A Proposed Algorithm for Data Measurements Synchronization in Wireless Sensor Networks

Sofia Fanarioti, Athanasios Tsipis, Konstantinos Giannakis, George Koufoudakis, Eleni Christopoulou, Konstantinos Oikonomou
Department of Informatics
Ionian University, Corfu, Greece
Email: {sofiafanar, atsipis, kgiann, gkoufoud, hristope, okon}@ionio.gr

Ioannis Stavrakakis
National and Kapodistrian University of Athens, Greece - Universidad Carlos III de Madrid and IMDEA Networks Institute, Spain
Email: ioannis@di.uoa.gr

***Abstract*—The proliferation of wireless sensor networks supports nowadays numerous areas of everyday life and activities. Each sensor node senses some data of interest (e.g., humidity, temperature) and the corresponding measurements need to be tagged with the reference time (usually sink node's time) they took place, in order to be further analyzed. Given that clocks deviate, time synchronization is a challenging problem in this network environment. In this paper, the focus is on synchronizing the particular measurements in a per hop basis as they are transmitted encapsulated in data packets towards the sink node, instead of synchronizing the node clocks, thus, inducing no extra overhead. In this direction, a synchronization algorithm is proposed here based on MAC layer time stamping. Analytical results regarding the average time deviation and the corresponding variance under the proposed algorithm show a dependency on the *residual time* (i.e., the time period a data packet remains with a node), the *distance* (i.e., the number of hops between the sink node and the sensor node that sensed the data initially) and the average *skew* deviation value. The effectiveness of the algorithm as well as the validation of the analytical results are demonstrated thoroughly using simulation results.**

***Index Terms*—Data Measurements Synchronization, Wireless Sensor Networks, Clock Deviation.**

I. Introduction

Wireless sensor networks have experienced a wide proliferation the last decade [1], [2] in numerous areas of everyday life and activities (e.g., monitoring pollution, precision agriculture etc.) mostly due to the deployment of low-cost devices that implement the sensor network. Each sensor node senses some data of interest (e.g., humidity, temperature) and the corresponding measurements need to be tagged with the actual time they took place, i.e., the *timestamp*, in order to be further analyzed (e.g., to exploit time and space dimensions). Since clocks deviate as time passes, e.g., [3], [4], the timestamp given by the node that sensed the data may not be a correct one compared to a central *reference clock*, thus prohibiting further processing of the obtained data measurements.

In such systems, the reference clock is usually the clock of the *sink* node, i.e., the particular node that collects all sensed data from all network nodes in a multi-hop manner. Most approaches in the literature deal with this *synchronization* problem by synchronizing all node clocks, e.g., [5], [6], the main idea being the exchange of messages among neighbor nodes e.g., [7], [8], or the use of GPS, e.g., [9]. More on related approaches that appear in the literature are included in Section II. These approaches introduce a certain overhead in the form of additional transmissions, while the latter ones increase the hardware cost due to the GPS requirement [9].

In this paper, instead of synchronizing the node clocks, the focus is on synchronizing the particular measurements in a per hop basis as they are transmitted encapsulated in data packets towards the sink. In particular, the *data measurements synchronization* approach followed here *updates* the timestamp of the obtained measurement in a per hop basis, thus no extra overhead is introduced.

The system model considered in this work assumes two types of clock deviation, i.e., the *simple skew model* and the *general clock model*, [10]. A data measurements synchronization algorithm is proposed here for synchronizing the corresponding measurements in a per hop basis based on MAC [11] layer time stamping as data packets are transmitted. In particular, each data packet has a *timestamp field* that corresponds to an *estimation* of the current node's clock for the particular measurement.

An analysis of the proposed algorithm is also presented here, focusing on *time deviation*, which is defined as the difference between the time stamp value when the data packet arrives at the sink and the reference clock value at the time the measurement was obtained. Analytic results about the average time deviation and the corresponding variance show a dependency on the *residual time* (i.e., the time period a data packet remains within a node), the *distance* (i.e., the number of hops between the sink node and the sensor node that sensed the data initially) and the average *skew* deviation value of the considered model. More specifically, it is shown that as distance increases, the average time deviation also increases. The same applies for the residual time, whereas for the average skew, the closer to one (i.e., optimal), the smaller the average time deviation.

