Technical performance assessment of the Ableton Link protocol over Wi-Fi*

LUCA TURCHET AND EDOARDO RINALDO
(luca.turchet@unitn.it) (edoardo.rinaldo@studenti.unitn.it)

Department of Information Engineering and Computer Science, University of Trento, Trento, Italy

To date, Ableton Link is the most widely adopted synchronization protocol for musical applications based on Wi-Fi networks. However, the limitations of Link over Wi-Fi in terms of scalability are not known, an understanding that may be useful to designers of musical ecosystems involving many nodes to be synchronized. In this paper we present four experiments aiming at investigating how the protocol performance is affected by the number of connected devices, the kind of Wi-Fi access point utilized, and by connecting or disconnecting the nodes. Results showed the reliability of the protocol only for a limited number of nodes, which was 22 for a consumer-grade portable router and 41 for a mesh network created by two high-end access points. The protocol performances were found to decrease with the number of devices and when nodes connected or disconnected. Furthermore, the performances of Link are tightly bounded to that of Wi-Fi, which can vary significantly from day to day depending on network load and interferences. Taken together, our findings indicate that Link over Wi-Fi is not suitable for ensuring synchronization in ecosystems with a high number of nodes, and call for new wireless technologies suitable for large scale synchronizations in co-located settings.

0 INTRODUCTION

The efficient and accurate operation of several applications in networks of distributed computational devices require a synchronized notion of time [1]. The individual network nodes are equipped with a local clock from which events may be accurately scheduled. Such a clock is derived from an embedded crystal or an oscillator circuit, which however tend to drift with respect to the clocks on other network nodes, due to imperfections in the timing hardware [2]. Even if different devices in the network would initially share the same clock, they need a resynchronization procedure from time to time. Various systems and protocols have been proposed within the Internet of Things (IoT) field to address the issue of establishing an accurate, network-wide notion of time, which is crucial for scenarios demanding precise temporal coordination [3, 4]. A particular focus has been placed on minimizing synchronization errors between nodes of wireless sensor networks (see e.g., as the Flooding Time-Synchronization Protocol [5], PulseSync [6] and variations of it [7]).

Synchronization is a central aspect within the emerging subfield of the IoT paradigm called the Internet of Musical Things (IoMusT) [8]. In several IoMusT applications, the Musical Things (i.e., devices serving a musical purpose, such as smart musical instruments [9]) that are present in an ecosystem require a tight synchronization in order to create musically relevant outcomes enjoyable by both musicians and audiences. This issue impacts networked music performances in both wide and local area networks and in both wired and wireless networks [10, 11].

Today, the majority of systems adopted for musical and creative applications in wireless local area networks (WLANs) typically utilize Bluetooth (originally IEEE 802.15.1), ZigBee (IEEE 802.15.4) or Wi-Fi (IEEE 802.11b/g/n/ac), and in particular they use protocols to exchange musical messages between devices such as MIDI or Open Sound Control [12, 13, 14]. Synchronization aspects in WLANs have been addressed by various studies in the literature [15, 10, 16, 8, 17]. An example is represented by the approach based on HTML5 proposed in [18] to synchronize mobile based applications leveraging the Web Audio framework.

To date, the most widely adopted synchronization protocol for musical applications within Wi-Fi based WLANs is *Link*[1], a de-facto standard developed by the company Ableton [19]. Link is a protocol that allows one to synchronize musical beat, tempo, phase, and start/stop commands

*Corresponding author: Luca Turchet; e-mail: luca.turchet@unitn.it

[1] http://ableton.github.io/link/

across multiple applications running on one or more Musical Things. The applications running on Musical Things connected to a WLAN are able to discover each other automatically and form a musical session in which each participant can start or stop while still staying in time, change the tempo, as well as join or leave without disrupting the session.

However, to the best of authors' knowledge at present there is no study conducted to assess the limitations of Ableton Link. This paper aims to address this gap. Specifically, we are interested in assessing the scalability of the Link protocol, i.e. how many nodes can be simultaneously handled within a standard Wi-Fi based WLAN. Whereas Ableton Link appears designed to support a limited number of devices, some IoMusT applications may require the synchronization of a large amount of devices, such as participatory live music systems involving the audience in the music creation process [20]. Understanding the limitations of this protocol is useful to provide guidelines to design IoMusT applications based on it and to envision possible avenues to advance its development.

In the reminder of the paper, we first provide an overview of Link's main features. Then we describe a set of experiments conducted to assess its limitations under different conditions, and discuss the reported results.

1 ABLETON LINK CONCEPTS

The main difference between Link and other approaches developed by the music technology community (e.g., JACK Transport [21], MIDI Beat Clock) is that it is not based on a master/client paradigm [19]. In systems based on a master/client communication protocol, the master provides a clock signal to a number of clients, and the master application is usually the only one that has control over tempo and transport state. This has the drawback that if the master fails, or the channel breaks, the clients are in an undefined state. Link introduces a different approach by creating a peer-to-peer network where all peers share a global time reference and a beat timeline. Any of the peers is empowered with the ability to introduce changes to the timeline in order to change the state of the session. Specifically, Link is based on the following concepts as reported on the company's website:

- **Tempo Synchronization.** All participants to the session can propose a change to the session tempo at any time. Whereas no single participant is responsible for maintaining the shared session tempo, each participant chooses to adopt the last tempo value that has been proposed on the network. Therefore, it is possible for participants' tempi to diverge during periods of tempo modification (this is particularly true during simultaneous modification by multiple participants). Nevertheless, this state is only temporary as the session will converge quickly to a common tempo after any modification.
- **Beat Alignment.** Beside having a common tempo it is important that participants in the session are synchronized on the same beat. Link guarantees that beats are aligned across the nodes within the network.
- **Phase Synchronization.** Whereas in most circumstances beat alignment is a necessary condition for playing in time, it is often not enough. When working with bars or loops, a musician may expect that the beat position within such a construct (i.e., the phase) be synchronized, resulting in alignment of bar or loop boundaries across participants. Link addresses this case.
- **Start/Stop Synchronization.** Link allows participants to share information on the musician's intent to start or stop transport with other peers that have the feature enabled.

Link is based on UDP, and uses the multicast group 224.76.78.75 and port 20808. Link clients join or leave the multicast group via Internet Group Management Protocol (IGMP), and every client continuously sends packets containing information to synchronize with other hosts. Link is released as open source and is currently implemented on various platforms and embedded in an increasing number of applications thanks to freely available APIs.

2 METHOD

To assess the Link protocol a number of aspects must be taken into account. First, different Link-enabled nodes may differ for the underlying audio processing architecture, which may impact the protocol performance measurements. Indeed to assess if multiple devices are playing in time, it is necessary to consider the moment at which their generated audio signals hit the output cable, not the time at which an audio callback is invoked at software level to generate the output (which is typically followed by buffering mechanisms or other delays due to digital-to-analog converters).

Second, the kind of access point (AP) utilized to create the network and regulate its traffic may impact the performance of the protocol, as APs may differ for the number of antennas, frequency bands supported, and algorithmic method to handle multicast. Third, the environment can have a huge impact on the Wi-Fi network conditions in terms of interferences due to the presence of other co-located APs or devices using the same frequency bands. Fourth, devices can join and leave the network as well as the tempo variations can occur as a result of the expressive needs of the performers.

In addition, it is important to consider the measures related to the perception of synchronicity of musical sounds, which can set an upper bound for which two out of synch sounds generated by as many Link-enabled devices can be perceptually tolerated by musicians. While research has found that performers using conventional instruments or hand clapping are generally able to maintain a stable tempo in presence of latencies up to 30 ms [10], this is not the scenario here investigated as the synchronization is handled entirely by machines and perceived by humans. It is well known that the temporal accuracy of the auditory system is very high. Humans can distinguish as separate two clicks

temporally distant even at 2 ms [22], but in the context of electronic music making this limit is certainly too strict.

We could not find in the literature a clear result of psychoacoustics experiments that investigated how the music making experience of musicians (and electronic musicians in particular) is hampered by a specific number of milliseconds for out of synch beats coming from different digital musical instruments. Based on results of previous experiments conducted in the networked music performance field [23] it is reasonable to assume that such a number depends on several factors, including the music type and tempo, as well as the envelope parameters (including duration) and timbre of the electronically generated sounds. Furthermore, also the intended use of the device running the protocol might play a role (e.g., for applications not requiring timely actions, such as moving a fader in a slow piece of music, the constrains could be relaxed, while a very low threshold would be needed for applications requiring very tight synchronization, for instance when an array of networked speakers collectively engage in some form of phase sensitive sound synthesis such as wave field synthesis).

Nevertheless, in absence of a proven threshold, we believe that it is reasonable to assume for it the same number of milliseconds found for the synchronization among performers of conventional instruments, that is 30 ms [10]. Nevertheless, in the case of the use of Link for music making, performers are distributed in the same room and, therefore, we also need to take into account the synchronization error due to the physical distance between performers (who might have the source of the sounds placed near them, e.g., a loudspeaker). Considering a range of about 3 meters and that sound travels at 343 m/s in air, we can account for about 10 ms due to the sound transmission between performers. Thus, subtracting this amount from the 30 ms due to previous results on the synchronization among performers of conventional instruments, we set to 20 ms the threshold for the Link protocol above which synchronicity can be considered to be deteriorating from the perceptual standpoint.

The consideration of the aspects mentioned above led to the following research questions: *i)* how does the protocol performance vary in function of the number of connected devices?; *ii)* what is the maximum number of nodes that Link can support before a perceptible decrease of synchronization occurs?; *iii)* how the change of AP affect the protocol performance?; *iv)* how is the protocol performance affected by connecting or disconnecting devices?

To address these questions we designed four experiments, which leveraged an ad-hoc created ecosystem and a measurement apparatus described in the next sections.

2.1 Ecosystem and measurement apparatus

The ecosystem and measurement apparatus involved in the experiments are illustrated in Figure 1. The ecosystem consisted of a number of Link-enabled devices supporting Wi-Fi 5GHz (IEEE 802.11ac). These were connected to the same Wi-Fi network, which was created by an AP. Specifically, a total of 75 Link-enabled devices were employed in the experiments, namely 24 Raspberry Pi 4 and 38 Raspberry Pi 3b+ (with Linux Raspbian Buster and using the internal sound interface), 9 Bela Mini (with Linux Xenomai, using the internal sound interface and an A6100-100PES Wi-Fi USB dongle), 2 laptops, 2 Android-based smartphones.

