Quantifying Direct Link Establishment Delay Between Android Devices

Tomás Lagos Jenschke, Marcelo Dias de Amorim, Serge Fdida

Quantifying Direct Link Establishment Delay Between Android Devices

Tomás Lagos Jenschke, Marcelo Dias de Amorim, and Serge Fdida
Sorbonne Université, CNRS, LIP6 – Paris, France
Emails: {tomas.lagos-jenschke, marcelo.amorim, serge.fdida}@lip6.fr

***Abstract*—The enormous success of direct communication applications has shed light on the practical interest of Device-to-device (D2D) communications. However, to set up a direct link between two neighboring nodes, they have first to detect each other, which introduces a delay before they can start sending and receiving data. The link establishment delay can be particularly unfavorable in situations of strong mobility, as the availability of the direct communication link depends on how long the devices stay within communication range of each other. This paper reports on our experiments to evaluate the link establishment delay. We focus on Android devices and use the Nearby Connection Application Programming Interface (API), which supports Bluetooth Classic and Bluetooth Low Energy (BLE) to perform link connectivity. In a nutshell, we observe that the link establishment delay requires several seconds to complete in the case of Bluetooth Classic and even tens of seconds for BLE.**

***Index Terms*—Device-to-device, neighbor discovery, Bluetooth.**

I. Introduction

Device-to-device (D2D) communications do not depend on fixed network infrastructure and therefore open up a range of applications that can be used to enhance existing network infrastructures or leverage them to perform independent services [1]. Among others, it is an attractive solution to generate local wireless networks as a substitute for cellular networks [2], [3], thus relieving the traffic load on base stations. Most importantly, we observe a renewed interest in D2D communications in far-edge architecture, opening a more efficient and consistent integration in global architecture.

In many scenarios, mobile nodes may establish direct links among them opportunistically, i.e., whenever they get close to each other. When a communication opportunity occurs, one of the goals is to maximize the time the nodes use to exchange data. Nevertheless, the nodes must first go through a link establishment phase. Though within wireless range, the nodes do not communicate valuable data during this period, leading to a waste of resources. Although many papers have investigated the contact time among nodes [4], [5], very few have focused on the connection establishment phase.

We investigate *experimentally* the time it takes for devices to perform neighbor discovery before connecting. We considered Android devices running the Nearby Connections Application Programming Interface (API) provided by Google Developers [6]. We focus on the link establishment procedures of both Bluetooth Classic and Bluetooth Low Energy (BLE). The choice of the technology is essential as it impacts the link establishment delay. Although technologies such as Wi-Fi have better response times than other technologies, they are seldom considered for the neighbor detection phase because they introduce significant overhead [7]. For example, the power consumption required by Wi-Fi is much higher than the one required by Bluetooth Classic [8]–[10]. Furthermore, given that connection establishment does not require large packets of information and assuming that a device automatically requests connectivity with several devices as it goes along, attempting multiple connections with certain technologies may lead to higher battery life consumption than others.

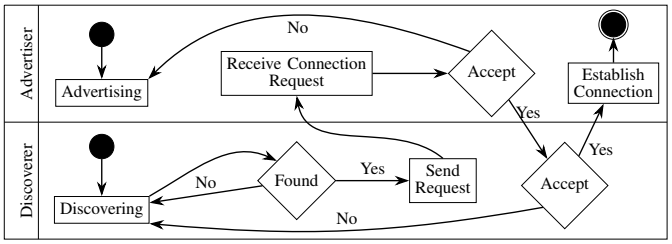


Fig. 1: D2D communication connectivity flow diagram.

To measure the connection establishment delay finely and collect other useful statistics about the D2D links, we have developed a dedicated application called AtomD. This application works with Nearby Connections API to analyze the behavior of D2D links in real environments. We present the features of AtomD in Section II. We run several measurement experiments in different conditions to introduce diversity in our analyses. In particular, we launch the neighbor detection and connection procedures for multiple distances separating the two nodes. Our goal here is to assess the influence of the signal strength on the link establishment delay. We also consider four different smartphone models of varying generations (OnePlus 5T, Samsung S20, Samsung S8, and Xiaomi Redmi 9T). As we will see in Section III, each has its specific behavior. Finally, we evaluate separately when a device operates in "discoverer" or "advertiser" modes. We consider all twelve arrangements of the devices and show that their modes also influence the results.

