Fast Converging Distributed Pulse-coupled Clock Synchronization for Half-duplex D2D Communications over Multipath Channels

Onur Karatalay*, Ioannis Psaromiligkos*, Benoit Champagne* and Benoit Pelletier†
*Department of Electrical and Computer Engineering, McGill University, Montréal, PQ, Canada.
†InterDigital Canada Ltée, Montréal, PQ, Canada.
Email: onur.karatalay@mail.mcgill.ca; ioannis.psaromiligkos@mcgill.ca;
benoit.champagne@mcgill.ca; benoit.pelletier@interdigital.com

***Abstract*—Clock synchronization is a fundamental problem in distributed device-to-device (D2D) communications as envisioned in fifth g eneration (5G) w ireless n etworks. T o achieve synchronization, especially in out-of-coverage scenarios where no common reference point is available, devices can use distributed phase-locked loops (DPLLs). Although DPLLs provide accurate synchronization in ideal conditions, i.e., additive noise channel without propagation delays, dispersive multipath channels may significantly d egrade t he s ynchronization performance. The choice of duplexing mode (full-duplex or half-duplex) also affects greatly the performance of a distributed synchronization scheme. Since full-duplexing is not yet a practical technology at the device side, in this work we consider distributed synchronization under the realistic assumption of half-duplex communication over multipath channels. We first present a new synchronization algorithm that allows devices to self-determine their transceiver mode, and then, we introduce a modified D PLL a lgorithm b ased o n i terative propagation delay estimation to improve synchronization performance. Numerical results show that when using the proposed algorithms, devices can achieve a steady-state timing error on the order of 1μs while allowing them to arbitrarily join or leave the synchronization process.**

I. Introduction

For reliable data transmissions, all entities in a communication network must become synchronized and remain in this state as long as they operate. In conventional wireless cellular networks, base stations (BSs) or access points broadcast a synchronization beacon to all devices in the network; hence, the devices can synchronize their clock to a common reference time. However, in distributed systems such as device-to-device (D2D) networks, where communication links are peer-to-peer, no common reference time is available to synchronize the devices [1]. Thus, in out-of-coverage scenarios where the devices cannot connect to BSs, as envisaged for the fifth g eneration (5G) cellular networks [2], devices should use distributed synchronization algorithms when D2D connections are established [3].

A detailed overview of distributed synchronization in wireless networks is presented in [6], where synchronization techniques are divided into two different classes, namely: packet-based and pulse-based (or pulse-coupled). Comparing the two, pulse-coupled synchronization is preferable at the initialization stage of D2D networks, especially for out-of-coverage scenarios. Indeed, packet demodulation is not possible during initial network deployment due to asynchronous clocks, whereas pulse-based synchronization at the physical (PHY) level can achieve frame synchronization with high precision while offering scalability [7].

Distributed phase-locked loops (DPLLs) [6], the most popular implementation of pulse-coupled synchronization, offers a promising solution for distributed networks, but it remains sensitive to propagation delays. In the literature, several studies on pulse-coupled distributed clock synchronization have focused on only minimizing clock errors due to random clock initializations (i.e., clock phase difference) or physical variations in the crystal oscillator (i.e., clock skew), which result in timing offset (TO) and carrier frequency offset (CFO). For example in [4], joint time and frequency synchronization for distributed networks is proposed, however, propagation delays are not considered. The convergence of clock skew and clock phase offset is investigated in [5] but environmental factors, such as radio propagation delays and changes in the number of connected devices, are not taken into consideration. It is known that multipath propagation creates ambiguity in the timing measurements that also severely degrades TO [8]. Consequently, mismatches in frame timing and collisions in data packets may occur which result in throughput reduction [10]. Thus, a clock synchronization algorithm must consider both clock imperfections and environmental factors to ensure reliable operation, i.e., that all devices can join the network arbitrarily and remain synchronized over time.

The duplexing mode (i.e., full-duplex versus half-duplex) is another important consideration in the design of a clock synchronization scheme, especially in multipath environments. In many works, e.g., [4], [8], clock synchronization is investigated under the assumption of full-duplex communications; however, full-duplex is not yet a mature technology especially at the device side. In [11], [12], half-duplex communication is considered by constantly and randomly changing the transceiver mode of each device. Although these approaches ensure synchronization in the absence of propagation delays, we have observed that randomly changing the transceiver mode during synchronization introduces random perturbations to the local clocks especially in the case of propagation delays.

This work was supported in part by the Natural Science and Engineering Research Councli under the Discovery Grant program and by InterDigital Canada Ltée and MITACS Canada under the MITACS Accelerate program.