The effectiveness of the algorithm as well as the validation of the analytical results are demonstrated thoroughly using simulation results. Various simulation scenarios have been considered regarding different topologies and the two mentioned types of deviation models (the simple skew model and the general clock model). It is shown that the analytical results

are in accordance with those obtained by the simulations.

In Section II past related work is included and the system model is described in Section III. The proposed algorithm is presented in Section IV and its analysis is included in Section V. The simulation results are presented in Section VI and the conclusions are drawn in Section VII. Various proofs are included in the appendices.

II. Past Related Work

Various clock synchronization protocols for wireless sensor networks have been proposed and a number of survey papers have classified these protocols according to various features and aspects. Sundararaman et al. [3] present a classification of clock synchronization protocols based on two kinds of features: synchronization issues and application-dependent features. The survey paper of Rhee et al. [4] reviews representative clock synchronization protocols used in wireless sensor networks and additionally describes several methods for estimating clock offset and skew. Specifically, the authors describe the maximum likelihood estimate of clock offset for both RBS (Reference Broadcasting Synchronization) [5] and TPSN (Timing-sync Protocol for Sensor Networks) [6] protocols, as well as novel methods which use nonparametric bootstrap and parametric bootstrap techniques and particle filtering techniques, in the case of TPSN.

Swain and Hansdah [12] present a survey paper, which considers not only features that reflect the structure of the networks and the global objectives that protocols try to achieve, but features that are associated with different phases of clock synchronization protocols in wireless sensor networks. In a different approach, Djenouri and Bagaa [13], as well as Sivrikaya and Yener [9] provide an insight on issues related to the implementation of synchronization protocols in wireless sensor networks and analyze real implementations of the synchronization protocols.

There are many clock synchronization protocols related to the work presented here, like the Delay Measurement Time Synchronization for wireless sensor networks (DMTS) [14], the Flooding Time Synchronization Protocol (FTSP) [15], the probabilistic clock synchronization service in sensor networks [16], the Time Diffusion Synchronization Protocol (TDP) [17] and the Asynchronous Diffusion protocol by [8]. More recent protocols include the Gradient Time Synchronization Protocol (GTSP) [18], the Average TimeSync (ATS) algorithm [19], the PulseSynch [20], the Round-Robin Timing Exchange (RRTE) protocol [7], the R^{4}Syn [21] and the 2LTSP [22].

III. System Definition

Each node in the network upon data generation, creates a data packet, encapsulates its local clock and forwards it over a pre-constructed path towards the sink node. It is assumed that the first node on this path is node 0, whereas the k^{th} node is the sink. The distance between node 0 and the sink node in terms of number of hops is k, as it can be seen in Fig. 1. It is assumed that all *packet delays* (denoted by τ) are equal for all nodes in the network and propagation delay is negligible.

Let node 0 generate a packet at time t_0 that will be forwarded to the 1$^{\text{st}}$ node towards the sink at time t_1, to the 2$^{\text{nd}}$ node at time t_2, to the $(k-1)^{\text{th}}$ at time t_{k-1} and finally to the k^{th} (i.e., the sink) at time t_k, as depicted in Fig. 1.

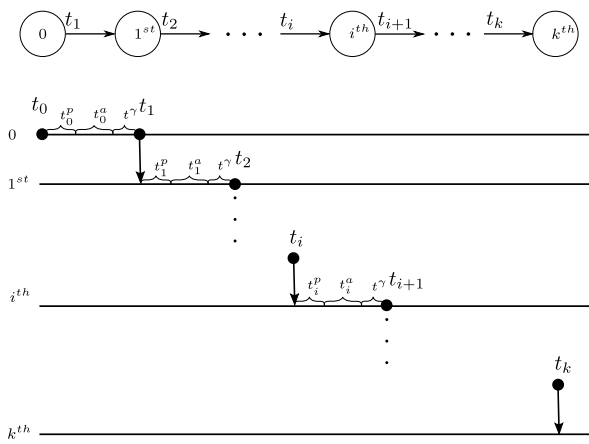

Fig. 1. Data packet path of k hops from node 0 towards the sink node (k^{th}).

