Received 10 December 2020; revised 8 January 2021; accepted 21 January 2021. Date of publication 26 January 2021; date of current version 11 February 2021.

Digital Object Identifier 10.1109/OJCOMS.2021.3054810

A Distributed Pulse-Based Synchronization Protocol for Half-Duplex D2D Communications

ONUR KARATALAY [1] (Graduate Student Member, IEEE), IOANNIS PSAROMILIGKOS [1] (Member, IEEE), BENOIT CHAMPAGNE [1] (Senior Member, IEEE), AND BENOIT PELLETIER [2]

[1] Department of Electrical and Computer Engineering, McGill University, Montreal, PQ H3A 0G4, Canada
[2] InterDigital Canada Ltée, Montreal, PQ H3A 3G4, Canada

CORRESPONDING AUTHOR: O. KARATALAY (e-mail: onur.karatalay@mail.mcgill.ca)

This work was supported in part by the Natural Science and Engineering Research Council (NSERC) of Canada under the Discovery Grant program, and in part by InterDigital Canada Ltée and MITACS Canada under the MITACS Accelerate program.

ABSTRACT In distributed device-to-device (D2D) communications, no common reference time is available and the devices must employ distributed synchronization techniques. In this context, pulse-based synchronization, which can be implemented by distributed phase-locked loops is preferred due to its scalability. Several factors degrade the performance of pulse-based synchronization, such as duplexing scheme, clock skew and propagation delays. Furthermore, in distributed networks, devices should be aware of the synchronization status of others in order to initiate data communications. To address these prevailing issues, we first introduce a half-duplex timing-advance synchronization algorithm wherein each device alternates between being a transmitter and receiver in their exchange of synchronization pulses at each clock period. Based on this algorithm, we propose a novel fully-distributed pulse-based synchronization protocol for half-duplex D2D communications in 5G wireless networks. The protocol allows participating devices to become aware of the global synchronization status, so that they can complete the synchronization process ideally at the same time and proceed to data communication. In simulation experiments over multi-path frequency selective channels, the proposed synchronization protocol is shown to outperform a benchmark approach from the recent literature over a wide range of conditions, e.g., clock skew, number of devices, and network topology.

INDEX TERMS Distributed synchronization, phase locked-loops, timing-advance, device-to-device communication, half-duplex, 5G.

I. INTRODUCTION

THE FIFTH generation (5G) of wireless networks is currently being actively deployed in various parts of the world. It is expected that new use cases will continue to be addressed for years to come by introducing new features into the specifications of 5G or future generations. Device-to-device (D2D) communications [1]–[3], i.e., direct connection between selected devices, is a functionality that has been introduced recently in Release 16 of the 3GPP specifications in support of vehicle-to-vehicle communications [4]. Hence, traffic load on base-stations (BSs) can be significantly reduced since they do not need to initiate, maintain or relay the connections between such devices [5]. Moreover, this feature is expected to be further developed to address use cases such as advanced vehicle-to-vehicle communications, extended coverage via device relaying, industrial applications and virtual reality. Specifically in [6], the authors study distributed learning for D2D communications, whereas in [7], downlink resource allocation and power control are considered for underlay D2D networks. However, there remain several challenges of their own, especially regarding how the devices initiate and maintain data transmission.

Synchronization is an essential step in establishing a connection in a digital communication system. In cellular networks, a fixed access point (AP) such as a BS regularly broadcasts a time signal so that user devices can synchronize themselves to a common reference clock. Such centralized schemes not only offer fast synchronization but also easily maintain it by regularly correcting the device clocks, which can diverge due to clock skews. In addition, the same AP can coordinate the devices during communication to compensate for propagation delays by adjusting their clocks, a technique known as timing-advance. However, in distributed systems such as wireless sensor networks (WSNs) or out-of-coverage D2D networks, the latter being one of the use case for D2D communications, no such fixed AP is available. In this type of scenarios, the aforementioned benefits of centralized synchronization can be retained by selecting, e.g., via an "elect-a-leader" algorithm [8], one of the devices to serve as the synchronization AP. Nonetheless, the synchronization performance depends heavily on the choice of the AP and on the quality of the channels between the AP and the rest of the devices. In addition, if the AP loses connectivity with a part of the network or leaves the network, then the AP selection process should be re-initiated.

An alternative approach is distributed synchronization, wherein devices exchange synchronization signals according to a predefined strategy, or protocol, allowing them to reach a consensus on a common reference time [9]. Although typically slower than centralized synchronization, distributed synchronization is more robust against connectivity failures and network changes due to mobility. Hence, it may be better suited for WSNs or out-of-coverage D2D networks where such conditions are prevailing [10]. Nevertheless, the technical aspects of a network determine the design of the synchronization algorithm. Specifically, in a typical WSN, information flows towards a single sink node, while data communication is low rate and sporadic with relaxed guarantees in terms of latency and reliability. In contrast, D2D communication by nature requires high data rate, low-latency and reliable communication between arbitrary devices. Therefore, to satisfy these requirements, distributed D2D networks must employ a reliable, fast and efficient synchronization algorithm, which should also mitigate the effect of propagation delays and multipath channels by properly using timing-advance [11].

A. RELATED WORKS AND MOTIVATION

The prevalent approaches for distributed synchronization can be divided into two main categories, namely: packet-based and pulse-based [10]. Packet-based synchronization is a medium access control (MAC) layer-based approach relying on the exchange of timestamps encoded in packets [12]–[15]. It requires collision-free transmission of the packets on a random access channel and their subsequent successful decoding. As such it suffers from delays due to packet queuing and re-transmissions and, more importantly in the context of D2D communications, it exhibits high energy expenditure, high latency and poor scalability. Pulse-based synchronization, in contrast, is a physical layer-based approach where the timing information is encoded in the transmission time of physical layer pulses. Local clock updates are done by processing the received superposition of timing pulses transmitted by neighboring devices. This approach, which naturally capitalizes on the broadcast nature of wireless channels, can overcome the above-mentioned limitations of packet-based approach. Thus, pulse-based is often preferred over packet-based synchronization in distributed wireless networks [9], [10].

Pulse-based synchronization is typically implemented by distributed phase locked-loops (DPLL) [16], [17]. The performance of DPLL is affected by the duplexing scheme employed by the devices. Full-duplex communication significantly decreases the synchronization time compared to half-duplex due to simultaneous signal transmission and reception, and has been considered by several authors [9], [18]. However, implementation of full-duplex technology, especially at the mobile devices poses a number of practical issues in terms of cost, complexity and power consumption [16], [17]. Due to the additional power required for self-interference cancelation mobile devices with a limited battery life cannot currently afford to operate in the full-duplex mode [19]. Hence, half-duplex communication is a more practical implementation choice for distributed networks and is likely to remain so in the near future.

Several other factors limit the performance of DPLL, such as the quality of the crystal oscillators whose frequency may drift with temperature fluctuations. This effect, known as clock skew, alters the perceived rate of signal transmission and reception arbitrarily over time. Furthermore, the effects of clock phase and propagation delays are entangled within the superimposed timing pulses. In order to achieve synchronization, the aforementioned effects should be estimated from the received pulses and removed by updating the device clocks accordingly [17]. However, this estimation becomes highly challenging as the received pulses are not only altered by these effects, but also by the very mechanisms used to correct them, which could lead to instability.

In the literature, it is often assumed that: the device clocks are frequency synchronized (i.e., there is no clock skew) [9], [16]–[18], there are no propagation delays, and the network size (i.e., the number of devices) is static [20]. However, signal propagation delays do exist and D2D devices can arbitrarily join or leave the network; hence, these assumptions are not valid in a realistic scenario. In [21], a synchronization method is proposed, in which clock skew and clock phases are corrected with respect to a selected reference node which does not participate in the synchronization process. Another approach is proposed in [18], where the devices first estimate the propagation delays to their neighbors and then transmit these estimates to a centralized fusion center; in turn, the center informs all the devices about the delay estimates within the network. Consequently, synchronization is achieved after pre-compensation for propagation delays is applied, i.e., timing-advance. In the absence of a centralized fusion center as, for example, in an out-of-coverage D2D network, such global time-advance compensation is impractical. Clock phases, clock skews and propagation delays in this case should be jointly estimated in a distributed manner, thereby allowing devices to synchronize their clocks individually by means of timing-advance.

Furthermore, to initiate data communication in a distributed D2D network, the participating devices should simultaneously terminate the synchronization process. However, the devices are not aware of the synchronization status of others in a distributed network and the time it takes to reach synchronization might vary for each device. If some of the devices stop their synchronization process later than others, the presence of ongoing timing pulses might trigger the synchronized devices to re-start this process [17]. Thus, in a fully distributed wireless network, the devices should be aware of the global synchronization status, so that they can terminate the synchronization process ideally at the same time and proceed to data communication.

Once the synchronization process is completed, some of the devices might become idle to conserve energy, while others might begin data communication. In this case, the idle devices may lose synchronization with respect to the communicating ones since there is no mechanism to inform them about their status. To prevent this, the devices must periodically re-transmit timing pulses, i.e., discovery beacons, to remain part of the network [22], [23]. However, this will cause a disturbance among the synchronized devices and increase energy consumption in the overall network. Hence, synchronization should be maintained as long as possible without exchanging timing pulses to reduce the frequency of unnecessary re-initialization.

B. CONTRIBUTION AND PAPER ORGANIZATION

In this article, motivated by the above considerations, we first introduce a timing-advance synchronization algorithm wherein each device alternates between the transmitter and receiver modes in their exchange of synchronization pulses at each clock period. Based on this algorithm, we then propose a novel fully-distributed pulse-based synchronization protocol for half-duplex D2D communications in 5G networks. This work significantly extends upon our previous contributions in [17], [24], where we focus on a simplified version of the problem by neglecting the presence of clock skew and its influence on data communication. Specifically, our main contributions in this work are summarized as follows:

- We propose an online estimation technique which jointly tracks the synchronization errors due to clock phases, clock skews and propagation delays in multipath channels. We incorporate the obtained estimates in the conventional DPLL clock update and propose a half-duplex timing-advance synchronization algorithm which compensates for *all* these effects in a distributed manner. In particular, we show analytically that the proposed compensation mechanism contributes to reducing the synchronization error at each iteration.
- We conceive a distributed synchronization protocol in the form of a state diagram, which can be easily implemented on each device. In this protocol, the participating devices acquire the synchronization status of others by a coordinated exchange of pulses, and terminate this process as soon as the overall network is synchronized. Our proposed protocol also allows the devices already in operation to detect the presence of new devices joining the network at any time, and to re-synchronize themselves by including the new ones.
- After the network is synchronized, the devices can either initiate data communication or stay idle and only operate as a receiver to conserve power. At this point, to maintain synchronization in the network without exchanging timing pulses, the devices predict their relative clock time based on estimated values of the synchronization parameters. Thus, they can preserve the already achieved synchronization without unnecessarily re-initiating this process.
- The complete integrated protocol is evaluated by means of computer simulations based on 5G channel models and under different conditions of operation, i.e., clock skew, number of devices, and network topologies, including full mesh and partial mesh. Our extended results show that the proposed protocol offers better synchronization performance than a benchmark approach from the current literature [16], even for partial-mesh topology.

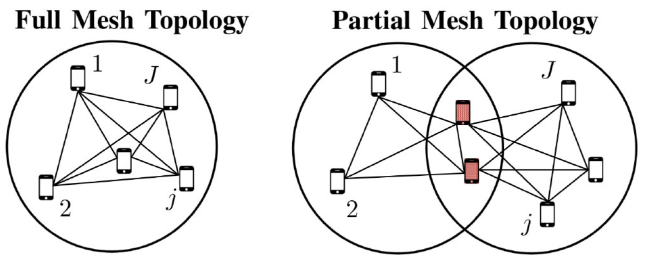


FIGURE 1. Fully or partially connected D2D networks. In the partial mesh topology, common devices (shown in red), act as a relaying node during synchronization.

The rest of the paper is organized as follows. In Section II, we present the system model and the problem statement for distributed D2D synchronization. In Section III, we propose a distributed half-duplex synchronization algorithm that uses timing-advance. In Section IV the overall synchronization protocol with its state diagram is described in detail. The performance of the proposed protocol is evaluated by means of computer simulations in Section V. Finally, Section VI concludes the paper.

II. SYSTEM MODEL AND PROBLEM STATEMENT

A. NETWORK SETUP AND CLOCK MODEL

We consider a distributed overlaying D2D network, which can be fully or partially connected, with J wireless devices indexed by $j \in \mathcal{J} = \{1, \ldots, J\}$ as illustrated in Fig. 1. We assume that the devices may join or leave the network at any time and that they do not have any information about the network, such as the number of nearby devices or their locations. Since there is no BS to provide a common timing reference, the devices synchronize their clocks in a fully distributed manner by exchanging timing pulses over radio frequencies at the physical layer.

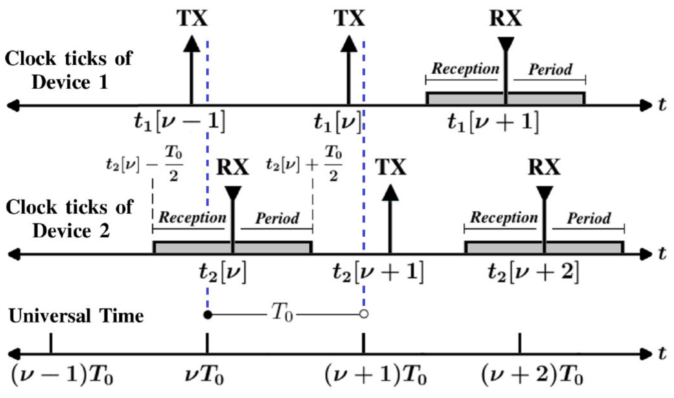


FIGURE 2. Clock ticks of two devices relative to the partitioned universal time axis. At the νth clock tick, device 1 is a transmitter as denoted by TX (upward arrow) whereas device 2 is a receiver as denoted by RX (downward arrow).

