NOD: Lightweight Continuous Neighbor Discovery on Everyday Devices

Hsiao-Yuan Chen(✉), Evan King, and Christine Julien

The University of Texas at Austin, Austin, TX 78712, USA
{littlecircle0730,e.king,c.julien}@utexas.edu

Abstract. Continuous neighbor discovery (CND) allows wirelessly enabled devices to build a continuously updated picture of nearby devices. Such situational awareness is useful in many circumstances. While feasible on commodity devices like smartphones and wearables devices, CND is often not enabled due to practical limitations and concerns. Therefore, protocol implementations for CND have been mostly relegated to purpose-built sensor network deployments. However, the emergence of lightweight IoT devices and the need to rapidly discover what capabilities are around has renewed interest in CND on commodity devices. An important principle of CND protocols is their ability to be parameterized to achieve probabilistic guarantees regarding timeliness and assuredness of discovery events. However, transitioning from controlled deployments to commodity devices raises issues in providing guarantees related to CND. We explore two such concerns: (1) packet loss due to communication competition and (2) the time required to transition between sending and scanning for beacons. We propose a new CND protocol, **NOD**, that accounts for the impacts of these practical issues. We validate **NOD** through extensive simulations framed by real-world deployment settings and a proof-of-concept implementation for Android. **NOD** considers real-world application environments, allowing applications to identify optimal settings for guaranteeing probabilistic discovery in practical scenarios.

Keywords: Continuous Neighbor Discovery · Bluetooth Low Energy

1 Introduction

Continuous neighbor discovery (CND) entails continuously using on-device communication capabilities to detect ambient devices and collect information about what other devices and capabilities are in the surroundings. Protocols for energy-efficient CND are useful in many circumstances. For example, CND was fundamental to digital contact tracing [6] as demanded by the COVID-19 pandemic. CND has also been suggested in IoT applications, including those for smart cities [16]. Historically, CND is primarily implemented in wireless sensor networks [1,2,4,12,14], but with emerging lightweight and energy-efficient

A. Zaslavsky et al. (Eds.): MobiQuitous 2023, LNICST 593, pp. 3–25, 2024.
https://doi.org/10.1007/978-3-031-63989-0_1

technologies on mobile devices, CND has expanded to include even ordinary smartphones, broadening the reach of these protocols. In particular, Bluetooth Low Energy has become a technology of choice for implementing CND [8,10].

Generically, CND protocols consist of a schedule that alternates between sending beacons to announce presence and listening for others' beacons. When a CND protocol is designed on a purpose-built sensor network, many practical concerns can be assumed or designed away. However, making CND work on commodity sensing devices and smartphones requires considering practical constraints that arise when low-level control of the radio states is not possible and the communication medium is shared. Protocols have started to consider these constraints, for instance BLEnd [8] considers the significant impact of packet collisions on the probability of discovering other nearby devices. However, BLEnd only considers collisions that occur with other devices as they also perform continuous neighbor discovery. Kindt et al. consider some properties of real radios that impact the correctness of discovery [11] but fail to consider the performance of neighbor discovery in very dense deployments where collisions are problematic.

As a result, diverse practical considerations remain unaddressed in the application of CND on everyday multi-purpose networks like those that connect smartphones with each other and with wirelessly available resources in their environments. We focus on two such practical considerations, which we refer to as *packet loss* and *warmup interval*. The first concern is caused largely by collisions that occur between wirelessly transmitting devices, whether the collisions happen because two neighbor discovery beacons collide or because a neighbor discovery beacon is impacted by other wireless communication in a busy wireless environment. In noisy environments like cities, homes, and offices, packets may be lost seemingly randomly. The second concern, which we term *warmup*, arises because CND protocols that rely on strict timing requirements of listening and beaconing neglect the fact that the wireless radio cannot immediately switch modes. Instead, there is a period of time that precedes the start of both listening and beaconing before the device can successfully receive or send data. This phenomenon has been recognized by others as well; it has also been referred to as *turnaround time* [11].

As CND shifts from MCU-class devices in sensor networks to smartphones, new protocols are necessary. We leverage existing CND design work, identify limitations when applied to smartphones, and develop new models and protocols for this context. Our research reveals significant impacts of packet loss and radio command timing on traditional CND protocols. Despite these protocols not being tailored for smartphones, we quantify these impacts on their performance. To address these issues, we introduce **Nod**[1], a CND protocol considering packet loss and warmup intervals. We construct an analytical model to evaluate **Nod** in dense, varied networks. Compared to applying traditional CND protocols on smartphones, **Nod** notably enhances CND performance, especially in discovery guarantees. Concretely, the contributions of this paper are:

[1] The name **Nod** is an homage to a subtle head-tilt upon meeting, a greeting that can be lightweight and ephemeral.

- We quantify the impact that practical considerations of packet loss and warmup have on discovery performance in existing continuous neighbor discovery protocols.
- We propose a new protocol and associated analytical model for neighbor discovery that addresses the real-world issues of packet loss and warm-up interval. The new model builds on existing work to provide a guaranteed discovery probability within a specified discovery latency.
- We demonstrate the improvements possible with a protocol that implements our updated model and quantify tradeoffs relative to energy consumption.

The remainder of this paper is organized as follows. Section 2 provides motivation and related work. In Sect. 3, we empirically demonstrate the impact of packet loss and warmup interval in existing CND protocols. Section 4 presents **Nod**, a new protocol and associated analytical model of CND that accounts for packet loss and warmup intervals, and Sect. 5 shows the performance of **Nod**, given a model that considers real-world behavior.

2 Background and Related Work

Given the interest in CND in wireless sensor networks, diverse CND protocols have been proposed. Protocols can be generally separated into two groups: slotted protocols and slotless ones. Both aim to optimize neighbor discovery performance against energy consumption by defining a *schedule* that defines when a participating node should listen and when it should beacon. Slotted protocols break time into equal sized chunks called *slots* and schedule periods of advertising and scanning based on these slots. Slotless protocols, while maintaining a repeating fixed-length schedule, offer more flexibility. In this section, we describe a few protocols in each class; we use three of these protocols in our motivating empirical study in Sect. 3.