978-1-5386-7568-7/18/$31.00 ©2018 IEEE

In this paper we focus on pulse-coupled synchronization for D2D communications over multipath channels. First, we propose a simple yet effective half-duplex synchronization algorithm which enables devices to independently determine their transceiver mode in an orderly manner. Second, we introduce a modified DPLL clock update rule based on iterative propagation delay estimation to mitigate synchronization errors. When combined, these two approaches enable accurate and reliable synchronization under realistic multipath propagation conditions, while allowing scalability and robustness to perturbations in the network as demonstrated by simulation-based studies.

II. System Model and Problem Statement

A. Network Setup and Clock Model

We consider out-of-coverage clock synchronization in a D2D network comprising K wireless devices indexed by $i \in \{1, ..., K\}$, where devices may join or leave the network at any time. Following [6], the physical clock time of the i^{th} device can be modeled as $t_i(t) = \alpha_i t + \theta_i$, where t is the universal time, α_i is the clock skew and $\theta_i \in [0, T_0)$ is the clock phase. In this work, we assume for simplicity that $\alpha_i = 1$, although generalizations are possible. A discrete logical clock is obtained by uniformly sampling the physical clock at times $t = \nu T_0$, i.e., $t_i[\nu] = \nu T_0 + \theta_i$, where T_0 is the clock period and $\nu \in \mathbb{Z}$ is the discrete-time index. We refer to time $t_i[\nu]$ as the ν^{th} *clock tick* of the i^{th} device, where the clock tick differences between the devices create a TO between the i^{th} and the j^{th} device, $\Delta t_{ij}[\nu] = t_i[\nu] - t_j[\nu]$. Furthermore, it is convenient to partition the universal time axis into a sequence of non-overlapping time slots $[\nu T_0, (\nu+1)T_0)$, so that each time slot contains a unique clock tick $t_i[\nu]$ from each device.

B. Signal Model

We consider a half-duplex communication scenario, in which at each clock tick, a device is either transmitting or receiving as illustrated in Fig. 1. We denote by $\mathcal{T}_\nu$ and $\mathcal{R}_\nu$ the sets containing the indices of the transmitting and receiving devices at the ν^{th} clock tick, respectively.

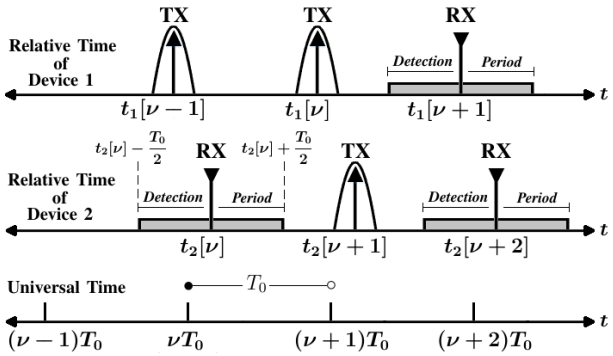


Fig. 1. Illustration of signaling for distributed pulse-coupled half-duplex synchronization. The figure shows the sequence of clock ticks of two devices relative to the universal time t. Here, during the ν^{th} time slot, device 1 acts as a transmitter while device 2 is a receiver, where they are denoted by TX and RX, respectively.

At the ν^{th} tick of its own clock, a device $i \in \mathcal{T}_\nu$ broadcasts a synchronization signal $x(t - t_i[\nu])$, where

$$x(t) = \sum_{n=0}^{N_s-1} s[n] g(t - nT_p). \tag{1}$$

In (1), $s[n] \in \mathbb{C}$ is a synchronization sequence of length N_s, $g(t) \in \mathbb{R}$ is a unit-energy baseband pulse, and T_p is the pulse spacing assumed to satisfy $N_s T_p \ll T_0$. A common approach (that we also follow in this work) is to construct the synchronization signal using Zadoff-Chu (ZC) sequences [13]. As it was shown in [4], in the presence of TO and CFO it is beneficial to form $s[n]$ by concatenating two ZC sequences of length N having a root index u of the opposite sign. That is:

$$s[n] = \begin{cases} e^{-\mathrm{j}\frac{\pi}{N}u(n-N)^2}, & 0 \le n \le N-1 \\ e^{\mathrm{j}\frac{\pi}{N}un^2}, & N \le n \le 2N-1 \end{cases} \tag{2}$$

in which case, $N_s = 2N$ in (1) and $\mathrm{j} = \sqrt{-1}$. This $s[n]$ yields a synchronization signal $x(t)$ whose discrete-time equivalent (obtained by sampling at a period T_s) has two parts: $x_-[k]$ corresponding to the ZC sequence with the root index $-u$, followed by $x_+[k]$ corresponding to the ZC sequence with the root index $+u$. Akin to [4], we set $u = 1$ for all devices.

Prior to transmission, the synchronization signal from the i^{th} device is upconverted to passband using a carrier frequency f_i, yielding $\widetilde{x}_i(t, \nu) = x(t - t_i[\nu]) e^{\mathrm{j}2\pi f_i t}$. In practice, the carrier frequency used by different devices may be slightly different, which leads to CFO as discussed later.

