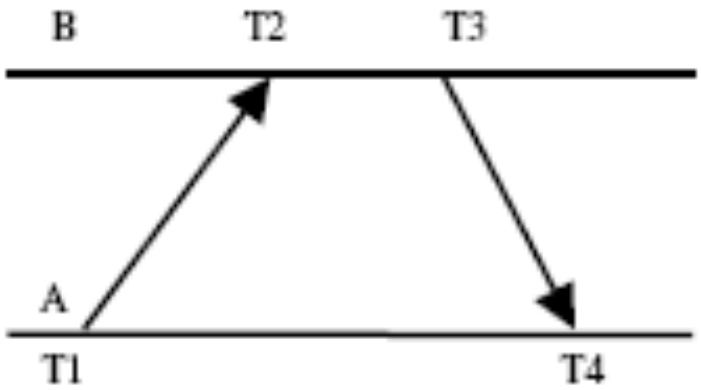
Protocols Taxonomy

- External vs. 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, etc.

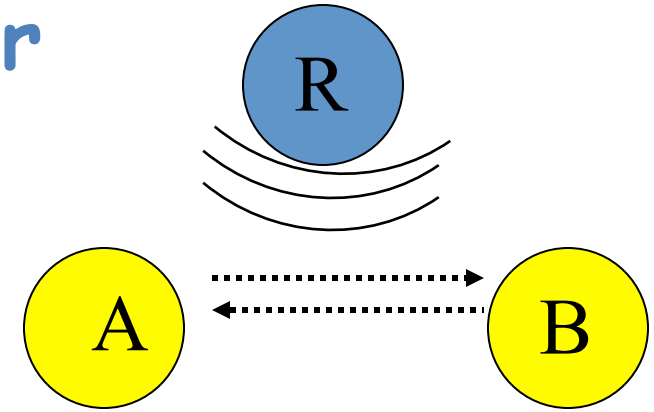
- Master-slave vs. peer-to-peer
- Sender-to-receiver vs. receiver-to-receiver

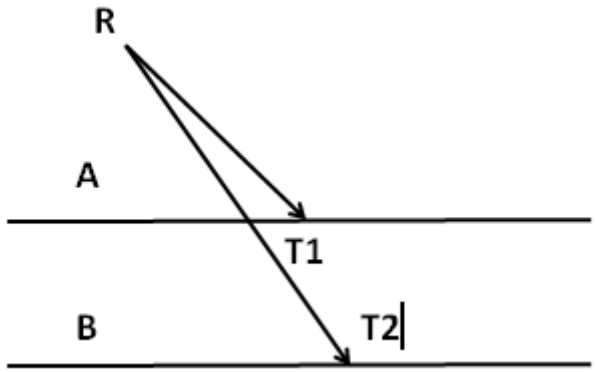
Sender-to-receiver vs. receiver-to-receiver





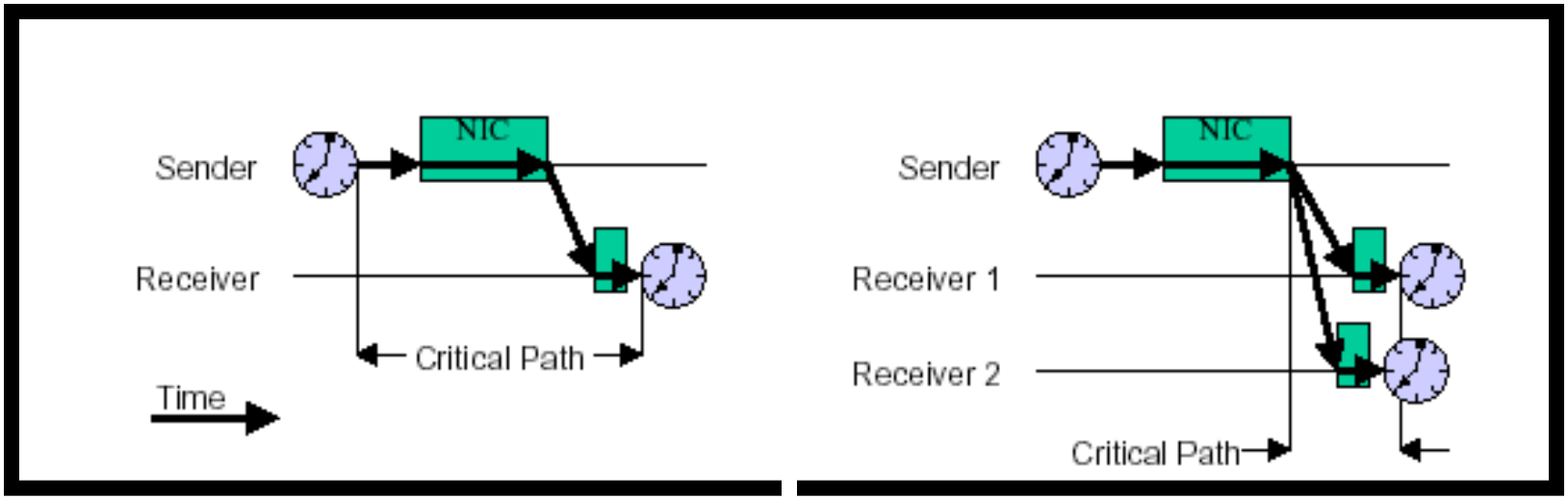
Sender/Receiver





Receiver/Receiver

Sender-to-receiver vs. receiver-to-receiver



send/rec

rec/rec

Reducing the time critical path

- Send Time Latency
- Access Time Latency
- Prorogation Time Latency
- Receive Time Latency

Example of Receiver-to-Receiver Protocol

Reference Broadcast Synchronization (RBS)

J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in Proc. 5th USENIX Symp. OSDI, Dec. 2002, pp. 147–163.

Reference Broadcast Synchronization (RBS)

- The first that introduces the receiver-to-receiver principle to time synch in WSN
- Local synchronization: one reference is used to synchronize nodes in its vicinity

Relative synchronization

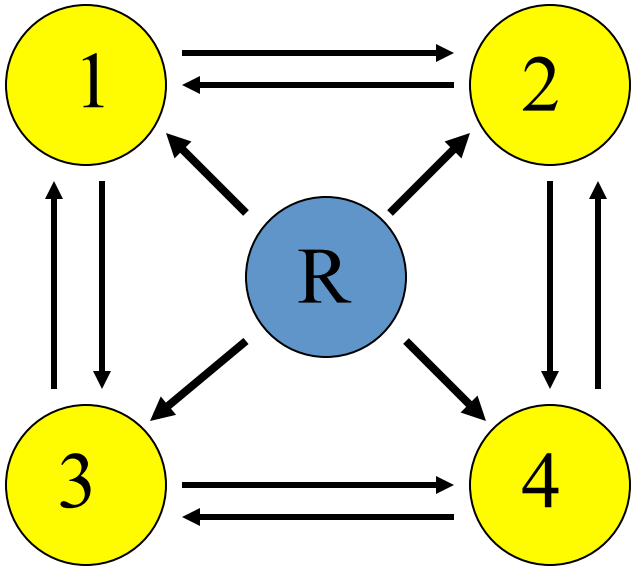
C1(t)=a*C2(t)+b.

a: **relative** skew between C1 & C2

b: **relative** offset

Simplified model (offset-only)

C1(t)=C2(t)+b



Reference Broadcast Synchronization (RBS)

Basic idea to estimate phase offset for non-deterministic receivers:

- Transmitter broadcasts **m** reference packets (m samples)
- For each couple of nodes N1, N2

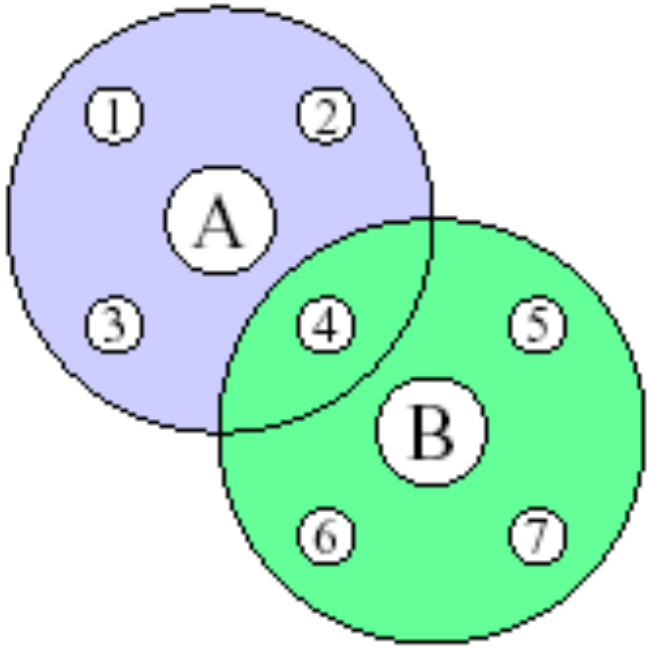
N1 (resp. N2) gets, $T_{1,i}$ (resp. $T_{2,i}$), $i \in \{1...m\}$

N1 estimates:

- In Offset only model: $b = 1/m \sum(T_{1,i} - T_{2,i})$
- In Offset-skew model: a and b using linear regression

Reference Broadcast Synchronization (RBS)

- Allows both MAC and high layer timestamping
- Multi-hop extension.



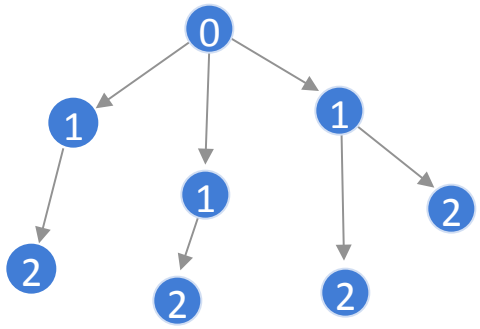
Example of Sender-to-Receiver Protocol

Time-sync Protocol for Sensor Networks (TPSN)

S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in Proc. 1st Int. Conf. Embedded Netw. ACM SenSys, 2003, pp. 138–149.

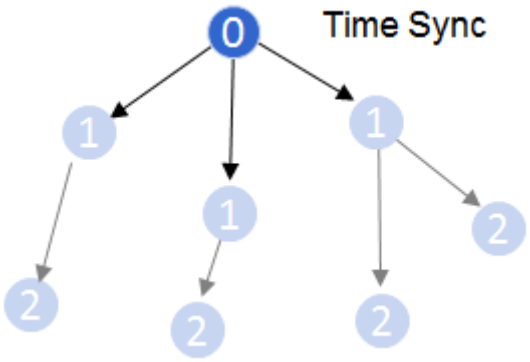
TPSN

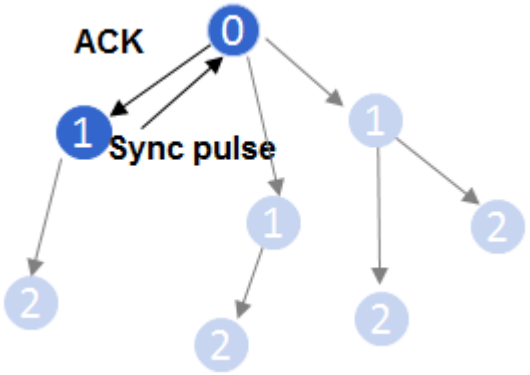
- *Initialization phase: Breadth-first-search flooding*
  - Root node at level 0 sends out a *level discovery* packet
  - Receiving nodes which have not yet an assigned level, set their level to the received level +1, start a random timer
  - After the timer is expired, a new level discovery packet will be sent
  - Random times: to avoid collisions
  - Outcome of this step: tree construction



TPSN

- *Synchronization phase*
  - Root node issues a *time-sync* packet which triggers a random timer at all level-one nodes
  - Every node at level 1 initiates two-way message exchange (send-rec synchronization) to its parent for synchronization after the random timer is expired using synch pulse.
  - Every node at level 1 gets synchronized to the root by calculating the offset (offset only model). b= ((T2-T1)-(T4-T3))/2. And **it updates its clock**.
  - Likewise, the synch pulse packets from level-one nodes trigger random timers at level two, and the process is repeated until all nodes are synchronized to the root.