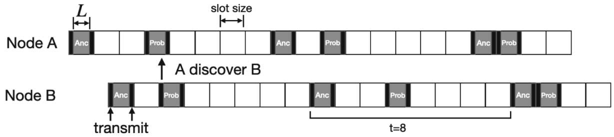
Generically, CND protocols try to balance three concerns:

λ: a discovery latency, i.e., the time between when a neighboring node appears and when it is discovered;

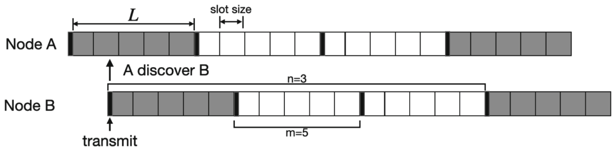
ρ_Λ: a discovery probability, i.e., the probability of discovering a neighbor within some target discovery latency Λ; and

η: the duty cycle of neighbor discovery, i.e., the fraction of time a device's radio actively participates in CND.

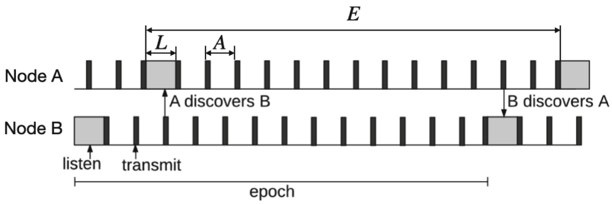
Slotted Neighbor Discovery Protocols. In a slotted protocol, the length of a slot is often determined by practical aspects of the radio. The protocols differ most substantially on how they choose to use slots (i.e., during which slots the protocol is *active* and which slots it is *idle*) and how they schedule beaconing and listening activities within active slots. Some of the earliest protocols achieved good performance in choosing active slots probabilistically [13,17]. The analytical models of these protocols could then assure probabilistic discovery by examining the probability of overlap in two devices' active slots.



(a) A Searchlight schedule with $t = 8$. Anchor slots are labeled as 'Anc' and Probe slots are labeled as 'Prob'.



(b) A Nihao schedule with $m = 5$ and $n = 3$



(c) An F-BLEnd schedule

Fig. 1. Example schedules for existing protocols

These protocols often abstract the details of activities *internal* to an active slot, assuming that overlapping active slots between two nodes lead to discovery. Many protocols overlook the impact of collisions: if three nodes have overlapping active slots, they are assumed to discover each other, despite practical radio limitations. The Birthday protocol [13] is an exception; its model does consider potential discovery event losses due to multiple aligned active beaconing slots. In our analysis in Sect. 3, we use two slotted protocols: Searchlight [1] and Nihao [14], selected for their recognition and distinct approaches to allocating time for listening versus beaconing.

Figure 1(a) illustrates a Searchlight [1] schedule. It features two active slots per period (with period length t slots), placing beacons at each active slot's start and end, with a listening interval in between. This structure ensures a discovery chance when active slots overlap. Searchlight uses anchor slots at each period's start and probe slots in a random order within the first $t/2$ non-anchor slots, ensuring eventual overlap between one node's anchor slot and another's probe slot. For instance, in Fig. 1(a), Node A follows the pattern {3, 2, 1, ...}, and Node B {2, 3, 1, ...}.

Figure 1(b) shows the Nihao schedule [14]. Nihao is characterized by two parameters, m and n and follows a repeating pattern every $m * n$ slots, which we refer to as an *epoch*. During each epoch, each node schedules a beacon at the beginning of every m slots, and then each node listens for a duration of m slots after their first beacon in the epoch. This pattern repeats for the duration of the

Nihao schedule. This schedule ensures nodes discover each other as long as the start times of two nodes do not align with each other. Specifically, if the start time of one node is at least b (the length of a beacon) before or after another node's start time, then the two nodes have a chance to discover each other.

Slotless Neighbor Discovery Protocols. In contrast to slotted protocols, slotless protocols are not driven by slots of fixed length. Instead, they define a repeating schedule of beaconing and listening [8,10]. Some protocols schedule beacons independently of listening, so a beacon may be scheduled in the middle of a listen, in which case the node stops listening, sends the beacon, then starts listening again [10]. In general, the key parameters of a slotless protocol are the frequencies of listen and beacon activities and the length of a schedule epoch.

Because slotless protocols have similar timing constraints relative to our practical considerations, we choose a single such protocol for study: BLEnd [8]. As shown in Fig. 1(c), an *epoch* in BLEnd starts with a period of listening, immediately followed by a period of advertising. The full BLEnd protocol advertises for the remainder of the epoch, as depicted in Fig. 1(c). The lengths of the listen period (L) and the advertising interval, (A) (i.e., the length of time between the start time of two successive beacons) are designed to ensure the protocol's discovery latency and probability targets. BLEnd can be configured by defining the length of an epoch, E (which can be of an arbitrary duration because BLEnd is slotless), and the length of the advertising interval, A. The length of a listen period is derived from the advertising interval. Adjusting E and A in BLEnd can respond to different application contexts, including the expected number of nearby nodes or to achieve different goals in terms of ρ_{Λ}.

Practical Considerations for CND. Several aspects of wireless communication affect CND performance; we consider two facets, both of which are important in making CND compatible with real-world devices. The first challenge of these, *packet loss*, refers to the fact that beacons may be not received, even after accounting for collisions with other neighbor discovery beacons. These packet losses may be due to interference with other communication in the environment, because of idiosyncrasies of the mobile operating systems, or several other causes. For instance, in [15], the authors demonstrate that, even without additional BLE devices competing for the wireless medium, there is still a packet loss rate of more than 13% for off-the-shelf smartphones. Additional experiments demonstrate packet loss rates that are between 4.55% to 39.47% [5].