The physical clock of the jth device is modeled as $t_j(t) = \alpha_j t + \theta_j$ [25], where t is the universal time, α_j is the clock skew, and $\theta_j \in [0, T_0)$ is the clock phase with T_0 being the clock period. A discrete logical clock is obtained by uniformly sampling the physical clock at times $t = \nu T_0$, that is:

$$t_j[\nu] = t_j(\nu T_0) = \alpha_j \nu T_0 + \theta_j \tag{1}$$

where $\nu \in \mathbb{N}$ is the discrete-time index. We refer to time $t_j[\nu]$ as the νth *clock tick* of the jth device. It is convenient to partition the universal time axis into a sequence of non-overlapping time slots $[\nu T_0, (\nu+1)T_0)$. In practice, α_j differs from 1 by a very small amount, on the order of a few parts per million (ppm) [26], and it is therefore safe to assume that each time slot contains a single clock tick $t_j[\nu]$ as shown in Fig. 2. In the ideal case of no propagation delay, we define the time-offset (TO) between the ith and the jth devices as the minimum clock tick difference, that is:

$$\Delta t_{ij}[\nu] = \min_{\eta \in \mathcal{V}_\nu} \left| t_i[\eta] - t_j[\nu] \right| \tag{2}$$

where for convenience, we define the set $\mathcal{V}_\nu = \{\nu, \nu \pm 1\}$. In effect, $\Delta t_{ij}[\nu]$ can be interpreted (for this ideal case) as the synchronization error between the devices i and j.

B. HALF-DUPLEX SIGNALING MODEL

We assume half-duplex communication, where a device can only transmit or receive at any given time. We define the transceiver mode of the jth device at the νth clock tick as $M_\nu^j \in \{\text{TX}, \text{RX}\}$ indicating whether the device operates in transmitter mode (TX) or in receiver mode (RX), as depicted in Fig. 2. We let $\mathcal{T}_\nu$ and $\mathcal{R}_\nu$ denote the mutually exclusive index sets of transmitter and receiver devices at the νth clock tick, respectively.

If, at a given clock tick ν, a device operates as a transmitter, i.e., $i \in \mathcal{T}_\nu$, it broadcasts a time-shifted synchronization signal $x(t - t_i[\nu])$ with $x(t)$ defined as:

$$x(t) = \sum_{n=0}^{N_s-1} s[n] g\big(t - nT_p\big) \tag{3}$$

where $g(t) \in \mathbb{R}$ is a normalized baseband pulse and T_p denotes the pulse spacing with $N_s T_p \ll T_0$. In (3), $s[n]$ is a synchronization sequence of length $N_s = 2N$ constructed by concatenating two Zadoff-Chu (ZC) sequences of length N with root indices u and $-u$ [18], that is:

$$s[n] = \begin{cases} e^{\mathrm{j}\frac{\pi}{N}un^2}, & 0 \le n \le N-1 \\ e^{-\mathrm{j}\frac{\pi}{N}u(n-N)^2}, & N \le n \le 2N-1 \end{cases} \tag{4}$$

where u and N are coprime, and $\mathrm{j} = \sqrt{-1}$. By constructing the synchronization sequences as in (4), the effect of carrier frequency offsets (CFO) is decoupled from TO estimation [18].

A receiver device $j \in \mathcal{R}_\nu$ listens for broadcasted synchronization signals over the *reception period* $[t_j[\nu] - \frac{T_0}{2}, t_j[\nu] + \frac{T_0}{2})$, centered at its own clock tick $t_j[\nu]$, as illustrated in Fig. 2. The received signal at the jth device is:

$$y_j(t) = \textstyle\sum_{\eta \in \mathcal{V}_\nu} \sum_{i \in \mathcal{T}_\eta} x(t - t_i[\eta]) * h_{ij}(t) + w_j(t) \tag{5}$$

where $h_{ij}(t) = \sum_{p \in \mathcal{P}} \rho_{ijp} \delta(t - \tau_{ijp})$ is the impulse response of the multipath channel between the ith and jth device, $p \in \mathcal{P} = \{1, \ldots, P\}$ is the path index, P is the number of resolvable paths, assumed to be the same for all devices, and $\delta(\cdot)$ is the Dirac delta function. Additionally, $\rho_{ijp} \in \mathbb{C}$ and $\tau_{ijp} \in \mathbb{R}_+$ are the complex gain and propagation delay, respectively, of the pth path, while operator $*$ denotes convolution and $w_j(t)$ is an additive noise term. Note that depending on the clock phase of the receiver device, the received signal may contain signal contributions not only from the νth clock tick but also from the adjacent ones, i.e., $(\nu \pm 1)$th. Thus, the outer summation in (5) takes all possible signal contributions into account; however, it does not span beyond the $(\nu - 1)$th time slot as the propagation delays are assumed to be much smaller than the clock period, that is, $\tau_{ijp} \ll T_0$.

Finally, the jth receiver device samples (5) at time instances kT_s during the reception period, where $T_s \ll T_0$ is the sampling period, $k \in \mathcal{K} = \{-K, \ldots, -1, 0, 1, \ldots, K\}$ is the discrete-time index, and $K = \lfloor \frac{T_0}{2T_s} \rfloor$. The resulting sampled signal at the νth clock tick is expressed as:

$$\begin{aligned} y_j[k;\nu] &= y_j\big(kT_s + t_j[\nu]\big) \\ &= \sum_{\eta \in \mathcal{V}_\nu} \sum_{i \in \mathcal{T}_\eta} \sum_{p \in \mathcal{P}} \rho_{ijp} x\big(kT_s + t_j[\nu] - t_i[\eta] - \tau_{ijp}\big) + w_j[k] \end{aligned} \tag{6}$$

where $w_j[k]$ is the discrete-time noise process.

C. DPLL CLOCK UPDATE

At the νth clock tick, the jth receiver device cross-correlates (6) with the two distinct parts of the synchronization signal (available locally) to decouple the effect of CFO in TO estimation [18]:

$$R_{y_j x_\pm}[l, \nu] = \sum_{k \in \mathcal{K}} y_j[k;\nu] x_\pm[k - l]^* \tag{7}$$

where l is the integer lag and superscript * denotes complex conjugation. The first correlation is with $x_+[k] = \sum_{n=0}^{N-1} s[n]g(kT_s - nT_p)$, which is obtained from (3) by retaining pulses with index $0 \le n \le N-1$ and sampling every T_s, while the second correlation is with $x_-[k] = \sum_{N}^{2N-1} s[n]g(kT_s - nT_p)$, which is obtained in a similar way but for $N \le n \le 2N-1$. Hence, the subscripts $\pm$ in the signals $x_\pm[k]$ indicate which ZC root index (i.e., $+u$ or $-u$) is used to construct them.

The jth device then uses a weighted average across lags l to obtain two different preliminary TO estimates [9]:

$$q_j^\pm[\nu] = \frac{\sum_l lT_s |R_{y_j x_\pm}[l,\nu]|^2}{\sum_l |R_{y_j x_\pm}[l,\nu]|^2}. \tag{8}$$

Here, $q_j^+[\nu]$ provides an estimate of the average TO seen by the jth device, while $q_j^-[\nu]$ provides a similar estimate but with a positive offset of NT_p due to the definition of $x_-[k]$ in (7). Then, the jth device combines the estimates in (8) to obtain the desired weighted average TO estimate as:

$$\widehat{\Delta t}_j[\nu] = \tfrac{1}{2}\Big(q_j^+[\nu] + q_j^-[\nu] - NT_p\Big). \tag{9}$$

In the following, this quantity will be interpreted as the synchronization error experienced by the jth receiver at the νth clock tick. Hence, the jth device updates its clock tick for the $(\nu+1)$th time slot as dictated by DPLL [10]:

$$t_j[\nu+1] = t_j[\nu] + \alpha_j T_0 + \epsilon \widehat{\Delta t}_j[\nu], \;\; j \in \mathcal{R}_\nu \tag{10}$$

where $\epsilon > 0$ is a scaling parameter. We note that by using the DPLL clock update, the devices implicitly change their clock phases, which makes the clock phase time-variant, i.e., $\theta_j \equiv \theta_j[\nu]$. Therefore, to emphasize this point, we re-write the discrete logical clock in (1) as $t_j[\nu] = \alpha_j \nu T_0 + \theta_j[\nu]$, $\forall j \in \mathcal{J}$. In contrast to (10), for the ith transmitter device, no correction is made and the clock tick is updated as:

$$t_i[\nu+1] = t_i[\nu] + \alpha_i T_0, \;\; i \in \mathcal{T}_\nu. \tag{11}$$

D. PROBLEM STATEMENT

When devices join the network, unpredictable differences in clock phases, clock skews and propagation delays lead to synchronization errors. To further elaborate on this point, let us analyze the weighted average TO expression in (9). Based on the signal model in (6), this can be approximated as [10]:

$$\begin{aligned}\widehat{\Delta t}_j[\nu] &\approx \sum_{\eta\in\mathcal{V}_\nu}\sum_{(i,p)\in\mathcal{D}_j^{\nu,\eta}} \mu_{ijp}\big(t_i[\eta] + \tau_{ijp}\big) - t_j[\nu] \\ &= \underbrace{\sum_{\eta\in\mathcal{V}_\nu}\sum_{(i,p)\in\mathcal{D}_j^{\nu,\eta}} \mu_{ijp}\big(t_i[\eta] - t_j[\nu]\big)}_{\overline{\Delta t}_j[\nu]} + \underbrace{\sum_{\eta\in\mathcal{V}_\nu}\sum_{(i,p)\in\mathcal{D}_j^{\nu,\eta}} \mu_{ijp}\tau_{ijp}}_{\beta_j[\nu]}\end{aligned} \tag{12}$$

where $\mathcal{D}_j^{\nu,\eta} = \{(i,p) \in \mathcal{T}_\eta \times \mathcal{P}: |t_i[\eta] + \tau_{ijp} - t_j[\nu]| \le \frac{T_0}{2}\}$ is the set of pairs formed by the index of transmitter devices and the path indices contributing to the received signal of the jth receiver device during the reception period centered at its νth clock tick. In (12), μ_{ijp} is the normalized channel gain of the pth path between the ith and jth devices given by $\mu_{ijp} = |\rho_{ijp}|(\sum_{\eta\in\mathcal{V}_\nu}\sum_{(i,p)\in\mathcal{D}_j^{\nu,\eta}} |\rho_{ijp}|)^{-1}$. Moreover, $\widehat{\Delta t}_j[\nu]$ can be written as a sum of two terms: the first term $\overline{\Delta t}_j[\nu]$, is the weighted average of clock tick differences between the contributing transmitters and the jth receiver, which includes the effects of the relative clock phases and clock skews. The second term $\beta_j[\nu]$ is a weighted average of the propagation delays seen from the jth device; we will refer to this term as *bias*.

Using (12), the DPLL clock update in (10) becomes:

$$t_j[\nu+1] \approx t_j[\nu] + \alpha_j T_0 + \epsilon\big(\overline{\Delta t}_j[\nu] + \beta_j[\nu]\big), j \in \mathcal{R}_\nu. \tag{13}$$

Even if the device clocks were perfectly aligned initially, i.e., $t_i[0] = t_j[0]\ \forall i,j \in \mathcal{J}$, and consequently $\overline{\Delta t}_j[0] = 0$, the clocks might start deviating from each other as ν increases due to differences in clock skews, i.e., $\alpha_i \neq \alpha_j$, so that $\overline{\Delta t}_j[\nu] \neq 0$ for $\nu > 0$ in general. Furthermore, the propagation delays, which are always positive by nature, introduce an additional error due to the bias $\beta_j[\nu]$. Hence, our first objective in this work is to reduce the effects of clock phases, clock skews and bias by means of distributed timing-advance synchronization. Ultimately, for ν sufficiently large, the maximum synchronization error for the overall network in the practical case with propagation delays should not exceed a pre-defined threshold λ_{sync}, that is:

$$\max_{\substack{i,j\in\mathcal{J}\\ \eta\in\mathcal{V}_\nu}} \big|t_i[\eta] + \tau_{ij1} - t_j[\nu]\big| \le \lambda_{\text{sync}} \tag{14}$$

where τ_{ij1} is the delay of the first path and the maximum is over all $i,j \in \mathcal{J}$ as well as $\eta \in \mathcal{V}_\nu$.

In distributed networks, the devices are not aware of the synchronization status of others and they do not generally experience the same synchronization error. Therefore, the devices cannot stop the synchronization process simultaneously by just relying on their own error estimates, as given by (9). In fact, if some devices stop this process earlier than others, they might become asynchronous with respect to the remaining devices still running DPLL; while if some devices stop later than others, the presence of ongoing timing pulses might trigger the synchronized devices to re-start the process. Hence, our second objective aims for distributed coordination among the devices to let them be aware of the overall network synchronization status and, ideally, terminate the synchronization process at the same time.

After synchronization, some of the devices may become idle to conserve energy instead of immediately initiating data communication, in which case the idle devices may become asynchronous with the rest of the network. To prevent this, they must periodically re-initiate the synchronization process. However, frequent re-initialization will lead to a disturbance among the synchronized devices and increased energy consumption in the overall network. Thus, our third and final objective is to maintain synchronization as long as possible without exchanging timing pulses to reduce the frequency of re-initialization.