Estimation and Signal Processing Issues

Estimators

Y.-C. Wu, Q. M. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," IEEE Signal Process. Mag., vol. 28, no. 1, pp. 124–138, Jan. 2011.

- Delay of messages is a random process → estimation methods
- The literature of estimators of skew/offset parameters for time synchronization in WSN can be divided according to the assumed distribution:
- i) Gaussian, ii) Exponential, and iii) arbitrary distributions.

Estimators

Gaussian Delays

- Several methods have been used: minimum variance unbiased estimator (MVUE), the best linear unbiased estimator (BLUE), the maximum-likelihood estimator (MLE), and the least square estimator (LSE).
- MLE and LSE are the most largely used in the literature
- The optimal estimators are relatively easy to derive for Gaussian delays, where the MVUE, MLE, BLUE LSE all coincide.

Y.-C. Wu, Q. M. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," IEEE Signal Process. Mag., vol. 28, no. 1, pp. 124–138, Jan. 2011.

Estimators

Exponential Delays

- For exponential delays and the sender-to receiver approach: analytically demonstrated that the MLE is better than the MVUE when the means of the uplink and downlink delays are very similar, and that the MVUE becomes better when they are dispersing.

Y.-C. Wu, Q. M. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," IEEE Signal Process. Mag., vol. 28, no. 1, pp. 124–138, Jan. 2011.

Estimators

Arbitrary Delays,

- Methods used: linear programming, bootstrap bias correction, and composite particle filtering.
- These estimators are robust when delay distributions are unknown, and they can adapt to different delay distributions.
- Reported that the MSE performance of bootstrap bias corrected estimate is better than the exponential MLE when applied to non-exponential delays. (Gamma with two degrees of freedom used for illustration).

Y.-C. Wu, Q. M. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," IEEE Signal Process. Mag., vol. 28, no. 1, pp. 124–138, Jan. 2011.

Recommended References

Estimation issues

Y.-C. Wu, Q. M. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," IEEE Signal Process. Mag., vol. 28, no. 1, pp. 124–138, Jan. 2011.

Kyoung-Lae Noh et al., Novel Clock Phase Offset and Skew Estimation Using Two-Way Timing Message Exchanges for Wireless Sensor Networks. IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. 55, NO. 4, APRIL 2007

Other Review papers

B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: A survey," Ad Hoc Networks, (3) 3, 2005.

Fikret Sivrikaya and Bülent Yener Time Synchronization in Sensor Networks: A Survey, IEEE Networks Magazine, August 2004.

Osvaldo Simeone, et al. Distributed Synchronization in Wireless Networks, IEEE SIGNAL PROCESSING MAGAZINE, Sep 2008.

Implementation Issues

"The only source of knowledge is experience." Albert Einstein

Implementation Challenges

Fast Drifting

- Cost reduction that inevitably requires the use of cheap, but less reliable, components.
- Most motes include two clocks:

i) Internal clock: allows for microsecond-level granularity but also with relatively high drifting.

ii) External clock: more stable but usually has low frequency and thus provides a weak granularity. Low frequency is used in practice even when the available frequency is high, as the clock operates even in power saving and sleep modes.

Implementation Challenges

Fast Drifting

- E.g., crystals used in MICAz and TelosB have 32.768-kHz frequency, i.e., 30.5 μs resolution, while the internal oscillator frequency is at the order of 8MHz.
- The external clock has the advantage of running when the node turns to the sleep mode, contrary to the RC internal clock. This justifies the low frequency use for the sake of power saving.

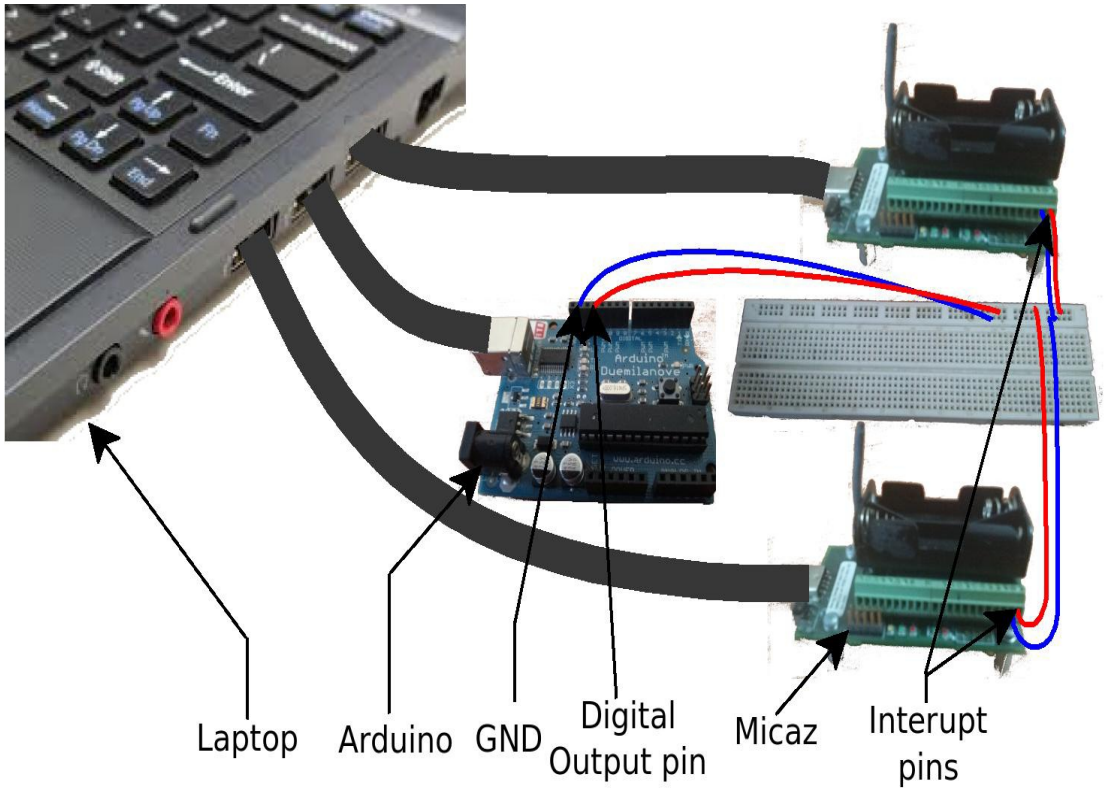
Implementation Challenges

Fast Drifting

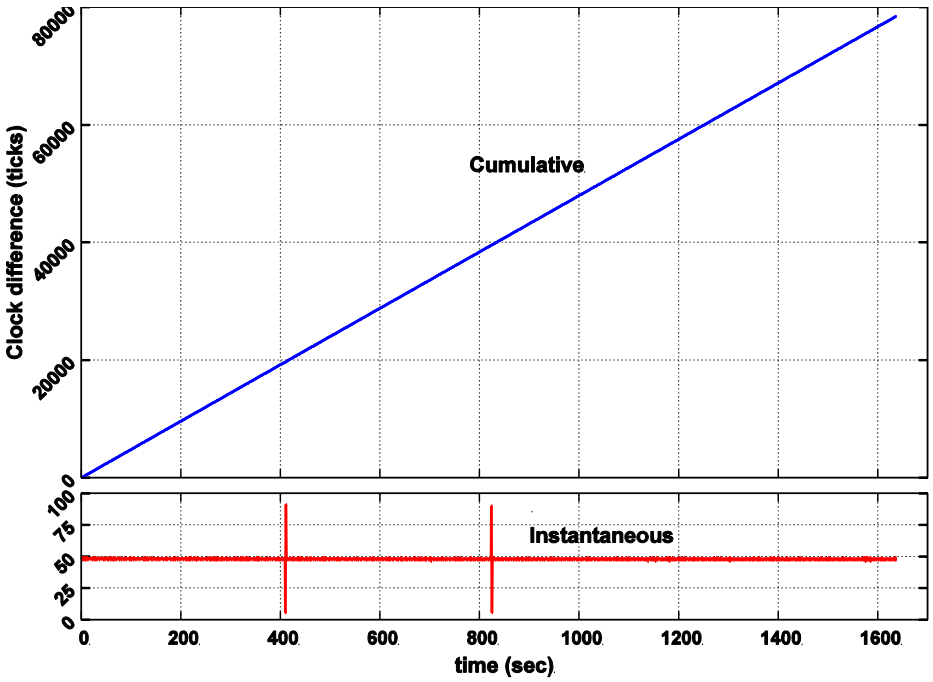
- To investigate the clock drift, we performed an extensive experimental study with MICAz motes.

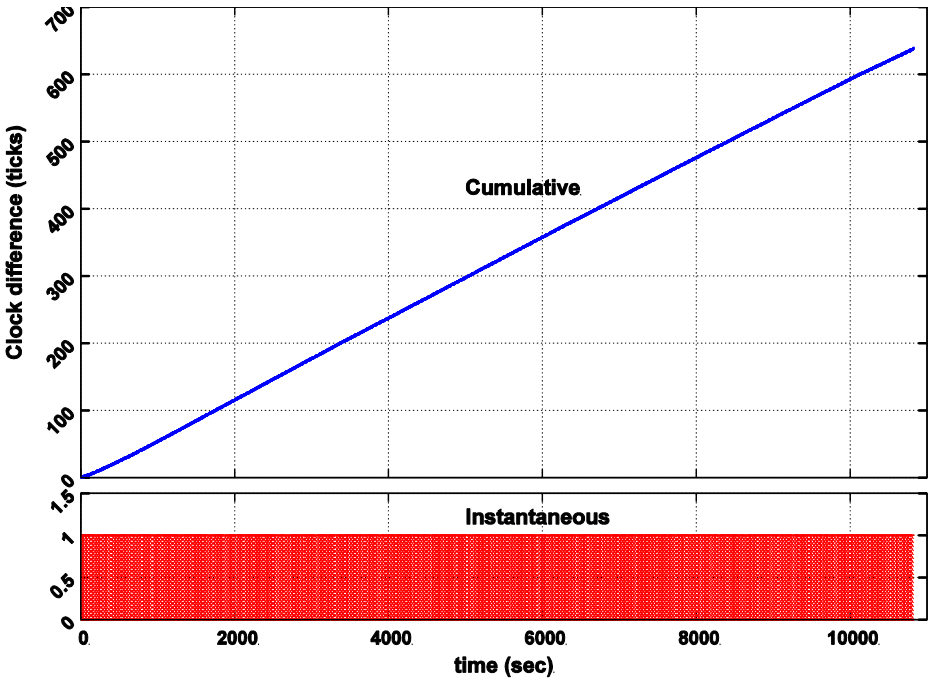
- Relative drift between two motes has been investigated.

- Both internal and external clocks.



