Clock (time) synchronization in wireless networks

Ali Sahraee

April 2022

1 Introduction

Time synchronization in wireless networks can be applied to improve routing, power/energy consumption, and lifetime of the wireless network systems [11, 9]. Less collision can cause less re-transmission of data and hence less energy consumption for the network [11, 9]. In the following, we review some of the synchronization techniques and protocols.

1.1 Uncertainty-driven approach

This method is aiming for an empirical measurement of three key parameters of the network which affect the long-term clock synchronization. The key parameters are: 1) synchronization rate, 2) history of past synchronization beacons, and 3) the estimation scheme. In [3], they also analyze the interplay between the parameters to estimate the long-term clock drifts and design a rate-adaptive, energy-efficient long-term time synchronization algorithm, known as the rate-adaptive time synchronization (RATS) protocol. The objective of RATS protocol is to provide an optimal window time for sampling with respect to the application-specific error bound. While at runtime, RATS calculates the synchronization sampling period and the prediction error to adapt the re-synchronization rate to keep error within an application-specific bound. In this regard, RATS exploits the multiplicative increase and multiplicative decrease (MIMD) strategy. A MIMD is a simple linear scheme that converges to a sampling rate with minimal energy consumption and can adapt to changing environmental or system conditions. If prediction error is above the upper threshold, the sampling period is decreased multiplicatively. If prediction error is under the lower threshold, the sampling period is increased multiplicatively. If the prediction error is between the two thresholds, the sampling period remains the same. Experimental results show a significant energy consumption reduction compared to the best fixed-rate synchronization scheme while providing synchronization precision.

1.2 Lucarelli's protocol [5]

This protocol is proposed for large-scale, dense sensor networks. This protocol aims for converging to a synchronized state based on bi-directional nearest-neighbor coupling. Each sensor node, in the network, contains a state variable x_i ranging from 0 to 1 with a certain rate. The sensor generates a pulse signal when $x_i = 1$ and resets to 0. A sensor node that receives its neighbor's pulse signal will update the state variable of itself to $x_i + \epsilon g_{ij}(x_i)$ where ϵ is the coupling constant and $g_{ij}(x_i)$ is the coupling function between sensors i (receiver) and j (generator) with a positive value between [0,1]. In other words, for two adjacent sensors i and j:

$$x_j = 1 \implies \begin{cases} x_i \to x_i + \epsilon g_{ij}(x_i), & if \quad x_i + \epsilon g_{ij}(x_i) < 1 \\ x_i \to 0, & otherwise \end{cases} \tag{1}$$

It is guaranteed that by exploiting this protocol the network will converge to synchronicity over time.

1.3 Reachback firefly algorithm (RFA) [10]

This algorithm is a decentralized synchronicity algorithm implemented on TinyOS-based motes. RFA considers realistic effects of sensor network communication, such as message loss and delays. This implementation and algorithm is based on a mathematical model proposed in [6] which describes the spontaneous synchronization process of fireflies and neurons. Firefly synchronization is robust and can adapt to environmental changes and system modifications, such as losses, adding nodes, and link changes. In this protocol, each node acts as an oscillator with a fixed time period T. Each node has a local time t which can be incremented from 0 to T. When t reaches the time period T, the node emits a signal and resets the local time to 0. A node which hears its neighbor's signal will shorten its own time to fire by a function called the firing function and a small constant ϵ. Nodes in the network will synchronize, over time, to a common phase and firing pulse.

1.4 Timing-sync protocol for sensor network (TPSN) [2]

TPSN provides network-wide time synchronization in a sensor network. This scheme is proposed based on a conventional sender–receiver synchronization approach. TPSN consists of two phases, a level discovery phase and a synchronization phase. In the level discovery phase, the algorithm generates a hierarchical topology in the network. A level is assigned to every sensor node in the hierarchical structure. A level i sensor can communicate with at least one sensor from level $i - 1$. In this hierarchical structure, the level 0 node is called root and it is unique. The root node would initiate the second phase when the hierarchical structure is established. In the synchronization phase, each node synchronizes itself with a sensor node that is one level lower in the structure.

Finally, all nodes would be synchronized with the root node and once the root node is synchronized, the whole network is timed synchronized.

1.5 Clock-sampling mutual network synchronization (CSMNS) [7]

CSMNS is a distributed and autonomous network time-synchronization (NTS) approach. It has been proposed for the support of QoS-aware protocols in wireless Ad Hoc networks. This approach is based on the non-hierarchical influence of nodes in obtaining the network-wide time-synchronization of the clocks that supports single and multi-hop communication. In this approach, time synchronization is achieved independent of marking a node as a centralized node or using a special external or internal circuitry (e.g., master clocks) to send continuous pulses, such as the ones located in the Global Positioning System (GPS) or any cluster of broadcasting, centralized or reference nodes. It explicitly exchanges timing information among the nodes through IEEE 802.11 periodic beacon transmission. Hence, it is compatible with IEEE 802.11 physical layer and over-the-air procedures.