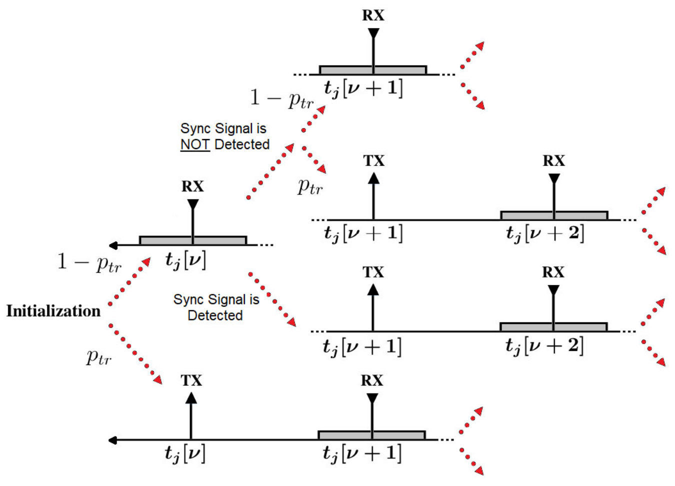


FIGURE 3. **Concept of the alternating transceiver mode.**

III. TIMING-ADVANCE SYNCHRONIZATION ALGORITHM

In this section, we introduce a timing-advance synchronization algorithm that will later be used to develop a full-fledged synchronization protocol. First, we introduce a half-duplex method, whereby each device alternates between the transmitter and receiver modes in their exchange of timing pulses at each clock tick. Second, we modify the DPLL clock update to achieve timing-advance synchronization in a distributed manner. Third, we present an online bias estimation technique, which will be incorporated into the modified DPLL. Finally, the overall algorithm is given from the perspective of a single device.

A. ALTERNATING TRANSCEIVER MODE

In distributed half-duplex synchronization, some devices broadcast timing pulses, while the others listen for the broadcasted signals during their own reception period to update their clock. However, if the devices randomly decide their transceiver mode at each clock tick as in [16], then the set $\mathcal{D}_j^{\nu,\eta}$ of transmitter devices and path indices seen by the jth receiver device, as defined in (12), will evolve unpredictably over time. In turn, this will result in arbitrary fluctuations in the weighted average TO estimates $\widehat{\Delta t_j}[\nu]$ [24].

To eliminate this source of randomness in the set $\mathcal{D}_j^{\nu,\eta}$, we propose a method called *alternating transceiver mode*, which enables devices to determine their transceiver mode independently in a systematic manner. The underlying concept of the method is illustrated in Fig. 3. When a device first joins the network at the νth clock tick, it randomly initializes its transceiver mode, where the probability of being a transmitter, denoted as $p_{\text{tr}} \in (0,1)$, is pre-determined and the same for all devices. If a device operates as a transmitter, it broadcasts its synchronization signal and then for the next clock tick, it changes its mode to become a receiver. If instead the device operates as a receiver, it listens for broadcast synchronization signals and attempts to determine whether or not such a signal is present during its reception period. This detection is performed by comparing the cross-correlation in (7) to a threshold value, as further explained in Section IV. In the case of signal detection (whose probability depends on several factors, e.g., the number of devices, p_{tr}, and the SNR) the device alternates its mode at the next clock tick to operate as a transmitter; otherwise, it randomly re-determines its transceiver mode based on p_{tr} as above.

Based on the diagram in Fig. 3 and assuming that the probability of signal detection for a receiver is high, the devices will cluster themselves into TX and RX groups and alternate between them at each clock tick. We have observed this behavior in our simulations under representative conditions of operation for Long-Term Evolution (LTE) [27], as further discussed in Section V. Hence, the synchronization signals are exchanged between the same groups of transmitter and receiver devices, which has two important implications. First, the clocks of each device are corrected at every two clock ticks by using DPLL as given in (10). Consequently, clock skew can only alter the device clocks for no more than one clock tick before being compensated, which helps speed up synchronization. Second, in the case of time-invariant channels, the bias term in (12) becomes constant, i.e., $\beta_j[\nu] = \beta_j[\nu+2]$, since the jth receiver is always affected by the same combination of transmitters and propagation paths, i.e., $\bigcup_\eta \mathcal{D}_j^{\nu,\eta}$ remains constant as ν is incremented to $\nu+2$.

B. MODIFIED DPLL

Timing-advance synchronization can reduce the effect of propagation delays. Under the alternating transceiver mode, the receiver devices expect signals from the same group of transmitters at every two clock ticks. Hence, we let the receiver devices take proactive actions by modifying the DPLL clock update in (10) as follows:

$$t_j[\nu+1]=t_j[\nu]+\alpha_j T_0+\epsilon\widehat{\Delta t_j}[\nu]-2\widehat{\beta_j}[\nu], \quad \forall j \in \mathcal{R}_\nu \quad (15)$$

where $\widehat{\beta_j}[\nu]$ is an estimate of the bias term in (12). Meanwhile the clock update of transmitter devices remains unchanged, as already given by (11).

To motivate the introduction of the term $-2\widehat{\beta_j}[\nu]$ in (15), we consider a simple case wherein two devices, labeled as D1 and D2, exchange synchronization signals under the alternating transceiver mode over a single path channel with delay $\tau > 0$. Assume that at the νth clock tick, D1 and D2 operate as transmitter and receiver, respectively, and that their clocks are perfectly aligned without clock skew, i.e., $t_1[\nu] = t_2[\nu]$ and $\alpha_1 = \alpha_2 = 1$ as illustrated in Fig. 4. After broadcasting its signal, D1 updates its clock based on (11) as $t_1[\nu+1] = t_1[\nu] + T_0$, and then switches to the receiver mode. Meanwhile, due to the propagation delay, the corresponding synchronization error at D2 is $\widehat{\Delta t_2}[\nu] = \tau$. Assuming that D2 has a perfect bias estimate, i.e., $\widehat{\beta_2}[\nu] = \tau$, it updates its clock based on the modified DPLL (15) as $t_2[\nu+1] = t_2[\nu] + T_0 - \tau$, where ϵ is set to 1 for simplicity. At the $(\nu+1)$th clock tick, D2 which now operates as a transmitter, broadcasts its synchronization signal. Due to the

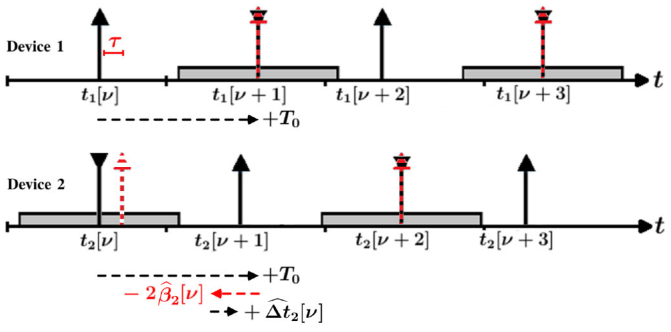


FIGURE 4. Timing-advance synchronization in a distributed manner with propagation delay τ (upward dashed arrow represents the arrival time of a synchronization signal).

additional term $-2\widehat{\beta}_j[\nu] = -2\tau$, the synchronization error at D1, which now operates as a receiver, will be $\widehat{\Delta t}_1[\nu+1] = 0$.

More generally, by employing the modified DPLL along with alternating transceiver mode, clocks of the receivers will be advanced in time with respect to the clocks of the transmitters at every tick. Therefore, the devices can achieve timing-advance synchronization in a distributed manner.

C. ESTIMATION OF THE BIAS

Application of the modified DPLL in (15) requires an estimate of the bias term $\beta_j[\nu]$. For estimating this term, the jth device can only rely on its TO estimate $\widehat{\Delta t}_j[\nu]$ from (9), which includes contributions from both $\overline{\Delta t}_j[\nu]$ and $\beta_j[\nu]$, as seen from (12). Furthermore, as each device updates its clock based on $\widehat{\Delta t}_j[\nu]$ by using the recursive DPLL in (15), future TO estimations will be affected by previous clock corrections. Hence, conventional time averaging techniques applied to $\widehat{\Delta t}_j[\nu]$ might fail to yield an unbiased estimate. Besides, the averaging time required by conventional techniques to achieve the desired accuracy may negatively impact the overall synchronization time. Therefore, estimation of $\beta_j[\nu]$ becomes challenging.

Alternatively, we can capitalize on the fact that, by using the alternating transceiver mode, the bias seen from a receiver device over time-invariant channels becomes constant. By exploiting this special property of $\widehat{\beta}_j[\nu]$, the jth receiver device may try to isolate it from $\overline{\Delta t}_j[\nu]$ while updating its clock based on (15). Since by using DPLL, $\overline{\Delta t}_j[\nu]$ tends to 0 as ν increases in the absence of propagation delays [10], estimation of the bias can be achieved by seeking to iteratively reduce the absolute synchronization error, i.e., $|\widehat{\Delta t}_j[\nu+2]| \leq |\widehat{\Delta t}_j[\nu]|$. To this end, we therefore propose the following online bias estimation technique:

$$\widehat{\beta}_j[\nu] = \widehat{\beta}_j[\nu-2] + \gamma_j \text{sgn}\big(\widehat{\Delta t}_j[\nu]\big) \tag{16}$$

where sgn$(\cdot)$ is the sign function, i.e., the jth receiver device applies corrections to its bias estimate with a small step size $\gamma_j \in \mathbb{R}_+$ based on the sign of its current TO estimate. Specifically, if $\widehat{\Delta t}_j[\nu] > 0$ (< 0), then the weighted average of the signal contributions from transmitters to the jth device is advanced (is lagging) in time with respect to its own clock

Algorithm 1 Timing-Advance Synchronization

```
1:  Initialize $M^j_{\nu_0}$ (based on $p_{\text{tr}}$) and $\widehat{\beta}_j[\nu_0] = \widehat{\beta}^{\text{init}}$
2:  for $\nu = 0, 1, 2, \ldots$ do
3:      if $M^j_\nu = \text{TX}$ then
4:          Broadcast the synchronization signal as in (3)
5:          Advance the clock as in (11)
6:          Become receiver: $M^j_{\nu+1} = \text{RX}$
7:      else
8:          if Synchronization signal is detected then
9:              Estimate the average TO as in (9)
10:             Update the clock as in (15)
11:             Update the bias estimate as in (16)
12:             Become transmitter: $M^j_{\nu+1} = \text{TX}$
13:         else
14:             $M^j_{\nu+1} = \begin{cases} \text{TX}, & \text{with } p_{\text{tr}} \\ \text{RX}, & \text{with } 1-p_{\text{tr}} \end{cases}$
15:             Advance the clock as in (11)
16:         end
17:     end
18: end
```

tick. Hence, the jth device increases (decreases) its previous estimate $\widehat{\beta}_j[\nu-2]$ by γ_j in order to reduce its future TO estimate $|\widehat{\Delta t}_j[\nu+2]|$. When a device joins the network at tick ν_0, it may initialize its bias estimate $\widehat{\beta}_j[\nu_0] = \widehat{\beta}^{\text{init}}$ based on the expected physical delay in the network. For instance, the initial value can be set to $\widehat{\beta}^{\text{init}} = \frac{d}{c}$, where d is the average distance between the devices and c is the speed of light.

In Appendix A, we analytically show that the application of (16) in a multi-device network leads to a reduction of the synchronization error at each device as ν increases. In practice, we have found that this technique can be improved further by using a dynamic step size instead of a fixed one, that is, $\gamma_j[\nu] = a\gamma_j[\nu-2] + b$, where $a \in (0, 1)$ and $b \in \mathbb{R}_+$ is a constant increment.

D. PROPOSED ALGORITHM

Each device independently implements the above procedures for distributed timing-advance synchronization after joining the network at the ν_0th clock tick. In Algorithm 1, these procedures are summarized and presented from the perspective of the jth device, assuming without loss in generality that it joins the network at the $\nu_0 = 0$th clock tick.

In Appendix B, we study the rate of decrease of TO when Algorithm 1 is applied to a simplified scenario consisting of two devices. Furthermore, in Appendix C, we derive the expected value of TO when the devices run Algorithm 1 in the same simplified scenario. Therefore, based on the effect of various parameters such as γ_j, p_{tr} and ϵ used in the algorithm, we can choose the values that yield the best performance in terms of error reduction.

IV. PROPOSED SYNCHRONIZATION PROTOCOL

Based on the timing-advance synchronization algorithm, we first propose a synchronization protocol and present it over a state-transition diagram and then we give its complexity analysis. The underlying idea of the protocol is to create coordination among the participating devices to ensure a simultaneous termination of the synchronization process and let them initiate data communication. Therefore, to coordinate the devices in a distributed manner, we propose to use two different synchronization signals; the first signal is utilized for error reduction, whereas the presence of the second signal is a declaration of the synchronization status of a device to the network.

We construct these two signals based on (3)-(4), which can be distinguished by their ZC index u. Specifically, all the transmitter devices broadcast the first signal by using the ZC index $u = u_1$ to decrease their synchronization error below a threshold. Once they achieve this, the devices start broadcasting the second signal by switching to the ZC index $u = u_2$. If a receiver device detects only the second signal but not the first one, then it knows that all the contributing devices are synchronized. In other words, the use of the ZC root index u_2 by a device serves as a declaration to the rest of the network that it has deemed itself synchronized.

Since the devices use two synchronization signals, the cross-correlated received signal at a device can be one of four kinds:

- It contains no synchronization signals at all (this may occur when there are no transmitter devices).
- It only contains synchronization signals using root index u_1 (this occurs when all transmitter devices try to reduce their own synchronization errors);
- It only contains synchronization signals using root index u_2 (this occurs when all the transmitter devices deem themselves to be synchronized).
- It contains a mix of signals using u_1 and u_2 (this occurs when some devices are still reducing their errors while others are deem themselves to be synchronized).

Therefore, a receiver device should be able to reliably detect the presence or absence of the ZC root index u_1 or u_2 within the received signal by performing the cross-correlation in (7), which we re-define as $R_{y_j x^u_\pm}[l, \nu] = \sum_{k\in\mathcal{K}} y_j[k; \nu] x^u_\pm[k-l]^*$ to emphasize u.

