A Distributed Algorithm for Network-Wide Clock Synchronization in Wireless Sensor Networks

Aitzaz Ahmad*, Davide Zennaro†, Lorenzo Vangelista‡, Erchin Serpedin§, Hazem Nounou§, and Mohamed Nounou§

*Department of Electrical and Computer Engineering, Texas A&M University, College Station, Texas, USA 77840
†Cisco Systems Italy srl, Milano-Vimercate, Italy
‡Department of Information Engineering, University of Padova, Padova, Italy
§Texas A&M University at Qatar, Doha, Qatar

Abstract—Clock synchronization has become an indispensable requirement in wireless sensor networks due to its central importance in vital network operations such as data fusion and duty cycling, and has attracted considerable research interest recently. Assuming exponentially distributed random delays in a two-way message exchange mechanism, this work proposes a network-wide clock synchronization algorithm using a factor graph representation of the network. Message passing using the max-product algorithm is adopted to derive update rules for the proposed iterative procedure. A closed form solution is obtained for each node’s belief about its clock offset at each iteration. The proposed algorithm is completely distributed as the clock offset of each node is determined at the node itself, as opposed to employing a central processing unit to compute all offsets. Simulation results show that the application of the proposed message passing-based network-wide clock synchronization algorithm provides convergent estimates for both regular cycle-free and random topologies.

I. INTRODUCTION

Clock synchronization plays an important role in establishing a common time scale among nodes which is an essential requirement for distributed processing in wireless sensor networks (WSNs). In addition, clock synchronization is a building block for a number of algorithms and protocols. Deterministic channel access is based on a firm time agreement between nodes so that techniques such as time-division-multiple-access (TDMA) and frequency-division-multiple-access (FDMA) can be successfully applied. Target localization and tracking are other applications of WSNs in which clock synchronization is mandatory in order to obtain satisfactory results [1].

As highlighted in the survey [2], several clock synchronization protocols have been introduced so far in the literature. In the receiver-receiver synchronization protocol, two nodes that have received the same broadcast message from a reference node aim to synchronize with each other directly. A particular example is the reference broadcast synchronization (RBS) algorithm [3] where a reference node broadcasts its timestamps, and the neighboring nodes estimate the clock phase and frequency differences (offset and skew, respectively) among each other by exchanging their local observations and using linear regression. In a two-way message exchange mechanism, a node synchronizes with a reference node by exchanging timestamps, and uses these timestamps to estimate its offset and skew. An example of this approach is the timing-sync protocol for sensor networks (TPSN) [4], where first a hierarchical structure in the form of a tree is built, and then every sensor synchronizes itself to its parent node through a two-way message exchange mechanism. On the other hand, in a one-way message exchange, a reference node broadcasts its timing information to many listening nodes, which record the arrival times of these messages and use them to estimate their clock parameters with respect to the reference node. The flooding time synchronization protocol (FTSP) [5] constitutes an example of this mechanism where the listening nodes estimate their clock offset and skew with respect to the reference node via linear regression. Based on this model, [6] proposed a low-power consumption estimator of clock parameters, referred to as pairwise broadcast synchronization (PBS), in which listening nodes exploit messages exchanged between a pair of synchronizing nodes.