A device $j \in \mathcal{R}_\nu$ listens for the broadcasted synchronization signals for one clock period, $[t_j[\nu] - T_0/2, t_j[\nu] + T_0/2)$, centered at $t_j[\nu]$, called the detection period (see Fig. 1). The received passband signal is given by

$$\widetilde{y}_j(t) = \sum_{\eta \in \{\nu, \nu \pm 1\}} \sum_{i \in \mathcal{T}_\eta} \widetilde{x}_i(t, \eta) * h_{ij}(t) + w_j(t) \tag{3}$$

where $h_{ij}(t)$ denotes the impulse response of the wireless channel between the i^{th} and j^{th} device, $*$ denotes convolution and $w_j(t)$ is an additive noise term. A multipath model is assumed between any two devices, that is:

$$h_{ij}(t) = \sum_{p=1}^{P} \rho_{ijp} \delta(t - \tau_{ijp}) \tag{4}$$

where p is the path index, P is the number of resolvable paths (assumed to be the same for all pairs (i, j), for simplicity), $\rho_{ijp} \in \mathbb{C}$ and $\tau_{ijp} \in \mathbb{R}_+$ are the complex gain and propagation delay of the p^{th} path, respectively, and $\delta(\cdot)$ the Dirac delta function. We note that the received signal $\widetilde{y}_j(t)$ in (3) may contain contributions from synchronization signals transmitted not only during the ν^{th} time slot but also during the $(\nu-1)^{th}$ and $(\nu+1)^{th}$ slots. This is because the detection period may partially overlap with these intervals as its starting position depends on the clock phase θ_j within the ν^{th} time slot. However, signal contributions from the $(\nu+2)^{th}$ time slot and beyond cannot be detected

as the detection period even with the maximum θ_j cannot reach further than one time slot. In addition, considering the channel delay spread and propagation delays, which are assumed to be much smaller than T_0, the detection period cannot have signal contributions from the $(\nu-2)^{th}$ time slot and earlier either. Hence, the outer summation in (3) needs to account for only those contributing intervals, i.e., $\eta \in \{\nu-1, \nu, \nu+1\}$.

After downconversion with the local carrier frequency f_j, (3) is sampled at time instances kT_s with T_s being the sampling period. The resulting baseband discrete-time signal, which is truncated over the time interval $[t_j[\nu] - \frac{T_0}{2}, t_j[\nu] + \frac{T_0}{2})$, is

$$\begin{aligned} y_j[k;\nu] =& y_j(kT_s + t_j[\nu]) \\ =& \sum_{\eta\in\{\nu,\nu\pm1\}} \sum_{i\in\mathcal{T}_\eta} \sum_{p=1}^{P} \rho_{ijp} x(kT_s + t_j[\nu] - t_i[\eta] - \tau_{ijp}) \\ & \times e^{\mathrm{j}2\pi\Delta f_{ij} k T_s} + w_j[k] \end{aligned} \tag{5}$$

where $\Delta f_{ij} = f_i - f_j$ represents the CFO between the i^{th} and j^{th} device, and $w_j[k]$ is the discrete-time noise process. We note that the factor $e^{-\mathrm{j}2\pi f_i \tau_{ijp}}$ has been absorbed by ρ_{ijp} in (5) with no loss of generality.

C. DPLL Clock Update

To estimate the TO, the j^{th} receiver device first cross-correlates $y_j[k;\nu]$ with the synchronization signals $x_-[k]$ and $x_+[k]$ as follows:

$$R_{y_j x_\mp}[l;\nu] = \sum_{k\in\mathbb{Z}} y_j[k+l;\nu] x^*_\mp[k] \tag{6}$$

from which, two weighted averages across the lags l are subsequently obtained [8]:

$$q_j^\mp[\nu] \triangleq \frac{\sum_l l |R_{y_j x_\mp}[l;\nu]|^2}{\sum_l |R_{y_j x_\mp}[l;\nu]|^2}. \tag{7}$$

In (7), $q_j^-[\nu]$ yields the weighted average TO in terms of samples, that is, if $d_j[\nu]$ is the weighted average TO in continuous-time as seen from the j^{th} device, then $q_j^-[\nu] = f_s d_j[\nu]$, where $f_s = \frac{1}{T_s}$ is the sampling frequency. On the other hand, $q_j^+[\nu]$ is equal to the sum of the weighted average TO and the duration of $x_-[k]$ (both expressed in terms of samples). That is, $q_j^+[\nu] = f_s(d_j[\nu] + NT_p)$ due to the concatenation in (2). Thus, the continuous-time equivalent of the weighted average TO between the contributing transmitters and the j^{th} device is obtained from (7) as follows:

$$\Delta t_j[\nu] = \frac{\left(q_j^-[\nu] + q_j^+[\nu]\right) T_s - NT_p}{2}. \tag{8}$$

As shown in [4], the effect of CFO on the estimation of the average TO is canceled by using (8).