Implementation Challenges (Fast Drifting)

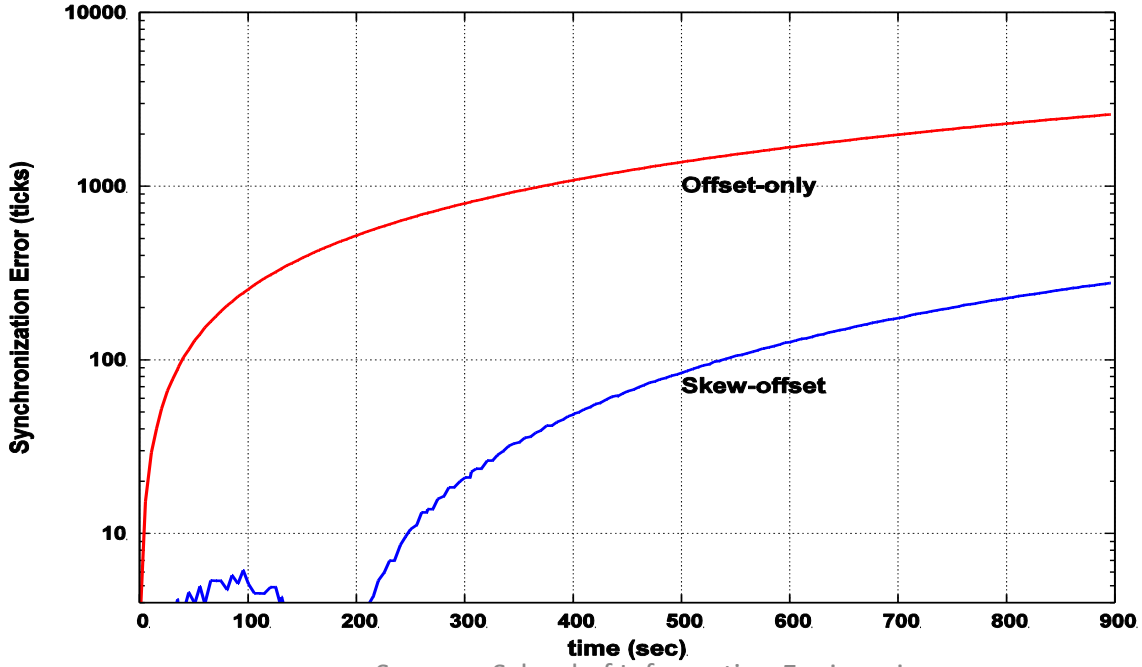




- External, relatively less drifting (more stable).
- Linear cumulative increase for both → Confirms suitability of linear model for skew estimation.

Implementation Challenges (Fast Drifting)

- Investigated the impact of skew estimation on long-term synchronization,
- Synchronization error between two MICAz motes has been empirically measured in both models (offset-only and skew-offset) and the sender-to-receiver approach.
- Synchronization messages exchanged for few rounds for estimation.
- At t=0, the synchronization process stopped, and the synchronization error measured.
- Confirms the skew estimation usefulness for long-term synchronization



Implementation Challenges

Mote limitations (memory and processing)

- Most simulations and mathematical analyses use tools such as MATLAB or C-based simulators at a high level of abstraction.
- Coding for a standard computer, running and collecting the results for analysis of thorough, but computation-costly, estimators.
- Completely ignore that this code for calculating estimators is to be run by tiny sensor motes with limited capacities.

Implementation Challenges

Mote limitations (memory and processing)

- E.g., the floating-point computation is **not supported** by most platforms (MICAz, TelosB, etc.); implemented as a library at the kernel of operating systems, e.g., TinyOS 2.1
- Our experience with TinyOS revealed several problems with this library when handling large numbers (clock values), and reimplementation of floating-point division calculation has been necessary.
- Estimators evolving sequences of clock values **multiplications** are to be avoided, (fast arithmetic overflow).
- Such estimators should be simplified and/or rewritten to fit motes limitations.

Challenges Implementation

Mote limitations (memory and processing)

- Most estimators rely on the collection of large samples to get statistical meaningful estimates. This would have an important memory footprint on the sensor motes.
- The use of limited window with possible moving average can help reducing the memory footprint. However, the results would be different from the nice theoretical performance, an issue that needs to be addressed by testbed evaluations.

Challenges Implementation

Mote limitations (memory and processing)

- Existing solutions such as *TinyECC2* of TinyOS and its NN library can be used to implement estimators while overcoming the problem of arithmetic overflow, as they enable customizing the variable size according to the user need. However, the memory space should be carefully managed since the memory footprint may increase dramatically.

Implementation Challenges

Delay Variation (Jitter)

- High **variation** for delays related to exchanging synchronization signals (wireless links), and handling them (mote limitation).
- Rec-to-rec reduces the jitter

Other practical approaches
- low-layer timestamping (MAC layer),
- Timestamping several bytes to normalize the in-node jitter (FTSP): considerably reduces the amplitude of the variation,
- Drawback: they make the implementation dependent on the specific platform and reduces the portability compared with high-layer timestamping.
- The big challenge caused by delay variation is when using timestamping at the higher-layers.

Implementation Challenges

Fault Tolerance

- Most theoretical and simulation-based evaluations suppose ideal scenarios and neglect aspects such as packet loss and node failure.
- Correct behavior in the presence of faulty nodes must be ensured.
- E.g., protocols using consensus for time update (such as averaging offsets) are affected by erroneous reports.
- More importantly, those based on round-robin operation may fail and stop working when a single node is down.

Implementation Challenges

Fault Tolerance

- Indications such as the absence of messages or responses from a node during a threshold time may be used as an indicator of faulty nodes.
- But, the threshold should be carefully selected to avoid possible false positives due to the *lossy* channels.
- In addition to faulty nodes, considering Byzantine behavior is challenging, where compromised nodes may report falsified timestamp values for attacking the synchronizing protocol.

Implementation Challenges

Accuracy Measurement

- Instantaneous local times (or estimates) at the synchronized nodes involved in the measurement should be captured at the ***same physical time.***
- Similar to the fundamental atomic snapshot in distributed systems.
- Existing distributed solutions do not apply due to the high variability of message exchange.

Implementation Challenges

Accuracy Measurement

- Trend: use an external hardware setting that has a latency at a lower order.
- This complicates, i) scalability investigation, ii) the use of testbeds installed in laboratories.

- Developing test-beds facilitating external hardware connection and allowing for low-level and interrupt handling programming will be useful.

Implementation Issues

Impact of Empirical Parameters

Impact of Empirical Parameters

- Empirical parameters have a significant impact on the protocol performance, which has been ignored in the theoretical studies.
- We investigated the elementary sender-to-receiver approach for single-hop synchronization and the joint skew–offset model.

Impact of Empirical Parameters

- Two MICAz motes with Arduino for capturing instantaneous clock value (similarly to the experiment described before).
- MLE for skew-offset estimation in Gaussian model.
- Nodes exchange two-way messages to construct a quadruple of timestamps that forms a sample. The process is then repeated for certain rounds to acquire a set of samples for estimation.

Impact of Empirical Parameters

- We varied two important parameters of the implementation,

i) the sample size (k): the number of rounds before estimation, and

ii) the period separating two rounds (T).

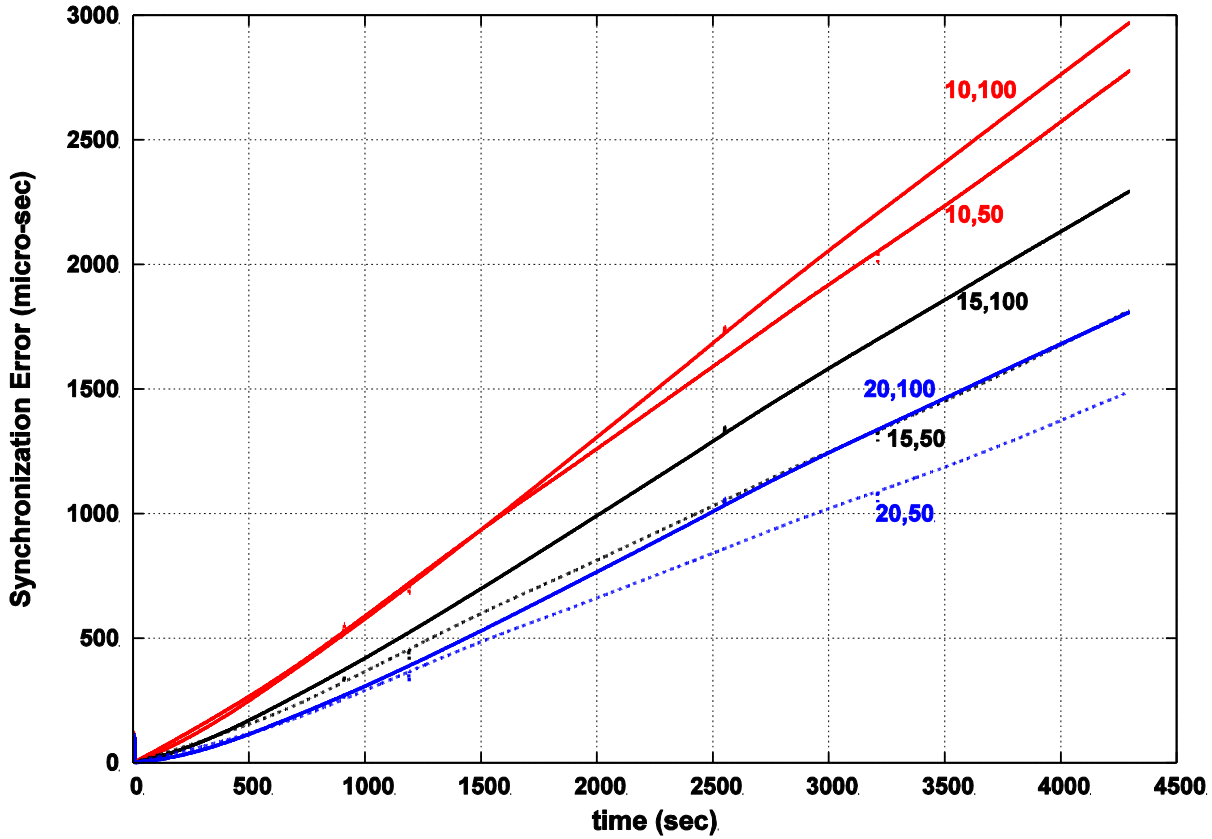
- Three values for k (10, 15, 20), and two for T (50 ms, 100 ms).

- Synchronization error has been measured for each variant of the implementation,
- "t = 0" is the time when the estimation is completed and the protocol execution is stopped

Impact of Empirical Parameters

Accuracy Measurement

- T has a significant impact. For all values of k, the precision of every variant with T = 50 (dashed lines) has better performance than the respective one with T = 100 (solid lines).



Impact of Empirical Parameters

- MLE relies on the assumption that the transmission/ reception delays follow a normal distribution with the same parameters, i.e., $N(\mu, \sigma^2)$.
- Theoretical analyses confirm MSE and CRLB inversely depends on the variance (σ^2), but no study investigated the issues that may affect σ^2.

Impact of Empirical Parameters

- We remarked in the experiment that the variance of the measured delays for executions with T = 50 were lower than those with T = 100 in all scenarios.

- This is justified by the fact that channel conditions vary over time, and that picking up samples in shorter periods is likely to encounter more similar conditions than picking them up over longer periods.

Impact of Empirical Parameters

- Strongly recommended to perform sample acquisition in one cycle (round) when using models relying on assumptions about the delay distribution (e.g., MLE), and the skew-offset model, notably in low duty-cycled applications.