In this work, our focus is on the classical two-way message exchange mechanism for clock synchronization, used in most of the synchronization algorithms. There is an extensive literature that makes use of this data exchange process in order to derive expressions for clock offset and skew estimators. A critical component of the clock synchronization problem is the accurate modeling of the random network delays that contaminate the data exchange process. By considering the clock offset as the cause of time disagreement between sensors, and assuming exponentially distributed network delays, [7] reported the maximum likelihood estimator (MLE) for the clock offset while an earlier study [8] stated the non-uniqueness of the MLE when both the mean of the random delays and the fixed portion of the overall delay are known. The expressions for both the best linear unbiased estimator using order statistics (BLUE-OS) and the minimum variance unbiased estimator (MVUE) are derived in [9] under the assumption of asymmetric exponentially distributed delays between uplink and downlink. By recasting the clock parameter estimation problem as a linear program, joint ML estimators of clock offset and skew are proposed in [10] for exponentially distributed network delays. The Gaussian distribution is assumed in [11] and [12] to jointly estimate the clock offset and skew for known and unknown propagation delay, respectively. A more general assumption is considered in [13], where the authors use the Gaussian mixture Kalman particle filter to cope with the presence of a general delay model represented as a mixture of several distributions. In general, the set of candidate distributions for modeling the network delays includes the Gaussian, exponential, Gamma [14] and Weibull distributions [15].
A natural extension of pairwise synchronization is to design algorithms for synchronizing sensor nodes across the whole network. A network-wide clock synchronization algorithm is proposed in [16] assuming both clock offset and fixed skew affecting the running behavior of the oscillators. A statistical analysis of the algorithm proposed in [16] is performed in [17]. The authors in [18] proposed a synchronization algorithm by assuming no initial clock offsets but time-varying skews among oscillators in the network. More recently, there has been a focus on the application of graphical models and message passing to the clock synchronization problem. Assuming Gaussian distributed network delays, the authors in [19] proposed the use of the sum-product algorithm in order to design a network-wide clock synchronization method, where clock offset is the only cause of time disagreement between sensors (i.e., no skew). Factor graphs are also used in [20] to derive a closed-form expression of the clock offset estimator in the presence of exponentially distributed network delays, while assuming that the clock offset evolves along a Gauss-Markov model. However, this analysis was limited to a pairwise offset estimation scenario where only two nodes attempt to synchronize with each other.

It was observed in [8] that a single server M/M/1 queue can represent the cumulative link delay in WSN, when the random delays are modeled as exponential random variables. Moreover, the Minimum Link Delay algorithm, proposed in [8], which shows good performance, was derived independently in [7] assuming exponentially distributed network delays. Therefore, the assumption of an exponential distribution seems to be well suited for modeling WSN link delays. It must also be emphasized that the sum-product algorithm used for Gaussian distributed delays in [19] is intractable for the case of exponentially distributed network delays. In this work, a factor graph representation of the sensor network, in conjunction with message passing, is used to design a network-wide clock synchronization algorithm assuming exponentially distributed network delays. This paper also extends the framework in [20] to the much more challenging case of network-wide synchronization. The max-product algorithm is used to design the message exchange rules between sensors. This leads to an iterative update of the belief, whose maximization gives rise to the expression for the clock offset estimator. A simulation study is then performed to show that the proposed algorithm converges in various network topologies of interest.

The paper is structured as follows. In Section II, the system model is described along with the two-way message exchange between pairs of nodes. The details of the proposed algorithm as well as the specific message exchange rules are provided in Section III. Numerical simulations are presented in Section IV. Finally, conclusions are drawn in Section V.

II. SYSTEM MODEL

Consider a strictly connected network of \(N+1\) sensor nodes that share a wireless channel to communicate. It is assumed that clock inconsistency between different nodes is due to the presence of clock offset, so that the time measured by node \(i\), denoted by \(T_i\), can generally be expressed as [1]

\[
T_i = t + \theta_i ,
\]

where \(t\) is the reference time and \(\theta_i\) represents the clock offset of node \(i\), \(i = 1, \ldots, N\). Clock offsets are assumed mutually independent and their prior beliefs, denoted by \(p_1(\theta_i)\), are considered uniform over the entire real axis. The clock of node 0 is assumed to provide the reference time to the network, therefore its clock offset is set to zero, i.e., \(\theta_0 = 0\) and \(p_0(\theta_0) = \delta(\theta_0)\). The communication network topology is considered time invariant and described by the link set \(L \triangleq \{(i,j) : \text{there is a link between nodes } i \text{ and } j\}\).

The two-way message exchange mechanism is depicted in Fig. 1 for the \(k\)-th message exchange. The data transmission is started by node \(i\), which sends a message to node \(j\) containing the timestamp \(T_{i,k}^1\) with respect to the clock of node \(i\). Node \(j\) records the time \(T_{j,k}^2\) at which this message is received, based on its own clock. After a predetermined period of time, node \(j\) replies with a message carrying the timestamps \(T_{j,k}^2\) and \(T_{j,k}^3\), the latter being the second message sending timestamp. Finally, node \(i\) receives this message at time \(T_{i,k}^4\) with respect to its own time scale. Since both nodes need all the timestamps for processing purposes, we assume that the first message in the \((k+1)\)-th round carries inside the timestamp \(T_{i,k}^4\) as well. The message exchange is repeated for \(k \in \{1, \ldots, K\}\), while node \(i\) finally sends an additional message to node \(j\) containing the timestamp \(T_{i,K}^4\), so that both nodes have all the timestamps.