The second challenge, *warmup interval*, relates to the fact that just because an application instructs a radio to change from one modality to another does not mean that the change happens instantaneously. On one hand, this is physically impossible, even when the application executes very close to the radio layer [11]. On the other hand, CND protocols ask radios to be in one of three states at any given time: beaconing, listening, or idle. The transition from any one of these states to any of the other three takes a non-negligible amount of time as a result of physical, application, and operating system constraints, during which the radio is not able to do anything (it can neither receive beacons nor send them). We capture two distinct *warmup* intervals: the time it takes a device to transition into listening (ψ_L) and the time it takes a device to transition into beaconing (ψ_A). We conducted measurements to empirically determine the warmup intervals of various devices, and the average warmup of 5 trials are presented in Table 1. The results demonstrate that warmup intervals range up to more than one hundred milliseconds. Compared to a common CND epoch length of a few seconds, the warmup interval is non-negligible and can dramatically impact discovery probability, as we see in the next section.

Table 1. Measured warmup period on different devices. Each value is averaged over 5 trials and shown with 95% confidence intervals.

	Google Pixel 3	Samsung S7	Huawei N29
Advertising warmup (ψ_A) (ms)	257.3 ± 23.5	4.23 ± 23.5	264.7 ± 10.6
Scanning warmup (ψ_L) (ms)	42.8 ± 5.4	149 ± 4.9	49 ± 1.5

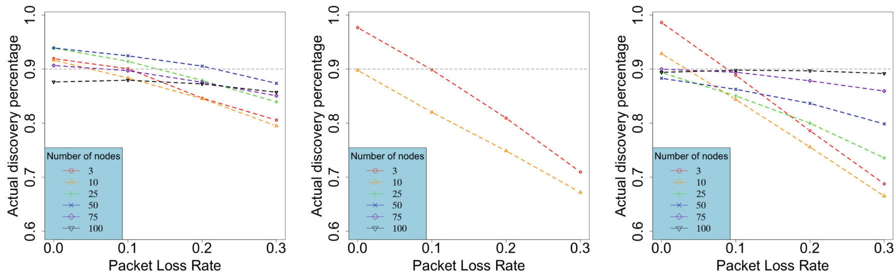
3 Neighbor Discovery in the Wild

While existing neighbor discovery protocols consider some realistic implications on performance, key challenges remain unaddressed. In particular, while the impact of collisions on discovery has been noted [8,9], neighbor discovery protocols have failed to address the potential impacts of random *packet loss* and communication *warmup intervals*. We empirically demonstrate the impact of these two concerns on discovery probability (ρ_A) and discovery latency (λ)—the two hard constraints of many CND protocols—on three neighbor discovery protocols.

To support our empirical analysis and evaluations, we developed a Java-based discrete event simulator for CND. It allows specifying various neighbor discovery schedules such as Searchlight [1], Nihao [14], or BLEnd [8], and simulates random schedules for nodes within range. The simulator[2] emulates BLE radio behavior typical in Android devices, assuming a 1.1 ms beacon length based on empirical data and the BLE standard [18]. While numerous network simulators exist, we opted to implement this simple event simulator because we are not interested in node mobility or the behavior of the network stack, which are the purposes of existing simulators. Instead, we are interested in probabilistic overlap of discovery events. Specifically, our simulator simply analyzes generic neighbor discovery schedules for a set of nodes that are all assumed to be within range of each other simultaneously. It accounts for collisions among neighbor discovery beacons, and it can also be adjusted to inject both random packet losses and arbitrary warmup intervals.

To assess the impact of packet loss, we studied CND protocol behavior with packet loss rates from 0% (no loss) to 30%, in 10% increments, based on the

[2] https://github.com/UT-MPC/CND_simulator.



(a) Searchlight (b) Nihao (c) BLEnd

Fig. 2. Impacts of packet loss on discovery probability. All protocols are configured to attempt to achieve 90% probability of discovery within 10 s. Random packet losses range from 0% of packets to 30% of packets (x-axis).

prevalence of such rates in smartphone communications [5]. For communication warmup intervals, including radio and OS/application delays, we used intervals of 0 (no warmup), 50 ms, 100 ms, 150 ms, and 200 ms, reflecting observed ranges (50–150 ms) with slight adjustments to 0–200 ms 1. We analyzed these factors across different node counts (3, 10, 50, 100, 150, 200) to represent various social scenarios, from home settings (3 nodes) to larger gatherings like concerts or parties (up to 200 nodes).

We set a target discovery latency $\Lambda = 10s$ and a discovery probability $\rho_\Lambda = 0.9$ to ensure high confidence in data retrieval while minimizing discovery time. This is particularly relevant in applications like opportunistic recommendation exchanges in smart cities [7], where quick device discovery (under 10 s) on transient interactions, like on a subway or street corner, is crucial for completing further steps like authentication and data exchange. Because we assume implementations that will function on commodity devices, we seek compatibility with Bluetooth Low Energy (BLE). Because the Android operating system (similar to other mobile operating systems) limits on advertising intervals for BLE, we adopt Android's *balanced* mode interval of 250 ms, balancing energy cost and delay. Consequently, we set Searchlight's slot size to 251.1 ms, accommodating two full beacons per slot. For Nihao, with beacons only at slot starts, the slot length is 250 ms. In BLEnd, we choose $A = 250$ ms.

Impacts of Packet Loss. Network packets can be lost for various reasons. While some losses, like collisions with other neighbor discovery beacons, can be modeled, others are random. Thus, a listener might not receive every beacon, even if it is actively scanning and within range. To evaluate the effect of packet loss on neighbor discovery in our protocols, we introduced random beacon loss into the simulation, observing protocol behavior amidst other devices also engaged in discovery. These losses are additional to those from collisions, which the simulator already models.