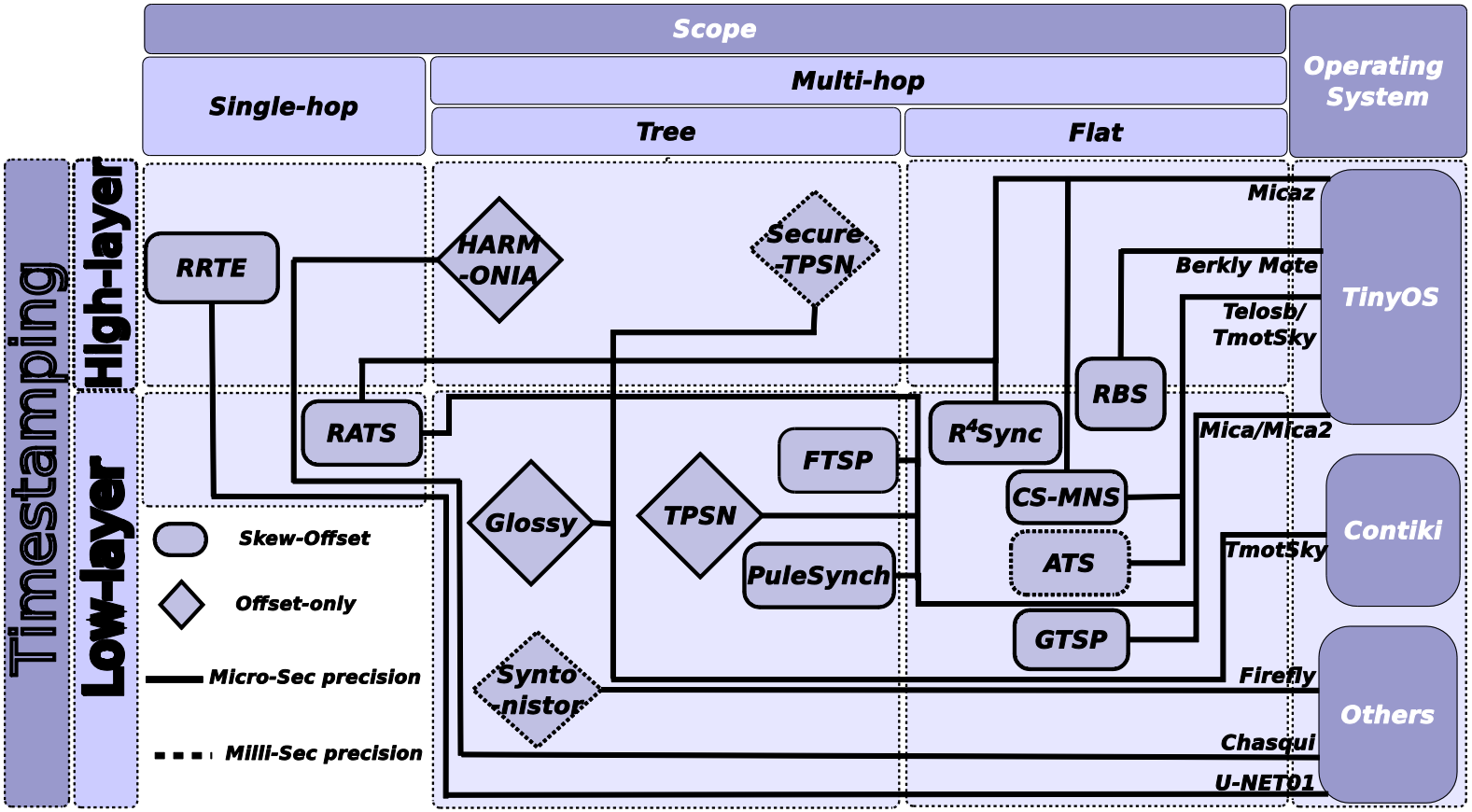
- The duty cycle may be established using a cycle with a synchronization phase(s) that should be long enough to fit operations needed to get the sample size k, followed by a set of cycles without a synchronization phase, rather than using a synchronization phase at the beginning of every cycle.

- The synchronization phase can be at the beginning of the cycle (which is the most common), or even split throughout the cycle; to avoid long latencies for data packets.

Implementation Issues

Some Implemented Protocols

Some Implemented Protocols (Taxonomy)



*D. Djenouri, M. Bagaa, M. "Synchronization Protocols and Implementation Issues in Wireless Sensor Networks: A Review", **IEEE Systems Journal**, Vol 10 Issue 2, pp 617-627, June 2016.*

Implemented Protocols

Some Lessons

Problem

- The internal clock does not run when in sleep mode. It cannot be used in applications such as coordinated duty cycling.

Solution

- The Virtual High-resolution Time (VHT) technique used by HARMONIA and Glossy enables the use of the high-granularity internal clocks to accurately synchronize the external clocks and then to make a precise rendezvous with the external clocks.
- Useful in applications that need precise rendezvous or synchronized coordinated actions, such as coordinated duty cycling (e.g., TDMA-like scheduling), but not those needing time measurement or conversion (timestamping), e.g., object tracking.

Implemented Protocols

Some Lessons

- Offset-only implementations can only be used in applications requiring sporadic delay-tolerant synchronization, where the synchronization can be performed instantly before the timestamping of the reported event.
- Empirical parameters (e.g., delay between sample acquisition), considerably affect the estimators' precision.
- MAC-layer timestamping: practical solution to reduce jitter, but comes at a reduced portability.

Implemented Protocols

Some Lessons

- Another practical technique is the use of several timestampings per synchronization message (packet) and averaging the resulted timestamps to normalize the jitter, e.g., FTSP and GTPS.
- This comes at a reduced portability and can only be implemented with byte-level interrupt triggering radios, e.g., the one of MICA2, and not the common packet-oriented radios, e.g., CC2420.

Implemented Protocols

Some Lessons

- FTSP is implemented with TinyOS official distribution.
- We realized that this implementation used low resolution interfaces and does not implement the multiple timestamping per packet, which is FTSP's main contribution.
- Many protocols have been compared with the TinyOS distribution of FTSP (e.g., CS-MNS), and confirms superiority at a low granularity of synchronization (several micros per tick).
- For accurate investigation, a more fair comparison would use byte-level interrupt triggering platforms and a revisited implementation.
- Many protocols are compared with high-layer timestamping implementation of RBS (e.g., TPSN), while using a low-layer timestamping
- Unfair! since RBS allows for both.

Some Contributions

Distributed protocol

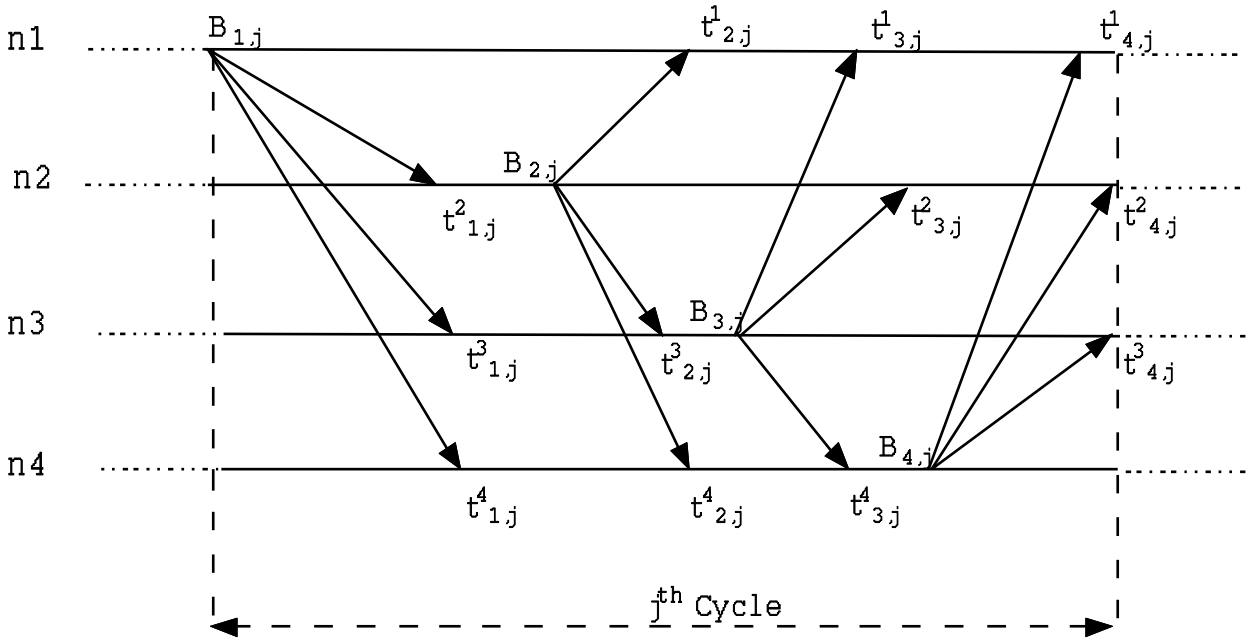
A distributed RBS-based protocol

Idea: distribute the reference's function to all nodes and integrate timestamp exchanges into beacons.

- No need of super nodes.
- Possibility of combining beacons and timestamps (no need for separate steps for timestamp exchange).
- Fast collection of samples

D. Djenouri, N. Merabtine, F.Z Mekahlia, M. Doudou. Fast distributed multi-hop relative time synchronisation protocol and estimators for wireless sensor networks Original Research Article ***Ad Hoc Networks, Elsevier*** *Publisher, Vol 11, vol 8, PP 2329-2344, October 2013.*

New Solution



- In each cycle, every node broadcasts one beacon that piggybacks the timestamps not submitted previously.
- E.g. $B_{1,j}$ *includes* $t1_{2,j-1}$, $t1_{3,j-1}$, $t1_{4,j-1}$, *while* $B_{4,j}$: $t4_{1,j}$, $t4_{2,j}$, $t4_{3,j}$
-

Complexity

- In one cycle, the proposed protocol provides (n-2) samples for each pair of nodes, which makes $O(n^3)$ in total.
- RBS provides one sample for each pair of nodes in a cycle, which results in $O(n^2)$ in total

Estimators

- Maximum likelihood estimators (MLE) and the corresponding Cramer-Rao lower bounds (CRLB) have been derived for offset/skew estimation, and Gaussian distribution.

Estimators

Homogeneous environment (symmetric delays)

- Reception delays, d_{ui}, d_{vi}, are r.v. $\sim N(\mu_0, \sigma_0^2)$
- $Xi = (d_{ui} - d_{vi}) \sim N(0, 2\sigma_0^2) = N(0, \sigma^2)$
- Using MLE method, the following estimators and Cramer Rao lower bounds (CRLB) have been derived.

$$\widehat{\alpha}_{mle} = \frac{\sum_{i=1}^{K} u_i \sum_{i=1}^{K} v_i - K \sum_{i=1}^{K} v_i u_i}{\left(\sum_{i=1}^{K} v_i\right)^2 - K \sum_{i=1}^{K} v_i^2},$$

$$\widehat{\beta}_{mle} = \frac{1}{K}\left(\sum_{i=1}^{K} u_i - \frac{\sum_{i=1}^{K} u_i \sum_{i=1}^{K} v_i - K \sum_{i=1}^{K} v_i u_i}{\left(\sum_{i=1}^{K} v_i\right)^2 - K \sum_{i=1}^{K} v_i^2} \sum_{i=1}^{K} v_i\right).$$

$$Var(\widehat{\alpha}) \geq (I^{-1})_{1,1} = \frac{K\sigma^2}{K \sum_{i=1}^{K} v_i^2 - \left(\sum_{i=1}^{K} v_i\right)^2},$$

$$Var(\widehat{\beta}) \geq (I^{-1})_{2,2} = \frac{\sigma^2 \sum_{i=1}^{K} v_i^2}{K \sum_{i=1}^{K} v_i^2 - \left(\sum_{i=1}^{K} v_i\right)^2}$$

Offset/skew

Heterogeneous environment (asymmetric delays)

- Reception delays have different parameters, $d_{ui} \sim N(\mu_1, \sigma_1^2)$, d_{vi}, are r.v. $\sim N(\mu_2, \sigma_2^2)$
- $Xi = (d_{ui} - d_{vi}) \sim N(\mu, \sigma^2)$, *where* $\mu = \mu_1 - \mu_2$, $\sigma^2 = \sigma_1^2 + \sigma_2^2$, *d: the difference of the fixed portions of the delays.*

$$(\widehat{\Theta}^{mle}_{skew}, \widehat{\Theta}^{mle}_{offset}) =$$

$$argmax(\log \mathcal{L}(\Theta_{skew}, \Theta_{offset} | X_1, ...X_K))$$

$$\widehat{\Theta}^{mle}_{skew} = \frac{K\sum_{i=1}^{K} t_{v,i}(t_{u,i} - d - \mu) - \sum_{i=1}^{K} t_{v,i} \sum_{i=1}^{K} (t_{u,i} - d - \mu)}{K\sum_{i=1}^{K} t^2_{v,i} - (\sum_{i=1}^{K} t_{v,i})^2},$$

$$\widehat{\Theta}^{mle}_{offset} = \frac{\sum_{i=1}^{K} t_{u,i} - \widehat{\Theta}^{mle}_{skew} t_{v,i} - d - \mu}{K}.$$