In this work, the fixed delay between nodes \(i\) and \(j\), denoted by \(d_{ij}\), is considered unknown but symmetric in uplink and downlink transmission. The random portion of delays in uplink and downlink are asymmetric and are denoted by \(X_{ij,k}\) and \(Y_{ij,k}\), respectively. The pair-wise message exchange between neighboring nodes \(i\) and \(j\) can be expressed as [19]

\[
\begin{align*}
T_{j,k}^2 &= T_{i,k}^1 + \theta_j - \theta_i + d_{ij} + X_{ij,k} \\
T_{i,k}^4 &= T_{j,k}^3 + \theta_i - \theta_j + d_{ij} + Y_{ij,k} .
\end{align*}
\]  (1)

where the random delays \(X_{ij,k}\) and \(Y_{ij,k}\) are assumed to be independent and exponentially distributed with mean \(1/\lambda\), \(\lambda > 0\). By recalling the steps performed in [8], the timestamps are processed to obtain

\[
\begin{align*}
U_{ij,k} &\triangleq T_{j,k}^2 - T_{i,k}^1 = d_{ij} + (\theta_j - \theta_i) + X_{ij,k} \\
V_{ij,k} &\triangleq T_{i,k}^4 - T_{j,k}^3 = d_{ij} - (\theta_j - \theta_i) + Y_{ij,k} .
\end{align*}
\]  (2)

An additional message may not be necessary if piggybacking is adopted.
It was shown in [7] that the maximum likelihood estimator (MLE) for the clock offset difference \((\theta_j - \theta_i)\) is given by

\[
S_{ij} = \frac{U_{ij}(1) - V_{ij}(1)}{2},
\]

where \(U_{ij}(1)\) and \(V_{ij}(1)\) denote the first order statistics of the data sets \(\{U_{ik}, k = 1, \ldots, K\}\) and \(\{V_{jk}, k = 1, \ldots, K\}\), respectively. It was proved in [15] that \(U_{ij}(1)\) and \(V_{ij}(1)\) constitute a sufficient statistic for estimating \((\theta_j - \theta_i)\), and that (3) is also the uniformly minimum variance unbiased (UMVU) estimator for \((\theta_j - \theta_i)\). Based on the aforementioned properties, it suffices to work with pre-processed data \(S_{ij}\) in order to make the network-wide algorithm computationally sustainable and scalable.

Using the definition of \(S_{ij}\) in (3) and the system model expressed in (2), we can write

\[
S_{ij} = (\theta_j - \theta_i) + Z_{ij},
\]

where \(Z_{ij} \sim \mathcal{N}(0, 1/2K\lambda)\), i.e., \(Z_{ij} \sim \mathcal{N}(0, 1/2K\lambda)\). The factor 1/K comes from computing the minimum of \(K\) independent exponential random variables. Consequently, \(S_{ij} \sim \mathcal{N}(\theta_j - \theta_i, 1/2K\lambda)\), so that its probability density function (pdf) can be expressed as

\[
p(S_{ij}|\theta_i, \theta_j) = K\lambda \exp(-2K\lambda|S_{ij} - \theta_j + \theta_i|).
\]

The goal of this work is to infer the value of the clock offset \(\theta_i\) for all \(i\), using data \(S_{ij}\) gathered from the message exchanges between nodes \((i, j) \in L\). Clearly, an estimator for \(\theta_i\) does not depend solely on the data exchanged between nodes \(i\) and \(j \in N_i\), with \(N_i\) being the set of neighbors of node \(i\). Rather, all nodes in the network play a role through data exchanges between pairs of neighbors. Inference about \(\theta_i\) can be obtained by

\[
\hat{\theta}_i \triangleq \arg \max_{\theta_i} p_i(\theta_i|S_i)
\]