Finally, the j^{th} receiver device updates its clock as dictated by DPLL [8]:

$$t_j[\nu+1] = t_j[\nu] + T_0 + \epsilon \Delta t_j[\nu],\ \forall j \in \mathcal{R}_\nu. \tag{9}$$

where $\epsilon \in (0, 1]$ is a scaling term. The devices that are operating as transmitters only advance their clock by T_0 as follows:

$$t_i[\nu+1] = t_i[\nu] + T_0,\ \forall i \in \mathcal{T}_\nu. \tag{10}$$

D. Problem Statement

From the signal model given in (5), $\Delta t_j[\nu]$ in (8) has the form of a weighted average TO:

$$\begin{aligned} \Delta t_j[\nu] \approx& \sum_{(\eta,i,p)\in\mathcal{D}_j^\nu} \mu_{ijp}(t_i[\eta] - t_j[\nu] + \tau_{ijp}) \\ =& \underbrace{\sum_{(\eta,i,p)\in\mathcal{D}_j^\nu} \mu_{ijp}\big((\eta-\nu)T_0 + \theta_i - \theta_j\big)}_{\overline{\Delta t_j}[\nu]} + \underbrace{\sum_{(\eta,i,p)\in\mathcal{D}_j^\nu} \mu_{ijp}\tau_{ijp}}_{\overline{\tau_j}[\nu]} \end{aligned} \tag{11}$$

where $\mathcal{D}_j^\nu = \left\{(\eta,i,p) \in \bigcup_{\eta=\nu-1}^{\nu+1}\{\eta\} \times \mathcal{T}_\eta \times \mathbb{N} : |t_i[\eta] - t_j[\nu] + \tau_{ijp}| \le \frac{T_0}{2}\right\}$ is the set of transmitter devices contributing to the signal received during the detection period centered at the ν^{th} clock tick of the j^{th} receiving device. Furthermore, $\mu_{ijp} = \frac{|\rho_{ijp}|}{\sum_{(\eta,i,p)\in\mathcal{D}_j^\nu} |\rho_{ijp}|}$ are weights given by the normalized channel gain between the i^{th} transmitter and the j^{th} receiver device. Finally, $\overline{\Delta t_j}[\nu]$ and $\overline{\tau_j}[\nu]$ are the weighted average of relative clock differences with respect to the j^{th} device and the bias due to the propagation delays, respectively. The latter is time-varying due to new devices joining the network as well as the half-duplex communication scheme, and non-zero unless $\mathcal{D}_j^\nu = \emptyset$.

Hence, the DPLL update rule in (9) for the j^{th} receiver device becomes:

$$t_j[\nu+1] = t_j[\nu] + T_0 + \epsilon\big(\overline{\Delta t_j}[\nu] + \overline{\tau_j}[\nu]\big),\ \forall j \in \mathcal{R}_\nu. \tag{12}$$

To achieve global synchronization within the network, we would like to eliminate the bias $\overline{\tau_j}[\nu]$. However, this is equivalent to either removing the propagation delays which is impossible, or having $\mathcal{D}_j^\nu = \emptyset, \forall \nu$, which simply means that there are no synchronization signals transmitted. Thus, the objective of this work, is to minimize the effect of the bias, in the presence of multipath transmissions and a time-varying set $\mathcal{D}_j^\nu$ that is, in general, non-empty.

III. Proposed Method

In this section, we propose a new half-duplex synchronization algorithm which enables devices to determine their transceiver mode in a fully distributed but structured manner leading to a controlled evolution of $\mathcal{D}_j^\nu$ over time. Then, we capitalize on the latter to introduce a new propagation delay compensation algorithm to reduce the bias, and hence, improve the synchronization performance.

A. Alternating Transceiver Mode Algorithm

In a distributed network with half-duplex communications, the devices should choose their transceiver mode at each clock tick individually and without any coordination. In our proposed algorithm, when a device joins the network, it initializes its transceiver mode randomly. Specifically, it becomes a transmitter with probability p_{tr} or a receiver with

probability $1 - p_{tr}$. If a device starts as a receiver, it first checks whether a synchronization signal is present in the received signal. Note that the auto-correlation of two ZC sequences with the same root index exhibits a peak value equal to N [9]. Since unit-energy pulses are used in the broadcasted sequences in (1), the maximum amplitude of the auto-correlation equals the sequence length N in the case of a noiseless flat channel. However, due to noise, multipath fading and the superposition of multiple synchronization signals, the peak value of (6) fluctuates around N. Thus, we propose to use the following detection rule:

$$\max_{l}(|R_{y_j x_{\mp}}[l;\nu]|) \geq N/2. \tag{13}$$

If such detection occurs, the device estimates the average TO according to (6)-(8) and updates its clock as in (12). Then, it switches its transceiver mode to become a transmitter for the next clock tick $\nu + 1$. However, if the device does not detect any synchronization signal, it only advances its clock by T_0 for the $(\nu+1)^{th}$ clock tick, where it re-selects its transceiver mode randomly as in the initialization stage. The reason behind this behavior is two-fold: Noisy clock updates due to weak signal reception are avoided and, perhaps more importantly, the situation where all devices are either in receiver or transmitter mode becomes highly unlikely. On the other hand, if a device operates as a transmitter, it broadcasts its synchronization signal, advances its clock by T_0, and then, switches its transceiver mode to become a receiver at the $(\nu+1)^{th}$ clock tick.

With the proposed alternating transceiver mode algorithm, synchronization signals are exchanged between the same clusters of transmitter and receiver devices as long as a synchronization signal is detected. In other words, from a specific j^{th} receiver's point of view, the set $\mathcal{D}_j^{\nu}$ is mainly static over ν.

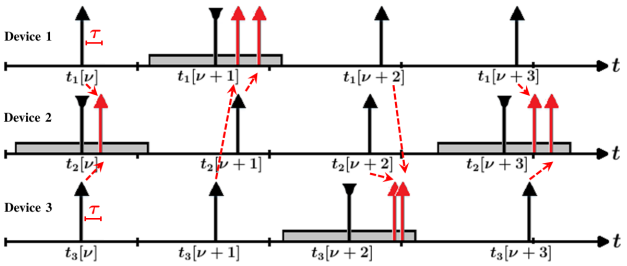
To illustrate the behavior of the proposed algorithm we compare it in Fig. 2 to the random transceiver mode algorithm proposed in [12], where the devices determine their transceiver mode randomly at *each* clock tick and become a transmitter with probability p_{tr} or a receiver with probability $1 - p_{tr}$. As seen in Fig. 2a, in the case of selecting the transceiver mode randomly, the average TO seen from the receivers may fluctuate greatly from one clock tick to the next, which, in turn, results in increased synchronization errors. In sharp contrast, with the proposed method the average TO estimates and clock updates evolve over time in an orderly manner (cf. Fig. 2b) which facilitates fast convergence with a deterministic bias.

B. Modification to DPLL Clock Update

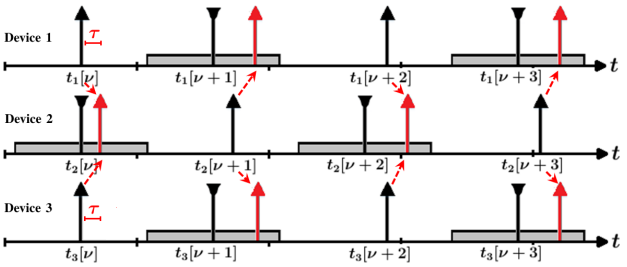
In order to mitigate the bias in the average TO estimate in (11), we propose to modify the DPLL clock update rule as follows:

$$t_j[\nu+1] = t_j[\nu] + T_0 - 2\hat{\phi}_j[\nu] + \Delta t_j[\nu], \ \forall j \in \mathcal{R}_\nu \tag{14}$$

where $\hat{\phi}_j[\nu]$ is an estimate of the bias at the ν^{th} clock tick. In (14), we set $\epsilon = 1$. An iterative method for the tracking of the bias is discussed in the next subsection.



(a) Random transceiver mode algorithm.



(b) Alternating transceiver mode algorithm.

Fig. 2. Comparison between the random transceiver mode selection algorithm and the proposed algorithm for pulse-coupled clock synchronization in the presence of propagation delays. A network of 3 devices is assumed, located at equal distances from each other. The propagation delay from one device to any other device is τ. Hence, broadcasted beacons are detected with a delay as depicted with dashed-arrows within the receiver's detection period (depicted in gray). Note that at the ν^{th} clock tick Device 2 is a receiver while the other two devices are transmitters.

If a receiver device at the ν^{th} clock tick detects a synchronization signal, it updates its clock as in (14). Then, it will act as a transmitter at the $(\nu+1)^{th}$ clock tick. On the other hand, transmitting devices will always alternate their transmission mode to become receivers, where their next clock tick is given in (10). Thus, the receiver devices at the ν^{th} clock tick "time-advance" their clocks prior to broadcasting their beacons at the $(\nu+1)^{th}$ clock tick. This improves the alignment of the beacons as detected by the receiver devices at the $(\nu+1)^{th}$ clock tick which in turn reduces the bias in (11) at future ticks. Note that if the estimate $\hat{\phi}_j[\nu]$ was subtracted once, the effect of the bias would still be present at the next clock tick.