Our experiments reveal several observations. First, the link establishment process requires a few seconds when using Bluetooth Classic and several seconds (frequently above ten seconds) when using BLE. Secondly, devices from different brands behave differently, and some of them show clear insta-

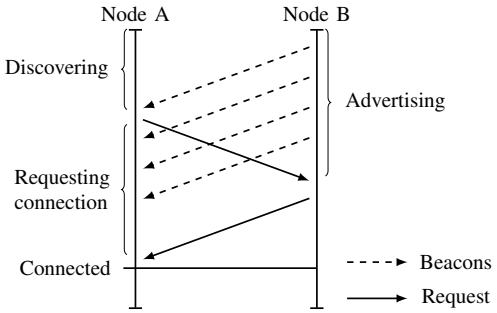


Fig. 2: Process to establish a connection, where Node A is a Discoverer and Node B is an Advertiser.

bility. Also, the behavior changes whether the device operates as a discoverer or an advertiser node. Another observation is that, contrary to what one would expect, the distance separating the nodes does not impact the link establishment delay. In conclusion, we point out in this paper that the link establishment phase is far from negligible. Therefore, network designers should consider this time when conceiving forwarding algorithms and protocols for opportunistic networks.

The remainder of this paper is structured as follows. We present in Section II the context and functionalities of the Nearby Connections API that we use in our experiments. We also present AtomD, a dedicated measurement application to assess the behavior of the D2D links. We present the details of the experimental setup in Section III, including the environmental conditions and device models. In Section IV, we give the results to estimate the impact of the connection establishment delay on the communication opportunity. Related works are the topic of Section V. Finally; we conclude the paper and list topics for future investigation in Section VI.

II. Context and Measurement Methodology

Nearby Connections is a high-level peer-to-peer (P2P) API developed by Google that acts as a medium-agnostic socket. It uses Bluetooth Classic or BLE to perform neighbor discovery and mainly Wi-Fi Direct (WFD) to perform data transfers. As shown in Figure 1, Nearby Connections assigns the participating devices with roles. Advertisers announce their presence, while Discoverers detect nearby Advertisers; when a Discoverer finds an Advertiser, it sends a connection request. If they both agree to connect, they establish a P2P link so that both parties can send and receive data.

We focus on capturing and analyzing the different stages of the connectivity process using Bluetooth Classic and BLE technologies. We measure the time it takes for a Discoverer to detect an Advertiser, as well as the time it takes for these two devices to establish a connection. In Figure 2, we depict the message exchanges between nodes A and B to establish a D2D link. We measure the time that elapses from the moment node A (Discoverer) initiates the discovery process until the time it detects node B (Advertiser), plus the time that elapses from the instant node A sends a connection request until the instant it receives a response from B.

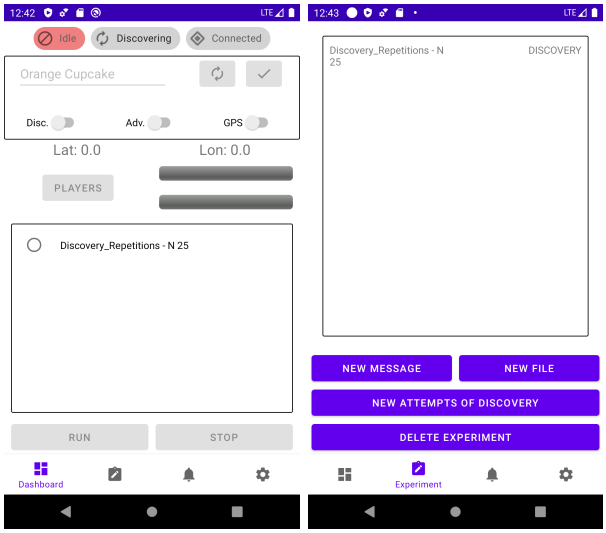


Fig. 3: AtomD's dashboard and experimental configuration.

To measure with precision the delays involved in the connection establishment phase, we have developed AtomD. AtomD uses the Nearby Connections API (see Section II) to generate D2D and manage connections between neighboring devices.[1] More specifically, AtomD establishes Nearby Connections as a service, allowing devices to perform background and foreground D2D communications. Furthermore, as shown in Figure 3, the AtomD interface allows the user to set the device as either "Disc." (Discoverer) or "Adv." (Advertiser). In addition, it allows translating a device ID into a user-readable name and displaying the GPS location of the device.