where the a-posteriori pdf, \(p_i(\theta_i|S_i)\), is given by

\[
p_i(\theta_i|S) = \int_{\theta_i} p(\theta|S) d\theta_i,
\]

where \(\theta \triangleq [\theta_1, \ldots, \theta_N]\), the symbol \(\hat{\theta}_i\) denotes the vector composed of all the offset variables except \(\theta_i\), and \(S\) is the (antisymmetric) \((N+1) \times (N+1)\) matrix composed of entries \(S_{ij}\) given by (3) if \((i, j) \in L\), and zero otherwise. The computation of \(p(\theta_i|S)\) in (6) is mathematically cumbersome due to multidimensional integration operations. The next section shows how the problem of inferring a clock offset \(\theta_i\) is solved by obtaining the pdf \(p(\theta_i|S)\) in (6) via belief propagation on a suitably defined factor graph.

### III. The Belief Propagation Algorithm

The overall a-posteriori pdf in (6) can be factorized using Bayes’ rule and the independence of the link delays in (1) as well as the independence of the clock offsets, as follows

\[
p(\theta|S) \propto \prod_{(i,j) \in L} h_{ij}(\theta_i, \theta_j) \prod_{i=0}^N p_i(\theta_i),
\]

where \(h_{ij}(\theta_i, \theta_j) \triangleq p(S_{ij}|\theta_i, \theta_j)\). The factorization in (7) naturally leads to a factor graph representation of the problem. By representing random variables with circles and factors with squares, a factor graph such as the one in Fig. 2 can be easily obtained [21]. Message passing on the resulting factor graph is then applied to obtain approximately (since the graph typically has cycles), the posterior pdf \(p_i(\theta_i|S)\), which can then be maximized to yield the clock offset \(\theta_i\).

We use max-product message passing algorithm to infer \(\theta_i\) in (6), primarily due to its greater analytical tractability in our synchronization problem, as compared to sum-product message passing algorithm. In the max-product algorithm, the message \(m_{\theta_i \rightarrow h_{\ell}}(\theta_i)\) from the variable node \(\theta_i\) to the factor node \(h_{\ell}\) is given by the product of the prior \(p_i(\theta_i)\) and the incoming messages from all factor nodes except the \(\ell\)-th one, i.e., [22]

\[
m_{\theta_i \rightarrow h_{\ell}}(\theta_i) = p_i(\theta_i) \prod_{j \in N_i, j \neq \ell} m_{h_{ij} \rightarrow \theta_i}(\theta_i).
\]

At the factor node, the marginalization process is performed using the ‘max’ operator, so that the message \(m_{h_{\ell} \rightarrow \theta_i}(\theta_i)\) from the factor node \(h_{\ell}\) to the variable node \(\theta_i\) is given by

\[
m_{h_{\ell} \rightarrow \theta_i}(\theta_i) = \max_{\theta_i} \left[ m_{h_{ij} \rightarrow \theta_i}(\theta_i) h_{ij}(\theta_i, \theta_j) \right].
\]

Fig. 2 also shows an example of message exchange in a factor graph. At node \(i\), the belief about \(\theta_i\), obtained from all the connected factor nodes as well as the prior, is given by

\[
b_i(\theta_i) = p_i(\theta_i) \prod_{j \in N_i} m_{h_{ij} \rightarrow \theta_i}(\theta_i).
\]
This belief \( b_i(\theta_i) \) represents the posterior pdf \( p_i(\theta_i|S) \) in (6). An estimate of \( \theta_i \) is then obtained as
\[
\hat{\theta}_i = \arg \max_{\theta_i} b_i(\theta_i) .
\] (11)