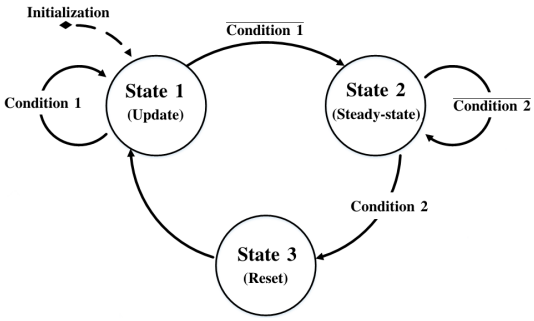


Fig. 3. State transitions of the proposed iterative bias tracking algorithm. Transitions happen at each clock tick. $\overline{\textbf{Condition x}}$ denotes the contrary of **Condition x**.

C. Bias tracking

The iterative nature of the DPLL algorithm where $\Delta t_j[\nu]$ is used to correct the device clocks and, hence, affects $\Delta t_j[\nu+1]$ calls for an iterative approach for tracking the bias $\overline{\tau_j}[\nu]$ using $\Delta t_j[\nu]$. Note that we only observe the cumulative effect of the bias and the weighted average relative clock differences in $\Delta t_j[\nu]$. However, to isolate the bias from the observed quantity we can capitalize on the fact that, due to the proposed alternating transceiver mode algorithm, it is stationary over ν. The proposed iterative bias estimation method is described by the state transition diagram in Fig. 3.

Initialization: Set

$$\hat{\phi}_j[0] = \hat{\phi}_j^{init} \tag{15}$$

where $\hat{\phi}_j^{init}$ is the initial estimate of the bias, which can be chosen in a variety of ways. In this work, assuming that the devices are randomly deployed with coordinates X, Y, e.g., $X, Y \sim \mathcal{U}(0, d)$, we set

$$\hat{\phi}_j^{init} = \frac{1}{c}\mathbf{E}(\sqrt{X^2 + Y^2}) \tag{16}$$

where c is the speed of light, and $\mathbf{E}(\cdot)$ denoted expectation. Since the initial estimate is a rough approximation of the propagation delays, it can be set to a different value depending on the position distributions of the devices.

State 1: Update of the estimate:

$$\hat{\phi}_j[\nu+1] = \begin{cases} \hat{\phi}_j[\nu] - \delta, & \Delta t_j[\nu] < 0 \\ \hat{\phi}_j[\nu] + \delta, & \Delta t_j[\nu] > 0 \end{cases} \tag{17}$$

where $\Delta t_j[\nu]$ is calculated using (8), δ is the step-size for bias tracking, and λ_{th} is the synchronization error threshold, both of which will be discussed later.

We keep revisiting State 1, i.e., updating the bias estimate, until Condition 1 is satisfied, that is:

Condition 1: $|\Delta t_j[\nu]| \leq |\Delta t_j|_{min}$ **or** $|\Delta t_j|_{min} > \lambda_{th}$

where $|\Delta t_j|_{min}$ is the minimum value of the weighted average TO encountered up to the ν^{th} iteration, which can be less than λ_{th}.

State 2: Keep the estimate corresponding to the minimum error $|\Delta t_j|_{min}$:

$$\hat{\phi}_j[\nu+1] = \hat{\phi}_j[\nu] \tag{18}$$

until **Condition 2** which signifies a perturbation in the system, e.g., a new device joins the network, is satisfied.

Condition 2: $\big||\Delta t_j[\nu]| - |\Delta t_j|_{min}\big| > \lambda_{th}$

State 3: Reset the estimate:

$$\begin{aligned} |\Delta t_j|_{min} &= \Delta t_j[\nu] \\ \hat{\phi}_j[\nu+1] &= \hat{\phi}_j[\nu] = \hat{\phi}_j^{init} \end{aligned} \tag{19}$$

The main idea behind the proposed bias tracking algorithm is that properly compensating the bias should lead to small $|\Delta t_j[\nu]|$, $\forall j$ as $\nu \to \infty$. The $\pm\delta$ updates of $\hat{\phi}_j[\nu]$ in (17) adjust $\hat{\phi}_j[\nu]$ towards decreasing $|\Delta t_j[\nu]|$. For instance, if the average TO $\Delta t_j[\nu]$ is negative, then $\hat{\phi}_j[\nu]$ should be reduced by a small amount δ, since the subtraction of $2\hat{\phi}_j[\nu]$ with a negative $\Delta t_j[\nu]$ in (14) will produce a larger negative $|\Delta t_j[\nu]|$ at the next clock tick. The adjustment of $\hat{\phi}_j[\nu]$ continues until the average TO becomes small enough, i.e., $|\Delta t_j[\nu]| \approx |\Delta t_j|_{min}$ with $|\Delta t_j|_{min} \leq \lambda_{th}$, in which case $\hat{\phi}_j[\nu]$ is kept fixed as long as $|\Delta t_j[\nu]|$ remains in the vicinity of $|\Delta t_j|_{min}$. However, the receiver devices should detect sudden changes (see **Condition 2**) and act accordingly, which may happen due to a perturbation in the network such as when new devices join the network. In this case, the tracking begins anew by replacing the current estimate with the initial value $\hat{\phi}_j^{init}$.