Each of the 75 devices run a Pure Data application, which connected to the Link protocol and emitted a short click sound in response to each beat. Pure Data was configured with a block size of 64 samples and a sample rate of 44100 Hz. The utilized Pure Data external implementation[2] quantized the clicks to the start of the next audio block (i.e., the clicks were not sample accurate). Therefore, in our evaluations it is necessary to consider that the block size contributed to the timing error in a node's synchronization, which in the worst case amounts to 1.45 ms.

We also used two kinds of APs to assess the impact of the AP in modulating the protocol performances, precisely the TP-Link TL-WR902AC and the HPE Aruba AP-304. These two kinds of APs have both 5GHz and 2.4GHz radio, but differ for the number and type of antennas, as well as for the maximum number of nodes supported. Whereas TL-WR902AC is a simple, portable, and single-antenna router, designed for a small number of clients, the AP-304 is a professional-grade high-performance AP, which has three total antennas. Multiple AP-304 can be configured to form a mesh network and, although it is not a router, it is designed to support layer 3 (Network Layer) capabilities such as routing independently multicast and IGMP groups. As suggested by [13] for higher Wi-Fi performance in the context of interactive musical performances, we configured them with no encryption.

The 75 devices were distributed in a 7-meter diameter. For the experiment involving the TL-WR902AC, this AP was placed approximatively at the middle of the nodes. The other three experiments involved a mesh network of 2 AP-304, which were hanging on the ceiling at the opposite ends of the diameter in which nodes were distributed.

The measurement apparatus comprised three RME Fireface UFX+ soundcards, which shared the same clock via a dedicated wordclock cable and were connected via as many USB cables to a MacBook Pro laptop. This gives a total of 24 analog input channels that can be used to record the sound generated by as many Link-enabled devices. For this purpose, we utilized 24 identically configured Raspberry Pi 4, which were connected to the soundcards via jack cables. We then created a Pure Data application to record the 24 channels in .wav files for further analysis. Recordings were performed at a sample rate of 44100 Hz, therefore resulting in a precision of around 0.0227 milliseconds between samples.

In addition the measurement apparatus comprised 2 laptops. The first one run a Pure Data patch that automatically changed the Link's beats per minute (BPM) as described

[2] https://github.com/libpd/abl_link

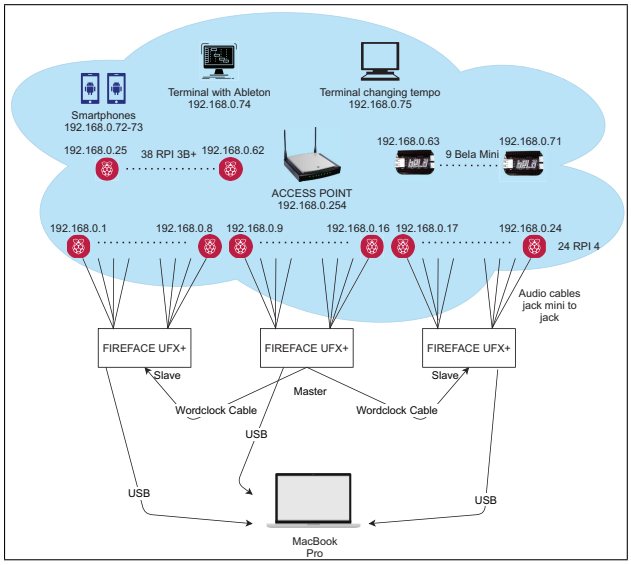

Fig. 1. Schematic diagram of the ecosystem and measurement apparatus.

below, and the second one monitored the total device number, the BPM changes, and the connection of devices.

2.2 Procedure

Four experiments were conducted to stress-test the protocol, which provide examples of various network settings and load situations that may resemble a real performance configuration with varying degrees of complexity. All test were conducted under "ideal" conditions as far as the Wi-Fi networks traffic is concerned, namely in a laboratory of University of Trento in July 2020 during the period of the social distances restrictions due to the COVID-19 pandemics, which allowed only a few people to be present in the university. Therefore, our measurements were intended to serve as a baseline useful for future comparisons with other network conditions.

Experiment 1. The first experiment aimed at understanding the protocol performances when using a consumer grade AP and how the investigated synchronization error measures changed with the number of connected devices. For this purpose we used the TLWR902AC portable router.

To stress-test the protocol we developed a Pure Data patch running on a laptop, which automatically changed the BPM according to the following pattern:

1. from 20 to 200 BPM in 2.5 seconds;
2. wait for 20 seconds;
3. from 200 to 20 BPM in 5 seconds;
4. from 20 to 200 BPM in 5 seconds;
5. wait for 20 seconds;
6. from 200 to 20 BPM in 10 seconds;
7. from 20 to 200 BPM in 10 seconds;
8. wait for 50 seconds;
9. repeat.

This pattern aimed at representing tempo changes happening with different speeds, replicating possible behaviors of human performers. It was repeated for 3 minutes in each recording. Recordings started from 4 nodes and progressively increased by one unit the number of nodes reaching 22, for a total of 19 recordings (22 was the maximum number of nodes that the AP supported, which was found experimentally). Each recording contained around 500 clicks for each device.