### A. Message Computation

At the first iteration, denoted by \( t = 0 \) in the superscript, for all non-reference nodes, i.e., \( i \neq 0 \), (8) is simply equal to the prior, since no messages have arrived from neighboring nodes yet. This implies that
\[
m_{\theta_i \to h_{\ell i}}(\theta_i) = p_i(\theta_i) ,
\]
while the subsequent message from the factor node \( h_{\ell i} \) to the variable node \( \theta_i \) is given by
\[
m_{\theta_i \to h_{\ell i}}(\theta_i) = \max_{\theta_i} \left[ m_{\theta_i \to h_{\ell i}}(\theta_i) h_{\ell i}(\theta_i, \theta_{\ell}) \right] = \max_{\theta_i} \left[ p_i(\theta_i) p(S_{\ell i}|\theta_i, \theta_{\ell}) \right] \propto 1 ,
\] (12)
where (12) follows from (5) and the fact that \( p_i(\theta_i) \) is uniform. On the other hand, for the reference node \( i = 0 \), the message sent by the variable node \( \theta_0 \) to the factor node \( h_{0\ell} \), \( \ell \in \mathcal{N}_0 \), can be written as
\[
m_{\theta_0 \to h_{0\ell}}(\theta_0) = p_0(\theta_0) = \delta(\theta_0) .
\]
The subsequent message from the factor node \( h_{0\ell} \) to the variable node \( \theta_i \) can be expressed as (cf. (9))
\[
m_{h_{0\ell} \to \theta_i}(\theta_i) \propto \max_{\theta_0} \left[ m_{h_{0\ell} \to \theta_i}(\theta_0) h_{0\ell}(\theta_0, \theta_{\ell}) \right] = \max_{\theta_0} [\delta(\theta_0) p(S_{0\ell}|\theta_0, \theta_{\ell})] \propto \exp(-2K\lambda|\theta_\ell - S_{0\ell}|) ,
\] (13)
where (13) follows from (5). This implies that the message \( m_{h_{0\ell} \to \theta_i} \) is proportional to a Laplace pdf with mean \( S_{0\ell} \) and parameter \( 1/2K\lambda \). In other words, at the first iteration, the messages coming to the variable node \( \theta_i \) from the factor nodes other than \( h_{0\ell}, \) i.e., \( m_{h_{qu} \to \theta_i} \) with \( q \in \mathcal{N}_\ell, q \neq 0 \) are just constants given by (12).

At the second iteration \( t = 1 \), the variable node \( \theta_i \) uses (8) to compute the message
\[
m_{\theta_i \to h_{\ell i}}(\theta_i) = p_\ell(\theta_i) \cdot \prod_{q \in \mathcal{N}_\ell, q \neq j} m_{h_{qu} \to \theta_i}(\theta_i) \propto m_{h_{0\ell} \to \theta_i}(\theta_i) .
\] (14)
As a consequence, the message from the factor node \( h_{\ell j} \) to the variable node \( \theta_j \) can be expressed as
\[
m_{h_{\ell j} \to \theta_j}(\theta_j) = \max_{\theta_j} \left[ m_{h_{\ell j} \to \theta_j}(\theta_j) h_{\ell j}(\theta_j, \theta_{\ell}) \right] = \max_{\theta_j} \left[ m_{h_{0\ell} \to \theta_i}(\theta_i) p(S_{\ell j}|\theta_j, \theta_{\ell j}) \right] ,
\] (15)
where (15) follows from (7) and (14). The following lemma provides a closed form expression for the message \( m_{h_{\ell j} \to \theta_j} \), and also serves as a stepping stone for the computation of the generic message sent by a factor node to a variable node at the general iteration index \( t \).

**Lemma 1:** The message \( m_{h_{\ell j} \to \theta_j}^{(1)} \) is proportional to a Laplace pdf with mean \( W_{\ell j}^{(1)} \) and parameter \( 1/2K\lambda \), i.e.,
\[
m_{h_{\ell j} \to \theta_j}^{(1)}(\theta_j) \propto \exp\left(-2K\lambda|\theta_j - W_{\ell j}^{(1)}|\right) .
\] (16)
where \( W_{\ell j}^{(1)} \triangleq S_{\ell j} + S_{0\ell} \) is to be interpreted as the message received by the node \( \theta_\ell \) from its neighbors except the node \( \theta_j \) at iteration \( t = 1 \).