AtomD displays the current state of the device out of three possibilities. The first, Idle, indicates that the device does not have an instance to generate a D2D connection. The second, Discovering, indicates that the device has created a new instance for D2D communication and is currently doing beacon listening or beacon broadcasting. Finally, the state Connected corresponds to the case where two devices have already discovered each other and have made the corresponding connection requests to establish a D2D connection.

Among the experiments that the user can generate, we include the transmission of a single message over data chunks, the transmission of a fixed-size binary file over multiple data chunks, and the execution of multiple automatic link establishment procedures through a Discoverer node.

Focusing on executing multiple automatic link establishment procedures, AtomD stores the time instants in nanoseconds of the neighbor detection and the connection establishment procedures in a database. In particular, it stores the instant at which a device starts its discovery instance, the instant at which it discovers a device, the instant at which it sends a connectivity request, and the instant at which these devices establish the D2D connection.

Once the experiment starts, AtomD performs the connec-

[1]Details on AtomD can be found on https://github.com/tlagos1/AtomD.

Fig. 4: AtomD in two devices close to each other.

tivity iterations set by the user. It should be noted that in addition to these experiments, AtomD's modular design allows the incorporation of developer-defined add-ons.

III. Experimental Setup

We performed a set of experiments to evaluate the time it takes for two devices running to establish a D2D link using both Bluetooth Classic and BLE. More specifically, we evaluate the delay for a Discoverer node to perform neighbor detection and connection establishment with an Advertiser node for various distances setting them apart. In our experiments, we configured AtomD in P2P point-to-point mode (see Section II). We used one of its pre-configured experiments consisting of 25 link establishment attempts per pair of devices in which one takes the role of the Discoverer and the other the Advertiser. In average, the experiment takes $2.5\,\mathrm{s}$ with Bluetooth Classic and $9.5\,\mathrm{s}$ with BLE.

To account for the impact of distance on the link establishment delay, we varied the distance separating the Advertiser from the Discoverer. The Discoverer is placed at coordinate p_0, while the Advertiser is placed at p_0 to p_5. The distance separating $p_1, \ldots, p_5$ from p_0 are, respectively, $20\,$, $40\,$, $60\,$, $80\,$, and $100\,\mathrm{m}$. In Figure 4, we show the configuration when both devices are at p_0.

To assess the influence of the devices' specificities, we decided to run the experiments with multiple brands and models. For the results we will show below, we used the following devices:

- OnePlus 5T: Android 10, Qualcomm MSM8998 Snapdragon 835, Bluetooth (5.0, A2DP, LE, aptX HD).
- Samsung S20 FE 5G: Android 10, Qualcomm SM8250 Snapdragon 865 5G, Bluetooth (5.0, A2DP, LE).
- Samsung S8: Android 9, Exynos 8895 - EMEA, Bluetooth (5.0, A2DP, LE, aptX).
- Xiaomi Redmi 9T: Android 10, Qualcomm SM6115 Snapdragon 662, Bluetooth (5.0, A2DP, LE).

We run experiments for all possibilities of Advertisers and Discoverers, which leads to a total of 12 combinations. Unfortunately, because of limitations of the Samsung S8, this device could only operate in Discoverer mode when running BLE. Thus, we had in total 12 combinations when evaluating Bluetooth Classic and 9 combinations when evaluating BLE. Out of curiosity, taking into account that the human factor in switching devices takes approximately $5\,\mathrm{min}$, time required to perform a set of measurements for a specific distance is approximately $1.2\,\mathrm{h}$ for Bluetooth Classic and $1.3\,\mathrm{h}$ for BLE.

IV. Experimental results

We have considered data with a z-score of 3, which we represent using boxplots in Figures 5 to 12. We present the results in pairs of plots: the boxplot data on the left correspond to the values obtained with Bluetooth Classic, while the boxplots on the right show the values obtained with BLE.