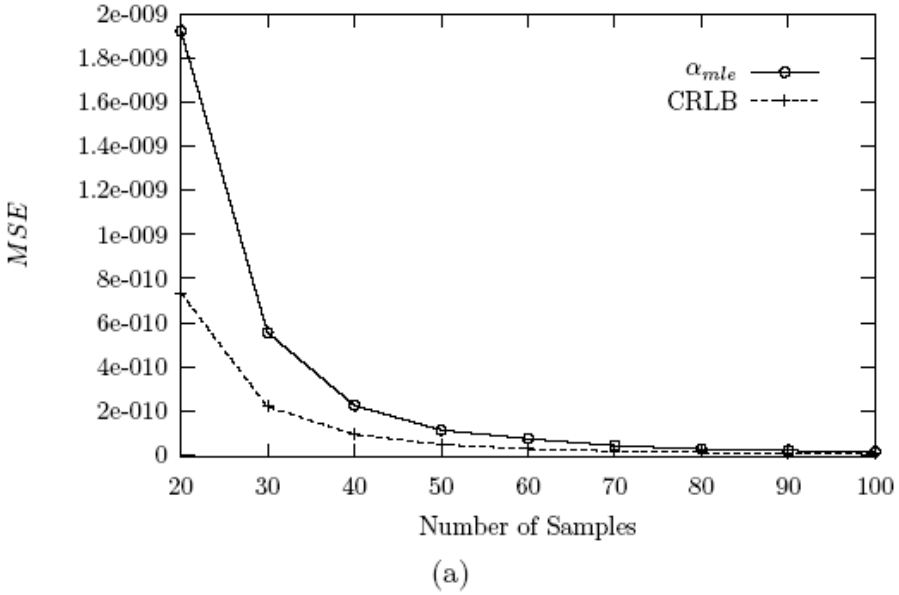
$$Var(\widehat{\Theta}_{skew}) \geq (I^{-1})_{1,1}$$

$$(I^{-1})_{1,1} = \frac{K\sigma^2}{K\sum_{i=1}^{K} t^2_{v,i} - (\sum_{i=1}^{K} t_{v,i})^2}$$

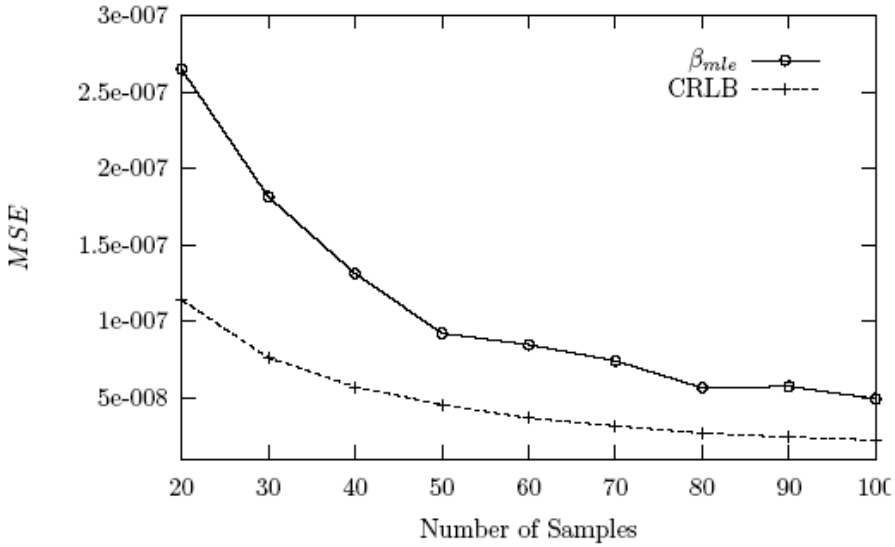
$$Var(\widehat{\Theta}_{offset}) \geq (I^{-1})_{2,2}$$

$$(I^{-1})_{2,2} = \frac{\sigma^2 \sum_{i=1}^{K} t^2_{v,i}}{K\sum_{i=1}^{K} t^2_{v,i} - (\sum_{i=1}^{K} t_{v,i})^2}$$

Numerical Analysis



(a)



(b)

Multi-hop Extension

- On demand;
- ASA a node initiates a multi-hop communication that requires synchronization between the end-points, intermediate routers may forward local relative synchronization parameters to the communicating nodes and allow them to calculate multi-hop relative parameters.
- Local parameters are used to calculate MH parameters.
- Any routing protocol may be used.

Multi-hop Extension Estimators

$$t_{n_i} = \alpha_{n_{i+1} \to n_i} t_{n_{i+1}} + \beta_{n_{i+1} \to n_i}, i \in \{1, ..., h-1\}$$

By successive substitutions of $t_{n_{i+1}}$ expressions in t_{n_i} equations ($i \in \{h-2, ..., 1\}$), the following may be obtained,

$$t_{n_1} = (\prod_{i=1}^{h-1} \alpha_{n_{i+1} \to n_i}) t_{n_h} + \sum_{i=2}^{h-1} [(\prod_{j=2}^{i} \alpha_{n_j \to n_{j-1}}) \beta_{n_{i+1} \to n_i}] + \beta_{n_2 \to n_1}.$$ Consequently,

$$\alpha_{n_h \to n_1} = \prod_{i=1}^{h-1} \alpha_{n_{i+1} \to n_i}, \tag{10}$$

$$\beta_{n_h \to n_1} = \sum_{i=2}^{h-1} [(\prod_{j=2}^{i} \alpha_{n_j \to n_{j-1}}) \beta_{n_{i+1} \to n_i}] + \beta_{n_2 \to n_1}. \tag{11}$$

Multi-hop Extension CRLB

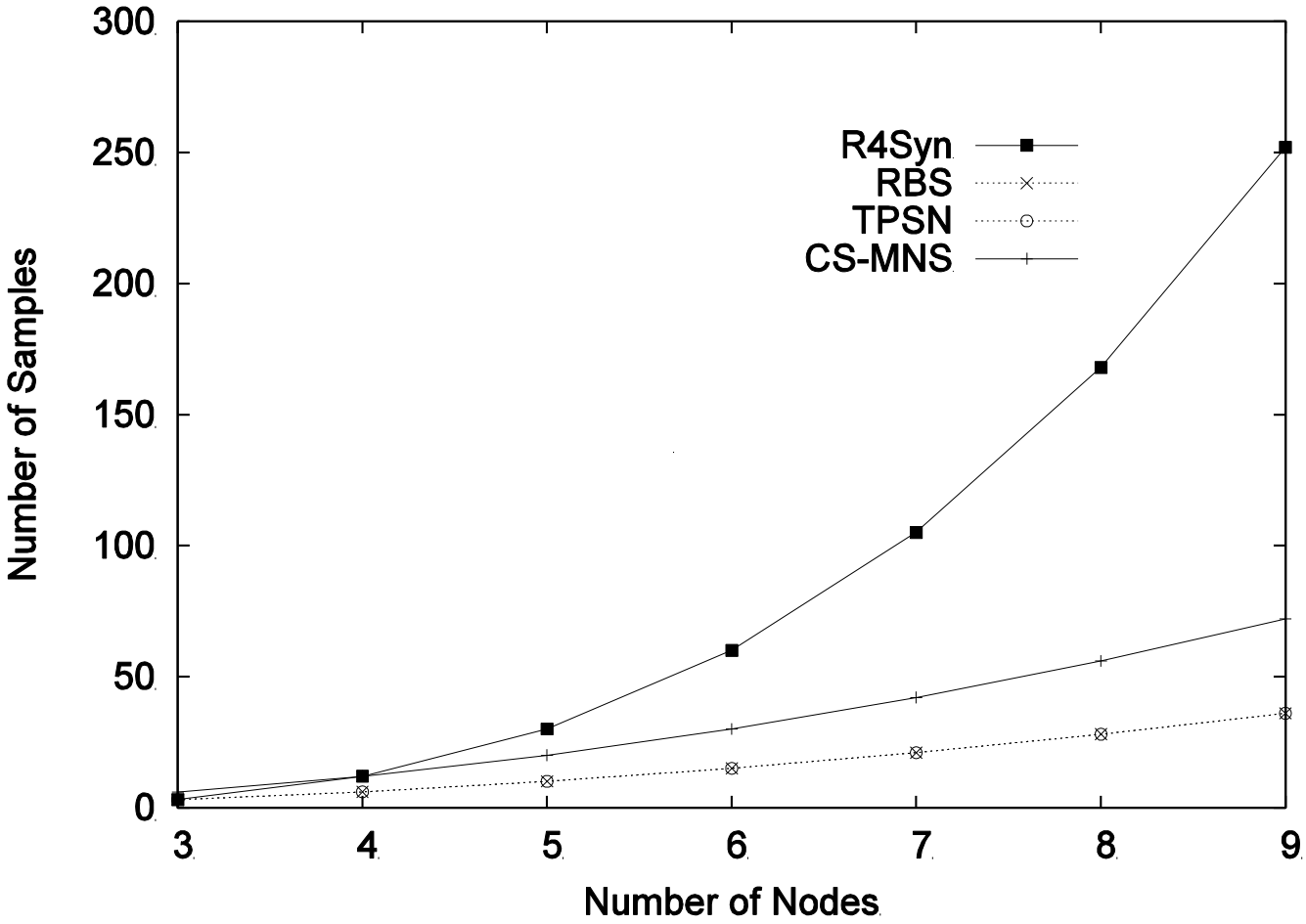
$$(I^{-1})_{1,1} = \frac{\sigma^2 \sum_{i=1}^{K} \sum_{j=2}^{h} \frac{1}{(\prod_{m=1}^{j-1} \alpha_m)^2}}{\sum_{i=1}^{K} \sum_{j=2}^{h} \left(\frac{v_{i,j}}{\prod_{m=1, m\neq j}^{h} \alpha_m}\right)^2 \sum_{i=1}^{K} \sum_{j=2}^{h} \frac{1}{(\prod_{m=1}^{j-1} \alpha_m)^2} - \left(\sum_{i=1}^{K} \sum_{j=2}^{h} \frac{v_{i,j}}{\prod_{m=2, m\neq j}^{h} \alpha_m \prod_{m=1}^{j-1} \alpha_m}\right)^2},$$

$$(I^{-1})_{2,2} = \frac{\sigma^2 \sum_{i=1}^{K} \sum_{j=2}^{h} \left(\frac{v_{i,j}}{\prod_{m=2, m\neq j}^{h} \alpha_m}\right)^2}{\sum_{i=1}^{K} \sum_{j=2}^{h} \left(\frac{v_{i,j}}{\prod_{m=1, m\neq j}^{h} \alpha_m}\right)^2 \sum_{i=1}^{K} \sum_{j=2}^{h} \frac{1}{(\prod_{m=1}^{j-1} \alpha_m)^2} - \left(\sum_{i=1}^{K} \sum_{j=2}^{h} \frac{v_{i,j}}{\prod_{m=2, m\neq j}^{h} \alpha_m \prod_{m=1}^{j-1} \alpha_m}\right)^2},$$