**Proof:** See [23].

At a general iteration \( t \), it is clear that several neighboring nodes of a particular node \( i \) that have updated their beliefs after receiving some communication from the reference node, either directly or indirectly, will be sending messages similar to (16). Messages from such nodes will be termed as non-constant messages in the sequel. Hence it follows that at a general iteration \( t \), (8) is a product of a number of Laplace distributions, taking the form
\[
m_{\theta_i \to h_{\ell i}}^{(t)}(\theta_i) \propto \exp\left(-2K\lambda \sum_{n=1}^{r_{i,\ell}^{(t)}-1} |\theta_i - W_{\ell i}^{(t)}(n)|\right) ,
\] (17)
where \( r_{i,\ell}^{(t)} - 1 \) is the number of neighbors of node \( i \) other than node \( \ell \), that have sent non-constant messages at iteration \((t-1)\). The sequence \( \{W_{\ell i}^{(t)}(n)\} \) denotes the non-constant messages sent by neighbors of node \( i \) other than node \( \ell \). Without loss of generality, \( \{W_{\ell i}^{(t)}(n)\} \) are assumed sorted in an increasing order, i.e., \( -\infty < W_{\ell i}^{(t)}(1) \leq W_{\ell i}^{(t)}(2) \leq \ldots \leq W_{\ell i}^{(t)}(r_{i,\ell}^{(t)} - 1) < +\infty \). It must be remarked that \( \{W_{\ell i}^{(t)}(n)\} \) are fully determined by data (cf. (16) for \( t = 1 \)).

Using (5), (7) and (17), the message (9) at iteration \( t \) can be written as
\[
m_{\theta_i \to h_{\ell i}}^{(t)}(\theta_i) = \max_{\theta_i} \left[ m_{\theta_i \to h_{\ell i}}^{(t)}(\theta_i) h_{\ell i}(\theta_i, \theta_{\ell}) \right] \propto \max_{\theta_i} \exp \left[ -2K\lambda \left(|\theta_i - \theta_{\ell} + S_{\ell i}| + \sum_{n=1}^{r_{i,\ell}^{(t)}-1} |\theta_i - W_{\ell i}^{(t)}(n)|\right) \right] .
\] (18)

The following lemma provides an approximate closed form expression for the message \( m_{h_{\ell i} \to \theta_j} \) at iteration \( t \).

**Lemma 2:** The message \( m_{h_{\ell i} \to \theta_j}^{(1)} \) can be approximated by a Laplace pdf with mean \( S_{\ell j} + C_{i \to \ell}^{(t)} \) and parameter \( 1/2K\lambda \), and can be expressed as
\[
m_{h_{\ell i} \to \theta_j}^{(1)}(\theta_j) \sim \exp\left(-2K\lambda\left|\theta_j - (S_{\ell j} + C_{i \to \ell}^{(t)})\right|\right) ,
\] (19)
where \( \sim \) should be read “approximately proportional to”, and
with
\[ C_{i,t}^{(t)} = \begin{cases} W_{i,t}^{(t)} \left( \frac{r_{i,t}^{(t)}}{2} \right), & \text{even } r_{i,t}^{(t)}, \\ \frac{1}{2} \left[ W_{i,t}^{(t)} \left( \frac{r_{i,t}^{(t)} + 1}{2} \right) + W_{i,t}^{(t)} \left( \frac{r_{i,t}^{(t)} - 1}{2} \right) \right], & \text{odd } r_{i,t}^{(t)}. \end{cases} \]

Proof: See [23].

An example of message propagation between sensor nodes is shown in Fig. 3, in which it is assumed that the messages from the variable nodes to the factor nodes are not actually transmitted as they only serve as intermediate steps for computing the messages in (19), which are then exchanged among neighboring nodes. In the example shown in Fig. 3, the sequence \( \{W_{i,t}^{(t)}(n)\} \) is given by
\[ \{W_{i,t}^{(t)}(n)\} = \left\{ S_{\alpha i} + C_{\alpha \rightarrow i}^{(t-1)}, S_{\beta i} + C_{\beta \rightarrow i}^{(t-1)}, S_{\gamma i} + C_{\gamma \rightarrow i}^{(t-1)} \right\}. \]
This sequence is then sorted to obtain \( W_{i,t}^{(1)} \), \( W_{i,t}^{(2)} \) and \( W_{i,t}^{(3)} \). The quantity \( C_{i,t}^{(t)} \) is then determined using (20). Equation (20) essentially requires calculating the exact or approximate median of \( \{W_{i,t}^{(t)}(n)\} \), depending on \( r_{i,t}^{(t)} \). This explains the need for sorting this sequence as mentioned earlier.