In the plots, the x-axis corresponds to the distance, in meters, separating the Advertiser from the Discoverer (see Section III). In the case of Bluetooth Classic, the experiments were performed up to a distance of $100\,\mathrm{m}$, while with BLE they were performed up to $20\,\mathrm{m}$ (beyond this distance, the two nodes do not detect each other). The y-axis gives the delay, in seconds, for the link establishment process, which includes both the neighbor detection procedure (Figures 5 to 8) and the connection request procedure (Figures 9 to 12). In addition, we present the mean values of each box in the header of the graph and show them as a green triangle in each set.

It is also important noticing that all graphs correspond to statistics from the Discoverer's point of view. The Discoverer's name is shown at the top left of the graph, while the scenario is specified at the top right.

A. Neighbor detection delay

From the results obtained with the OnePlus 5T (Figure 5), we see that the average delay for neighbor detection via Bluetooth Classic is around $1\,\mathrm{s}$ and $3\,\mathrm{s}$. In contrast, the average delay in detecting the Samsung S8 at $100\,\mathrm{m}$ is $5.7\,\mathrm{s}$, indicating dependency on the device. As for the neighbors detected by the OnePlus 5T, the delay of the Redmi 9T was the shortest, with an average value of $1.7\,\mathrm{s}$. In addition, the data dispersion of the Redmi 9T is lower than its counterparts at $0\,\mathrm{m}$, $80\,\mathrm{m}$, and $100\,\mathrm{m}$. Finally, 75% of the neighbor detection delays that the OnePlus 5T observed with the other devices were less than $4\,\mathrm{s}$. In contrast, the OnePlus 5T with BLE requires 689% longer to discover an advertiser than the Bluetooth Classic, with 75% of its delays above $10\,\mathrm{s}$.

As for the Redmi 9T (Figure 6), we can observe that the average neighbor detection delay of each of its cases using Bluetooth Classic is between $1\,\mathrm{s}$ and $4\,\mathrm{s}$. Moreover, most of the neighbor detection delays with the Redmi 9T show that 75% of the data are below $4\,\mathrm{s}$. On the contrary, with BLE, we can observe that the Redmi 9T needs 50.37% less time to find a device at $0\,\mathrm{m}$ than the One Plus 5T.

Continuing with the Samsung S20 (Figure 7), we can see that, like the OnePlus 5T and the Redmi 9T, 75% of its neighbor detection delays using Bluetooth Classic do not exceed $4\,\mathrm{s}$. However, its average neighbor detection delay has a minor deviation than the other devices. In fact, the variance is $2\,\mathrm{s}$ for the Samsung S20, $0.45\,\mathrm{s}$ for the Redmi 9T, $1\,\mathrm{s}$ for the OnePlus 5T, and $1.36\,\mathrm{s}$ for the Samsung S8. Regarding

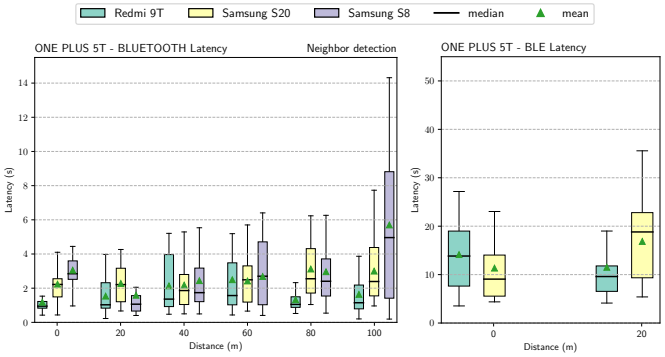


Fig. 5: Bluetooth Classic and BLE discovery latency using the One Plus 5T as a discovery device.

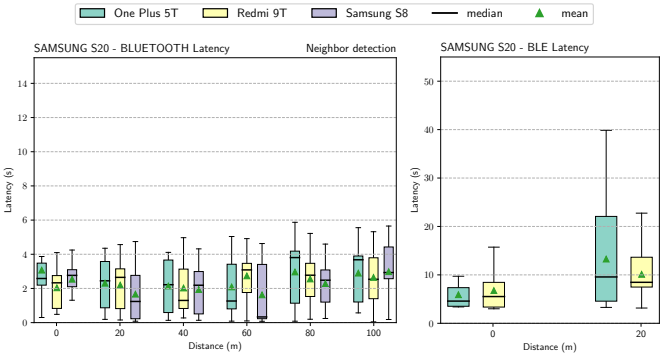


Fig. 7: Bluetooth Classic and BLE discovery latency using the Samsung S20 as a discovery device.