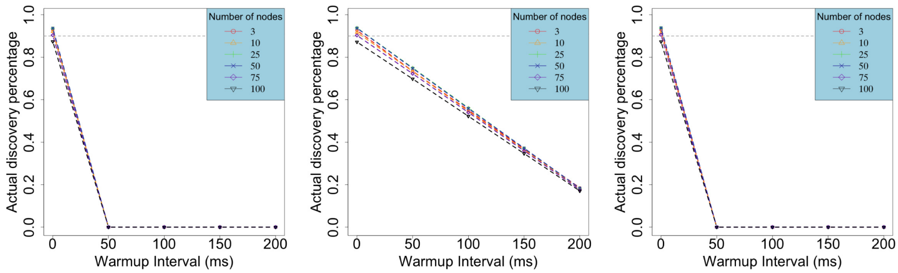
Figure 2 shows how CND protocols are impacted by random packet losses. First, on average, discovery probability decreases as packet loss rate (the x-axis) increases for all three protocols. Searchlight's performance is relatively consistent across all densities of nodes; BLEnd performs *better* as the number of surrounding nodes increases. This is due to the fact that BLEnd are tuned to *talk more* so they are more sensitive to collisions between neighbor discovery beacons, which is even more impactful with larger numbers of nodes. As beacons are randomly lost, collisions go down, which actually improves performance. As Fig. 2 shows, not explicitly considering packet loss in tuning a protocol for CND misses the mark for the target performance measures.

One observation is the lack of data for Nihao with more than 10 nodes. This is due to Nihao's inability to reach the desired 0.9 discovery probability, considering the detrimental effects of beacon collisions. Nihao and Searchlight's fixed beacon positions at each slot's start and end lead to similar scheduling across epochs. While Searchlight uses a probe slot to mitigate this, Nihao's beacons remain consistently positioned, heightening collision risks. With more nodes, collisions intensify, hindering Nihao's ability to meet the target discovery probability.

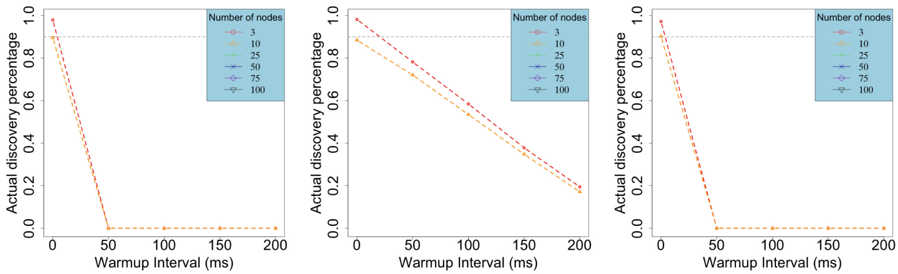
Impacts of Warmup Interval. The second real-world device characteristic we examine is the *warmup interval*, the time required for a device to transition to a transmitting or listening state, encompassing delays from the physical radio, operating system, and application-level. Typically, CND protocols like BLEnd [8], Searchlight [1], and Nihao [14] assume zero transition time, with precise scheduling of beacons. Kindt et al. [11] have recognized the impact of *turnaround time* on continuous neighbor discovery. The Searchlight authors also noted this and introduced a pre-slot to start service earlier as a workaround.

Figure 3 illustrates the impact of increasing warmup intervals on the protocols in three scenarios. First, we explore an advertising warmup interval (ψ_A) with values $0, 50, 100, 150, 200$. Second, we consider a scanning warmup interval (ψ_L) with the same values. Lastly, we assess the combined effects of ψ_A and ψ_L, using five pairs: (0, 0), (50, 50), (100, 100), (150, 150), (200, 200). Nodes cannot scan or advertise during warm-up intervals, so discovery probability inversely relates to warm-up length; listening capability diminishes as the warm-up duration increases relative to the original listening period. Searchlight and Nihao, scheduling beacons at slot edges, fail to send beacons with any advertising warmup, leading to a drop to 0 discovery probability when $\psi_A \neq 0$ in Fig. 3(a), (c), (d), (f). BLEnd, advertising continuously post-scanning, circumvents this issue.

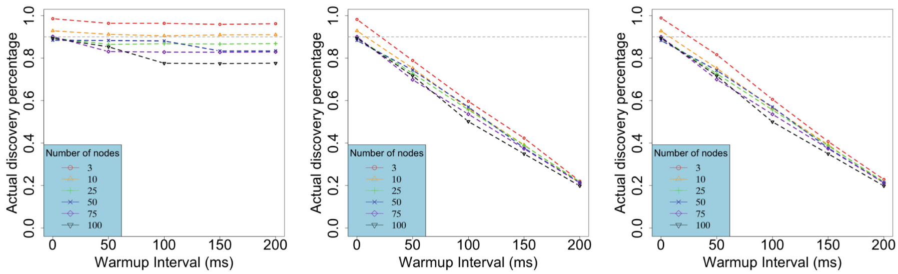
Figure 4 shows the impact of scanning warm-up and packet loss on discovery probability and lifetime for 50 nodes using BLEnd. We estimated energy consumption assuming CND as the sole activity, based on current draw (including warm-up) and power data from [3]. While power profiles vary by device, relative magnitudes are expected to follow similar trends. As Fig. 4(a) demonstrates, increasing packet loss or warm-up severity decreases discovery probability. The figure illustrates how these two factors can result in a notable reduction in the overall discovery probability, emphasizing the importance of considering both practical issues when designing and analyzing beacon scheduling protocols. Under the same conditions, a node's lifetime actually *increases*, indicating that



(a) Searchlight with ψ_A (b) Searchlight with ψ_L (c) Searchlight with ψ_A and ψ_L