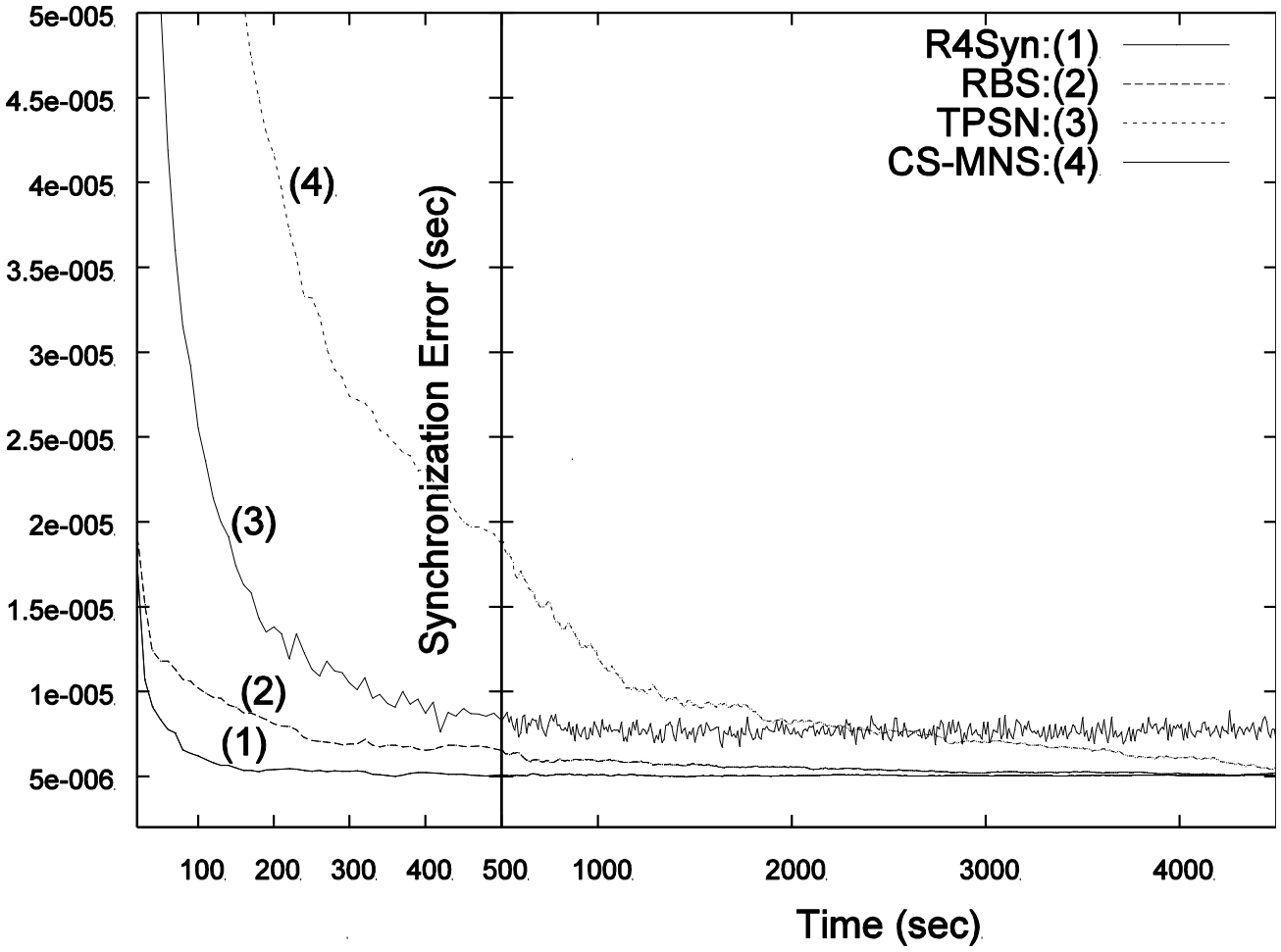
Simulation and Comparison

- Comparative simulation study vs. RBS (receiver-to-receiver) and TPSN (sender-to-receiver), and CS-MNS.
- First, the protocols are compared in single-hop environment, then in multi-hop.
- Each point of the following plots is the average of 200 measurements, and has been obtained with 99% confidence interval.

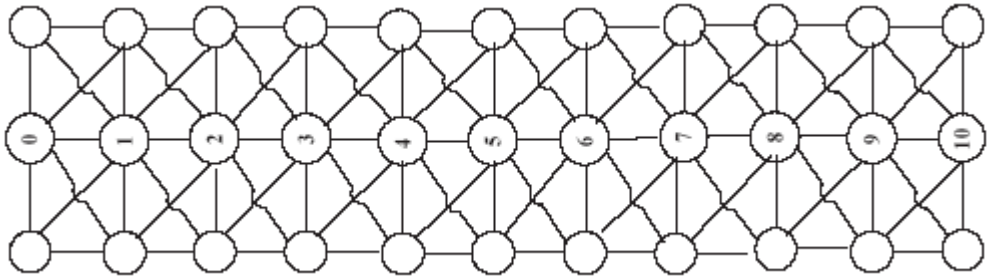
Simulation Results



Simulation Results

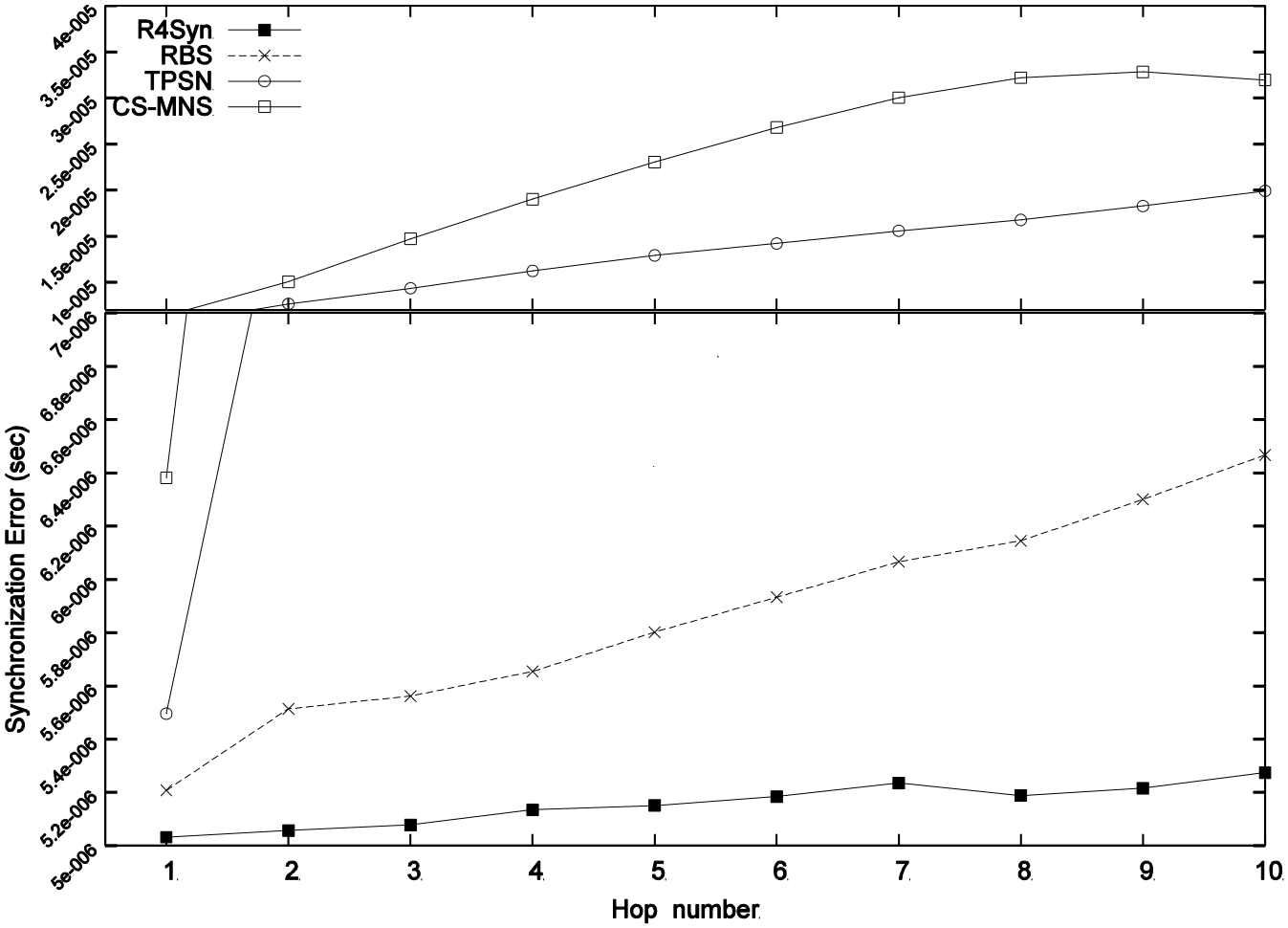


Multi-hop Precision



- Grid topology simulated, and synchronization precision between nodes 0 and 10 (10 hops) measured.

Simulation Results



Implementation

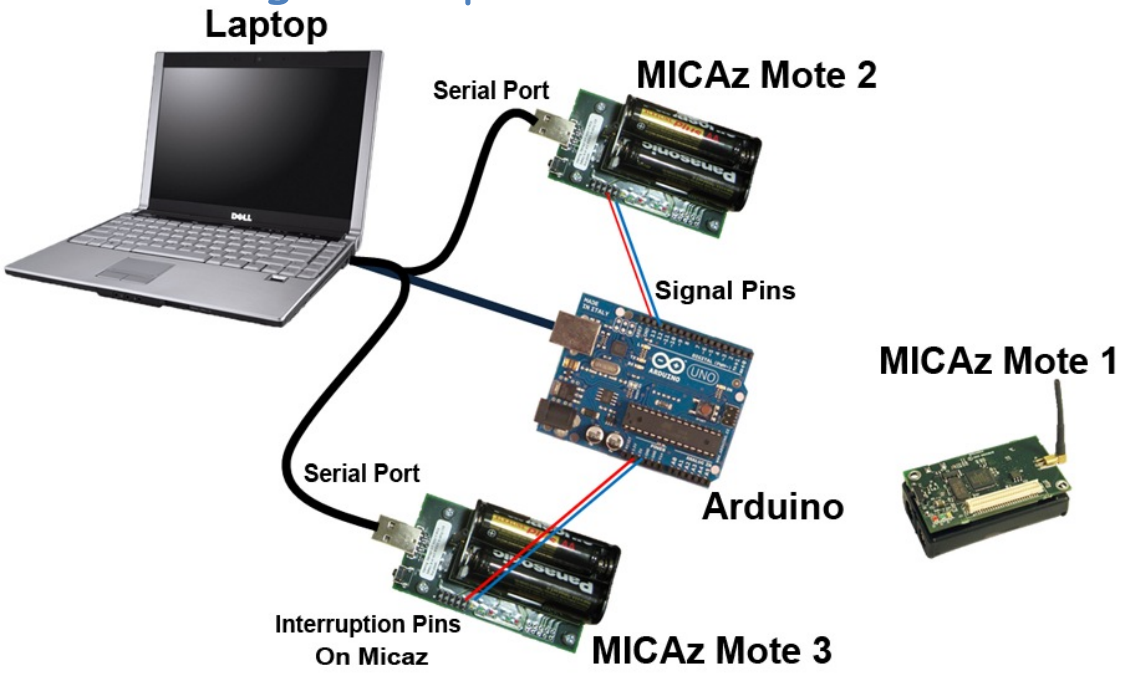
Implementation

- R[4]Syn has been implemented on MICAz and TinyOS.
- IEEE 802.15.4 radio (CC2420), which is a ZigBee compliant radio
- Problems for long number that are required for skew implementation → re-implement float calculation.
- *Problem implementing skew estimator formula:*