Experiment 2. The goal of the second experiment was to understand to which extent the protocol performance is affected when involving a high number of nodes and considering a long period of utilization. As the TLWR902AC router allowed only up to 22 devices, we used two Aruba AP-304s in a network mesh configuration. We found experimentally that this setup allowed for up to 74 nodes. The test involved the same patterns of tempo changes used in experiment 1. It lasted 2 hours (to resemble a realistic duration of a live concert). The total number of clicks per device was 21000.

Experiment 3. The third experiment aimed at investigating what is the maximum number of nodes that Link can support before a perceptible decrease of synchronization occurs. The two Aruba AP-304 were utilized. Using the same Pure Data patch involved in the previous experiments, we performed 12 recordings of 30 minutes using a progressive increase in the number of devices, from 26 to 74. Every recording contained 5000 clicks per device.

Experiment 4. The fourth experiment, which involved the two Aruba AP-304, addressed the research question about how the protocol performance is affected by connecting or disconnecting the nodes. For this purpose, we developed a Pure Data patch, which automatically changed the total number of active nodes and the BPM according to the following pattern, when connecting new devices:

1. from 200 to 20 BPM in 2.5 seconds;
2. from 20 to 200 BPM in 2.5 seconds;
3. wait for 20 seconds;
4. from 200 to 20 BPM in 5 seconds;
5. from 20 to 200 BPM in 5 seconds;
6. wait for 20 seconds;
7. from 200 to 20 BPM in 10 seconds;
8. from 20 to 200 BPM in 10 seconds;
9. wait for 15 seconds;
10. turn on 4 more devices (which takes around 20 seconds to boot and join the network using the protocol);
11. wait for 60 seconds.

An analog pattern was used when considering the removal of the devices from the network. The node number varied from 26 to 64, when connecting new devices (in the time span of about 30 minutes), and vice versa when disconnecting them. This recording contained 11600 clicks for every device. Notably, in this experiment we could not reach the maximum number of 74 devices as the protocol did not allow a higher number than 64 in that case.

3 ANALYSIS

A C# program was coded to iterate the .wav files sample by sample and fill a CSV file with the starting time of each click for each of the 24 channels. The peak picking method simply consisted in checking the amplitude of each sample above a certain threshold (as the click sounds were all identical) and adopting a debouncing period of 10 ms.

To measure the protocol performance we calculated the maximum and the mean absolute deviation (MAD) synchronization error between corresponding clicks recorded on each channel. The maximum synchronization error corresponds to the temporal distance between the furthest clicks. The MAD synchronization error is defined by the following formula for a set $\{x_1, x_2, ..., x_n\}$:

$$MAD = \frac{1}{n}\sum_{i=1}^{n} |x_i - m(X)| \quad (1)$$

where $m(X)$ is the measure of central tendency, in our case the mean value of the set.

Figure 2 provides a graphical representation of the two measures (considering just 4 channels for the sake of easiness of representation). The maximum synchronization error allows to understand how out of synch the devices can be from each other. Notably, this measure can be affected even by a single outlier. On the other hand the MAD provides a general depiction of the status of all nodes. In the first and third experiment we aimed at understanding the distribution of these two measures of synchronization error (along with their standard error) for the varying number of nodes involved, while in the second and fourth experiment we aimed at unraveling their temporal distribution.

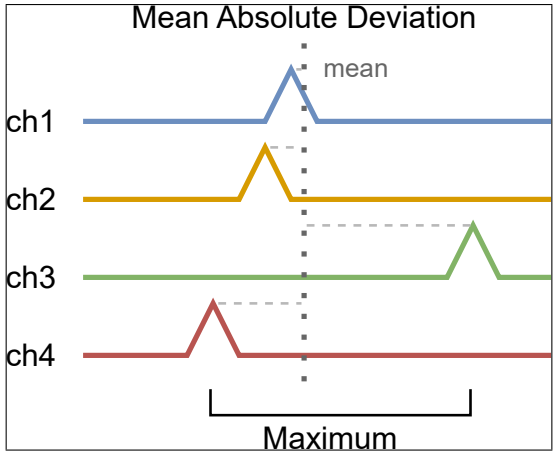

Fig. 2. Graphical representation of the maximum and mean absolute deviation synchronization errors between recorded clicks (4 channels are illustrated, in the actual analysis 24 were utilized).

Only a few outliers were identified and were manually removed. Notably, we noticed that under some experimental conditions, some devices did not synchronize immediately following a tempo change (both rapid and slow), but they took some time during which they kept their previous tempo. This led in some cases to high synchronization errors. Furthermore, an in-depth analysis on the conditions under which the maximum synchronization errors occur revealed that their cause is the rapid tempo changes. This is due to the fact that the propagation of tempo changes in the protocol takes some time when the network is crowded and when the tempo change is rapid.

4 RESULTS

Maximum nodes number. We found that there is a maximum number of devices that can be successfully connected to the protocol. This limitation depends mainly on the utilized AP, where high performance AP can form a mesh network and support a higher number of devices compared to average APs.

In particular, whereas for the Aruba AP-304 when a node tries to join exceeding the maximum number allowed the connection is refused, for the TLWR902AC the connection is accepted but the whole network becomes slow and unresponsive. Using the AP TLWR902AC, in the condition of a restricted number of devices the average throughput of a Link node was between 0.1 and 0.6 Mbps and the delay of ping messages between devices under 100 ms (20 ms on average). Whenever the device count passed the limit the ping increased to 500-2000 ms, the throughput to 0.9-1.5 Mbps and other devices lose sync or disconnect. Furthermore, we noticed that if any device is turned off, the exceeding one will connect shortly after, the network will quickly return to normal functioning, and the protocols will work normally again. Although the exceeding node is not synchronizing with others, nor appearing in the device count provided by the Link facilities, the IGMP group table on the router will still contain the device's IP. This limit can vary at different times or changing the position of the nodes.