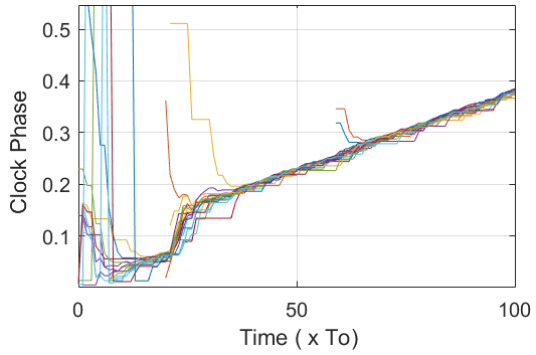
TABLE I. System parameters used in the simulations.

Parameter Description	Symbol	Value
Number of Devices	K	15 and 20
Scaling Term of DPLL Filter	ϵ	0.5
Zadoff-Chu Index	u	1
Zadoff-Chu Sequence Length	N	31
Clock Period	T_0	120μs and 10 ms
Maximum Network Distance	d	500 m
Operating Frequency	f	2 GHz
Delay Compensation Step Size	δ	33 ns
Synchronization Error Threshold	λ_{th}	1.5 μs
Transceiver mode selection probability	p_{tr}	0.5

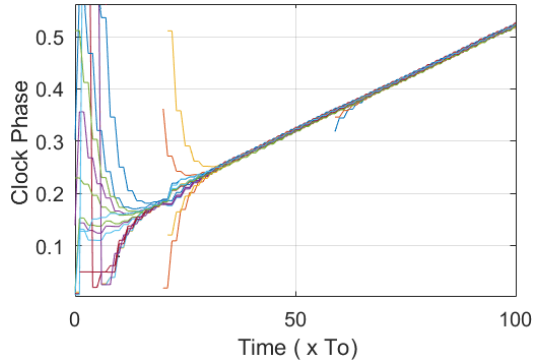
IV. Numerical Results

The proposed synchronization algorithm is implemented for a stationary network deployment over a Manhattan grid in a dense urban environment [14]. The rest of the system parameters are set according to the technical specifications for LTE-Advanced (LTE-A) D2D proximity services [15]. The system parameters along with the algorithm parameters are summarized in Table I. First, we study the evolution of the local clocks of the devices as indicated by their clock phases θ_i. For visualization purposes, we change the clock period to 120μs, since propagation delays for a D2D network is on the order of μs, and consider a network with 20 devices. In Fig. 4, we compare the proposed half-duplex synchronization algorithm (without compensating the propagation delays) with the random half-duplex transceiver mode as transmitter probability being $p_{tr} = 0.5$. Note that in both plots, there is an upward drift due to the propagation delay-induced bias $\overline{\tau_j}[\nu]$. However, in the proposed algorithm, convergence to a common clock phase is faster and more robust to perturbations caused by new devices joining the network at the 20^{th} and 60^{th} clock ticks.

Then, we examine the synchronization error $|\Delta t_j[\nu]|$ of the proposed half-duplex synchronization algorithm combined with propagation delay compensation. We compare the proposed methods with random transceiver mode with different transmitter probabilities p_{tr}. In Fig. 5 we show the evolution of the maximum synchronization error over 15 devices. As we can see, the proposed algorithm achieves not only faster convergence, but also lower steady-state error. Even though multiple new devices join the network and cause perturbations at the 25^{th} clock tick in Fig. 5, synchronization error is rapidly compensated and the steady-state error drops back to 1μs. On the other hand, in the case of the random transceiver mode algorithm (which by design



(a) Random transceiver mode algorithm with $p_{tr} = 0.5$.



(b) Alternating transceiver mode algorithm.

Fig. 4. Clock convergence using DPLLs under the effect of propagation delays.