Considering the properties of ZC sequences, auto-correlation of two synchronization signals that have the same root index yields a peak value N at lag l, i.e., $|R_{x^{u_1}_\pm x^{u_1}_\pm}[l, \nu]| = N$ [28]; however, cross-correlation of such two signals with different root indices yields $|R_{x^{u_1}_\pm x^{u_2}_\pm}[l, \nu]| = \sqrt{N}$ [29]. Hence, after cross-correlating the received signal in (7) with both $x^{u_1}_\pm[k]$ and $x^{u_2}_\pm[k]$ at the νth clock tick, the jth receiver device calculates the following decision statistic for each $u \in \{u_1, u_2\}$:

$$\psi(\nu, u) = \max_l \left(|R^u_{y_j x_\pm}[l, \nu]|\right) \tag{17}$$

and compares it to a detection threshold λ_{det}. Note that in LTE, ZC sequence length N is 839 [29], and considering

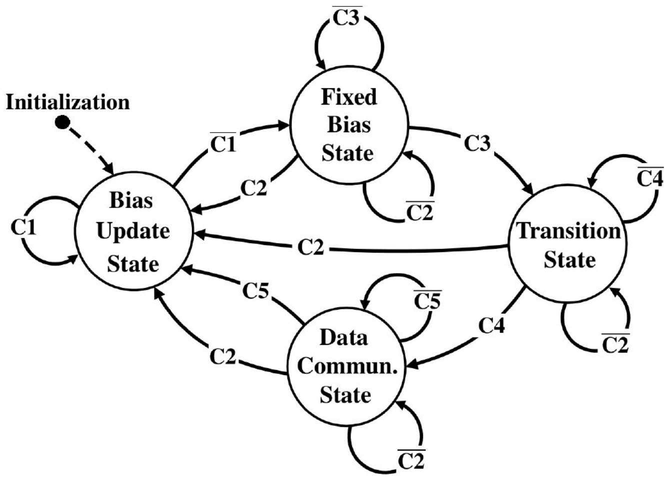


FIGURE 5. State-transition diagram of the proposed protocol. Transitions happen at each clock tick under different conditions; the conditions and their complements are denoted by Ci and $\overline{\text{Ci}}$, respectively, where i ∈ {1, 2, . . . , 5}.

the typical noise levels, we set this detection threshold as $\lambda_{\text{det}} = \frac{N}{2}$ in our work. Accordingly, the receiver makes one of the following four decisions:

$$\begin{aligned} \mathrm{D}^{00}_\nu &: \psi(\nu, u_1) < \lambda_{\text{det}} \ \wedge \ \psi(\nu, u_2) < \lambda_{\text{det}} \\ \mathrm{D}^{10}_\nu &: \psi(\nu, u_1) \geq \lambda_{\text{det}} \ \wedge \ \psi(\nu, u_2) < \lambda_{\text{det}} \\ \mathrm{D}^{01}_\nu &: \psi(\nu, u_1) < \lambda_{\text{det}} \ \wedge \ \psi(\nu, u_2) \geq \lambda_{\text{det}} \\ \mathrm{D}^{11}_\nu &: \psi(\nu, u_1) \geq \lambda_{\text{det}} \ \wedge \ \psi(\nu, u_2) \geq \lambda_{\text{det}} \end{aligned} \tag{18}$$

where $\wedge$ is the logical conjunction and the superscripts of the decisions indicate the presence, i.e., 1, or the absence, i.e., 0, of the ZC root indices u_1 and u_2, respectively. Decision D^{00}_ν corresponds to the case where no synchronization signal is detected. In response, the device re-establishes its transceiver mode for the next tick based on p_{tr} and advances its clock by one clock period. For any of the other three decisions, the receiver device forms its final TO estimate as follows:

$$\widehat{\Delta t_j}[\nu] = \begin{cases} \widehat{\Delta t}^{u_1}_j[\nu], & \mathrm{D}^{10}_\nu \\ \widehat{\Delta t}^{u_2}_j[\nu], & \mathrm{D}^{01}_\nu \\ \frac{1}{2}\left(\widehat{\Delta t}^{u_1}_j[\nu] + \widehat{\Delta t}^{u_2}_j[\nu]\right), & \mathrm{D}^{11}_\nu \end{cases} \tag{19}$$

In the above, the superscript to the weighted average TO estimate $\widehat{\Delta t}^{u_1}_j[\nu]$ or $\widehat{\Delta t}^{u_2}_j[\nu]$ indicates from which ZC root index it is obtained. In the case of D^{11}_ν, the jth device forms its final TO estimate by using the arithmetic mean of its individual TO estimates $\widehat{\Delta t}^{u_1}_j[\nu]$ and $\widehat{\Delta t}^{u_2}_j[\nu]$, since it is consistent with the definition of (9) as a weighted average.

In what follows we describe the proposed protocol over a state-transition diagram as given in Fig. 5 from the point of view of the jth device.

Initialization: When the jth device joins the network, it stores its initial transceiver mode $M^j_{\nu_0}$. Furthermore, the jth device initializes Γ_j, ξ_j and ζ_j, which will be used as error level, stopping and clock skew control counters, respectively, as:

$$\Gamma_j = 0, \ \xi_j = 0 \ \text{ and } \zeta_j = 0 \tag{20}$$

The purpose of using the counters throughout the protocol is to provide a controlled evolution between the states by comparing them to pre-defined thresholds.

A. BIAS UPDATE STATE

In this state, the jth device tries to iteratively estimate its bias to reduce its synchronization error while updating its clock. Therefore, the device runs Algorithm 1 with $u = u_1$ as long as the following condition is satisfied:

C1: $|\widehat{\Delta t_j}[\nu]| \leq |\Delta t_j|_{\min} \vee |\Delta t_j|_{\min} > \lambda_{\text{sync}}$

where $\vee$ is the logical disjunction, $|\Delta t_j|_{\min}$ is the smallest value of the weighted average TO encountered up to the νth iteration and λ_{sync} is the pre-defined synchronization threshold. Note that $|\Delta t_j|_{\min}$ can be less than λ_{sync}. Hence, if the synchronization error stays above the desired threshold or it keeps decreasing in absolute value, then the device does not change its state.

B. FIXED BIAS STATE

In this state, the absolute value of the synchronization error of the device stops decreasing. Therefore, the device stops updating its bias estimate and fixes it to the one that yields the smallest error $|\Delta t_j|_{\min}$ as:

$$\widehat{\beta_j}\,[\nu + 2] = \widehat{\beta_j}[\nu] \tag{21}$$

From then on, the device only uses this bias estimate in Algorithm 1, i.e., (16) is replaced by (21). Meanwhile, the device increases its error level counter Γ_j but only when it operates as a receiver device:

$$\Gamma_j = \Gamma_j + 1, \text{ if } M_\nu^j = \text{RX} \tag{22}$$

In order for a device to leave this state, there are two conditions. The first condition captures the effect of a perturbation in the system, e.g., a new device joining the network, which can be detected by checking for a sudden change in the weighted average TO estimate as follows:

C2: $||\widehat{\Delta t_j}[\nu]| - |\Delta t_j|_{\min}| > \lambda_{\text{sync}}$

In this way, the proposed protocol can operate with dynamic network size.

Whereas the second condition indicates whether or not the device deems itself synchronized. Specifically, the decrease in the weighted average TO of the jth device, i.e., $|\widehat{\Delta t_j}[\nu]|$, might be temporarily due to noise in the TO estimate or some devices leaving the network, hence, the device should not consider itself to be synchronized immediately. Instead, the device observes the change in $|\widehat{\Delta t_j}[\nu]|$ by using condition $\overline{\textbf{C2}}$ and assumes synchronization only after this condition is satisfied for λ_{cons} consecutive clock ticks, that is:

C3: $\Gamma_j \geq \lambda_{\text{cons}}$

C. TRANSITION STATE

In this state the device declares its synchronization status to the network. Although the device still continues to run Algorithm 1, it changes its ZC index as follows:

$$u \leftarrow u_2 \text{ if } M_\nu^j = \text{TX} \tag{23}$$

Therefore, if the device in this state operates as a receiver at the νth clock tick, i.e., $M_\nu^j = \text{RX}$, and detects the presence of the second synchronization signal but not the first one, it assumes that other devices are also synchronized, hence, it increases its stopping counter ξ_j as:

$$\xi_j = \begin{cases} \xi_j + 1, & \text{D}_\nu^{01} \vee \left(\text{D}_\nu^{00} \wedge \xi_j > 0\right) \\ 0, & \text{D}_\nu^{10} \vee \text{D}_\nu^{11} \end{cases} \tag{24}$$

Note that when $(\text{D}_\nu^{00} \wedge \xi_j > 0)$ is satisfied, the receiver device does not detect any signals, i.e., D_ν^{00}, yet it has a non-zero stopping counter. Since the device is in the transition state, this can be interpreted in two ways: either the previously detected devices left the network, or they switched to *Data Communication State*, where they essentially stop their synchronization process. In both cases, the device continues to increase its counter.

In contrast, a device operating as a transmitter at the νth clock tick, i.e., $M_\nu^j = \text{TX}$, cannot detect any signals, therefore, cannot make any decisions about the presence of the synchronization signals. Hence, we propose to incorporate the initial transceiver mode $M_{\nu_0}^j$ to the decision process. Specifically, when a device operates as a transmitter at the νth clock tick and its initial mode was also transmitter, i.e., $M_\nu^j = M_{\nu_0}^j = \text{TX}$, then it increases its stopping counter. However, a transmitter device still needs to obtain decision statistics at the νth clock tick (17), therefore, we propose to rely on the decision statistic from the $(\nu - 1)$th clock tick, where the device was a receiver.

If $M_\nu^j = \text{TX}$:

$$\xi_j = \begin{cases} \xi_j + 1, & (M_{\nu_0}^j = \text{TX}) \wedge \text{D}_{\nu-1}^{01} \\ 0, & (M_{\nu_0}^j = \text{TX}) \wedge (\text{D}_{\nu-1}^{10} \vee \text{D}_{\nu-1}^{11}) \end{cases} \tag{25}$$

Similar to the previous state, perturbations must be detected, i.e., condition **C2**. However, since not every device necessarily has the same synchronization error, there might be some devices that are still trying to reduce their error by broadcasting the first signal with ZC index $u = u_1$ while not triggering condition **C2**. In this case, when the jth device in this state detects the first synchronization signal, it resets its stopping counter, i.e., sets $\xi_j = 0$. Therefore, it waits for the other devices to first synchronize themselves, then switch to the transition state, where they finally broadcast the second synchronization signal.

Note that when proceeding to the communication state, the devices should not lose synchronization with respect to each other. Specifically, this is important to avoid re-initiating the synchronization process for timing-advance communication, which will be explained later. Therefore, the next condition not only allows devices to terminate the synchronization process, ideally at the same time, but also enables them to initiate timing-advance data communication in a distributed manner.

C4: C4.1 $\vee$ **C4.2**

C4.1: $(M_\nu^j = M_{\nu_0}^j = \text{RX}) \wedge ((\xi_j > \lambda_{\text{stop}}) \vee (\text{D}_\nu^{00} \wedge \xi_j > 0))$

C4.2: $(M_\nu^j = M_{\nu_0}^j = \text{TX}) \wedge (\xi_j > \lambda_{\text{stop}})$

The device switches to the data communication state when its stopping counter exceeds the pre-defined stopping threshold λ_{stop} if and only if its current transceiver mode at the νth clock tick, i.e., M_ν^j, is the same as its initial transceiver mode, i.e., $M_{\nu_0}^j$. Specifically, if the device is a receiver (transmitter) and the first (second) sub-condition **C4.1** (**C4.2**) is satisfied, the device stops its synchronization process as a receiver (transmitter). Hence, the device is aware of whether or not its clock is advanced or regressed with respect to others when the synchronization process stops.

Note that in the case of no signal detection with a non-zero stopping counter, the device can safely assume that the contributing transmitter devices have already switched to the data communication state or left the network, hence, it can proceed to the next state.

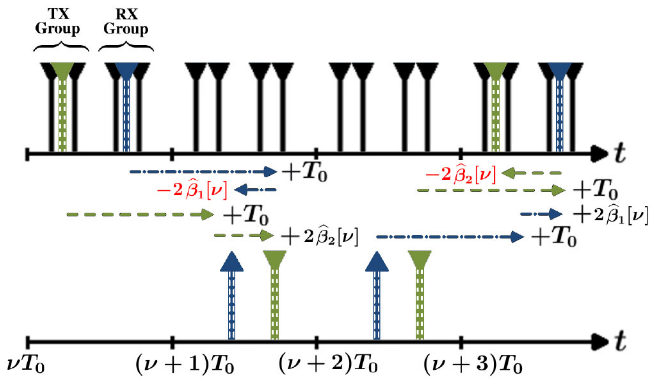

TX Group
RX Group
t
$+T_0$
$-2\widehat{\beta}_1[\nu]$
$-2\widehat{\beta}_2[\nu]$
$+T_0$
$+T_0$
$+2\widehat{\beta}_1[\nu]$
$+2\widehat{\beta}_2[\nu]$
$+T_0$
t
νT_0
$(\nu+1)T_0$
$(\nu+2)T_0$
$(\nu+3)T_0$

FIGURE 6. After synchronization, the devices are aware the group they are in, i.e., TX or RX group. Hence, they can re-arrange their clocks by only using their own bias estimate for timing-advance communication.