Let t_i^p denote the time required for processing a packet at node i and t_i^a the time that node i has to wait until the wireless channel becomes free before transmitting towards node $i+1$. Let t^γ denote the time required to transmit the preamble bits and the Start Of Frame (SOF) field (depicted in Fig. 2). Therefore, $\tau = t_{i+1} - t_i = t_i^p + t_i^a + t^\gamma$ and $t_i = t_0 + \tau i$.

Let $c_i(t)$ denote the clock of node i at time t and let the sink node's clock be the *reference clock* (i.e., $c_k(t) = t$). As already mentioned, there are two basic models that capture the behavior (i.e., the deviation) of a node's clock, i.e., the simple skew model and the general clock model [10]. The general clock model is given by,

$$c_i(t) = \beta_i(t)t + \theta_i, \tag{1}$$

while the simple skew model is given by ,

$$c_i(t) = \beta_i t + \theta_i, \tag{2}$$

where θ_i is the node's i clock *offset* at time $t = 0$ and $\beta_i(t)$ or β_i – described next – will be referred to hereafter as the *clock skew*. As it is shown in Eq. (1) and Eq. (2), under the general clock model, clock skew (i.e., $\beta_i(t)$) changes in time for node i, while under the simple skew model it remains constant (i.e., β_i) for the particular node i. For both models, assuming that offset θ_i is known and that the clock skew is equal to one, this corresponds to the ideal case where all clocks are synchronized.

However, clock skew ($\beta_i(t)$ or β_i depending on the model) are not expected to be ideal and therefore, the purpose in the sequel is to propose and analyze an algorithm for synchronizing the data measurements instead of the nodes' clock. In order to keep the notation as simple as possible, the general clock model is considered in the sequel since it reduces to the simple one when clock skew does not change by time. The

proposed algorithm in the sequel as well as the subsequent analysis can be applied to both models.

IV. The Proposed Algorithm

Algorithm 1 presents the details of the per hop basis update of the timestamp field. After the transmission of the SOF field both nodes (i.e., node i and node $i+1$) read their local clock values i.e., $c_i(t_{i+1})$ and $c_{i+1}(t_{i+1})$, respectively (lines 7-9). Node i continues the transmission by giving values to the packet fields $c_i(t_{i+1})$, TS_i (timestamp) and Data, as depicted in Fig. 2.

Preamble	SOF	$c_i(t+1)$	TS_i	Data

Fig. 2. Data packet structure.

Upon successful transmission, node $i+1$ estimates the difference between its own clock and that of node i as $\Delta C_{i+1} = c_{i+1}(t_{i+1}) - c_i(t_{i+1})$ (line 14) and replaces the TS_i field by $TS_{i+1} = TS_i + \Delta C_{i+1}$ or, $TS_{i+1} = TS_i + c_{i+1}(t_{i+1}) - c_i(t_{i+1})$ (line 15). This process continues until the k^{th} node receives the packet. Similarly, the k^{th} node estimates the clock difference as $\Delta C_k = c_k(t_k) - c_{k-1}(t_k)$ and replaces the TS_{k-1} field by $TS_k = TS_{k-1} + c_k(t_k) - c_{k-1}(t_k)$.

Algorithm 1 The Proposed Data Synchronization Algorithm.

Var k: Sink node; i: Current node; `Data`: Local generated data; `Clock`: Local clock; $Preamble$: Preamble bits;
1: **Operate**:
2: **if** `Data` **then**
3: $TS :=$ `Clock`;
4: **if** $i \neq k$ **then** SendPacket(TS, `Data`); ▷ If not the sink, forward the data packet
5: **if** receive $< Preamble >$ **then** ReceivePacket;
6: **function** SendPacket(TS, `data`)
Var c; SOF: SOF field;
7: send $< Preamble, i+1 >$; send $< SOF, i+1 >$;
8: $c :=$ `Clock`;
9: send $< c, i+1 >$; send $< TS, i+1 >$; send $<$ `data`$, i+1 >$;
10: **function** ReceivePacket
Var c; c_R: Clock field; `data`: Data field; ΔC: Clock difference; SOF: SOF field; TS: Timestamp; `data`: Data field;
11: receive $< SOF >$;
12: $c :=$ `Clock`;
13: receive $< c_R >$; receive $< TS >$; receive $<$ `data` $>$;
14: $\Delta C := c - c_R$;
15: $TS := TS + \Delta C$;
16: **if** $i \neq k$ **then** SendPacket(TS, `data`); ▷ If not the sink, forward the data packet