Notably, we also found that the maximum number of devices that can be supported can vary on different days or at different time within the same day, even when using the exactly same devices and position. This variability is likely due to environmental variables such as interfering Wi-Fi networks or devices.

Experiment 1. Results of the first experiment, which involved the low-cost AP TLWR902AC, are illustrated in Figure 3. As shown, up to 22 nodes the two measures of synchronization error remain below 3.5 ms, although with a constant increase in the maximum synchronization error. The same experiment was not repeated with the advanced AP Aruba AP-304, but considering its higher quality and the results of experiment 3 for 26 nodes (see Figure 5) it is more than reasonable to expect that the performances do not significantly differ from the ones achieved with the AP TLWR902AC.

Experiment 2. In Experiment 2 we tried to connect as many devices as possible for a rather long period (2 hours). Results, illustrated in Figure 4, show that while the MAD synchronization error was on average below 16 ms for the whole duration of the recording (mean = 5.3 ms), the maximum synchronization error could reach even 150 ms (mean = 40.2 ms). The measure of the maximum synchronization error (average and maximum) exceeded the 20 ms threshold we set, and indeed an inspection using the ac-

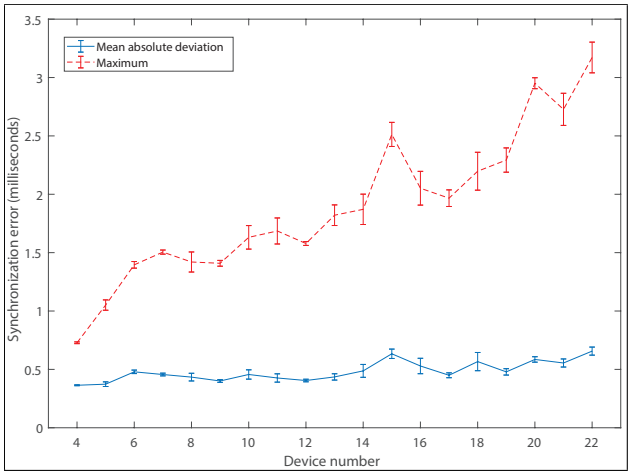

Fig. 3. Results of experiment 1. Mean absolute deviation and maximum synchronization errors for a number of nodes varying between 4 and 22, using the TLWR902AC access point.

tual sounds revealed a perceptible asynchronicity. Notably, the performances of the protocol deteriorated with time, especially in terms of the maximum synchronization error. Therefore, the experiment shows that even if in this network the protocol can handle 74 nodes, its performance is not good enough, and considering that we performed the test in a controlled environment it would likely be even worse in real-world applications.

Experiment 3. Figure 5 shows the results of the third experiment. As it is possible to notice, up to 41 nodes the maximum synchronization error remains under 11 ms, which is tolerable according to the threshold considered. However, with a higher number of devices the performance becomes clearly unstable and unpredictable. At 74 nodes, only the MAD is consistent with the results of experiment 2 for the first 30 minutes (see Figure 4), whereas the maximum synchronization error is on average lower indicating a high variability of this performance metric.

Experiment 4. The results of the fourth experiments are depicted in Figure 6. As it is possible to notice, the maximum synchronization error presents, above 41 devices, a set of spikes that can reach up to about 1300 milliseconds. This behavior is due to the combined effect of the number of the act of joining/leaving the network and the tempo change. It can be explained not only by internal mechanisms of the Link protocol, but also by effects due to the handling of multicast by the APs, which include a number of retransmissions due to dropped packets.

5 DISCUSSION AND CONCLUSION

In this work we documented the process of testing the scalability and performance of the Ableton Link protocol over Wi-Fi. We aimed at investigating the behavior of the protocol under stress conditions in order to understand whether it can be suitable for a complex usage like crowd participation and involving various kinds of tempo changes.

Results indicate that Link over Wi-Fi is not suitable for ensuring synchronization in ecosystems with a high number of nodes, even when involving high-end APs. Our findings showed the reliability of the protocol over Wi-Fi only for a limited number of nodes, which was 22 for a consumer-grade portable router and 41 for a mesh network created by two high-end AP. With Link every single node sends packets to the multicast group when joining/leaving the network or when a tempo change occurs, and that causes a growth of the required throughput for every device-antenna connection.

The quantity of antennas in the AP appears to be the main variables that affect the limit on the number of devices that can join the protocol. The number of devices per antenna is likely given by throughput (which changes mainly with the band), wireless standard, antenna gain, signal-to-noise ratio and beamforming [24]. It is reasonable to think that using newer technologies like 5G at high frequency bands a higher devices-per-antenna ratio will be observed. Nevertheless, the protocol performances decrease as the number of devices are added. Indeed there is a tendency for which the two measures of synchronization error increase (with some unpredictable fluctuations) as the number of devices (and, therefore, the transmission rates) increase on the network. This result is in line with those of the study reported in [25], which investigated the effect of number of nodes on the Wi-Fi performances (only using the 2.4 GHz band) in terms of synchronization error of packet delivery under stress conditions. Furthermore, the performances of the protocol over Wi-Fi in presence of a high number of devices are hampered by the nodes act of joining or leaving the Link session.