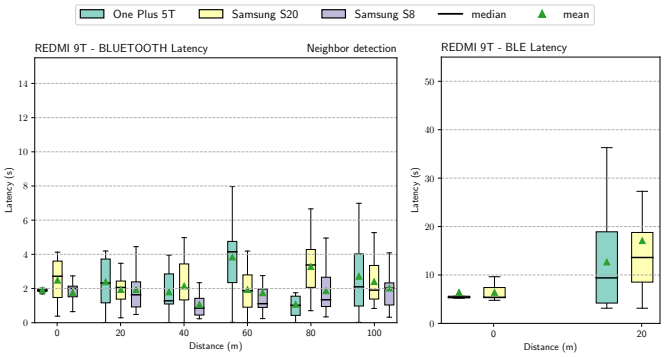


Fig. 6: Bluetooth Classic and BLE discovery latency using the Redmi 9T as a discovery device.

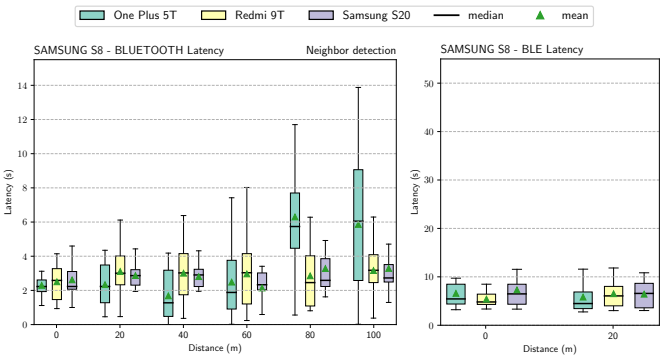


Fig. 8: Bluetooth Classic and BLE discovery latency using the Samsung S8 as a discovery device.

discovery delay via BLE, the Samsung S20 performs similarly to the Redmi 9T. Furthermore, as with Bluetooth Classic, the disparity in average discovery delay is narrower.

Finally, focusing on the Samsung S8 discovery delay data (Figure 8), 75% of the values per case using Bluetooth Classic remain below $4\,\mathrm{s}$, except for the OnePlus 5T at $80\,\mathrm{m}$ and $100\,\mathrm{m}$. Also, when comparing these results with the values in Figure 5, a similarity is observed at $100\,\mathrm{m}$, where we can intuit that the same physical impediment affected the results. Regarding BLE, the Samsung S8, unlike the other devices, maintains the neighbor detection delays as the devices move away. In fact, 75% of the delays in each case do not exceed $9\,\mathrm{s}$. Therefore, the Samsung S8 has the best performance to find an advertiser with BLE.

B. Connection establishment delay

In contrast to neighbor detection delays, connection establishment delays are faster. If we focus on the values obtained with the OnePlus 5T (Figure 9), we can see that, with Bluetooth Classic, the mean values in each case are between $0.5\,\mathrm{s}$ and $0.2\,\mathrm{s}$. Moreover, each mean value maintains a slight difference in most cases as the devices move away. In the case of the transmission speed of each connection establishment, the graph shows that the OnePlus 5T has faster responses with the Samsung S20, followed by the Redmi 9T and finally the Samsung S8. As for the data obtained with BLE, contraryly to the neighbor detection delays, the connection establishment delays are four times faster than when using Bluetooth Classic, with average values between $0.08\,\mathrm{s}$ and $0.07\,\mathrm{s}$.

Moving on to the Redmi 9T (Figure 10), with Bluetooth Classic, 75% of the delays obtained by the Samsung S20 and the OnePlus 5T are between $0.3\,\mathrm{s}$ and $0.2\,\mathrm{s}$. As for the Samsung S8, its delay increases as it moves away from the Redmi 9T, placing its average delays between $0.25\,\mathrm{s}$ and $0.44\,\mathrm{s}$. Concerning its performance with BLE, it performs similarly to the Redmi 9T, with faster values compared to those obtained with Bluetooth Classic. However, unlike the previous case, the Redmi 9T is $0.03\,\mathrm{s}$ slower.