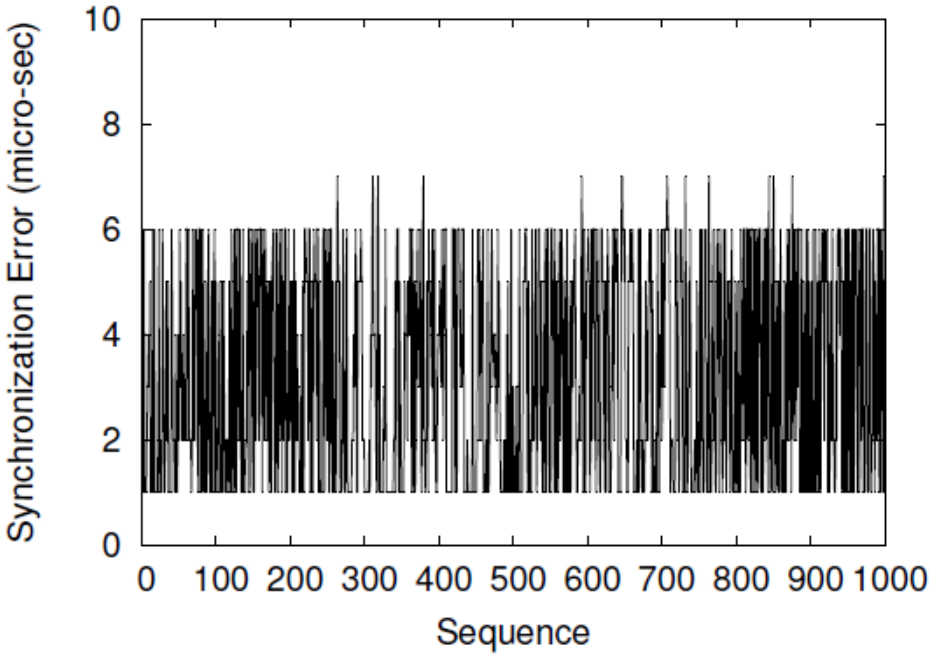
$$\widehat{\alpha}_{mle} = \frac{\sum_{i=1}^{K} u_i \sum_{i=1}^{K} v_i - K \sum_{i=1}^{K} v_i u_i}{\left(\sum_{i=1}^{K} v_i\right)^2 - K \sum_{i=1}^{K} v_i^2} \longrightarrow \frac{\sum_{j=1}^{K}\sum_{i=1}^{K}(u_j v_i - u_i v_i)}{\sum_{j=1}^{K}\sum_{i=1}^{K}(v_j v_i - v_i^2)}$$

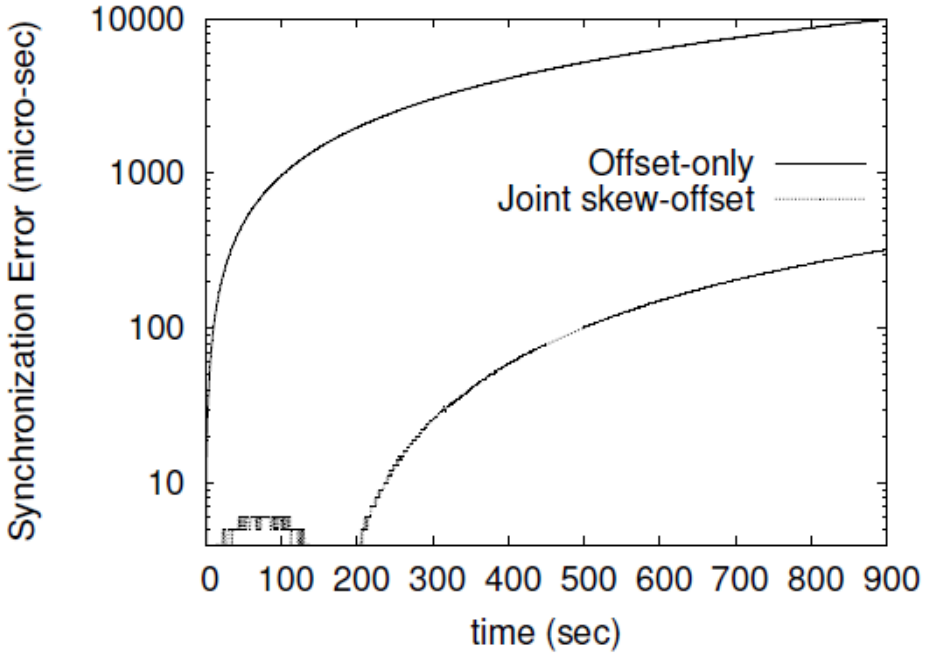
Preliminary Experiments

- Three nodes running the protocol for precision calculation.
- Low-level implementation, pins, interrupt handler.
- *Arduino micro-controller has been used to trigger simultaneous reading on both nodes.*
- Interrupt associated to the pin has been given a high priority for handling interupts.



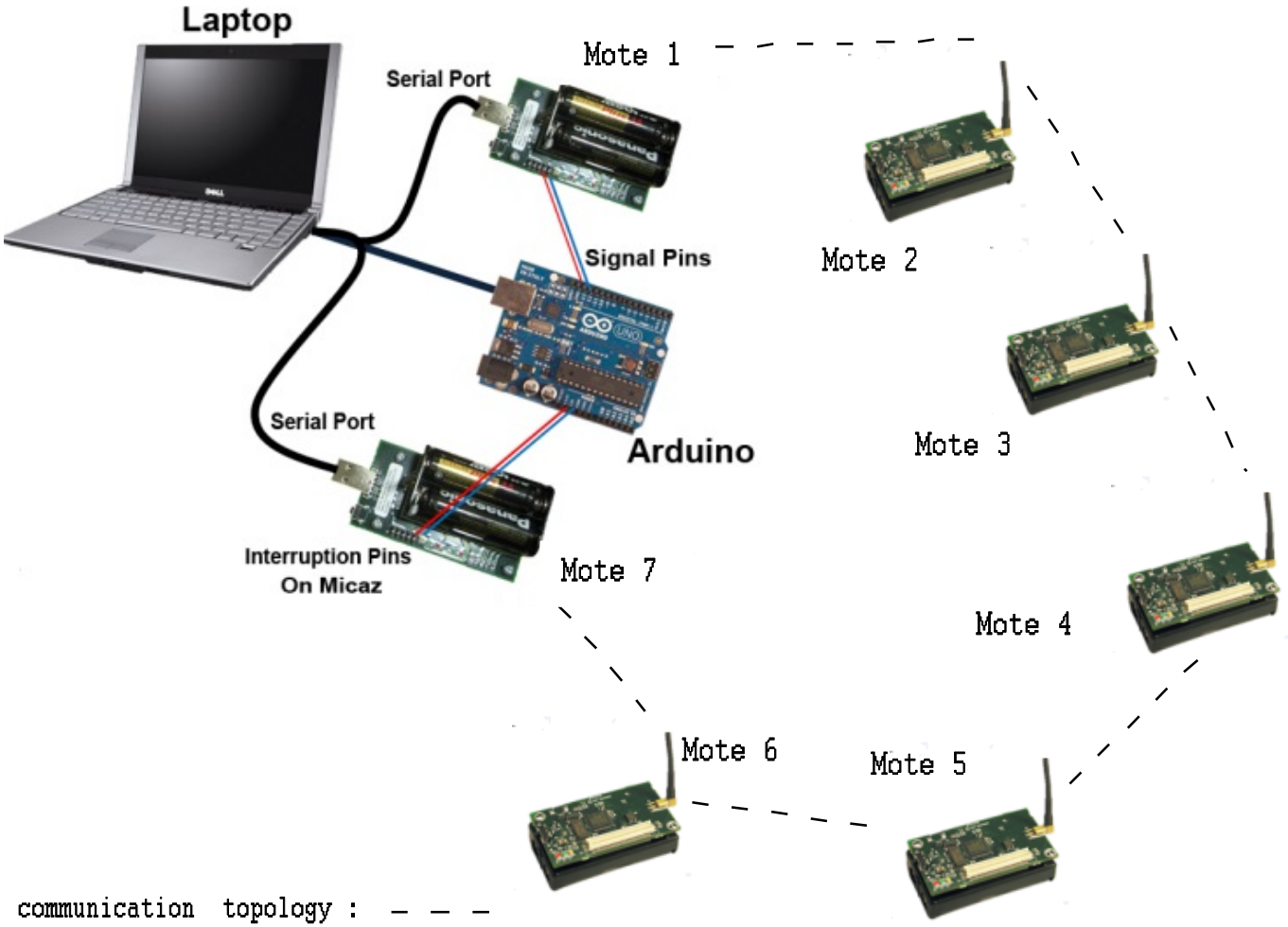
Experiment Results (Synchronization Error)



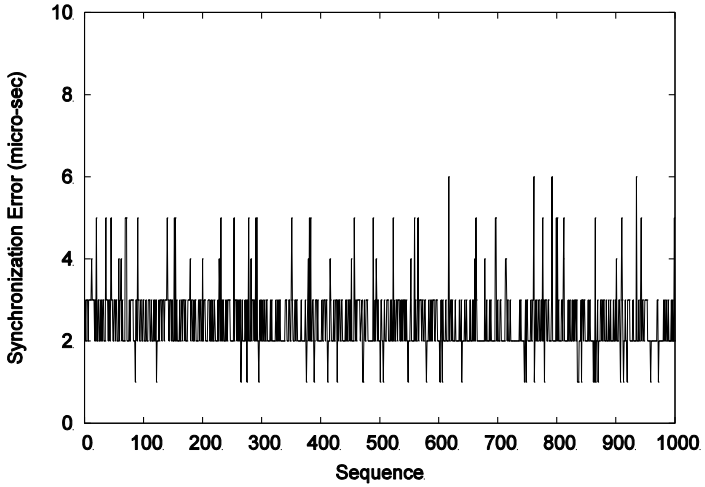


Multi-hop Experiments

- Force multi-hop communication via a controlled topology

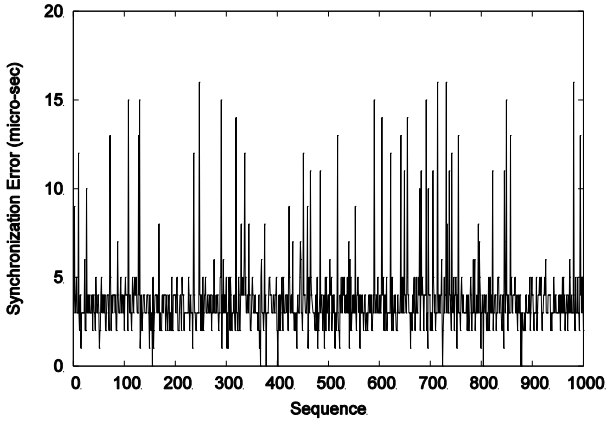


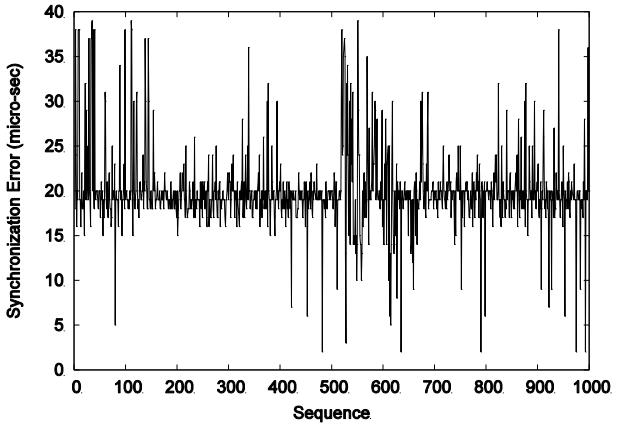
Multi-hop Experiment



1-hop

2-hop





6-hop

Error precision in μs.