It is worth noticing that the measures reported in this study related to an ideal condition, i.e., an empty office in a building with few people and, therefore, little interference from other networks and devices. The protocol performance may vary significantly from day to day depending on network usage in a crowded environment, as well as by the presence of electro-magnetic interferences. Therefore, we should consider that outside the controlled environment, in actual application settings such as performance environments, the performances here reported could be worse.

Link performances are intrinsically bounded to the Wi-Fi performances. The fact that the performance of Wi-Fi can vary drastically due to external conditions such as network load or interferences, is a problematic aspect for practical live music application scenarios, a concern also expressed by other researchers in the music technology field [25]. In future work we plan to investigate the exact causes of the decrease of performances of Ableton Link. For this purpose, it is necessary to devise a methodology capable of isolating the potential issues, in order to understand for instance whether the decreases of performance found on this study can mostly be attributed to an algorithmic basis, the network/operating system/audio stack of the specific systems under study, or a combination thereof. For instance, instead of using Wi-Fi it is in principle possible to run the tests here reported over ethernet or 5G.

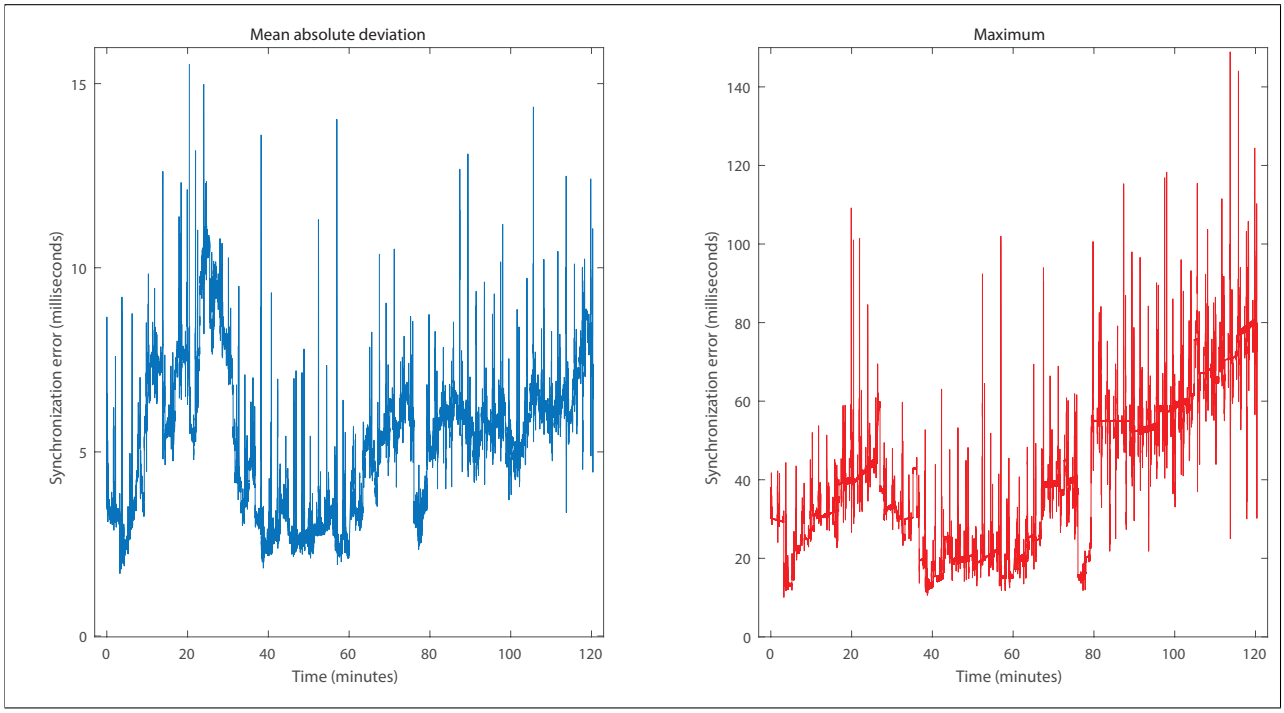

Fig. 4. Results of experiment 2. Mean absolute deviation (left) and maximum (right) synchronization errors for a recording of 2 hours involving 74 nodes and the mesh network generated by two Aruba AP-304 access points.

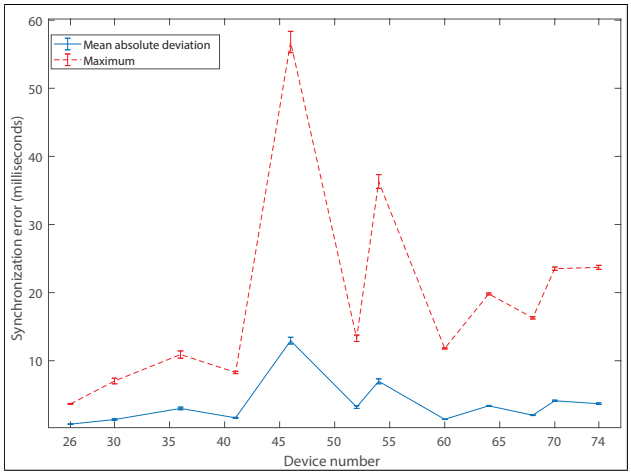

Fig. 5. Results of experiment 3. Mean absolute deviation (left) and maximum (right) synchronization errors in function of the number of devices (12 separate recordings of 30 minutes), using two Aruba AP-304 access points.