B. A Synchronization Algorithm

At each iteration, node 0 sends to its neighbors the mean of the distribution described by (13), which is \( S_{0\ell} \) with \( \ell \in N_0 \). On the other hand, from iteration 0 onward, the other nodes send no data to its neighbors unless their beliefs are different from the uniform priors. At the beginning of step \( t \), node \( i \neq 0 \) receives 1 \( r_{i,t}^{(t)} \) non-constant values \( S_{j} + C_{j \rightarrow i}^{(t-1)} \), where \( j \in N_i \) (if \( i \) is not receiving anything from a particular neighbor \( j \), it will also be assumed that the message \( m_{h_j \rightarrow i} \) is constant, and therefore useless) and it sorts all these values to obtain a sequence2 \( \{W_{i,t}^{(t)}(n)\} \). The following theorem provides a closed form solution of the belief \( b_i^{(t)} \) at iteration \( t \).

**Theorem 1:** The belief \( b_i^{(t)} \) of a node about its clock offset can be expressed as
\[ b_i^{(t)}(\theta_i) = p_i(\theta_i) \cdot \prod_{j \in N_i} m_{h_j \rightarrow i}(\theta_i) \times \exp \left( -2K\lambda \sum_{n=1}^{r_{i,t}^{(t)}} |\theta_i - W_i^{(t)}(n)| \right), \]

Proof: The proof simply follows from (10) and Lemma 2, and the fact that the prior pdf \( p_i(\theta_i) \) is uniform. \( \blacksquare \)

1It must be noted that \( r_{i,t}^{(t)} - 1 \) refers to the number of neighbors of node \( i \), other than node \( \ell \), that send non-constant messages at iteration \( t \), while \( r_{i,t}^{(t)} \) indicates the number of neighbors of node \( i \) that send non-constant messages.

2It must be noted that \( \{W_{i,t}^{(t)}(n)\} \) refers to the sorted sequence of messages received from neighbors of node \( i \), other than node \( \ell \), at iteration \( t \), while \( \{W_{i,t}^{(t)}(n)\} \) indicates the sorted sequence of messages received from neighbors of node \( i \).

---

**Fig. 3:** Example of message circulation between sensor nodes: node \( i \) computes the message to send to the node \( \ell \) at time \( t \) based on the information messages received at time \( t-1 \) from the nodes \( \alpha, \beta \) and \( \gamma \) and the information-less constant message received from node \( \delta \).

At this point, by maximizing the belief (21), node \( i \) can compute the estimate (11) of \( \theta_i \) as follows
\[ \hat{\theta}_i^{(t)} = \arg \max_{\theta_i} b_i^{(t)}(\theta_i) = W_i^{(t)} \left( \left\lfloor \frac{r_{i,t}^{(t)}}{2} \right\rfloor \right), \]
where \( \lfloor x \rfloor \) is the smallest integer not less than \( x \). The maximizer has been computed through techniques similar to those used for proving Lemma 2. Lastly, node \( i \) transmits all the messages \( S_{i\ell} + C_{i \rightarrow \ell} \), computed according to (19) and (20), to its neighboring nodes \( \ell \).

The estimate \( \hat{\theta}_i^{(t)} \) is updated according to (22) until the first update of less than \( \varepsilon \% \) is encountered, where \( \varepsilon \) is the desired error margin. This time instant is denoted by \( t_i^{(t)} \) at node \( i \), and the estimate \( \hat{\theta}_i^{(t)} \) is kept unchanged for all \( t > t_i^{(t)} \). After this instant, node \( i \) stops sending messages to its neighbors.

The steps of the algorithm are summarized in Table I. It must be emphasized that the algorithm is completely distributed, since the only offset of the node \( i \) is determined at the node itself, as opposed to employing a central processing unit to compute all offsets.