Algorithm 1 is capable of estimating the clock difference between two nodes assuming no or negligible propagation delay. Each node that receives a packet, increases the TS field by the clock difference between its clock and the sender. Thus, when a packet has been received by the k^{th} node, the value of the TS field is equal to $TS_k = TS_{k-1} + \Delta C_k$. As it is proved in Appendix A,

$$TS_k = t_k - \sum_{i=0}^{k-1} \left[c_i(t_{i+1}) - c_i(t_i) \right]. \tag{3}$$

The timestamp when the packet has arrived at the sink node is derived by Eq. (3) and it corresponds to the sink node's estimation of the time of the data generation at node 0 (i.e., t_0).

V. Analysis

The focus of the analysis is on *time deviation* at the sink node (the k^{th}). Let ϵ_k denote the time deviation of the timestamp at the sink node. Then, $\epsilon_k = TS_k - t_0$ and by replacing TS_k from Eq. (3) it follows that, $\epsilon_{k,0} = t_k - t_0 - \sum_{i=0}^{k-1} \left[c_i(t_{i+1}) - c_i(t_i) \right]$. As it is proved in Appendix B,

$$\epsilon_{k,0} = \tau k - \tau \sum_{i=0}^{k-1} \left[(i+1)\beta_i((i+1)\tau) - i\beta_i(i\tau) \right]. \tag{4}$$

The next step is to study the accuracy of Algorithm 1 by studying the expected value and the variance of time deviation, i.e., $\mathbb{E}\left[\epsilon_{k,0}\right]$ and $\sigma^2\left[\epsilon_{k,0}\right]$, respectively. Let the distribution of the skew function β_i remain the same across all nodes for the same time step t, and for all time steps t for each node. Let μ_β and σ_β^2 denote the corresponding mean and variance.

Considering Eq. (4) and the linearity property of the expected value [23], it follows that $\mathbb{E}\left[\epsilon_{k,0}\right] = \tau k - \tau \sum_{i=0}^{k-1} \left[(i+1)\mathbb{E}\left[\beta_i((i+1)\tau)\right] - i\mathbb{E}\left[\beta_i(i\tau)\right] \right]$. Given the previous assumption with respect to the distribution of $\beta_i(t)$, it follows that $\mathbb{E}\left[\epsilon_{k,0}\right] = \tau k - \tau \sum_{i=0}^{k-1} \left[(i+1)\mu_\beta - i\mu_\beta \right]$ or, $\mathbb{E}\left[\epsilon_{k,0}\right] = \tau k - \tau \sum_{i=0}^{k-1} \mu_\beta$ or, $\mathbb{E}\left[\epsilon_{k,0}\right] = \tau k - \tau k \mu_\beta$ or,

$$\mathbb{E}\left[\epsilon_{k,0}\right] = \tau k \left(1 - \mu_\beta\right). \tag{5}$$

Considering Eq. (4), the variance of time deviation is, $\sigma^2\left[\epsilon_{k,0}\right] = \sigma^2\left[\tau k - \tau \sum_{i=0}^{k-1} \left[(i+1)\beta_i((i+1)\tau) - i\beta_i(i\tau) \right]\right]$. As it is proved in Appendix C,

$$\sigma^2\left[\epsilon_{k,0}\right] = k\tau^2\sigma_\beta^2. \tag{6}$$

Both Eq. (5) and Eq. (6) help evaluate the performance of the proposed Algorithm 1 with respect to the time deviation. As it is concluded from Eq. (5), the mean value of time deviation $\mathbb{E}\left[\epsilon_{s,0}\right]$ increases linearly with packet delay τ and the number of hops k between the particular node and the sink node. The mean value of the skew function $\beta_i(t)$ also plays an important role. When $\mu_\beta \to 1$, this allows for $\mathbb{E}\left[\epsilon_{k,0}\right] \to 0$. However, $\mu_\beta \to 1$ means that $\beta_i(t) \to 1$ and eventually, the offset by the hardware of each node should be negligible (i.e., close to zero). Regarding the variance, as given by Eq. (6), it is affected by the number of hops k, the packet delay τ and variance σ_β. As before, when $\beta_i(t) \to 1$, it is expected that $\sigma_\beta \to 0$ and eventually, $\sigma^2\left[\epsilon_{k,0}\right] \to 0$.