Wireless technology offers many benefits to the design of new musical interfaces and IoMusT ecosystems compared to the cabled-based counterparts, including increased mobility, dynamic network formation, increased flexibility of ecosystem design and development, as well as ease of deployment. Nevertheless, despite these benefits and the importance of accurate timing in music, today there are relatively few widely applicable synchronization solutions for Wi-Fi networks available to computer music practitioners. This calls for novel wireless technologies enabling IoMusT applications capable of supporting various kinds of synchronized interactions between performers as well as between performers and audiences. With respect to this, the deployment of 5G infrastructures appears to be a promising, yet unexplored solution for the creation of synchronized applications in IoMusT ecosystems, in both co-located and remote settings [26].

6 REFERENCES

[1] I. F. Akyildiz, M. C. Vuran, *Wireless Sensor Networks*, vol. 4 (John Wiley & Sons) (2010).

[2] Y. Wu, Q. Chaudhari, E. Serpedin, "Clock Synchronization of Wireless Sensor Networks," *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 124–138 (2011 Jan.).

[3] Q. Li, D. Rus, "Global Clock Synchronization in Sensor Networks," *IEEE Trans. Comput.*, vol. 55, no. 2, pp. 214–226 (2006 Feb.).

[4] F. Sivrikaya, B. Yener, "Time Synchronization in Sensor Networks: A Survey," *IEEE Netw.*, vol. 18, no. 4, pp. 45–50 (2004 July-Aug.).

[5] M. Maróti, B. Kusy, G. Simon, Á. Lédeczi, "The flooding time synchronization protocol," presented at the *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, pp. 39–49 (2004).

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

[7] K. S. Yildirim, A. Kantarci, "Time Synchronization Based on Slow-Flooding in Wireless Sensor Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 1, pp. 244–253 (2013 Jan.).

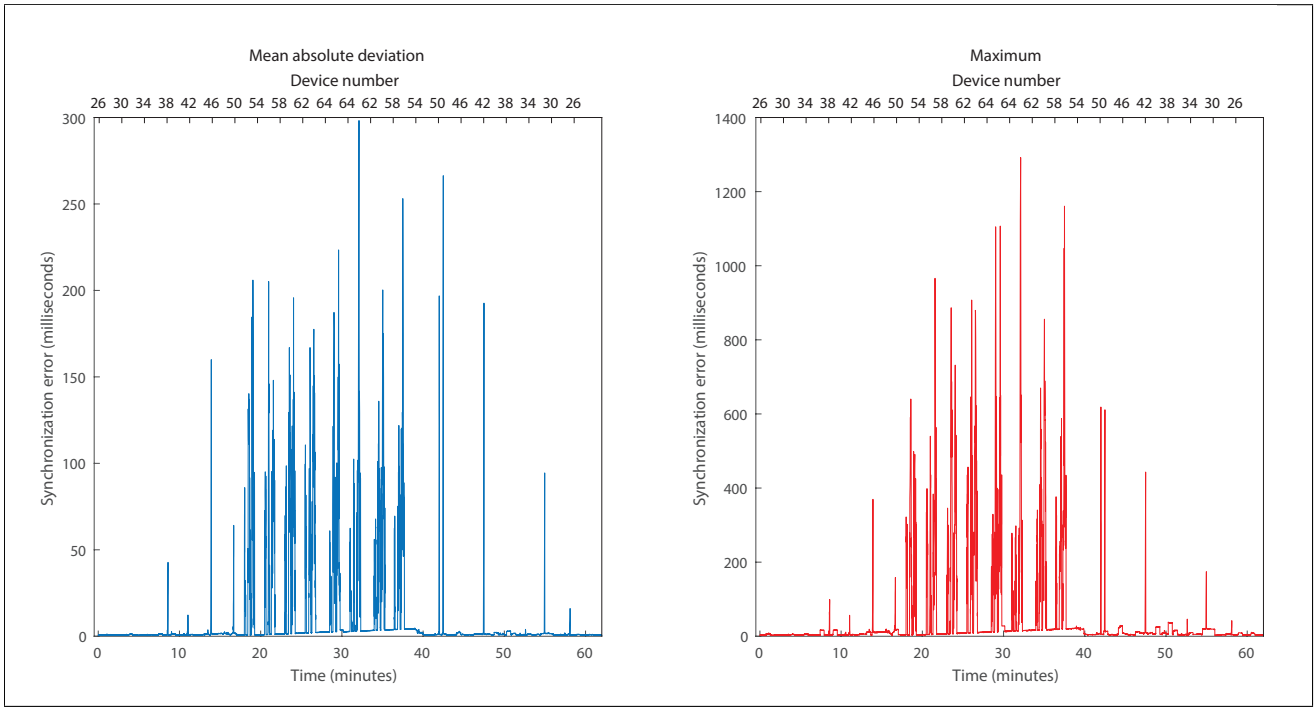

Fig. 6. Results of experiment 4. Mean absolute deviation (left) and maximum (right) synchronization errors when involving a variable number of devices joining and leaving the network and using two Aruba AP-304 access points.