---

**IV. SIMULATION RESULTS**

This section presents numerical simulation results to assess the performance of the proposed algorithm in structured as well as random topologies. The local estimate of clock offset error (3) is based on \( K = 4 \) message exchanges. Results are averaged both with respect to the instances of the random delays and with respect to the clock offsets, generated uniformly in the interval \([-30,30]\). The algorithm runs for a maximum of 150 iterations but the update of \( \hat{\theta}_i^{(t)} \) (22) is stopped when estimate updates of less than \( \varepsilon = 5\% \) are encountered. The MSE is given by
\[ \text{MSE}(t) = \frac{\sum_{i=1}^{N} \text{MSE}_i(t)}{N} \]
TABLE I: The Synchronization Algorithm.

\[ F = \{ 0 \}. \]

For \( t \) from 0 on

Node \( i = 0 \) sends to its neighbors a packet with means \( S_{i\ell}, \ell \in N_0 \), according to (13).

For each node \( i \notin F \) in parallel to node 0

Node \( i \) computes the belief \( b_i(t) \) according to (21) from data \( W_i(t) \) received from its neighbors.

Node \( i \) computes the estimate \( \hat{\theta}_i(t) \) according to (22).

If \( |\hat{\theta}_i(t) - \hat{\theta}_i(t-1)|/\hat{\theta}_i(t-1) > \varepsilon \)

Then

\[ \hat{\theta}_i(t) = \hat{\theta}_i(t-1). \]

Else

\[ \hat{\theta}_i(t) = \hat{\theta}_i(t-1). \]

End If

End For

End For

Fig. 4: Connectivity structure of a 4-node chain and 9-node mesh grid topology. Node ‘0’ is the reference node.

where \( \text{MSE}_i(t) = \mathbb{E} \left[ \left( \hat{\theta}_i(t) - \theta_i \right)^2 \right] \).

The plots in this section show the MSE in chain graphs (CGs), mesh grids (MGs) and random geometric graph (RGG) topologies. An example of the connectivity structure of CG and MG is depicted in Fig. 4 for illustration purposes.

Fig. 5 depicts a comparison of MSE for the CG, MG and RGG with \( N = 99 \). For the CG, the reference node is assumed positioned at one of the two ends of the chain, while for the MG, the reference node is assumed to lie at one corner of the mesh grid. For RGGs, the nodes are assumed uniformly distributed on a disc of area 1, and a communication radius of \( \zeta = 0.25 \) is chosen in order to allocate a considerable number of neighbors, \( N_{\text{neg}} \), for each node, where \( N_{\text{neg}} = 0.25\pi N \).

The purpose of this plot is to compare the MSE for different graphs keeping the number of nodes \( N \) fixed. The plot clearly shows that the MSE decreases at the fastest rate when the underlying graph is random geometric. This comes from the better communication properties of the RGG, since the average number of neighbors is considerable and the information coming from the reference node can quickly disseminate across all the nodes in the network. Secondly, as expected, the number of iterations needed for getting a uniformly precise estimate in the CG is bigger in comparison with MG.

Fig. 6 shows a comparison of the MSE of the proposed algorithm with the one in [19] (dashed lines) for a MG by varying the parameter \( \lambda \) of the random delays distribution, while keeping \( N = 99 \). The algorithm in [19] was proposed for Gaussian noise. The plot shows the comparison when the random delays are exponentially distributed. It can be observed that our proposed algorithm yields lower MSE for all values of \( \lambda \). It can also be seen how the floor value of the MSE decreases as \( \lambda \) increases. The same observation holds for the CG and the
Finally, to assess the communication complexity of the proposed algorithm, Fig. 7 shows the total number of packets exchanged between nodes from the first iteration until the instant at which the algorithm stops running for all nodes in the network. It can be observed that the complexity of the algorithm is almost linear in terms of number of packets exchanged for all three topologies considered, thus making it well suited for implementation in a real WSN.

V. Conclusions

This paper studies the problem of network-wide clock synchronization in a WSN. Assuming that the variable portion of the link delays is exponentially distributed, an iterative factor graph-based distributed algorithm is proposed for network-wide clock synchronization. The update rules of the proposed algorithm are derived by message passing using the max-product algorithm. The MSE performance of the algorithm is studied for structured as well as random network topologies. It is observed that the random geometric graph (RGG) exhibits faster convergence rates compared to the chain graph (CG) and mesh grid (MG). In order to characterize the communication overhead of the algorithm, a study of the number of transmitted packets is also performed. It is observed that the number of packets exchanged increase linearly with the network size. Therefore, the algorithm is well suited for clock synchronization of a sensor network.

REFERENCES