(d) Nihao with ψ_A (e) Nihao with ψ_L (f) Nihao with ψ_A and ψ_L

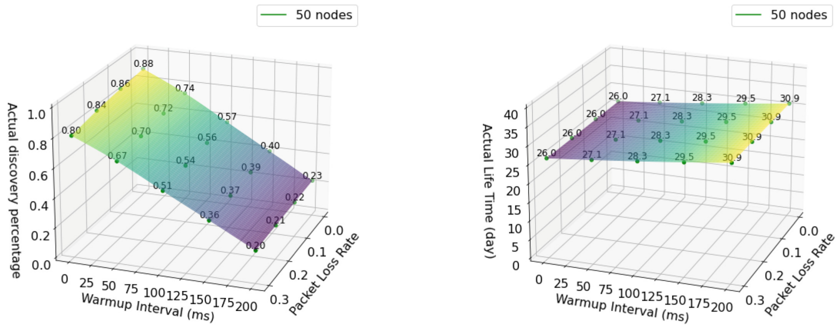


(g) BLEnd with ψ_A (h) BLEnd with ψ_L (i) BLEnd with ψ_A and ψ_L

Fig. 3. Impacts of different warm-up intervals on discovery probability. All protocols are configured to attempt to achieve 90% probability of discovery within 10 s. Warmup intervals range from 0 to 200 ms (x-axis).

the protocol is doing less work than the model anticipates (i.e., because of the warmup intervals, the radio is spending less time actively scanning or receiving).

Continuous neighbor discovery protocols designed for commodity smartphones must consider these practical impacts, whereas protocols tuned for MCU-class devices with tight timing constraints can get away with ignoring them. We next introduce a revised CND model and protocol that generates schedules in a way that is sensitive to random packet loss and arbitrary warm-up intervals.



(a) Discovery Probability (b) Lifetime

Fig. 4. Impact on discovery probability and lifetime of BLEnd with consideration in the presence of both packet loss and scanning warmup for 50 nodes. All protocols are configured to attempt to achieve 90% probability of discovery. Warm-up intervals range from 0 ms to 200 ms for both ψ_A and ψ_L. Packet loss rates range from 0 to 0.3.

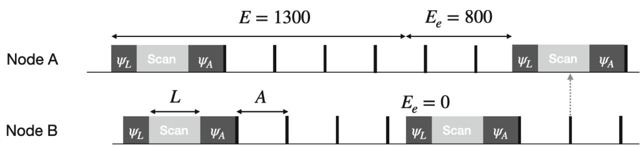


Fig. 5. Two nodes with different **Nod** schedules.

4 Solving Discovery with Real-World Practicalities

We next present a new model for continuous neighbor discovery that addresses the packet loss and warmup interval issues encountered in practical scenarios. Based on the results in Sect. 3, we adopt a slotless approach, as its less rigid and more flexible scheduling is better able to overcome these practicalities. We first introduce the basics of probabilistic slotless discovery and then use this as a basis for our new model that accounts for packet losses and warmup intervals.

4.1 Nod: CND with Constrained Schedules

To address practical issues such as packet loss and warmup intervals, we propose a new slotless protocol called **Nod**, as depicted in Fig. 5.

In **Nod**, a node follows a repeating pattern of epochs, each varying in duration, expressed as $E_r + E_e$. Here, E_r is a constant (*root* epoch), and E_e is an *extra* part, randomly chosen from $[0, E_r]$. This design makes each **Nod** epoch's length randomly and independently fall in $[E_r, 2E_r]$. An epoch begins with a listening interval L and continues with advertising at a fixed interval A.

Notice that the schedules depicted in Fig. 5 account for ψ_A and ψ_L so that the design can account for their impacts on discovery probability. We discuss three of these design decisions in detail in the following. In particular, we first examine how ψ_L impacts the necessary settings for L. We then examine the effect of decoupling the length of an epoch and A, the advertising interval. Finally, we discuss the rationale of choosing E_e to be in the range $[0, E_r]$ to probabilistically ensure that two nodes whose schedules are synchronized in one epoch are unlikely to be synchronized in the next epoch, boosting the probability of discovery.

Ensuring a Sufficient Functional Listening Interval. Figure 6(a) shows a schedule from a naïve slotless protocol with a fixed epoch length and no consideration of warmup intervals. To ensure that two nodes, in isolation, are guaranteed to discover one another if one's listen interval overlaps the other's advertising interval, the schedule needs to ensure that L is sufficiently long to capture an entire beacon regardless of the shift between the two schedules. To ensure this, one must make $L \geq A + b$ [8] (as shown in Fig. 6(a))[3]. However, existing CND protocols assume a zero time warmup interval to transition from an idle state to a scanning one. Table 1 shows this is not the case; indeed the transition time is in the same order as commonly chosen listening intervals. Therefore, as Fig. 6(b) shows, a significant fraction of the listen interval can actually be *non-functional*, which is the direct cause of the steep declines we see in Figs. 3(b), (e), and (h). Fortunately, mitigating this impact is quite simple—we simply extend the size of the listening interval to include ψ_L as shown in Fig. 6(c), i.e., $L = A + b + \psi_L$, ensuring that the functional listening period remains at $A + b$. Effectively, we extend the listen interval (L) to ensure that an epoch's active scanning time is guaranteed to capture at least one of any pair of successive beacons of any neighboring node, reconfirming a design tenet of the original schedule.

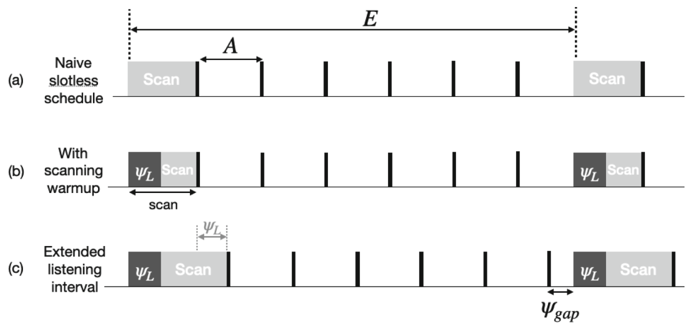


Fig. 6. Illustration of schedules in different scenarios: (a) original schedule (without warmup); (b) original schedule (with a scanning warmup); (c) **Nod** extends the listen interval (L) to restore the full functional scanning period

[3] In fact, when employing BLE radios, $L \geq A + b + s$, where s is some random slack added to the advertising interval and outside of the control of the application [18]. We account for s in our models and implementation but omit it here for simplicity.