VI. Simulation Results

A simulation program in C++ has been developed in order to generate geometric random graph topologies with 1000 uniformly and independently distributed nodes in $[0,1]\times[0,1]$ plane and a link between any pair exists if their euclidean distance is equal to or smaller than the connectivity radius r_c. The considered topologies correspond to $r_c = 0.06, 0.1, 0.2$ and are considered as a simple model suitable for representing wireless sensor network topologies. Wireless transmissions take place over an error-free channel with a bit rate of 100 kbps and an underlying nonpersistent CSMA protocol [11]. Each simulation scenario lasts 1000 seconds and the presented simulation results correspond to the average of ten independent runs for both skew models.

For both models, the skew values ($\beta_i(t)$ and β_i, respectively) are uniformly distributed within range $[0.990,\ldots,1.000]$. For the particular case that the skew function $\beta_i(t)$ is uniformly distributed with minimum value $\beta_{\min}$ and maximum value $\beta_{\max}$, the mean value and the variance of $\beta_i(t)$ are $\mu_\beta = \frac{\beta_{\min}+\beta_{\max}}{2}$ and $\sigma^2_\beta = \frac{(\beta_{\max}-\beta_{\min})^2}{12}$, respectively. Thus, Eq. (5) can be written as,

$$\mathbb{E}_U\left[\epsilon_{k,0}\right] = \tau k\left(1-\frac{\beta_{\min}+\beta_{\max}}{2}\right). \tag{7}$$

Fig. 3 depicts average values of time deviation (in ms) under the proposed Algorithm 1 as a function of the number of hops for the simple skew model for three different topologies. It is obvious that as the number of hops k increases, the average time deviation increases linearly. It is interesting to see that the analytical results, corresponding to Eq. (7) and depicted as a dotted line, capture the behavior of the system. For each depicted scenario (a, b and c), three different cases with respect to packet delay τ have been considered, i.e., for (a) $\tau = 8.32$ ms, 12.96 ms and 15.81 ms; for (b) $\tau = 8.33$ ms, 12.32 ms and 16.11 ms; and for (c) $\tau = 8.32$ ms, 13.98 ms and 17.99 ms. As observed, as τ increases, the average time deviation of the proposed algorithm increases. Still, the analytical results capture the system's behavior.

Fig. 4 shows average values of time deviation (in ms) under the proposed Algorithm 1 as a function of the number of hops for the clock model for three different topologies. As already shown for the case of the simple skew model, the average time deviation increases linearly with the number of hops k. For each depicted scenario (a, b and c), three different cases with respect to packet delay τ have been considered, i.e., for (a) $\tau = 9.04$ ms, 12.61 ms and 16.24 ms; for (b) $\tau = 8.72$ ms, 12.29 ms and 16.11 ms; and for (c) $\tau = 9.12$ ms, 13.09 ms and 16.85 ms. As observed, as τ increases, the average time deviation of the proposed algorithm increases. Still, the analytical results capture the system's behavior.

VII. Conclusions

The problem of synchronization in wireless sensor networks is addressed in this paper under a different perspective. In particular, the main idea presented here in the form of a synchronization algorithm was to synchronize data measurements as they are transmitted towards the sink node in a per hop basis instead of synchronizing the node clocks, thus inducing no extra overhead. The proposed algorithm was analyzed and simulation results demonstrated the algorithm's effectiveness in the considered environment. Furthermore, it was shown that the analytical results are in accordance with the simulations.

Acknowledgments

This work was supported in part by project "A Pilot Wireless Sensor Networks System for Synchronized Monitoring of Climate and Soil Parameters in Olive Groves," (MIS 5007309) which is partially funded by European andNational Greek Funds (ESPA) under the Regional Operational Programme "Ionian Islands 2014-2020." In addition, this work was supported in part by the European Commission as part of the ReCRED project (Horizon H2020 Framework Programme of the European Union under GA number 653417), by the Chair of Excellence UC3M - Santander Program and by the National and Kapodistrian University of Athens (S.A.R.G.).