D. DATA COMMUNICATION STATE

In this state, the device can conserve energy by only operating as a receiver until it initiates data communication. Therefore, the device sets its transceiver mode to receiver:

$$M_\nu^j \leftarrow \text{RX} \tag{26}$$

and updates its clock similarly to (11). However, if the device anticipates data communication, then it relies on how condition **C4** was satisfied, i.e., whether or not its clock is advanced or regressed at the termination of synchronization process. Specifically, $M_{\nu_0}^j = \text{RX}$ means that the device is in the RX group which is suitable for data reception, while $M_{\nu_0}^j = \text{TX}$ that the device in the TX group which is suitable for data transmission. On the contrary, if the device is in the group that does not match the anticipated communication type, i.e., $M_{\nu_0}^j = \text{RX}$ but the device has data to transmit or $M_{\nu_0}^j = \text{TX}$ but it expects to receive data, then any two devices in this case would need to re-initiate their own synchronization process for proper timing-advance, which would cause a network-wide perturbation. In order to avoid this, the proposed protocol allows such devices to re-arrange their clocks for proper timing-advance without actively transmitting and receiving synchronization signals as follows:

$$t_j[\nu+1] = \begin{cases} t_j[\nu] + \alpha_j T_0 - \widehat{\beta}_j[\nu], M_{\nu_0}^j = \text{RX} \ \wedge \ B_j = 1 \\ t_j[\nu] + \alpha_j T_0 + \widehat{\beta}_j[\nu], M_{\nu_0}^j = \text{TX} \ \wedge \ B_j = 0 \end{cases} \tag{27}$$

where $B_j \in \{0, 1\}$ indicates that the jth device anticipates data transmission or reception, when it is one or zero, respectively. By applying (27), the jth device uses its own bias estimate to adjust its clock to approach the vicinity of the clocks of the opposite group in time as shown in Fig. 6. Note that if the device applies (27), then it can simply revert this procedure to return its original TX or RX group after stopping data communication.

Overall, (27) is useful in terms of extending the achieved synchronization by allowing devices to use timing-advance communication without unnecessarily exchanging timing pulses. Therefore, for the proposed protocol, clock-skew is the only factor affecting re-initialization of the synchronization process. Indeed, in a practical system, the presence of clock skew imposes an upper bound on how long devices can stay synchronized. Since in the communication state, the jth device does not apply the DPLL clock update in (15), its clock might diverge from the clocks of other devices after some period of time due to clock skew. To prevent this, the device keeps track of how long it stays in this state by increasing its clock skew control counter at each clock tick as $\zeta_j \leftarrow \zeta_j + 1$, which is then compared to clock skew control threshold λ_{skew} as follows:

C5: $\zeta_j \geq \lambda_{\text{skew}}$

If ζ_j exceeds the allowed limit, then the device re-initiates its synchronization process by resetting its variables as follows:

$$\Gamma_j = 0, \ \ \xi_j = 0 \text{ and } \zeta_j = 0 \tag{28}$$

$$|\Delta t_j|_{\min} = \widehat{\Delta t_j}[\nu] \tag{29}$$

$$\widehat{\beta}_j[\nu] = \widehat{\beta}^{\text{init}} \tag{30}$$

$$u \leftarrow u_1 \tag{31}$$

$$M_\nu^j \leftarrow M_{\nu_0}^j \tag{32}$$

We note that the condition **C5** is different than **Initialization**, where the devices are setting all their variables to 0 and determining $M_{\nu_0}^j$ randomly. However, in **C5**, the jth device sets its current transceiver mode to its initial transceiver mode $M_{\nu_0}^j$. Hence, if the devices switch to the bias update state due to the condition **C5**, then re-synchronization should take much less time since the effect of clock skew cannot cause drastic deviations on the device clocks.

E. COMPLEXITY ANALYSIS

To derive the complexity analysis of the proposed protocol, we first analyze the required operations in Algorithm 1. When a device operates as a transmitter at the νth clock tick, i.e., $M_\nu^j = \text{TX}$, constructing the synchronization signal as in (3) requires $\mathcal{O}(2N)$ operations. Then, the clock update in (11) only takes one multiplication and one addition, hence,

TABLE 1. System parameters.

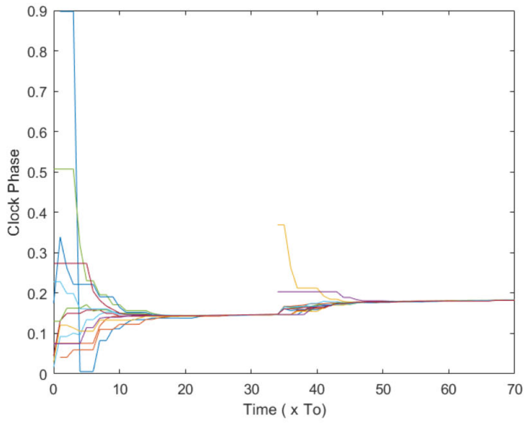
Parameter Description	Symbol	Value
Number of Devices	J	$2, 5, 8, 14$
Scaling Term of DPLL	ϵ	1
Zadoff-Chu Index	u_1, u_2	$7, 13$
Zadoff-Chu Sequence Length	N	839
Clock Period	T_0	$1\ ms$
Pulse Spacing	T_p	$0.1\ \mu s$
Sampling Period	T_s	$3\ ns$
Maximum Network Distance	d	500 m
Operating Frequency	f	2 GHz
Bias Estimate Initialization	$\widehat{\beta}^{\text{init}}$	$.86\ \mu s$
Step Size Initialization	γ^{init}	$33\ ns$
Step Size Slope	a	$.98$
Step Size Increment	b	$3\ ns$
Probability of Being a Transmitter	p_{tr}	$\{0.1, 0.5, 0.9\}$
Signal Detection Threshold	λ_{det}	$\frac{N}{2}$
Synchronization Error Threshold	λ_{sync}	$1.5\ \mu s$
Consecutive Clock Tick Threshold	λ_{cons}	2
Clock Skew Control Threshold	λ_{skew}	10
Stopping Threshold	λ_{stop}	2
Number of Resolvable Paths	P	4
Rayleigh Fading Scaling Parameter	-	1
Rician Fading Noncentrality Parameter	-	1
Rician Fading Scaling Parameter	-	1

the complexity is relatively small. If a device operates as a receiver at the νth clock tick, i.e., $M_\nu^j = \text{RX}$, it first estimates the average TO in (9), which requires computing a cross-correlation as in (7), and then forming two different preliminary TO estimates as given by (8). Hence, the total operations needed for (7) have $\mathcal{O}(2K)$ complexity related to complex multiplications, whereas each preliminary TO estimate, i.e., $q_j^{\pm}[\nu]$, in (8) takes $\mathcal{O}(8K^2)$ operations. Therefore, the overall computation complexity of Algorithm 1 at the νth clock tick is $\mathcal{O}(16K^2+2N)$. Now considering the overall computation complexity of the proposed protocol, forming the final average TO estimate in (19) doubles the amount of operations due to utilization of two distinct ZC sequences, hence, the computational complexity at the νth clock tick becomes $\mathcal{O}(32K^2+2N)$. However, the rest of the computation complexity of the proposed protocol is relatively small since it only requires memory registers and comparisons between the assigned values.

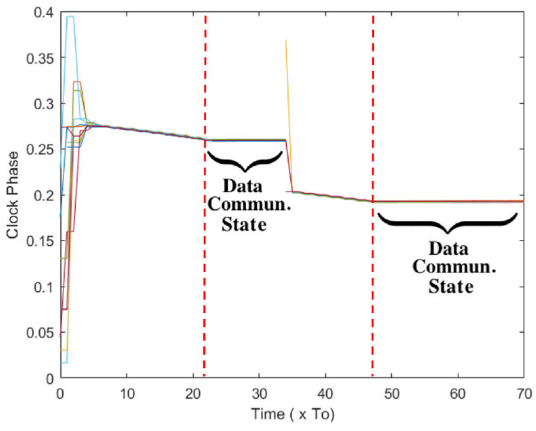
V. SIMULATION RESULTS

In this section, the proposed synchronization protocol is evaluated by means of computer simulations based on METIS 5G channel models and under different conditions of operation, i.e., clock skew, number of devices, and network topologies, including full mesh and partial mesh.

The complete synchronization protocol for a dense stationary D2D network is implemented in MATLAB. Unless otherwise is specified, we consider a network of 14 devices, where 12 of them initialize at $\nu = 0$ with 2 more devices joining the network at $\nu = 33$. We use a channel model according to the Manhattan grid scenario in [30] with Rician fading for the line of sight path (if present) and Rayleigh fading for the other paths. The SNR at the input of the correlator is fixed at 15dB. The rest of the system parameters are chosen accordingly from [31] and given in Table 1.



(a) Random transceiver mode [16] ($p_{\text{tr}} = 0.5$).



(b) Proposed synchronization protocol ($p_{\text{tr}} = 0.5$).

FIGURE 7. A single realization of clock phase evolution with a perturbation at the 33rd clock tick.

A. CLOCK PHASE CONVERGENCE

First, we study how fast our protocol reduces the synchronization error compared to [16] under the exact same network conditions. In Fig. 7(a) we show a single realization of the clock phase evolution for the algorithm in [16] where the devices randomly choose their transceiver mode based on $p_{\text{tr}} = 0.5$. Similarly in Fig. 7(b), the devices employ the proposed protocol with $p_{\text{tr}} = 0.5$. We note that the algorithm in [16] transmits synchronization signals for the duration of the experiment, while the proposed protocol only transmits until *Data Communication State* as illustrated in Fig. 7(b). As we can see, during these states, the proposed protocol not only reduces the difference in relative clock phases faster but also achieves a smaller deviation. Furthermore, our protocol quickly reacts to the perturbation that occurs at $\nu = 33$ when new devices join the network, and compensates for it better than [16].

Importantly, by using the proposed protocol, the devices are aware of the global synchronization status which allows

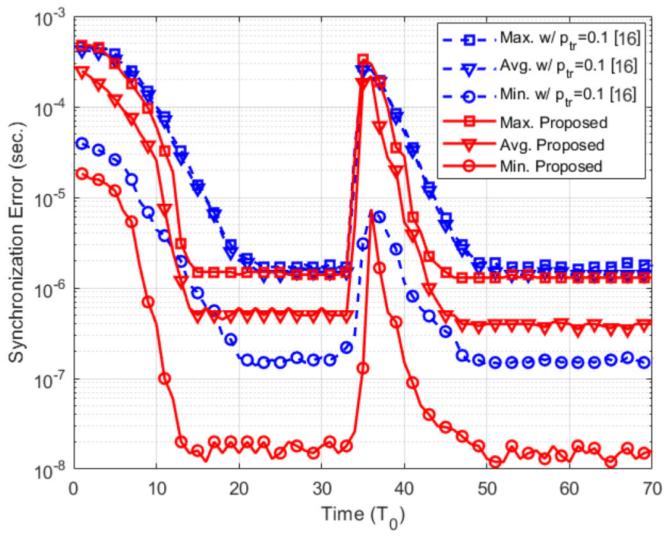


FIGURE 8. Synchronization error comparison; the proposed protocol vs the random transceiver mode in [16] when $J = 14$.

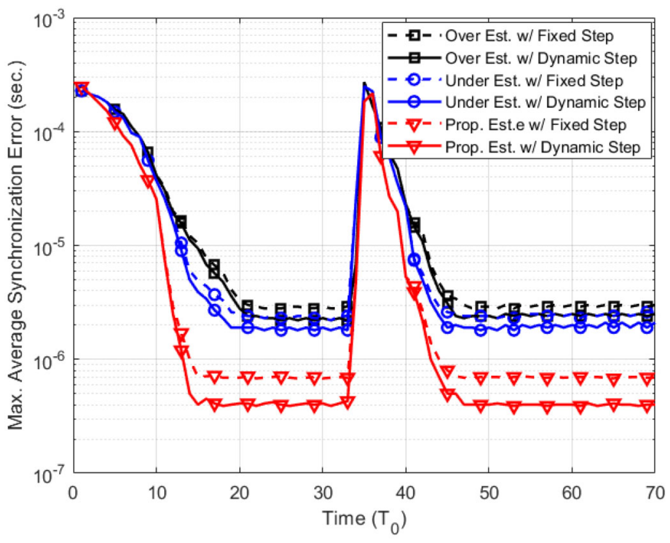


FIGURE 9. Synchronization performance for dynamic and fixed step size under different bias initializations.

them to terminate the synchronization process simultaneously at $\nu = 22$ and proceed to data communication state as shown in Fig. 7(b). However, when using the algorithm in [16], the devices are not aware of the synchronization status of others, hence, they cannot simultaneously terminate this process and proceed to data communication as intended.

B. SYNCHRONIZATION PERFORMANCE BEFORE DATA COMMUNICATION

To quantify the synchronization performance, we introduce three synchronization performance metrics: maximum, minimum and average synchronization errors at a given clock tick ν, which take propagation delays into account. These metrics are defined as follows:

$$\Delta t^{max}_{\text{sync}}[\nu] = \max_{(\eta,i,j)\in\mathcal{S}_\nu} \left|t_i[\eta] + \tau_{ij1} - t_j[\nu]\right| \quad (33)$$

$$\Delta t^{min}_{\text{sync}}[\nu] = \min_{(\eta,i,j)\in\mathcal{S}_\nu} \left|t_i[\eta] + \tau_{ij1} - t_j[\nu]\right| \quad (34)$$

$$\Delta t^{avg}_{\text{sync}}[\nu] = \max_{j\in\mathcal{R}_\nu} \left|\frac{1}{|\mathcal{A}^\nu_j|} \sum_{(\eta,i)\in\mathcal{A}^\nu_j} \left(t_i[\eta] + \tau_{ij1}\right) - t_j[\nu]\right| \quad (35)$$

In (33) and (34), $\mathcal{S}_\nu = \bigcup_{\eta\in\mathcal{V}_\nu}\{(\eta, i, j)\colon i \in \mathcal{T}_\eta,\ j \in \mathcal{R}_\nu, |t_i[\eta] + \tau_{ij1} - t_j[\nu]| \leq \frac{T_0}{2}\}$ is the set of triplets containing the indices of transmitters from the ηth clock tick that are contributing to the received signal of the jth device at the νth clock tick. Furthermore in (35), we consider the maximum of the average synchronization error, hence, the set $\mathcal{A}^\nu_j = \bigcup_{\eta\in\mathcal{V}_\nu}\{(\eta, i)\colon i \in \mathcal{T}_\eta, |t_i[\eta] + \tau_{ij1} - t_j[\nu]| \leq \frac{T_0}{2}\}$ includes the index of the transmitter devices from the ηth clock tick detected by the jth device at the νth clock tick.