That Last Beacon. In Figs. 5 and 6, the independence of epoch length and advertising interval leads to a fractional space, ψ_{gap}, at each epoch's end. One proposed solution is to position a beacon just before the listen interval [8], but warmup intervals hinder this approach. Firstly, the scanning can't immediately succeed the last beacon due to ψ_L. Secondly, the irregular timing between the hypothetical 'last' beacon (dashed in Fig. 7) and its predecessor necessitates an additional ψ_A, but ψ_{gap} might be shorter than ψ_A itself. Hence, rather than trying to 'fix' the schedule, we acknowledge the diminished discovery likelihood due to ψ_{gap}, which, along with ψ_A and ψ_L, constitutes another non-functional part of the discovery schedule. In particular, given that the end of the epoch is equally likely to fall anywhere within some advertising interval, the resulting expected value of ψ_{gap} is $\frac{A}{2}$.

Fig. 7. Illustration of ψ_{gap} and its potential incompatibility with ψ_A

Overlapping Epoch Start Times. With fixed epoch lengths, if one node chooses a start time that is nearly the same as another, the two nodes will never discover one another, effectively because they will always be aligned to be scanning at the same time. However, when we consider how the warmup intervals extend the fraction of an epoch that is non-functional, this potential alignment problem can be devastating to any effort to achieve high discovery probabilities.

Even with the extension of the listening interval, a node still has a lower chance of being discovered compared to when no warmup interval is present. As illustrated in Fig. 8, with the presence of a warmup interval, an advertiser must initiate its schedule either $\psi_L + \psi_{gap}$ after the scanner's start time or ψ_A before the scanner's start time, in order for one of its beacons to fall into the scanner's listening interval. That is, if an advertising neighbor's schedule start time is within the red block of the scanner's schedule in Fig. 8, discovery will not happen.

To tackle the consistent shift issue between nodes, we introduce irregular epoch lengths in **Nod**. We construct an epoch from two parts: E_r, a shared *root* interval, and E_e, a random addition chosen from $[0, E_r]$. Thus, an epoch's length, $E = E_r + E_e$, falls between $[E_r, 2E_r]$. We cap E_e at E_r to optimize the chance for a node to move out of the non-functional area in subsequent epochs. Selecting a smaller maximum for E_e might allow nodes to shift out of the non-functional area, but with low probability and potential for quick re-entry. A larger maximum causes epochs to wrap around, proving ineffective. Thus, we set E_e's maximum equal to E_r. This gives nodes an equal opportunity to shift within

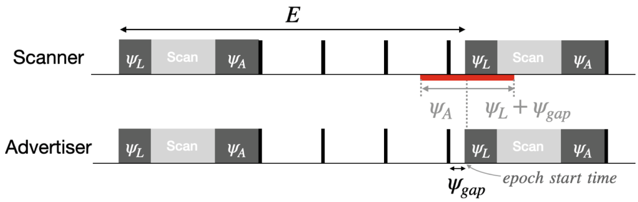


Fig. 8. Illustration of the non-functional area caused by the warmup interval and the removal of the last beacon right before next epoch.

the E_r range. The random epoch extension greatly increases discovery rates, key to **Nod**'s success in high discovery probabilities and handling warm-up intervals and packet loss. Table 2 details these parameters.

Table 2. Parameters used to model **Nod**.

Foundational Schedule Parameters	
N	number of nodes in the collision domain
Λ	target discovery latency (ms)
b	length (in time) of a beacon (ms)
E	epoch length (ms)
A	advertising interval (ms)
L	listening interval length (ms) ($L = A + s + b$)
γ	number of potentially colliding beacons
Parameters to Capture Real World Practicalities	
ℓ	packet loss rate
ψ_L	scanning warmup interval length (ms)
ψ_A	advertising warmup interval length (ms)
ψ_{gap}	expected value of gap between the last beacon and the next epoch (ms)
ψ	non-functional time in an epoch (ms), $\psi = \psi_L + \psi_A + \psi_{gap}$
E_{avg}	expected value of the length of a **Nod** epoch (ms)

4.2 Analytical Model for Nod

In this section, we propose a model to analyze the expected discovery probability using the **Nod** schedule described in the previous section, given the presence of packet loss and warm-up intervals. With the model, we can identify the optimal configuration, i.e., the schedule parameter E_r, that achieves the desired discovery probability while minimizing energy consumption. We base our model on the first few steps of the BLEnd model [8] which captures the impact of beacon collisions on discovery probability. From this basic model, we continue to consider the impacts of warmup intervals and random packet losses.

The Basic Model of Probabilistic CND. The model's goal is to define P_d (a "probability of discovery"), expressed in terms of N, the number of expected neighbors, Λ, the target discovery latency, and b, the length of a beacon (1.1 ms for a 31 byte BLE beacon [18]). The optimal scheduling, aimed at achieving Λ with minimal energy, is determined by two parameters: E_r, the root epoch length, and A, the advertising interval. A is set to 250 ms due to constraints from the Android operating system; the main task is to find the optimal E_r.

The probability of discovery, P_d, is also dependent on a value γ. γ represents the number of potential colliding beacons from neighboring nodes. While a simple assumption might suggest $\gamma = N - 2$ (excluding the sender and receiver), in reality, γ tends to be slightly higher. This is due to the positioning of beacons within nodes' schedules, especially when a target beacon occurs at the start or end of a listening interval, potentially overlapping with other beacons.