Appendix A
Derivation of Eq. (3)

The update is $TS_k = TS_{k-1} + \Delta C_k$ or, $TS_k = TS_{k-1} + [c_k(t_k) - c_{k-1}(t_k)]$ or, $TS_k = TS_{k-2} + [c_{k-1}(t_{k-1}) - c_{k-2}(t_{k-1})] + [c_k(t_k) - c_{k-1}(t_k)]$ or, $TS_k = TS_0 + [c_1(t_1) - c_0(t_1)] + [c_2(t_2) - c_1(t_2)]\ldots + [c_{k-1}(t_{k-1}) - c_{k-2}(t_{k-1})] + [c_k(t_k) - c_{k-1}(t_k)]$ or, $TS_k = TS_0 - c_0(t_1) - [c_1(t_2) - c_1(t_1)] - [c_2(t_3) - c_2(t_2)]\ldots - [c_{k-1}(k_s) - c_{k-1}(t_{k-1})] + c_k(t_k)$. Finally, $TS_k = c_k(t_k) - \sum_{i=1}^{k-1}[c_i(t_{i+1}) - c_i(t_i)] - [c_0(t_1) - TS_0]$. Note that TS_0 is node's 0 timestamp of the generated data at time $t = t_0$ (i.e., $TS_0 = c_0(t_0)$) and also k^{th} node's clock is the reference clock (i.e., $c_k(t) = t$ since the k^{th} node is the sink). Thus, the previous expression can be written as, $TS_k = t_k - \sum_{i=1}^{k-1}[c_i(t_{i+1}) - c_i(t_i)] - [c_0(t_1) - c_0(t_0)]$ or, $TS_k = t_k - \sum_{i=0}^{k-1}[c_i(t_{i+1}) - c_i(t_i)]$.

Appendix B
Derivation of Eq. (4)

Given, $\tau = t_{i+1} - t_i = t_i^p + t_i^a + t^\gamma$ and $t_i = t_0 + i\tau$, the time deviation of TS_k can be written as, $\epsilon_{k,0} = \tau k - \sum_{i=0}^{k-1}[c_i(t_0 + (i+1)\tau) - c_i(t_0 + i\tau)]$ and by replacing $c_i(t)$ from Eq. (1) it follows that, $\epsilon_{k,0} = \tau k - \sum_{i=0}^{k-1}[(t_0 + (i+1)\tau)\beta_i(t_0 + (i+1)\tau) + \theta_{i,0} - (t_0 + i\tau)\beta_i(t_0 + i\tau) - \theta_{i,0}]$ or, $\epsilon_{k,0} = \tau k - \sum_{i=0}^{k-1}[(t_0 + (i+1)\tau)\beta_i(t_0 + (i+1)\tau) - (t_0 + i\tau)\beta_i(t_0 + i\tau)]$. Without loss of generality it is assumed that $t_0 = 0$ thus, $\epsilon_{k,0} = \tau k - \tau\sum_{i=0}^{k-1}[(i+1)\beta_i((i+1)\tau) - i\beta_i(i\tau)]$.

Appendix C
Derivation of Eq. (6)

Considering Eq. (4), the variance of the time deviation is, $\sigma^2[\epsilon_{k,0}] = \sigma^2\left[\tau k - \tau\sum_{i=0}^{k-1}[(i+1)\beta_i((i+1)\tau) - i\beta_i(i\tau)]\right]$, or, $\sigma^2[\epsilon_{k,0}] = \tau^2\sigma^2\left[\sum_{i=0}^{k-1}[(i+1)\beta_i((i+1)\tau) - i\beta_i(i\tau)]\right]$.