Then with the Samsung S20 (Figure 11), the OnePlus 5T and the Samsung S8 with Bluetooth Classic maintain a similar behavior to the previous case (Figure 10). In contrast, the Redmi 9T presents a greater dispersion, with 75% of its values between $0.1\,\mathrm{s}$ and $0.3\,\mathrm{s}$, although with an average between $0.2\,\mathrm{s}$ and $0.3\,\mathrm{s}$. As for its average delays with BLE, the Samsung S20 has the lowest delay compared to other devices, with an average of $0.06\,\mathrm{s}$ per case.

Finally, considering the data obtained by the Samsung S8 (Figure 12), we can observe that, using Bluetooth Classic, the OnePlus 5T and the Samsung S20 have similar delays. Moreover, in each case tested with the Samsung S8, 75% of

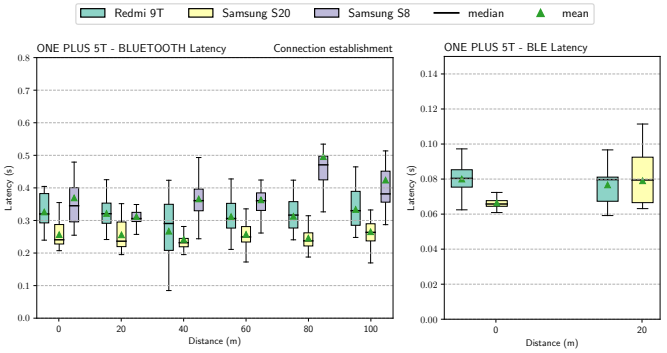


Fig. 9: Bluetooth Classic and BLE connection establishment delay using the OnePlus 5T as a discovery device.

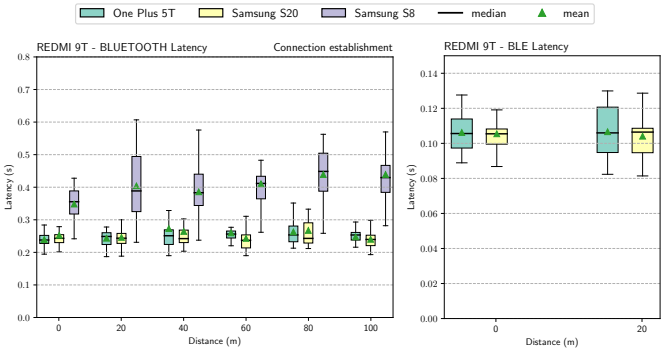


Fig. 10: Bluetooth Classic and BLE connection establishment delay using the Redmi 9T as a discovery device.

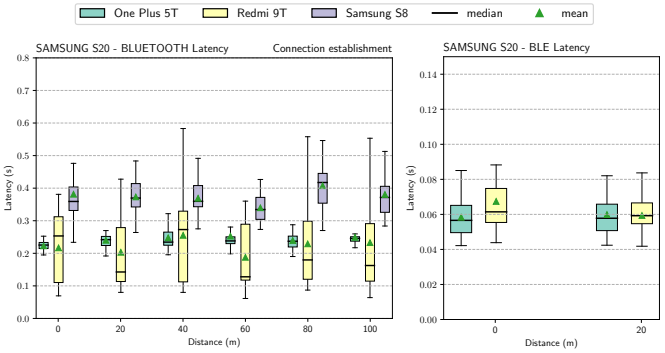


Fig. 11: Bluetooth Classic and BLE connection establishment delay using the Samsung S20 as a discovery device.

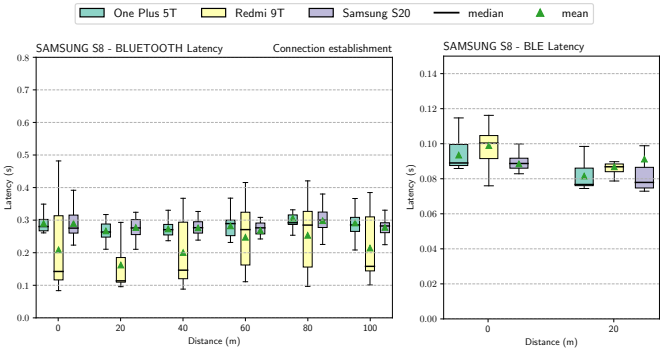


Fig. 12: Bluetooth Classic and BLE connection establishment delay using the Samsung S8 as a discovery device.

the delays are below $0.3\,\mathrm{s}$. Meanwhile, the Redmi 9T has a lower average delay than the other cases since 50% of its average values are between $0.1\,\mathrm{s}$ and $0.2\,\mathrm{s}$ at $20\,\mathrm{m}$ and $0.1\,\mathrm{s}$ and $0.3\,\mathrm{s}$ at the other distances. As for its performance with BLE, its average values remain between $0.9\,\mathrm{s}$ and $1\,\mathrm{s}$.