Given the above, we can use a relatively straightforward application of the birthday paradox to capture collisions and express the probability that a given node discovers a neighboring node as [8]:

$$P_d = \left(1 - \frac{2b}{A}\right)^{\gamma} \tag{1}$$

We next provide some brief intuition for Eq. 1. The design of $L = A + b$ ensures that when a neighboring node advertises during an epoch, at least one of its beacons will fall within the listener's functional interval. This basic model assumes that this beacon is guaranteed to be received by the listener unless it collides with another node's beacon. For modeling purposes, we treat these neighbor discovery collisions separately from more random packet losses, dealt with below. A collision occurs when beacons overlap. As nodes independently choose their schedule start times, there are $2b$ potential times leading to a collision with a neighbor's beacon, as illustrated in Fig. 9. Considering γ possible colliding beacons with independent start times, Eq. 1 gives the likelihood that a beacon is received without collision.

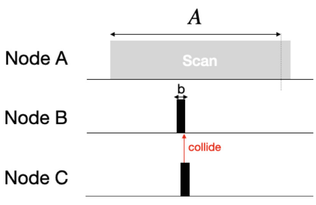


Fig. 9. The effective starting range of a beacon.

To ensure the possibility of discovery from schedule overlap, we must ensure that $E \leq \Lambda$. Otherwise it would be possible that a receiver's listen interval simply does not occur with Λ. Therefore, to express the probability that a node is discovered, not within a single epoch, but within the target discovery latency,

Λ, we need to consider the probability that the discovery event occurs in one of a set of contiguous epochs, plus a fraction of an epoch. Given that $\Lambda = kE + f$, where $f < E$, the probability of discovery within Λ, i.e., $P_{d,\Lambda}$ is [8]:

$$P_{d,\Lambda} = P_{d,kE} + P_{d,f} \tag{2}$$

where

$$P_{d,kE} = 1 - (1 - P_d)^k \tag{3}$$

and

$$P_{d,f} = (1 - P_d)^k \left(\frac{\Lambda - kE}{E} \right) P_d \tag{4}$$

Ultimately, the goal is to find a solution to Eq. 2 (i.e., values of E and A) that achieves the target discovery probability Λ with minimal energy consumption [8]. We use the basic model in Eq. 2 as a jumping off point for constructing schedules that account for both random packet losses and warmup intervals and modeling the probability of discovery of these new schedules.

Discovery Probability with Warmup Intervals. The model above suffices for schedules like those in Fig. 6(a). We next extend this model to consider the impacts of the schedule designs that include warmup intervals (ψ_L and ψ_A), the incomplete advertising interval at the end of an epoch (ψ_{gap}), and the random extension to the length of an epoch (E_e). These changes have two significant impacts. Both rely on the observation that ψ_L, ψ_A, and ψ_{gap} compound to create the non-functional block of time depicted in red in Fig. 8. Any neighbor node that chooses a schedule start time within this block of time will not have a beacon scheduled to overlap a receiver's functional listening interval. As a result, the number of potentially colliding beacons is reduced by a factor of $\frac{\psi}{E_{avg}}$ relative to the basic model, where E_{avg} refers to the expected value of the length of an epoch in **Nod**. Given our choice of E_e's range to be $[E_r, 2E_r]$, $E_{avg} = 1.5E_r$. Reflecting this in our model of discovery, we define γ_w, the number of potentially colliding beacons when one considers warmup interval impacts as:

$$\gamma_\psi = \left(1 - \frac{\psi}{E_{avg}}\right) \gamma \tag{5}$$

where γ is as defined above and in [8] and $\psi = \psi_L + \psi_A + \psi_{gap}$.

Second, and conversely, the probability of a node being discovered by a listener is *reduced* because beacons are not sent in the non-functional window represented by ψ. To account for these lost discoveries, we decrease the probability of discovering a node in a listening interval by the same factor, $\frac{\psi}{E_{avg}}$, effectively replacing Eq. 1 with:

$$P_{d,\psi} = \left(1 - \frac{\psi}{E_{avg}}\right) \left(1 - \frac{2b}{A}\right)^{\gamma_\psi} \tag{6}$$

We can use Eq. 6 to compute $P_{d,kE,\psi}$ and $P_{d,f,\psi}$ in a manner similar to Eqs. 3 and 4, which we combine as in Eq. 2 to express $P_{d,\Lambda,\psi}$, the probability any given node discovers any other given node within the window Λ, when one assumes warmup period impacts whose non-functional periods sum to ψ. Because in **Nod** E varies between E_r and $2E_r$, we must refine the constraint on the relationship between E and Λ and require that $2E_r \leq \Lambda$ for the same reasons as argued above (i.e., we must ensure that at least one listen interval falls within Λ.

Discovery Probability with Packet Loss. With the model of warmup intervals demonstrated, we next turn our attention to modeling the impact of random packet losses on the effectiveness of neighbor discovery. We model random packet losses uniformly: any beacon may experience a loss with a rate denoted as $0 \leq \ell \leq 1$. These losses have the same two notable impacts as above: the number of potentially colliding beacons diminishes by a factor of ℓ and the probability of discovering a node is reduced by a factor of ℓ due to the possibility of the target beacon itself being lost. Building on our work in the previous section, we account for the former by defining $\gamma_{\psi,\ell}$:

$$\gamma_{\psi,\ell} = (1-\ell)\left(1 - \frac{\psi}{E_{avg}}\right)\gamma \tag{7}$$

Since the impact of packet loss and warmup interval are independent of each other, we can aggregate their effects to derive the combined model. As a result, our final model expressing $P_{d,\Lambda,\psi,\ell}$, the probability of discovery in the presence of warmup intervals that aggregate to ψ and a packet loss rate of ℓ, is:

$$P_{d,\Lambda,\psi,\ell} = 1 - (1 - P_{d,\psi,\ell})^k + P_{d,f,\psi,\ell} \tag{8}$$

where

$$P_{d,f,\psi,\ell} = (1 - P_{d,\psi,\ell})^k \left(\frac{\Lambda - kE_{avg}}{E_{avg}}\right) P_{d,\psi,\ell} \tag{9}$$

and for the latter by defining $P_{d,\psi,\ell}$:

$$P_{d,\psi,\ell} = (1-\ell)\left(1 - \frac{\psi}{E_{avg}}\right)\left(1 - \frac{2b}{A}\right)^{\gamma_{\psi,\ell}} \tag{10}$$

5 Evaluation

To validate our approach, we used both simulations and a small scale real-world deployment as a proof-of-concept.