In Fig. 8 we compare the synchronization performance of the proposed protocol to [16] using these metrics. Since there is no stopping condition in [16], we set $\lambda_{\text{cons}} = \infty$ in the proposed protocol for a fair comparison, hence, the devices only operate in *Bias Update State* and *Fixed Bias State*. Furthermore, in [16], it is stated that the lower the p_{tr}, the better the synchronization performance. We confirmed this observation by trying several values of $p_{\text{tr}} \in \{0.1, 0.5, 0.9\}$ and we found that $p_{\text{tr}} = 0.1$ indeed yields the best performance. So, in Fig. 8 we use $p_{\text{tr}} = 0.1$ for the algorithm in [16]. We see that the proposed protocol reduces the synchronization error and reaches the steady-state much faster, while outperforming [16] in all performance metrics, i.e., (33), (34), (35). In addition, the proposed protocol adapts to the change in the network size, which occurs at $\nu = 33$, more rapidly. Furthermore, we can also observe that there is no fluctuation in the synchronization error as the devices switch from *Bias Update State* to *Fixed Bias State*, which shows that (19) produces reliable TO estimates.

In Fig. 9, we investigate the sensitivity of the proposed bias estimation in (16) under various initialization of $\widehat{\beta}^{\text{init}}_j = \frac{d}{c}$ by assuming ($\pm$%80) over and underestimates of the average distance d. We further study the performance of the dynamic step size $\gamma_j[\nu] = a\gamma_j[\nu - 2] + b$ and compare it to a fixed step size γ_j. As shown in Fig. 9, the proposed protocol still provides an average synchronization error of $3\mu s$ even in the presence of severe errors in the estimates of d. Furthermore, the dynamic step size leads to faster error reduction and a lower error floor.

In Fig. 10, we verify the analysis of the synchronization error reduction by Monte-Carlo simulations for different number of devices. It is shown that the analysis tracks well the synchronization error during the steady-state regime. Not surprisingly, the synchronization performance deteriorates as the number of devices increases. However, the performance is better with two devices compared to the setting for multiple devices since the TO estimate in (9) only includes the contribution from a single transmitter device.

Next, we consider the performance of the proposed protocol for the partial mesh topology as depicted in Fig. 1, where

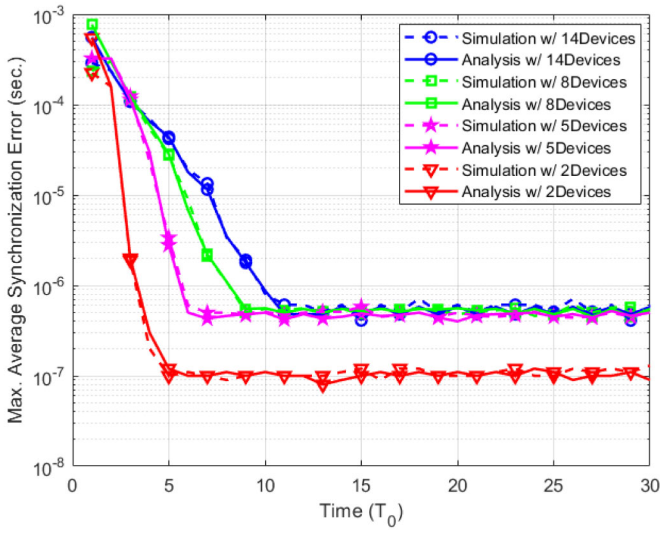


FIGURE 10. Verification of the analysis with different number of devices.

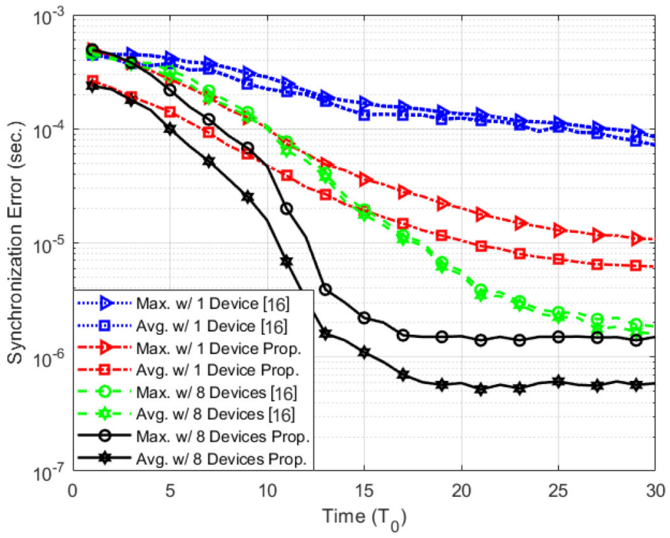


FIGURE 11. Max. and Avg. synchronization performance based on (33) and (35) for a partial mesh topology for one and eight common devices.

two physically separated device groups are synchronizing through the common devices that act as relaying nodes. In Fig. 11, we compare our protocol to [16] with $p_{\text{tr}} = 0.1$. We note that when the number of common devices decreases, the overall synchronization performance degrades as expected. More specifically, if the common devices do not operate as transmitters, which is more likely to happen in the case of random transceiver mode in [16], especially if p_{tr} is low, then physically separated devices cannot synchronize themselves. On the contrary, at each clock tick, the devices always alternate their transceiver mode in the proposed protocol, hence, even with a single common device, our protocol attains lower steady-state error.

C. TIMING ERROR DURING DATA COMMUNICATION

In Fig. 12, we consider the termination of the synchronization process to let the devices initiate timing-advance communication in a distributed manner. Thus, unlike Fig. 8, we set $\lambda_{\text{cons}} = 2$ so the devices leave *Fixed Bias State* after 2 consecutive clock ticks, and proceed to the next state, i.e., *Transition State*, where they will eventually terminate the synchronization process. We note that the performance metrics in (33)-(35) are not useful after the devices terminate synchronization since $\mathcal{T}_\nu = \emptyset$. Therefore, we investigate the timing error between the communicating devices once they initiate distributed timing-advance communication according to (27). To that end, we introduce two performance metrics. The first one is the maximum timing error over all pairs of potentially communicating devices:

$$\Delta t^{max}_{comm}[\nu] = \max_{(\eta,i,j)\in\mathcal{C}_\nu} \left| t_i[\eta] - \kappa_i \widehat{\beta}_i[\eta] + \tau_{ij1} - t_j[\nu] + \vartheta_j \widehat{\beta}_j[\nu] \right| \quad (36)$$

where $\mathcal{C}_\nu = \{(\eta,i,j) \in \mathcal{V}_\nu \times \mathcal{J} \times \mathcal{J} : i \neq j \wedge |t_i[\eta] + \tau_{ij1} - t_j[\nu]| \leq \frac{T_0}{2}\}$. In (36), the functions κ_i and ϑ_j indicate whether a device regresses or advances its clock, respectively, according to (27) and are defined as follows:

$$\kappa_i = \begin{cases} 0, & \text{if } M^i_{\nu 0} = \text{TX} \\ 1, & \text{if } M^i_{\nu 0} = \text{RX} \end{cases}$$

$$\vartheta_j = \begin{cases} 0, & \text{if } M^j_{\nu 0} = \text{RX} \\ 1, & \text{if } M^j_{\nu 0} = \text{TX} \end{cases}$$

The set $\mathcal{C}_\nu$ in (36) contains all triplets (η, i, j) where j is the index of a receiver device at clock tick ν, and i is the index of a device that might transmit to device j from the clock tick $\eta \in \mathcal{V}_\nu$.

The second metric is the maximum over all receivers of the average timing error between a given receiver and all potential transmitters:

$$\Delta t^{avg}_{comm}[\nu] = \max_{j\in\mathcal{R}_\nu} \left| \frac{1}{|\mathcal{H}^\nu_j|} \sum_{(\eta,i)\in\mathcal{H}^\nu_j} (t_i[\eta] - \kappa_i \widehat{\beta}_i[\eta] + \tau_{ij1}) - t_j[\nu] + \vartheta_j \widehat{\beta}_j[\nu] \right| \quad (37)$$

where the set $\mathcal{H}^\nu_j = \{(\eta, i) \in \mathcal{V}_\nu \times \{\mathcal{J} - \{j\}\} : |t_i[\eta] + \tau_{ij1} - t_j[\nu]| \leq \frac{T_0}{2}\}$ includes the pairs (η, i) where i is the index of a device that might transmit to device j from the clock tick $\eta \in \mathcal{V}_\nu$.

Note that in Fig. 12, we compare the timing error in two cases: with and without clock skew. We see that the timing error slightly increases in both metrics when the devices stop synchronization and use the clock arrangements in (27) as they switch to data communication state at the 25th and 56th clock ticks. Specifically, for the maximum timing error, the degradation is higher since the devices synchronized their clocks based on the weighted average TO and their bias estimates are the approximation of the average propagation delay with respect to the multiple transmitters.

In addition, clock skew increases the timing error since the devices do not employ DPLL update in data communication state. In Table 2, we show the change in the timing error due to the clock skew for different number of devices. We

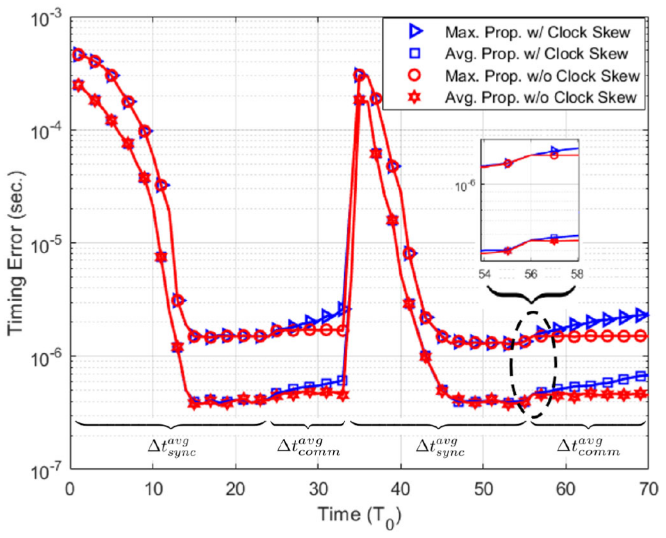


FIGURE 12. **Transitioning from synchronization to data communication in full mesh topology.**

TABLE 2. **Timing-advance data communication performance with clock skew.**

Comm. Performance	Sync. Stopped	After $1T_0$	After $5T_0$	After $9T_0$
Max Sync. Error (14 Dev.)	$1.5\mu s$	$1.7\mu s$	$1.98\mu s$	$2.6\mu s$
Avg Sync. Error (14 Dev.)	$.41\mu s$	$.48\mu s$	$.54\mu s$	$.61\mu s$
Max Sync. Error (8 Dev.)	$1.35\mu s$	$1.58\mu s$	$1.79\mu s$	$2.1\mu s$
Avg Sync. Error (8 Dev.)	$.39\mu s$	$.43\mu s$	$.5\mu s$	$.59\mu s$
Max Sync. Error (2 Dev.)	$.1\mu s$	$.13\mu s$	$.18\mu s$	$.21\mu s$
Avg Sync. Error (2 Dev.)	$.1\mu s$	$.13\mu s$	$.18\mu s$	$.21\mu s$

assume 20ppm crystal accuracy in our simulations. Here, the second column indicates the timing error right before the synchronization process stops and the third column is the timing error when data communication state starts. As time elapses without DPLL clock updates, the timing error increases. Specifically, after $9T_0$, the timing error is more than 1.5 times the error achieved at the end of the synchronization process. Hence, by using Table 2 and assuming that the synchronization error between any device should not exceed $2.5\mu s$, we should set $\lambda_{\text{skew}} = 10$ to re-initiate the synchronization process after $10T_0$. However, if there would not be clock skew, then the devices would use timing-advance data communication without needing to re-initiate the synchronization process.

VI. CONCLUSION

In this article, we proposed a novel, fully-distributed pulse-based synchronization protocol for half-duplex D2D communications in 5G networks, specifically for the out-of-coverage scenario. The new protocol allows devices to first synchronize themselves in a distributed manner and then simultaneously proceed to timing-advance data communication by maintaining the achieved synchronization. Our protocol also rapidly adapts the changes in the network size by allowing new devices to easily synchronize themselves to the network without disrupting the ongoing synchronization process. Finally, we note that the proposed protocol is not limited to out-of-coverage D2D communications and it can be implemented for any distributed system.