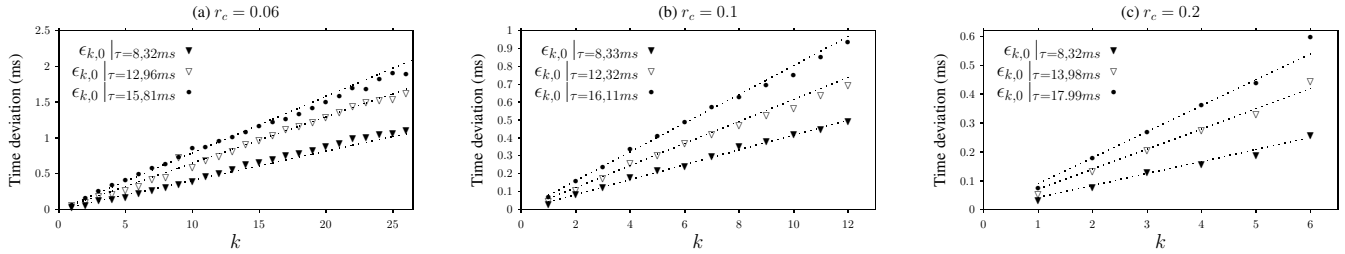

Fig. 3. Simulation results regarding the average time deviation as a function of the number of hops for $r_c = 0.06, 0.1$ and 0.2 for the simple skew model. The dotted lines correspond to the analytical results.

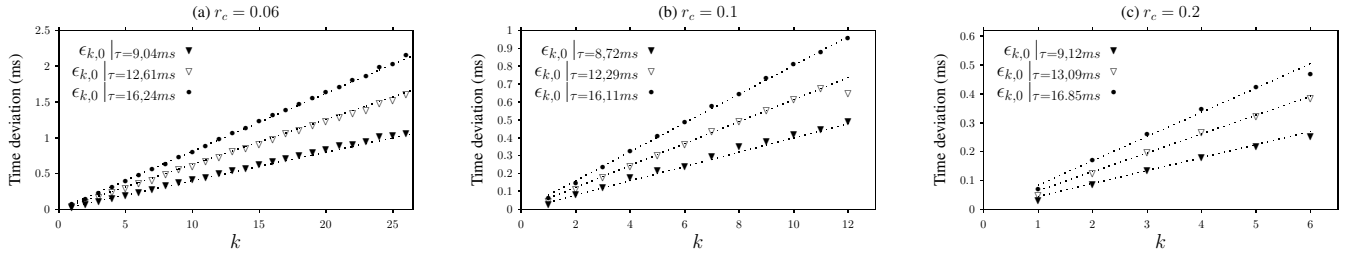

Fig. 4. Simulation results regarding the average time deviation as a function of the number of hops for $r_c = 0.06, 0.1$ and 0.2 for the general clock model. The dotted lines correspond to the analytical results.

Since $\beta_i(t)$ and $\beta_{j\neq i}(t)$ are uncorrelated it follows that their covariance equals to zero (i.e., $\rho\left(\beta_i(t), \beta_{j\neq i}(t)\right) = 0$), where $\rho(\cdot)$ is the covariance function. Also $\rho\left(\beta_i(t_m), \beta_i(t_{n\neq m})\right) = \sigma_\beta^2$ thus, $\sigma^2\left[\epsilon_{k,0}\right] = \tau^2 \sum_{i=0}^{k-1} \sigma^2\left[(i+1)\beta_i((i+1)\tau) - i\beta_i(i\tau)\right]$ or, $\sigma^2\left[\epsilon_{k,0}\right] = \tau^2 \sum_{i=0}^{k-1} \left[(i+1)^2\sigma^2\left[\beta_i((i+1)\tau)\right] + i^2\sigma^2\left[\beta_i(i\tau)\right] - -2(i+1)i\rho\left(\beta_i((i+1)\tau), \beta_i(i\tau)\right)\right]$ or, $\sigma^2\left[\epsilon_{k,0}\right] = \tau^2 \sum_{i=0}^{k-1} \left[(i+1)^2\sigma_\beta^2 + i^2\sigma_\beta^2 - 2(i+1)i\sigma_\beta^2\right]$ or, $\sigma^2\left[\epsilon_{k,0}\right] = \tau^2 \sum_{i=0}^{k-1} \left[(i+1-i)^2\sigma_\beta^2\right]$ or, $\sigma^2\left[\epsilon_{k,0}\right] = k\tau^2\sigma_\beta^2$.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer networks*, vol. 38, no. 4, pp. 393–422, 2002.

