Time Diffusion Synchronization Protocol for Wireless Networks

Weilian Su and Ian F. Akyildiz
Overview

- Motivation
- Constraints
- The Protocol
- Results
Motivation
Time Sync

• Why do we care?
  • coordination tasks such as tracking
  • often want to perceive events in the same time frame
  • many more powerful protocols depend upon time synchronization
  • e.g. MAC, localization, security, flow control
Constraints
What makes it hard?

- We can’t rely on existing solutions!
- NTP requires reliable links and nodes
- The Time-Sync Protocol for Sensor Networks (TPSN) requires a predefined, reliable hierarchy
What makes it hard?

- Temperature
- Phase noise
- Frequency noise
- Asymmetric delay
- Clock glitches
- Network partitioning
What makes it hard?

• Bottom line: we have to somehow achieve reliable time frames given unreliable clocks, links, nodes
The Protocol
At a high level

- The principle problem with other protocols is that they rely on particular nodes to be time servers or masters. We need a more robust solution, that:
  - automatically self-configures
  - is sensitive to energy requirements
Two cases

• If precise time servers are present, we want to incorporate their high-precision time estimates.

• If no precise time servers are present, the sensor network should still synchronize on a consistent time.
• If we can achieve an equilibrium time without time servers, then when time servers are present, we can simply use them for reference.

• Therefore the goal of TDP is to achieve a Universal Coordinated Time within the network.
What it looks like

Sensor nodes in the sensor field maintain at equilibrium time, which may experience gradual fluctuation in the sensor field and shift over time.
TDP Scheduling

- Each iteration of algorithm is self-contained.

- Does not run constantly. Performs several iterations, frequently reelecting master nodes in its active period.

- Does nothing in inactive period.
The algorithm(s)

- Every iteration is made up of several subparts:
  - Election/reelection of leaders
  - False ticker isolation
  - Load distribution
  - Peer evaluation
  - Time diffusion
  - Time adjustment
The algorithm(s)

\[ \text{ERP} = \text{Election/reelection of master/diffused leader node procedure} \]
\[ \text{FIA} = \text{False ticker isolation algorithm} \]
\[ \text{LDA} = \text{Load distribution algorithm} \]
\[ \text{PEP} = \text{Peer evaluation procedure} \]
\[ \text{TP} = \text{Time diffusion procedure} \]
\[ \text{TAA} = \text{Time adjustment algorithm} \]
\[ \text{CDA} = \text{Clock discipline algorithm} \]
Peer Evaluation

- Allows neighbor nodes to evaluate the stability of their own clocks by using Allan Variance.
- Relies on master nodes broadcasting timestamped messages, receiving replies, and finishing calculations for neighbors.
- Reset whenever new leaders elected.
Time Diffusion

• Creates a temporary tree-like structure, passed from one level to the next via Timing Information Handshake. Includes:
  • originating master node
  • current broadcaster
  • level depth
  • time
Time Adjustment

- Used on the tree created by the time diffusion procedure
- Nodes at every level update their times to more closely match the master's time according to level, difference, and local reliability.
Election

• Uses false ticker isolation to identify outliers and exclude them from leadership.

• Uses a load distribution algorithm to pick a node that won’t tax the network too heavily.

• Doesn’t necessarily succeed: if it doesn’t, do nothing this round.
Results
Convergence
Energy cost
Precision within a second.
Take-home ideas

• We can incorporate NTP-like hierarchies, but in a wireless sensor network, they have to be robust to sensor failures and changing conditions.

• When we adapt earlier ideas by using elections, we can get good results even in sensor networks.