APPENDIX A
REDUCTION OF TIMING OFFSET IN MULTI DEVICE SETTING

We consider distributed multi-device pulse-based synchronization over multipath channels with half-duplex technology. Consequently, there is no single timing reference for the clocks of the devices to converge to. Therefore, we analyze the reduction of timing offset, which is interpreted as the synchronization error by the devices. We assume each device runs the proposed protocol and, based on Algorithm 1, the devices keep alternating their transceiver mode at each clock tick ν after initialization. We will use the index j to denote a receiver device at clock tick ν and i to denote a transmitter device. These devices become transmitters and receivers, respectively, at the next clock tick, i.e., $j \in \mathcal{R}_\nu = \mathcal{T}_{\nu+1}$ and $i \in \mathcal{T}_\nu = \mathcal{R}_{\nu+1}$. We further assume that the network remains constant, i.e., no new devices join or leave the network, and channels are time-invariant. Therefore, the cardinality of the sets is $|\mathcal{T}_\nu| = T$ and $|\mathcal{R}_\nu| = R$. For simplicity of the analysis and with no loss of generality, we assume that the signal contributions are coming from the same time slot of the receiver. Hence, we have $\eta = \nu$ and the set of pairs formed by the index of transmitter devices and the path indices contributing to the received signal of the jth receiver device becomes $\mathcal{D}_j^{\nu,\nu}$. In addition, we use the superscript $\mathcal{T}$ and $\mathcal{R}$ on the device clock models to indicate their transceiver mode at the corresponding clock ticks. Thus, in the high SNR region, the weighted average TO estimate over multipath channels at the device $j \in \mathcal{R}_\nu$ equals to:

$$\widehat{\Delta t}_j[\nu] = \sum_{(i,p)\in\mathcal{D}_j^{\nu,\nu}} \mu_{ijp}\big(t_i^{\mathcal{T}}[\nu] + \tau_{ijp}\big) - t_j^{\mathcal{R}}[\nu]. \tag{38}$$

For the ith device $\widehat{\Delta t}_i[\nu+1]$ can be defined similarly. Then, the average TO estimate of the jth device when it becomes a receiver again is given as follows:

$$\begin{aligned}
\widehat{\Delta t}_j[\nu+2] &= \sum_{(i,p)\in\mathcal{D}_j^{\nu+2,\nu+2}} \mu_{ijp}\big(t_i^{\mathcal{T}}[\nu+2] + \tau_{ijp}\big) - t_j^{\mathcal{R}}[\nu+2] \\
&= \sum_{(i,p)\in\mathcal{D}_j^{\nu+2,\nu+2}} \mu_{ijp}\big(t_i^{\mathcal{R}}[\nu+1] + \alpha_i T_0 - 2\widehat{\beta}_i[\nu+1] \\
&\quad + \widehat{\Delta t}_i[\nu+1] + \tau_{ijp}\big) - t_j^{\mathcal{T}}[\nu+1] - \alpha_j T_0 \\
&= \sum_{(i,p)\in\mathcal{D}_j^{\nu+2,\nu+2}} \mu_{ijp}\big(t_i^{\mathcal{T}}[\nu] + 2\alpha_i T_0 - 2\widehat{\beta}_i[\nu+1] \\
&\quad + \widehat{\Delta t}_i[\nu+1] + \tau_{ijp}\big) - t_j^{\mathcal{R}}[\nu] - 2\alpha_j T_0 + 2\widehat{\beta}_j[\nu] - \widehat{\Delta t}_j[\nu]
\end{aligned} \tag{39}$$

where $t_i^{\mathcal{T}}[\nu+2]$ and $t_j^{\mathcal{R}}[\nu+2]$ are obtained from (15) and (11) at the clock tick $(\nu+1)$, respectively, knowing that the receiver device becomes transmitter and vice versa (see Section III-A). We have $\mathcal{D}_j^{\nu,\nu} = \mathcal{D}_j^{\nu+2,\nu+2}$, and using (38), the weighted average TO estimate in (39) is further simplified to:

$$\widehat{\Delta t}_j[\nu+2] = \sum_{(i,p)\in\mathcal{D}_j^{\nu+2,\nu+2}} \mu_{ijp}\big(\widehat{\Delta t}_i[\nu+1] - 2\widehat{\beta}_i[\nu+1] + 2\alpha_i T_0\big) + 2\widehat{\beta}_j[\nu] - 2\alpha_j T_0 \quad (40)$$

Similarly, the weighted average TO estimate at the ith device is:

$$\widehat{\Delta t}_i[\nu+3] = \sum_{(j,p)\in\mathcal{D}_i^{\nu+3,\nu+3}} \mu_{ijp}\big(\widehat{\Delta t}_j[\nu+2] - 2\widehat{\beta}_j[\nu+2] + 2\alpha_j T_0\big) + 2\widehat{\beta}_i[\nu+1] - 2\alpha_i T_0 \quad (41)$$

where it is simplified as in (40) by using $\widehat{\Delta t}_i[\nu+1]$ and for the sets $\mathcal{D}_i^{\nu+1,\nu+1} = \mathcal{D}_i^{\nu+3,\nu+3}$.

To track the error at the devices when they are receivers, we can generalize (40) for the device $j \in \mathcal{R}_\nu$ as follows:

$$\mathbf{y}[\nu] = \mathbf{W}(\mathbf{x}[\nu-1] - 2\mathbf{r}[\nu-1] + 2\mathbf{a}T_0) + 2\mathbf{p}[\nu-2] - 2\mathbf{b}T_0 \quad (42)$$

whereas the generalization of (41) for the device $i \in \mathcal{R}_{\nu+1}$ is given as:

$$\mathbf{x}[\nu+1] = \mathbf{V}(\mathbf{y}[\nu] - 2\mathbf{p}[\nu] + 2\mathbf{b}T_0) + 2\mathbf{r}[\nu-1] - 2\mathbf{a}T_0 \quad (43)$$

where $\mathbf{y}[\nu]$ and $\mathbf{x}[\nu+1]$ are the vectors that contain the weighted average TO estimates of the receiver devices in the sets $\mathcal{R}_\nu$ and $\mathcal{R}_{\nu+1}$, respectively. Here, the notation $[\,\cdot\,]^\top$ is the transpose of a vector. In addition, $\mathbf{W}$ and $\mathbf{V}$ are the matrices with compatible dimensions that contain the normalized channel weights of the jth and the ith devices, i.e., μ_{ijp} and μ_{jip}, respectively. Note that the row sums of $\mathbf{W}$ and $\mathbf{V}$ are normalized to one and since their product is a square matrix, it becomes a right stochastic matrix [32]. This feature will be used later in the proof. Furthermore, $\mathbf{p}[\nu]$ and $\mathbf{b}$ are the vectors that contain the biases and clock skews of the devices in $\mathcal{R}_\nu$, whereas $\mathbf{r}[\nu-1]$ and $\mathbf{a}$ are the vectors that contain the biases and clock skews of the devices in $\mathcal{R}_{\nu-1}$.

For simplicity of the analysis, we assume fixed step sizes as in (16), hence, online bias estimates are generalized for both $j \in \mathcal{R}_\nu$ and $i \in \mathcal{R}_{\nu+1}$ as follows:

$$\mathbf{p}[\nu] = \mathbf{p}[\nu-2] + \mathbf{m}\odot\mathrm{sgn}(\mathbf{y}[\nu-2])$$
$$\mathbf{r}[\nu+1] = \mathbf{r}[\nu-1] + \mathbf{n}\odot\mathrm{sgn}([\mathbf{x}[\nu-1]) \quad (44)$$

where $\odot$ is the Hadamard product, $\mathbf{m}$ and $\mathbf{n}$ are the vectors with compatible lengths, which contain the step sizes of the jth and the ith devices in the sets $\mathcal{R}_\nu$ and $\mathcal{R}_{\nu+1}$, respectively,

To continue the analysis, we assume that the synchronization error of each device is greater than the synchronization threshold chosen in the network, i.e., $|\mathbf{y}[\nu]| > \lambda_{\text{sync}}$ and $|\mathbf{x}[\nu+1]| > \lambda_{\text{sync}}$, where the absolute values and comparisons are element-wise. In other words, **C1** is satisfied, hence, the devices should try to decrease their errors until $\overline{\mathbf{C1}}$ is satisfied, where they stop updating their bias estimates and maintain the reduced error level.

In order to show the reduction in absolute value of the synchronization error, we compare the weighted average TO estimates at two consecutive clock ticks, i.e., $|\mathbf{y}[\nu+2]|$ and $|\mathbf{y}[\nu]|$ as ν increases. Hence, by using (43) in (42) at the $(\nu+2)$th clock tick, we obtain the following:

$$\begin{aligned}\mathbf{y}[\nu+2] &= \mathbf{W}(\mathbf{x}[\nu+1] - 2\mathbf{r}[\nu+1] + 2\mathbf{a}T_0) + 2\mathbf{p}[\nu] - 2\mathbf{b}T_0\\ &= \mathbf{WVy}[\nu] - 2(\mathbf{WV}-\mathbf{I})\mathbf{p}[\nu] - 2\mathbf{Wn}\odot\mathrm{sgn}(\mathbf{x}[\nu-1])\\ &\quad +2(\mathbf{WV}-\mathbf{I})\mathbf{b}T_0 \end{aligned}\quad (45)$$

where $\mathbf{I}$ is the identity matrix with compatible size. By subtracting $\mathbf{y}[\nu]$ from the both sides, we further obtain:

$$\begin{aligned}\mathbf{y}[\nu+2] &= \mathbf{y}[\nu] + (\mathbf{WV}-\mathbf{I})\overbrace{(\mathbf{y}[\nu] - 2\mathbf{p}[\nu] + 2\mathbf{b}T_0)}^{\mathbf{q}[\nu]} \\ &\quad +2\mathbf{W}(\mathbf{r}[\nu-1] - \mathbf{r}[\nu+1])\\ &= \mathbf{y}[\nu] + (\mathbf{WV}-\mathbf{I})\mathbf{q}[\nu] - 2\mathbf{Wn}\odot\mathrm{sgn}(\mathbf{x}[\nu-1]).\end{aligned}\quad (46)$$

The multiplication of $(\mathbf{WV}-\mathbf{I})$, which is a zero row-sum matrix, with a vector that has identical elements yields a zero vector. In this case, if all the elements in $\mathbf{q}[\nu]$ approaches the same values as ν increases, then $(\mathbf{WV}-\mathbf{I})\mathbf{q}[\nu] \rightarrow \mathbf{0}$, which is a zero vector. In order to prove that, we can re-arrange $\mathbf{q}[\nu]$ by using (42) and (43) through recursive iterations as follows:

$$\begin{aligned}\mathbf{q}[\nu] = \overbrace{\mathbf{WVWV}\overbrace{(\mathbf{y}[\nu-4] - 2\mathbf{p}[\nu-4] + 2\mathbf{b}T_0)}^{\mathbf{q}_2[\nu]}}^{\mathbf{q}_1[\nu]} &\\ -2\mathbf{WVWm}\odot\mathrm{sgn}(\mathbf{x}[\nu-5])&\\ -2\mathbf{WVn}\odot\mathrm{sgn}(\mathbf{y}[\nu-4])&\\ -2\mathbf{Wm}\odot\mathrm{sgn}(\mathbf{x}[\nu-3])&\\ -2\mathbf{n}\odot\mathrm{sgn}(\mathbf{y}[\nu-2])&\end{aligned}\quad (47)$$

Note that by multiplying the same stochastic matrices recursively, i.e., $\mathbf{WVWV}\cdots$, we obtain a right stochastic matrix that has identical elements in each column. Then, multiplication of $\mathbf{WVWV}\mathbf{q}_2[\nu] = \mathbf{q}_1[\nu] \approx [q_1[\nu], q_1[\nu], \ldots, q_1[\nu]]^\top$ approximates a vector that has identical elements. Thus, $(\mathbf{WV}-\mathbf{I})\mathbf{q}_1[\nu] \approx \mathbf{0}$. In addition, the remaining terms in $\mathbf{q}[\nu]$ include the fixed step sizes, which are the same for each device, and the sign of the error remains unchanged.[1] Therefore, they can be re-arranged such that -2$(\mathbf{WV}-\mathbf{I})\mathbf{z} = \mathbf{0}$, where $\mathbf{z} = \mathbf{Wm}\odot\mathrm{sgn}(\mathbf{x}[\nu-5]) = \mathbf{Wm}\odot\mathrm{sgn}(\mathbf{x}[\nu-3])$. Hence, we can conclude that $(\mathbf{WV}-\mathbf{I})\mathbf{q}[\nu] \rightarrow \mathbf{0}$ as ν increases and we left with the following:

$$\mathbf{y}[\nu+2] = \mathbf{y}[\nu] - 2\mathbf{Wn}\odot\mathrm{sgn}(\mathbf{x}[\nu-1]) \quad (48)$$

Note that sign of $\mathbf{y}[\nu]$ is dominated by $\mathbf{Wx}[\nu-1]$ as given in (42). Then, $\mathrm{sgn}(\mathbf{y}[\nu]) = \mathrm{sgn}(\mathbf{Wx}[\nu-1])$, or equivalently $\mathrm{sgn}(\mathbf{x}[\nu-1]) = \mathrm{sgn}(\mathbf{W}^\dagger\mathbf{y}[\nu])$, where $\mathbf{W}^\dagger$ is the pseudo-inverse of $\mathbf{W}$. By multiplying both sides in (48) with $\mathbf{W}^\dagger$,

1. Synchronization error must be decreased to zero before changing its sign

we obtain the following:

$$\mathbf{y}'[\nu+2] = \mathbf{y}'[\nu] - 2\mathbf{n} \odot \operatorname{sgn}\big(\mathbf{y}'[\nu]\big) \tag{49}$$

where $\mathbf{y}'[\nu] = \mathbf{W}^{\dagger}\mathbf{y}[\nu]$. Since the step-sizes are always positive, we can conclude that the quantity in (49) is reducing by $2\mathbf{n}$ at each clock tick. In other words, the synchronization error decreases in absolute value, that is $|\mathbf{y}'[\nu+2]| < |\mathbf{y}'[\nu]|$, where the vectors are compared element-wise.