C. Lessons learned

Let's compare the results obtained during this experimentation. First, we can state that the latency of the D2D link establishment depends directly on the search for beacons sent from an advertiser device. More specifically, running Bluetooth Classic on a device takes approximately seven times longer to find a beacon than to process a connection establishment. In contrast, with BLE, takes approximately 62 times longer. On the other hand, in terms of the neighbor detection performance, we can observe that Bluetooth Classic performs better than BLE, as it can operate up to distances of $100\,\mathrm{m}$ and is approximately 2.5 times faster. However, in terms of the connection establishment process performance, we can observe that BLE performs better in terms of delay, being approximately three times faster than Bluetooth Classic.

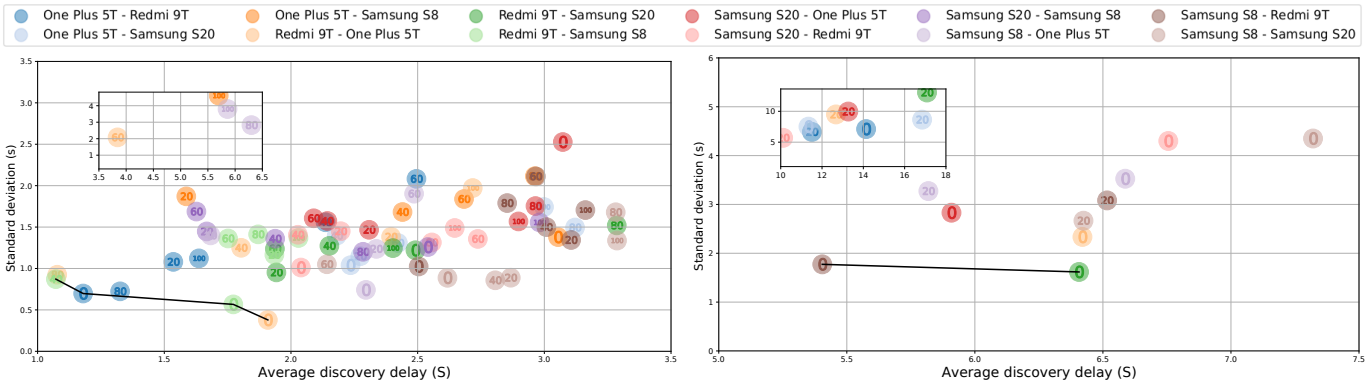
In terms of dispersion, Figure 13 shows that the delays per detection attempt using Bluetooth Classic are five times lower than using BLE. Furthermore, Figure 13a (Bluetooth Classic) shows that the delay range is between $1\,\mathrm{s}$ and $3.5\,\mathrm{s}$, while in Figure 13b (BLE), the average delays are between $5\,\mathrm{s}$ to $7.5\,\mathrm{s}$ and $10\,\mathrm{s}$ to $18\,\mathrm{s}$.

Finally, we can observe that regardless of the technology used to perform the link establishment, the delays observed do not vary as the devices move away from each other.

V. Related Work

Several studies evaluate the efficiency of D2D in different technologies and offer proposals to speed up the neighbor discovery process. Hayat et al. compare several in-band and out-of-band algorithm protocols, taking discovery latency, energy efficiency, and mobility into account [11]. Their study concludes that discovery through neighbor discovery affects the behavior of such algorithms and routing techniques. In a companion paper, the authors provide a list of possible ways to perform D2D neighbor discovery in underlying cellular networks, pointing out the limitations of existing mechanisms and proposing possible directions for future research [12].

Other studies evaluate the energy consumption for the link establishment process. Usman et al. present an energy consumption model for devices performing neighbor discovery using Wi-Fi Direct, showing that beacon exchange produces a significant energy cost [13]. In the literature, there are multiple criteria to select a specific neighbor from a set of potential neighbors depending on the metrics required by the target



(a) Bluetooth Classic.

(b) BLE.