[2] I. Akyildiz, T. Melodia, and K. Chowdhury, "A survey on wireless multimedia sensor networks," *Computer networks*, vol. 51, no. 4, pp. 921–960, 2007.

[3] B. Sundararaman, U. Buy, and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: A survey," *Ad Hoc Networks (Elsevier*, vol. 3, pp. 281–323, 2005.

[4] I.-K. Rhee, J. Lee, J. Kim, E. Serpedin, and Y.-C. Wu, "Clock synchronization in wireless sensor networks: An overview," *Sensors*, vol. 9, no. 1, pp. 56–85, 2009.

[5] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *ACM SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 147–163, 2002.

[6] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Timing-sync protocol for sensor networks," in *Proceedings of the 1st international conference on Embedded networked sensor systems*. ACM, 2003, pp. 138–149.

[7] Y.-H. Huang and S.-H. Wu, "Time synchronization protocol for small-scale wireless sensor networks," in *Wireless Communications and Networking Conference (WCNC), 2010 IEEE*. IEEE, 2010, pp. 1–5.

[8] Q. Li and D. Rus, "Global clock synchronization in sensor networks," *IEEE Transactions on computers*, vol. 55, no. 2, pp. 214–226, 2006.

[10] D. Veitch, S. Babu, and A. Pàsztor, "Robust synchronization of software clocks across the internet," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. ACM, 2004, pp. 219–232.

[9] F. Sivrikaya and B. Yener, "Time synchronization in sensor networks: a survey," *IEEE network*, vol. 18, no. 4, pp. 45–50, 2004.

[11] L. Kleinrock and F. Tobagi, "Packet switching in radio channels: Part I–Carrier sense multiple-access modes and their throughput-delay characteristics," *IEEE transactions on Communications*, vol. 23, no. 12, pp. 1400–1416, 1975.

[12] A. R. Swain and R. Hansdah, "A model for the classification and survey of clock synchronization protocols in WSNs," *Ad Hoc Netw.*, vol. 27, no. C, pp. 219–241, Apr. 2015.

[13] D. Djenouri and M. Bagaa, "Synchronization protocols and implementation issues in wireless sensor networks: A review," *IEEE Systems Journal*, vol. 10, pp. 617–627, 2016.

[14] S. Ping, "Delay measurement time synchronization for wireless sensor networks," *Intel Research Berkeley Lab*, vol. 6, pp. 1–10, 2003.

[15] M. Maróti, B. Kusy, G. Simon, and Á. Lédeczi, "The flooding time synchronization protocol," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*. ACM, 2004, pp. 39–49.

[16] S. PalChaudhuri, A. Saha, and D. B. Johnson, "Probabilistic clock synchronization service in sensor networks," *IEEE Transactions on Networking*, vol. 2, no. 2, pp. 177–189, 2003.

[17] I. S. Akyildiz and I. Su, "Time-diffusion synchronization protocols for sensor networks," *IEEE/ACM Transactions on Networking*, pp. 1626–1645, 2005.

[18] P. Sommer and R. Wattenhofer, "Gradient clock synchronization in wireless sensor networks," *2009 International Conference on Information Processing in Sensor Networks*, pp. 37–48, 2009.

[19] L. Schenato and F. Fiorentin, "Average timesynch: A consensus-based protocol for clock synchronization in wireless sensor networks," *Automatica*, vol. 47, no. 9, pp. 1878–1886, 2011.

[20] C. Lenzen, P. Sommer, and R. Wattenhofer, "Optimal clock synchronization in networks," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2009, pp. 225–238.

[21] D. Djenouri, "R^4Syn: Relative referenceless receiver/receiver time synchronization in wireless sensor networks," *IEEE Signal Process. Lett.*, vol. 19, pp. 175–178, 2012.

[22] G. Huang, A. Y. Zomaya, F. C. Delicato, and P. F. Pires, "Long term and large scale time synchronization in wireless sensor networks," *Computer Communications*, vol. 37, pp. 77–91, 2014.

[23] Papoulis, A., *Probability, Random Variables, and Stochastic Processes*, 3rd ed. Mcgraw-Hill College, 1991.