Finally, when the desired synchronization error is achieved, i.e., **C1** is satisfied, the devices switch to *Fixed Bias State*, hence, stop updating their bias estimates. In this case, $\mathbf{r}[\nu+1] = \mathbf{r}[\nu-1]$ or equivalently the term $\mathbf{n} \odot \operatorname{sgn}(\mathbf{x}[\nu-1])$ is no longer present in (44). Hence, from (48), we can conclude that the synchronization errors are reached a steady-state level that is smaller than or equal to the pre-defined synchronization error, i.e., $\mathbf{y}[\nu+2] = \mathbf{y}[\nu] \leq \lambda_{\text{sync}}$ as $\nu \to \infty$.

APPENDIX B
RATE OF TO REDUCTION

We consider a simplified scenario consisting of two devices labeled as D1 and D2 communicating over a flat reciprocal channel with propagation delay τ. We assume that the devices follow the alternating transceiver mode and based on p_{tr}, D1 operates as a transmitter, whereas D2 is a receiver at the νth clock tick. We further assume that the relative clock skew is negligible, i.e., $\Delta\alpha_{12} = \alpha_1 - \alpha_2 = 0$. Hence, the first TO estimate of D2 is equal to $\widehat{\Delta t}_{12}[\nu] = t_1[\nu] - t_2[\nu] + \tau$. However, $\widehat{\Delta t}_{12}[\nu]$ is the initialization error due to misaligned clock phases. To observe the effect of the protocol, we consider the next TO estimate of D2, which occurs at the $(\nu+2)$th clock tick (see Section III-A) as follows:

$$\begin{aligned}\widehat{\Delta t}_{12}[\nu+2] &= t_1[\nu+2] - t_2[\nu+2] + \tau \\ &= t_1[\nu+1] + \alpha_1 T_0 + \widehat{\Delta t}_{21}[\nu+1] - 2\widehat{\beta}_1[\nu+1] \\ &\quad - t_2[\nu+1] - \alpha_2 T_0 + \tau \\ &= t_1[\nu+1] - t_2[\nu+1] - \tau + \widehat{\Delta t}_{21}[\nu+1] \\ &\quad + 2\tau - 2\widehat{\beta}_1[\nu+1] \\ &= 2\big(\tau - \widehat{\beta}_1[\nu+1]\big)\end{aligned} \tag{50}$$

Note that the corresponding clock simplifications, i.e., $t_1[\nu+2]$ and $t_2[\nu+2]$, are derived respectively from (11) and (15) for a transmitter and a receiver device when $\epsilon = 1$. We further note that $\widehat{\beta}_1[\nu+1] = \widehat{\beta}^{\text{init}}$ and without loss in generality, we assume $\widehat{\Delta t}_{12}[\nu+2] > 0$. Similarly, at the $(\nu+4)$th clock tick, D2 estimates its TO again and it is given by

$$\begin{aligned}\widehat{\Delta t}_{12}[\nu+4] &= 2\big(\tau - \widehat{\beta}_1[\nu+3]\big) \\ &= 2\big(\tau - \widehat{\beta}_1[\nu+1] - \gamma \operatorname{sgn}\big(\widehat{\Delta t}_{21}[\nu+3]\big)\big) \\ &= \widehat{\Delta t}_{12}[\nu+2] - 2\gamma \operatorname{sgn}\big(\widehat{\Delta t}_{21}[\nu+3]\big)\end{aligned} \tag{51}$$

where the bias estimate of D1, i.e., $\widehat{\beta}_1[\nu+3]$, is simplified accordingly from (16) and we assume a fixed step size, i.e., γ, for updating the bias estimate. Note that TO estimate at each device must decrease to zero before changing sign, consequently, we have $\operatorname{sgn}(\widehat{\Delta t}_{21}[\nu+3]) = \operatorname{sgn}(\widehat{\Delta t}_{12}[\nu+4]) = \operatorname{sgn}(\widehat{\Delta t}_{12}[\nu+2]) = \operatorname{sgn}(\widehat{\Delta t}_{12}[\nu])$. Since we assume $\widehat{\Delta t}_{12}[\nu+2] > 0$, we can simplify (51) as $\widehat{\Delta t}_{12}[\nu+4] = \widehat{\Delta t}_{12}[\nu+2] - 2\gamma$. Finally, the change in the TO estimate of D2 for two consecutive clock ticks where it operates as a receiver is given by:

$$m = \frac{\widehat{\Delta t}_{12}[\nu+4] - \widehat{\Delta t}_{12}[\nu+2]}{(\nu+4) - (\nu+2) - 1} = -2\gamma \tag{52}$$

In Fig. 13, we plot the maximum synchronization error (33) against a straight line with negative slope as given by (52). The results show a very close match between the two curves, thereby supporting our analysis. For comparison, we also plot the synchronization error with adaptive step size.

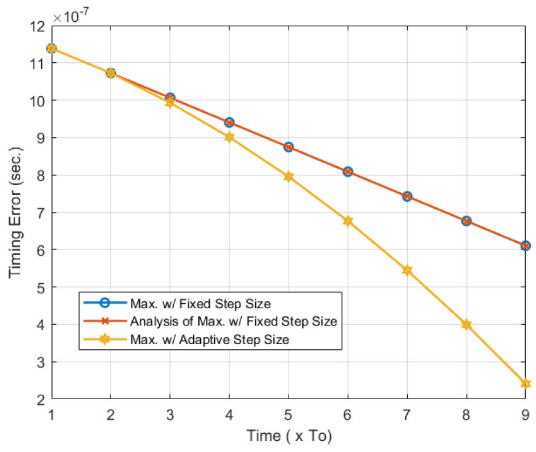


FIGURE 13. Synchronization performance based on the scenario in Appendix B with remaining parameters chosen from Table 1.

APPENDIX C
EXPECTATION OF TO ESTIMATE

We assume two devices labeled D1 and D2 communication over a flat reciprocal channel with propagation delay τ and the relative clock skew is negligible, i.e., $\Delta\alpha_{12} = \alpha_1 - \alpha_2 = 0$. In order for a device, say D2, to estimate the synchronization for updating its clock, it should operate as a receiver, while D1 should be a transmitter or vice versa. Therefore, the probability of this event is $p_e = 2p_{\text{tr}}(1-p_{\text{tr}})$. Then, the synchronization error at the νth clock tick for D2 is $\Delta t_{12}[\nu] = t_1[\nu] + \tau - t_2[\nu]$, whereas for D1, it is $\Delta t_{21}[\nu] = t_2[\nu] + \tau - t_1[\nu]$. Hence, the expected initial synchronization error can be given as $\Delta\theta = |t_1[\nu] - t_2[\nu]| = |\theta_1 - \theta_2| = |\Delta\theta_{12}| = |\Delta\theta_{21}|$. Now, based on the alternating transceiver mode and the clock updates according to (11) and (15), where we assume $\widehat{\beta}_{12}[\nu] = \widehat{\beta}_{21}[\nu] \approx \tau$ $\forall\nu$, the expected TO estimate at D2 (similar for D1) is equal to:

$$\mathbb{E}\big[\widehat{\Delta t}_{12}[\nu]\big] = \Bigg(\underbrace{(1-p_e)^{\nu}}_{\mathrm{P}_1} + \underbrace{p_e(1-\epsilon)^{(\nu-1)}}_{\mathrm{P}_2} + \underbrace{\sum_{k=1}^{\nu-1}(1-p_e)^k p_e (1-\epsilon)^{(\nu-1-k)}}_{\mathrm{P}_3}\Bigg)\Delta\theta, \forall \nu \geq 1 \tag{53}$$

Here, P_1 is the probability that the devices never operate at the opposite modes, which is the worst scenario since they cannot detect the error and update their clocks. Furthermore, P_2 is the probability that the devices operate on the opposite modes at the first clock tick and then start alternating between them, hence, it is the best scenario. Finally, P_3 is the probability that the devices start alternating their mode and update their clocks once they operate at the opposite modes, which is a mixed scenario. We numerically verify the validity of this analysis for values of $p_{\mathrm{tr}} \in [0, 1]$ and $\epsilon \in (0, 1]$, but did not to include the results in the paper due to space limitations.

REFERENCES

[1] B. A. Coll-Perales, J. Gozalvez, and J. L. Maestre, "5G and beyond: Smart devices as part of the network fabric," *IEEE Netw.*, vol. 33, no. 4, pp. 170–177, Jul./Aug. 2019,

[2] "Cisco visual networking index: Global mobile data traffic forecast update 2016–2021," San Jose, CA, USA, Cisco, White Paper, Feb. 2017.

[3] J. Cao *et al.*, "A survey on security aspects for 3GPP 5G networks," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 170–195, 1st Quart., 2020.

[4] *Technical Specification Group Radio Access Network, Stage 2 (Release 16)*, 3GPP Standard TS 38.300, Jul. 2020.

[5] M. N. Tehrani, M. Uysal, and H. Yanikomeroglu, "Device-to-device communication in 5G cellular networks: Challenges, solutions, and future directions," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 86–92, May 2014.

[6] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, "A joint learning and communications framework for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 1, pp. 269–283, Jan. 2021, doi: 10.1109/TWC.2020.3024629.

[7] Z. Yang, N. Huang, H. Xu, Y. Pan, Y. Li, and M. Chen, "Downlink resource allocation and power control for device-to-device communication underlaying cellular networks," *IEEE Commun. Lett.*, vol. 20, no. 7, pp. 1449–1452, Jul. 2016.

[8] O. Olabiyi, A. Annamalai, and L. Qian, "Leader election algorithm for distributed ad-hoc cognitive radio networks," in *Proc. IEEE Consum. Commun. Netw. Conf.*, Jan. 2012, pp. 859–863.

[9] 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. Commun.*, Jul. 2017, pp. 1–7.

[10] 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, Sep. 2008.

[11] M. J. Cannon, "On the design of D2D synchronization in 3GPP Release-12," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2015, pp. 633–638.

[12] A. Guchhait, "Maximum likelihood estimation of clock skew in sparse one-way packet transmissions for machine type communication applications," in *Proc. IEEE Int. Conf. Commun.*, May 2016, pp. 1–6.

[13] O. Al-Kofahi, "Evaluating time synchronization using application-layer time-stamping," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2016, pp. 1–6.

[14] H. Wang, H. Zeng, M. Li, B. Wang, and P. Wang, "Maximum likelihood estimation of clock skew in wireless sensor networks with periodical clock correction under exponential delays," *IEEE. Trans. Signal Process.*, vol. 65, no. 10, pp. 2714–2724, May 2017.

[15] A. K. Karthik and R. S. Blum, "Optimum full information, unlimited complexity, invariant, and minimax clock skew and offset estimators for IEEE 1588," *IEEE Trans. Commun.*, vol. 67, no. 5, pp. 3624–3637, May 2019.

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

[17] O. Karatalay, I. Psaromiligkos, B. Champagne, and B. Pelletier, "Fully distributed energy-efficient synchronization for half-duplex D2D communications," in *Proc. Symp. IEEE Pers. Indoor Mobile Commun.*, Sep. 2019, pp. 1–7.

[18] M. A. Alvarez and U. Spagnolini, "Distributed time and carrier frequency synchronization for dense wireless networks," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 4, no. 4, pp. 683–696, Dec. 2018.

[19] Z. Zhang, K. Long, A. V. Vasilakos, and L. Hanzo, "Full-duplex wireless communications: Challenges, solutions, and future research directions," *Proc. IEEE*, vol. 104, no. 7, pp. 1369–1409, Jul. 2016.

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

[21] N. Gresset and J. Letessier, "A random broadcast consensus synchronization algorithm for large scale wireless mesh networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2012, pp. 1573–1577.

[22] K. Doppler, C. B. Ribeiro, and J. Kneckt, "Advances in D2D communications: Energy efficient service and device discovery radio," in *Proc. IEEE Wireless VITAE*, Chennai, India, Feb. 2011, pp. 1–6.

[23] L. Song, D. Niyato, Z. Han, and E. Hossain, *Wireless Device-to-Device Communications and Networks*. Cambridge, U.K.: Cambridge Univ. Press, 2015.

[24] O. Karatalay, I. Psaromiligkos, B. Champagne, and B. Pelletier, "Fast converging distributed pulse-coupled clock synchronization for half-duplex D2D communications over multipath channels," in *Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT)*, Dec. 2018, pp. 123–128.

[25] Y. Wu, Q. Chaudhari, and E. Serpedin, "Clock synchronization of wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 124–138, Jan. 2011.

[26] P. Martí, J. Torres-Martínez, C. X. Rosero, M. Velasco, J. Miret, and M. Castilla, "Analysis of the effect of clock drifts on frequency regulation and power sharing in inverter-based islanded microgrids," *IEEE Trans. Power Electron.*, vol. 33, no. 12, pp. 10363–10379, Dec. 2018.

[27] "Evolved universal terrestrial radio access (E-UTRA); User Equipment (UE) radio transmission and reception," 3rd Gener. Partnership Project, Sophia Antipolis, France, Rep. TR 36.101, Mar. 2020.

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

[29] M. M. Mansour, "Optimized architecture for computing Zadoff-Chu sequences with application to LTE," in *Proc. IEEE Global Commun. Conf.*, Nov. 2009, pp. 1–6.

[30] *Mobile and Wireless Communications Enablers for the Twenty-Twenty Information Society (METIS), Deliverable D1.4 METIS Channel Models*, document ICT-317669-METIS/D1.4, METIS, Albuquerque, NM, USA, Feb. 2015.

[31] "Technical specification group radio access network; Study on LTE device-to-device proximity services; Sections A.2.1.1–A.2.1.2," 3rd Gener. Partnership Project, Sophia Antipolis, France, Rep. TR 36.843, Mar. 2014.

[32] P. A. Gagniuc, *Markov Chains, From Theory to Implementation and Experimentation*. Hoboken, NJ, USA: Wiley, 2017.