Fig. 13: Neighbor detection average delay vs. standard deviation where the distances are represented inside the circles.

function: transmission rate [14], energy consumption [15], and network lifetime [16]. However, few studies have focused on evaluating the delay performance of the neighbor detection process in real environments. The work the closest to ours is Medel and Brito's comparison of discovery methods between Wi-Fi Direct, Bluetooth Classic, and BLE technologies [7]. Cerio et al. contributed to the field by analyzing several scanning devices from different BLE chipset manufacturers and showing that they do not perform as expected in their scanning process, which the authors claim to have a severe impact on the performance of the discovery process [17].

VI. Conclusion

We focused on understanding the consequences of the delay that Bluetooth Classic and BLE take to discover a device and to establish a connection between them. We observe from experimentation that most of the delay in establishing a connection goes towards the beacon search to discover a neighboring device. Although BLE is more energy-efficient, the numbers show that Bluetooth Classic performs better, both in terms of range and speed, requiring approximately 2 s to discover a device within a 100 m range (where BLE takes between 8 and 20 s to detect a neighbor within 20 m). On the other hand, BLE has a better performance for exchanging point-to-point messages over short distances; thus, it is possible to use it in short-range D2D connection processes or other possible short-range D2D applications. We intend to run the experiments with more devices and in other locations (indoors and outdoors) as future work.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 957246.

References

[1] M. K. Pedhadiya, R. K. Jha, and H. G. Bhatt, "Device to device communication: A survey," *Journal of Network and Computer Applications*, vol. 129, 2019.

[2] L. Shan *et al.*, "Local information sharing system with wireless device-to-device communications," *IEEE Access*, vol. 8, 2020.

[3] F. Rebecchi *et al.*, "Data offloading techniques in cellular networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, pp. 580–603, 2019.

[4] P. U. Tournoux *et al.*, "The accordion phenomenon: Analysis, characterization, and impact on DTN routing," in *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, 2009, pp. 1116–1124.

[5] S. Gaito, E. Pagani, and G. P. Rossi, "Opportunistic forwarding in workplaces," in *Proceedings of the ACM workshop on Online social networks (WOSN)*, 2009, pp. 55–60.

[6] Google, "Nearby Connections – API Overview." [Online]. Available: https://developers.google.com/nearby/connections/

[7] A. Rodriguez Medel and J. M. Camara Brito, "A comparison among wi-fi direct, classic bluetooth, and bluetooth low energy discovery procedures for enabling massive machine type communications," in *Proceedings of the International conference on Internet of Things, Big Data and Security (IoTBDS)*, 2021, pp. 164–169.

[8] R. Friedman, A. Kogan, and Y. Krivolapov, "On power and throughput tradeoffs of wifi and bluetooth in smartphones," *IEEE Transactions on Mobile Computing*, vol. 12, 2013.

[9] A. Abedi, O. Abari, and T. Brecht, "Wi-le: Can wifi replace bluetooth?" in *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets'19)*, 2019, p. 117–124.

[10] G. D. Putra *et al.*, "Comparison of energy consumption in wi-fi and bluetooth communication in a smart building," in *Proceedings of the IEEE Computing and Communication Workshop and Conference (CCWC)*, 2017, pp. 1–6.

[11] O. Hayat *et al.*, "Device discovery in d2d communication: A survey," *IEEE Access*, vol. 7, 2019.

[12] O. Hayat, R. Ngah, and Y. Zahedi, "In-band device to device (d2d) communication and device discovery: A survey," *Wireless Personal Communications*, vol. 106, 2019.

[13] M. Usman *et al.*, "An energy consumption model for wifi direct based d2d communications," in *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.

[14] F. Ouyang *et al.*, "Collision resolving relay selection in large-scale blind relay networks," *Wireless Networks*, vol. 23, 2017.

[15] C. V. Anamuro *et al.*, "Energy-efficient discovery process for mmtc applications," in *Proceedings of the IFIP Wireless and Mobile Networking Conference (WMNC)*, 2019, pp. 79–86.

[16] A. Alsharoa *et al.*, "An energy-efficient relaying scheme for internet of things communications," in *Proceedings of the IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6.

[17] D. Perez-Diaz de Cerio *et al.*, "Analytical and experimental performance evaluation of ble neighbor discovery process including non-idealities of real chipsets," *Sensors*, vol. 17, 2017.