	1-hop	2-hop	3-hop	4-hop	5-hop	6-hop
Average	2.45	3.65	3.74	12.58	14.84	19.93
Worst case	6	12	16	19	29	39
Best case	1	0	0	8	3	2
Below average (%)	60.3	50.8	54.3	57.9	63.2	51.3

Estimator for Receiver-to-receiver Synchronization and Exponential delays

*Djamel Djenouri MLE for Receiver-to-Receiver Time Synchronization in Wireless Networks with Exponential Distributed Delays, **IEEE VTC**, Seoul, South Korea, May 2014.*

Existing Model for Reciver2receiver and Exponential delays

I Sari et al., "on the joint synchronization of clock offset and skew….."
IEEE Tran on Com, Vol 56, No5, 2008

- Sari et al. define the only model that considers joint skew/offset MLE estimation for receiver2receiver with exponential delay channels.

$$X[i] = T_1 + \theta_x + \beta_x \tau[i] + v_{x,\lambda_x}[i],$$
$$Y[i] = T_1 + \theta_y + \beta_y \tau[i] + v_{y,\lambda_y}[i],$$

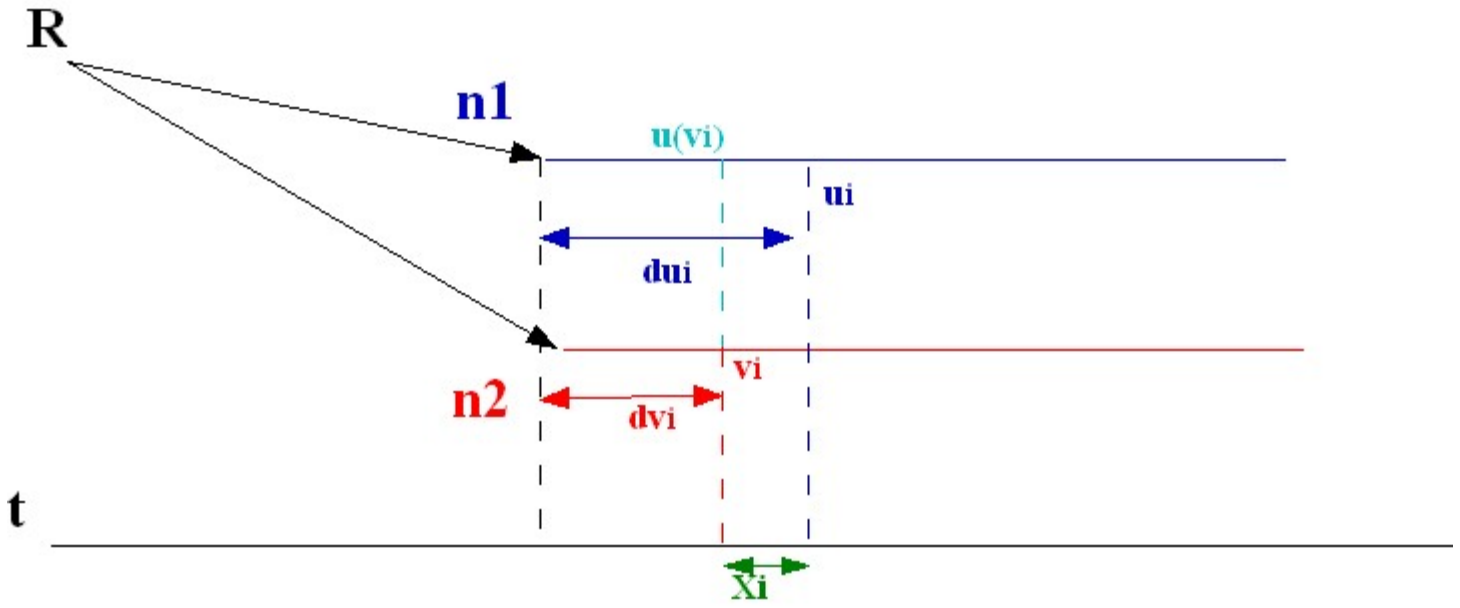
- X[i], Y[i]: the reception times at the two nodes of the i^{th} synchronization signal.
- T1: time at the sender when it sends the first synchronization signal,
- θ and β: offset and skew to the sender, $\tau[i]$: the difference between T_1 and the time of the i^{th} synchronization signal,
- $V_{.,\lambda[i]}$: noise.
- Estimate θx, θy, and βx, βy, separately, then calculate the relative parameters:
  $\theta = \theta x - \theta y$, $\beta = \beta x - \beta y$
- $V_{.,\lambda[i]}$: variables **inevitably include both transmission and reception delays**.
- **Deviates from the rec2rec concept**: Does not synchronize receiver directly, but through synchronization to the reference. Does not eliminates the sender's uncertainty.

Contribution Summary

- A model for receiver2receiver synchronization, both offset-only & joint skew/offset estimation
- It uses relative synchronization and eliminates the sender uncertainty.
- Directly synchronizes receivers,
- Estimators for both offset-only (analytical expression) & joint skew/offset (LP).

Model & Estimators

v_i: n_2 reception timestamp, u_i: n_1 reception timestamp
$u(v_i)$: n_2 estimate of n_1's time at time vi
X_i: delay difference (du_i-dv_i)
Delays, du_i and dv_i are Exp(λ), λ is unknown.
(du_i-dv_i) follows a Laplace distribution, Laplace(0, 1/λ).

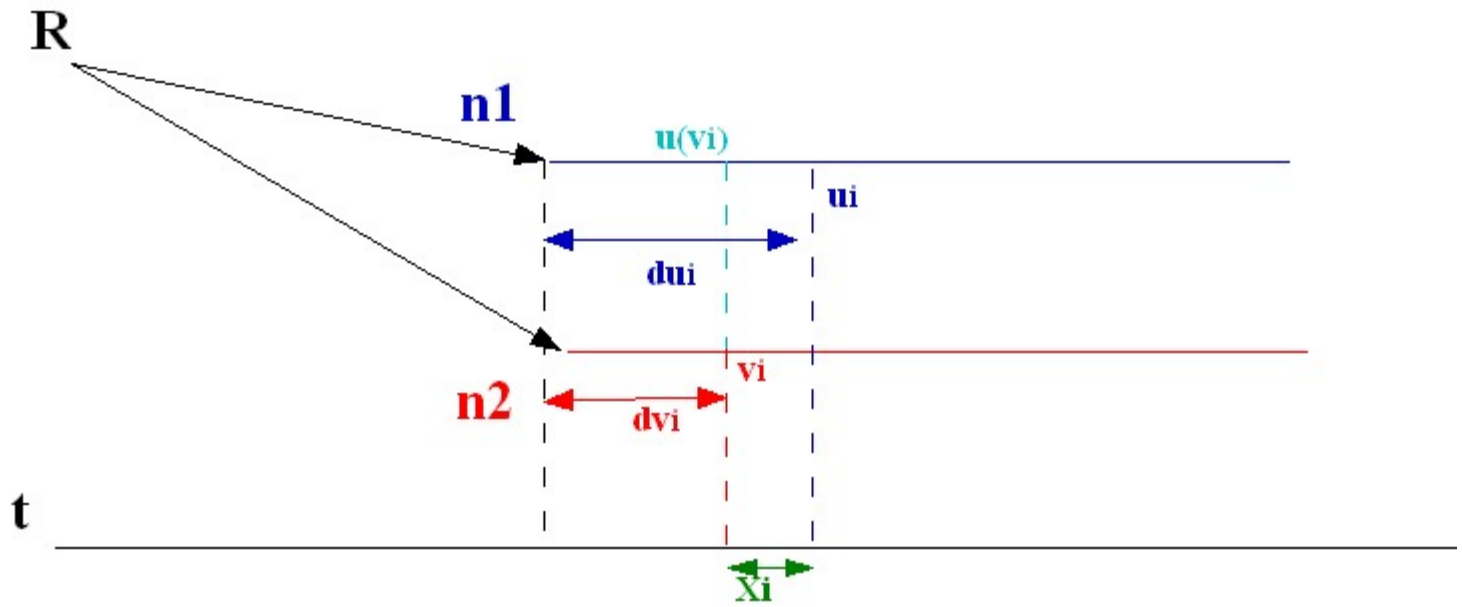


Model & Estimators

Offset-Only:

θ: relative offset

$u(v_i) = v_i + \theta = u_i - X_i \rightarrow X_i = u_i - v_i - \theta$



Model & Estimators

Offset-Only:

The likelihood function of K Laplace distributed samples, $\mathcal{L}(\theta|X_1, ...X_K)$, is given by,

$$\mathcal{L}(\theta|X_1, ...X_K) = \prod_{i=1}^{K} \frac{1}{2/\lambda} e^{\frac{-|X_i|}{1/\lambda}}$$

$$\mathcal{L}(\theta|X_1, ...X_K) = \prod_{i=1}^{K} \frac{\lambda}{2} e^{-\lambda|u_i - v_i - \theta|}. \quad (6)$$

The log-likelihood is then given by,

$$\ln(\mathcal{L}(\theta|X_1, ...X_K)) = n \ln(\lambda/2) - \lambda \sum_{i=1}^{K} |u_i - v_i - \theta|. \quad (7)$$

Maximizing Eq. (7) is equivalent to minimize $\sum_{i=1}^{K} |u_i - v_i - \theta|$.

Model & Estimators

Offset-Only:

- *(ui – vi)* denoted by *wi*,
- Order the *wi* and following the canonical order, the new samples are again denoted by w^i; $w^1 \leq w^2 \leq w^3 \ldots\ldots\ldots \leq w^k$

Minimizing the previous expression is equivalent to minimizing,

$\sum|\theta - w^i|$.

The minimum of such function has been given in:

M. M. Khashshan, "Minimizing the sum of linear absolute value functions on r," International Journal of Computational and Applied Mathematics (IJCAM), vol. 5, no. 4, pp. 537-540, 2010.

which is,

$$\begin{cases} i) w^{(K+1)/2}, & if \quad K \quad is \quad odd, \\ ii) [w^{(K/2)}, w^{K/2+1}] & if \quad K \quad is \quad even. \end{cases}$$

Model & Estimators

Joint Skew-Offset

α: relative skew, β: relative offset

$u(v_i) = \alpha v_i + \beta = u_i - X_i \rightarrow X_i = u_i - \alpha v_i - \beta$

The likelihood function for K samples, $\mathcal{L}(\alpha, \beta | X_1, ...X_K)$, becomes,

$$\mathcal{L}(\alpha, \beta | X_1, ...X_K) = \prod_{i=1}^{K} \frac{\lambda}{2} e^{-\lambda |u_i - \alpha v_i - \beta|}.$$

It results,

$$\ln(\mathcal{L}(\alpha, \beta | X_1, ...X_K)) = n \ln(\lambda/2) - \lambda \sum_{i=1}^{K} |u_i - \alpha v_i - \beta|.$$

Maximizing Eq. (12) is equivalent to minimize,

$$\sum_{i=1}^{K} |u_i - \alpha v_i - \beta|.$$

Model & Estimators

Joint Skew-Offset

This is similar to minimizing the least absolute deviation problem, which is nonlinear but can be transformed to a linear program (LP) by adding K variables, yi; i ∈ {1, …,K}, which permits to remove the absolute values.

K other variables, xi, are then added to transform inequalities into Equalities. β is supposed to be the difference of two non negative numbers (β1- β2).

D. G. Luenberger and Y. Ye, Linear and Nonlinear Programming, 3rd ed. Springer, 2008.

$$\begin{cases} min \sum_{i=1}^{K} y_i. \\ y_i - x_i + \alpha v_i - \beta_1 + \beta_2 = u_i, \forall i \in \{1, .., K\} \\ y_i - t_i - \alpha v_i + \beta_1 - \beta_2 = -u_i, \forall i \in \{1, .., K\} \\ y_i, x_i, t_i \geq 0, \forall i \in \{1, .., K\}, \alpha > 0, \beta_1, \beta_2 \geq 0. \end{cases}$$