[8] L. Turchet, C. Fischione, G. Essl, D. Keller, M. Barthet, "Internet of Musical Things: Vision and Challenges," *IEEE Access*, vol. 6, pp. 61994–62017 (2018), doi:https://doi.org/10.1109/ACCESS.2018.2872625, URL https://doi.org/10.1109/ACCESS.2018.2872625.

[9] L. Turchet, "Smart Musical Instruments: vision, design principles, and future directions," *IEEE Access*, vol. 7, pp. 8944–8963 (2019), doi:10.1109/ACCESS.2018.2876891, URL https://doi.org/10.1109/ACCESS.2018.2876891.

[10] C. Rottondi, C. Chafe, C. Allocchio, A. Sarti, "An Overview on Networked Music Performance Technologies," *IEEE Access*, vol. 4, pp. 8823–8843 (2016), doi:10.1109/ACCESS.2016.2628440, URL https://doi.org/10.1109/ACCESS.2016.2628440.

[11] L. Gabrielli, S. Squartini, *Wireless Networked Music Performance* (Springer) (2016), doi:10.1007/978-981-10-0335-6_5, URL https://doi.org/10.1007/978-981-10-0335-6_5.

[12] M. Wright, "Open Sound Control: an enabling technology for musical networking," *Organ. Sound*, vol. 10, no. 3, pp. 193–200 (2005).

[13] T. Mitchell, S. Madgwick, S. Rankine, G. Hilton, A. Freed, A. Nix, "Making the Most of Wi-Fi: Optimisations for Robust Wireless Live Music Performance," presented at the *Proceedings of the Conference on New Interfaces for Musical Expression*, pp. 251–256 (2014).

[14] L. Turchet, F. Antoniazzi, F. Viola, F. Giunchiglia, G. Fazekas, "The Internet of Musical Things Ontology," *Web Semant.*, vol. 60, p. 100548 (2020 Jan.), doi:https://doi.org/10.1016/j.websem.2020.100548, URL http://www.sciencedirect.com/science/article/pii/S1570826820300019.

[15] E. Brandt, R. Dannenberg, "Time in Distributed Real-Time Systems," presented at the *Proceedings of the International Computer Music Conference* (1999).

[16] S. Madgwick, T. Mitchell, C. Barreto, A. Freed, "Simple synchronisation for open sound control," presented at the *Proceedings of the International Computer Music Conference* (2015).

[17] R. Dannenberg, "O2: A Network Protocol for Music Systems," *Wirel. Commun. Mob. Comput.* (2019 May).

[18] J. Lambert, S. Robaszkiewicz, N. Schnell, "Synchronisation for Distributed Audio Rendering over Heterogeneous Devices, in HTML5," presented at the *Proceedings of the Web Audio Conference* (2016).

[19] F. Goltz, "Ableton Link–A technology to synchronize music software," presented at the *Linux Audio Conference*, pp. 39–42 (2018).

[20] O. Hödl, C. Bartmann, F. Kayali, C. Löw, P. Purgathofer, "Large-scale Audience Participation in Live Music Using Smartphones," *J. New Music Res.*, vol. 49, no. 2, pp. 192–207 (2020).

[21] J. Cáceres, C. Chafe, "JackTrip: Under the hood of an engine for network audio," *Journal of New Music Research*, vol. 39, no. 3, pp. 183–187 (2010).

[22] B. Moore, *An Introduction to the Psychology of Hearing* (Brill) (2012).

[23] C. Bartlette, D. Headlam, M. Bocko, G. Velikic, "Effect of Network Latency on Interactive Musical Performance," *Music Percept.*, vol. 24, no. 1, pp. 49–62 (2006 Sep.).

[24] N. Jindal, Z. Luo, "Capacity limits of multiple antenna multicast," presented at the *2006 IEEE International Symposium on Information Theory*, pp. 1841–1845 (2006).

[25] J. Wang, E. Meneses, M. Wanderley, "The Scalability of WiFi for Mobile Embedded Sensor Interfaces," presented at the *Proceedings of the Conference on New Interfaces for Musical Expression*, pp. 73–76 (2020).

[26] M. Centenaro, P. Casari, L. Turchet, "Towards a 5G Communication Architecture for the Internet of Musical Things," presented at the *IEEE Conference of Open Innovations Association (FRUCT)*, pp. 38–45 (2020).

THE AUTHORS

Luca Turchet

Edoardo Rinaldo

Luca Turchet is an Assistant Professor at the Department of Information Engineering and Computer Science of University of Trento. He received master degrees (summa cum laude) in Computer Science from University of Verona, in classical guitar and composition from Music Conservatory of Verona, and in electronic music from the Royal College of Music of Stockholm. He received the Ph.D. in Media Technology from Aalborg University Copenhagen. His scientific, artistic, and entrepreneurial research has been supported by numerous grants from different funding agencies including the European Commission, the European Institute of Innovation and Technology, the European Space Agency, the Italian Minister of Foreign Affairs, and the Danish Research Council. He is co-founder of Elk. His main research interests are in music technology, Internet of Things, human-computer interaction, and multimodal perception.

●

Edoardo Rinaldo holds a bachelor degree from the Department of Information Engineering and Computer Science of University of Trento. His research interests include embedded systems, music technology and the Internet of Musical Things.