In a network containing N nodes, each with a local clock that has a different time-drift coefficient and initial time, CSMNS tries to synchronize all the clocks and minimize the relative time-drift of the time process. Each node in the network sends its time process in periodic beacon transmission. Once a beacon transmission is received, the node computes the correction factor by calculating the difference between the timestamp of the received beacon and the timestamp of the local node. The node sets its local clock to the value of the adjusted timestamp if it is later than its own.

CSMNS-RMN[1] is an extension to CSMNS approach in which the number of nodes contending to send a beacon at every target beacon transmission time is reduced. Within the contention window, every node is contended to send its beacon. A node does not send its beacon If it receives another beacon before sending its own. For a while, all nodes in the same locality listen to a single node that wins the contention, called the Rotating Master node. In this approach, all nodes have equal opportunity to be a RM node. Using this approach, there is significant energy saving from beacon collision reduction.

1.6 Time synchronization (TSync) [1]

TSync is a novel lightweight, flexible, and comprehensive bidirectional time synchronization service for wireless sensor networks. It consists of two mechanisms.

First, a push mechanism for accurate and low overhead global time synchronization. This mechanism is a hierarchy referencing time synchronization (HRTS) protocol. HRTS allows a reference node to synchronize multiple sensor nodes. The HRTS protocol is detailed as follows:

[1] Clock-Sampling Mutual Network Synchronization Rotating Master Node

- **Step1:** The reference node initiates the synchronization process by broadcasting an announcement on the control channel. The reference node randomly specifies one of its children in the announcement and this child (Node N) jumps to the specified clock channel.
- **Step2:** N sends a reply to the reference node.
- **Step3:** The reference node computes the clock offset and broadcasts it on the control channel to all its surrounding sensor nodes.
- **Step4:** The surrounding nodes synchronize themselves and initiate the same process with their own downstream nodes.

Second, a pull mechanism for on-demand synchronization by individual sensor node. This mechanism is an individual-based time request (ITR) protocol. ITR protocol enables each node to synchronize with the surrounding environment or to obtain time independently. The ITR protocol is based on the Simple Network Time Protocol (SNTP) [8] but with multi-channel support to address the vulnerability of SNTP to variations in delay. The ITR protocol is detailed as follows:

- **Step1:** A sensor node transmits a query message on the control channel to get a clock channel for time synchronization. The clock channel is specified in the query message.
- **Step2:** The parent nodes transfer the query message upstream until it reaches a reference node, i.e, base station.
- **Step3:** The reference node sends an acknowledgment message to the specified clock channel. All nodes along the path will switch to the specified clock channel.
- **Step4:** The sensor node sends a synchronization request to the reference node on the specified clock channel.
- **Step5:** The reference node initiates the same procedure to send the time back to the sensor node.
- **End:** The sensor node synchronizes itself according to the reference node's feedback.

Multi-channel radios are used in TSync for frequency diversity to reduce packet collisions and interferences. Reducing the number of collisions decreases the variance in round trip delay. This improves the accuracy of time estimation.

1.7 Global synchronization [4]

Li and Rus [10] discuss three different global synchronization methods. We briefly review each one of them in the following.

1.7.1 All-node-based method

This method synchronizes all nodes along a specified cycle path using a message. A node initiates the process and sends a message along the cycle. Each recipient node records its local time and its order in the cycle. Then the message travels back to the first node and it sends another message along the cycle containing information about the start and end time of the previous message. Each node on the path adjusts its local time with the computed clock error.

1.7.2 Cluster-based method

A hierarchical structure is exploited in the cluster-based method to synchronize the network. First, nodes are divided into clusters where they synchronize their clock based on the cluster head's clock. Then the cluster heads' clocks are synchronized using the all-node-based synchronization method.

1.7.3 Fully localized diffusion based method

The fully localized diffusion-based method is based on achieving global synchronization by averaging all clock readings and setting each local clock in the network to the average time. A sensor node with a high clock value sends its time to all the adjacent nodes and decreases its own local clock value. Then all the recipient nodes which have a lower clock value read the time and increase their own local clock value. The wireless sensor network will be synchronized after a number of diffusion rounds.

1.8 Other methods

There are many other methods, classified by [9], used to address the synchronization issue that involves sensors adjusting their local clocks to a common time scale. The proposed methods are as follows:

- **Master–Slave synchronization:** In this method, a node in the network is assigned to be the master node and all other nodes are slave nodes. Each slave node synchronizes its local clock with the master node.