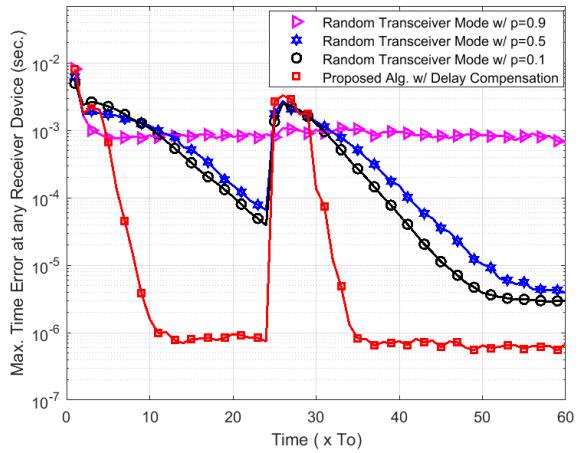


Fig. 5. Maximum clock synchronization error between 15 devices with clock period $T_0 = 10ms$.

does not include delay compensation) the devices experience slow convergence with higher synchronization error.

V. Conclusion

In this paper, we proposed a fully distributed half-duplex clock synchronization scheme over multipath channels for D2D networks. We numerically demonstrated that the proposed algorithm coupled with a new modification to the DPLL update can substantially mitigate the synchronization error in realistic setups as the ones specified for LTE Advanced D2D proximity services. Furthermore, we showed that the proposed algorithm is also scalable and robust to perturbations in the network. Although this paper focuses on D2D networks, the proposed half-duplex clock synchronization algorithm is not limited to D2D specifications and can be extended to any distributed system with half-duplex communication constraints such as wireless sensor networks.

References

[1] W. Sun, F. Brännström and E. G. Ström, "Network synchronization for mobile Device-to-Device systems," *IEEE Trans. on Commun.*, vol. 65, no. 3, pp. 1193–1206, Mar. 2017.

[2] R. I. Ansari, C. Chrysostomou, S. A. Hassan, M. Guizani, S. Mumtaz, J. Rodriguez, J. J. P. C. Rodrigues, "5G D2D Networks: techniques, challenges, and future prospects," *IEEE Systems Journal*, doi: 10.1109/JSYST.2017.2773633, Dec. 2017.

[3] N. Abedini, S. Tavildar, J. Li, and T. Richardson, "Distributed synchronization for device-to-device communications in an LTE network," *IEEE Trans. Wireless Commun.*, vol. 15, pp. 15471561, Feb. 2016.

[4] M. A. Alvarez, B. Azari, U. Spagnolini, "Time and frequency self-synchronization in dense cooperative network," in *Proc. Asilomar Conf. on Signals, Systems and Computers*, pp. 1811–1815, Nov. 2014.

[5] E. Garcia, S. Mou, Y. Cao, and D. W. Casbeer, "An event-triggered consensus approach for distributed clock synchronization," in *Proc. IEEE American Control Conf.*, pp. 279–284, May 2017.

[6] O. Simeone, U. Spagnolini, Y. Bar-Ness, and S. H. Strogatz, "Distributed synchronization in wireless networks," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 81–97, Sept. 2008.

[7] Y.-W. Hong, and A. Scaglione, "A scalable synchronization protocol for large scale sensor networks and its applications," *IEEE J. Select. Areas Commun.*, vol. 23, no. 5, pp. 1085–1099, May 2005.

[8] D. Tétreault-La Roche, B. Champagne, I. Psaromiligkos and B. Pelletier, "On the use of distributed synchronization in 5G device-to-device networks," in *Proc. IEEE Int. Conf. on Commun.*, pp. 1938–1883, Jul. 2017.

[9] D. Chu, "Polyphase codes with good periodic correlation properties," *IEEE Trans. Inf. Theory*, vol. 18, no. 4, pp. 531–532, Jul. 1972.

[10] K. Manolakis and W. Xu, "Time synchronization for multi-link D2D/V2X communication," *IEEE Vehicular Tech. Conf. (VTC-Fall)*, pp. 1–6, Sept. 2016.

[11] H. Han, J. Kim, H. P. Hyuck and M. Kwon, "An effective distributed synchronization method for device-to-device communications," in *Proc. IEEE Int. Conf. on Consumer Electronics*, pp. 346–347, Mar. 2017.

[12] M. A. Alvarez, U. Spagnolini, "Half-duplex scheduling in distributed synchronization," in *Proc. IEEE Int. Conf. on Commun.*, pp. 6240–6245, June 2015.

[13] M. M. U. Gul, X. Ma and S. Lee, "Timing and frequency synchronization for OFDM downlink transmissions using zadoff-chu sequences," *IEEE Trans. on Wireless Commun.*, vol. 14, no. 3, pp. 1716–1729, Mar. 2015.

[14] "Mobile and wireless communications enablers for the twenty-twenty information society (METIS)," Deliverable D1.4 METIS Channel Models, ICT-317669-METIS/D1.4, Feb. 2015. [Online]

[15] "Technical specification group radio access network; study on LTE device-to-device proximity services," 3rd Generation Partnership Project (3GPP), TR 36.843, Mar. 2014, Sections A.2.1.1 - A.2.1.2. [Online].