- **Peer-to-Peer synchronization:** In this method, each pair of nodes communicate with each other to exchange time information until the network is synchronized.

- **Clock correction:** In this method, synchronization proceeds in rounds and the interval between rounds is predefined. In each round, the first node that reaches the end of the synchronization round is picked as the reference node and all nodes should synchronize their clock rate to match the reference node.

- **Untethered clocks:** In this method, a common notion of time is achieved without synchronization. Local timestamps are exchanged between nodes and then compared using a table of parameters that identifies the relation between local clocks of nodes in the network.

- **internal synchronization:** This method uses a global time to minimize the local clock offset.
- **external synchronization** This method is based on a standard source of time.
- **Probabilistic synchronization:** In this method, a probabilistic guarantee is provided on the maximum clock offset with a failure probability that can be bounded or determined.
- **Deterministic synchronization:** In this method, a deterministic upper bound on the clock offset is guaranteed with certainty.
- **Sender-to-Receiver synchronization:** In this method, the sender node periodically sends its timestamp to the receiver. The receiver synchronizes its time with the sender's timestamp and then computes the round-trip message delay.
- **Receiver-to-Receiver synchronization:** In this method, receivers hear the same broadcast message. Then they exchange the timestamp at which they received the message. Each receiver computes the offset based on the collected timestamps.

References

[1] Hui Dai and Richard Han. "TSync: A Lightweight Bidirectional Time Synchronization Service for Wireless Sensor Networks". In: *SIGMOBILE Mob. Comput. Commun. Rev.* 8.1 (Jan. 2004), pp. 125–139. ISSN: 1559-1662. DOI: 10.1145/980159.980173. URL: https://doi.org/10.1145/980159.980173.

[2] Saurabh Ganeriwal, Ram Kumar, and Mani B. Srivastava. "Timing-Sync Protocol for Sensor Networks". In: *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems.* SenSys '03. Los Angeles, California, USA: Association for Computing Machinery, 2003, pp. 138–149. ISBN: 1581137079. DOI: 10.1145/958491.958508. URL: https://doi.org/10.1145/958491.958508.

[3] Saurabh Ganeriwal et al. "Estimating Clock Uncertainty for Efficient Duty-Cycling in Sensor Networks". In: *IEEE/ACM Transactions on Networking* 17.3 (2009), pp. 843–856. DOI: 10.1109/TNET.2008.2001953.

[4] Qun Li and D. Rus. "Global clock synchronization in sensor networks". In: *IEEE INFOCOM 2004.* Vol. 1. 2004, p. 574. DOI: 10.1109/INFCOM.2004.1354528.

[5] Dennis Lucarelli and I-Jeng Wang. “Decentralized Synchronization Protocols with Nearest Neighbor Communication”. In: *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems.* SenSys ’04. Baltimore, MD, USA: Association for Computing Machinery, 2004, pp. 62–68. ISBN: 1581138792. DOI: 10.1145/1031495.1031503. URL: https://doi.org/10.1145/1031495.1031503.

[6] Renato E. Mirollo and Steven H. Strogatz. “Synchronization of Pulse-Coupled Biological Oscillators”. In: *SIAM Journal on Applied Mathematics* 50.6 (1990), pp. 1645–1662. DOI: 10.1137/0150098. eprint: https://doi.org/10.1137/0150098. URL: https://doi.org/10.1137/0150098.

[7] C.H. Rentel and T. Kunz. “A clock-sampling mutual network time-synchronization algorithm for wireless ad hoc networks”. In: *IEEE Wireless Communications and Networking Conference, 2005.* Vol. 1. 2005, 638–644 Vol. 1. DOI: 10.1109/WCNC.2005.1424575.

[8] *Simple Network Time Protocol, (SNTP) version 4. IETF RFC 2030.*

[9] Bharath Sundararaman, Ugo A. Buy, and Ajay D. Kshemkalyani. “Clock synchronization for wireless sensor networks: a survey”. In: *Ad Hoc Networks* 3 (2005), pp. 281–323.

[10] Geoffrey Werner-Allen et al. “Firefly-Inspired Sensor Network Synchronicity with Realistic Radio Effects”. In: *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems.* SenSys ’05. San Diego, California, USA: Association for Computing Machinery, 2005, pp. 142–153. ISBN: 159593054X. DOI: 10.1145/1098918.1098934. URL: https://doi.org/10.1145/1098918.1098934.

[11] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. “Wireless sensor network survey”. In: *Computer Networks* 52.12 (2008), pp. 2292–2330. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2008.04.002. URL: https://www.sciencedirect.com/science/article/pii/S1389128608001254.