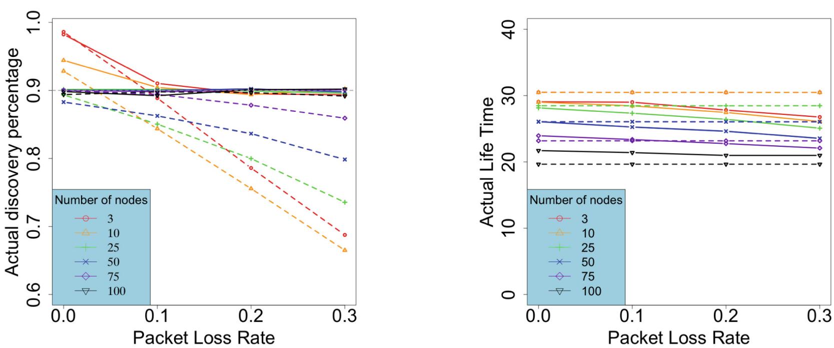
5.1 Comparative Simulations

In our simulations, we compare the model of **Nod** with the BLEnd model. We compare against BLEnd for two reasons: (1) it was the best performing of the existing models out of the box (Sect. 3) and (2) our resulting schedule is also slotless and similar in structure to the BLEnd schedule. We used the same discrete event simulator here as we used in Sect. 3. We also use the same settings for packet loss and warmup intervals, with the same application-motivated justifications as presented in Sect. 3.

For BLEnd, the published model [8] determined optimal parameters (E and A) for each setup. For **Nod**, we utilized the model from Sect. 4, running each configuration 500 times. Consistent with Sect. 3, the target discovery latency (Λ) was set to 10 s and the target discovery probability (ρ_Λ) to 0.9. Lifetime was computed using power profiles from [3]. Models were evaluated based on average discovery probability and lifetime (inverse of energy consumption).

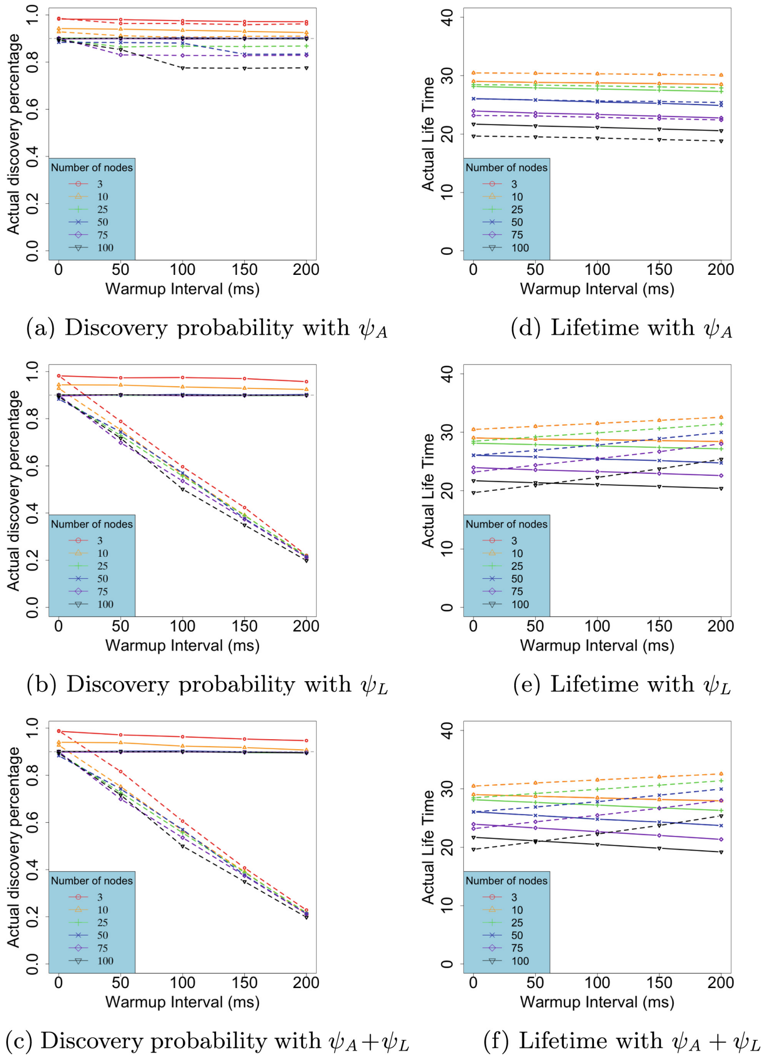
Random Packet Loss. Figure 10(a) shows how **Nod** performs in the presence of increasing rates of random packet loss. The figure compares **Nod** (solid lines) to BLEnd (dashed lines, the same as depicted in Fig. 2(c)). Notably, **Nod** shows a significant improvement in the discovery probability, meeting the target (0.9) discovery rate almost uniformly. In particular, the discovery rate is extremely stable across increasing numbers of nodes and increasing packet loss rate.

Of course, this resilience to packet loss comes at a cost, particularly in terms of energy consumption. As depicted in Figs. 10(b), it is evident that the system's lifetime decreases as the packet loss rate increases. This correlation arises due to the fact that higher packet loss rates result in lower discovery probabilities within a single listening interval. Consequently, to achieve the desired 0.9 discovery probability within the specified discovery latency, it becomes necessary to increase the total number of listening intervals (i.e., reduce epoch size and increase the number of epochs). Listening intervals generally consume more energy. As a result, the extended listening intervals lead to increased overall energy consumption of the system, resulting in a reduction in the system's lifetime.



(a) Discovery Probability (b) Lifetime

Fig. 10. Results of discovery probability and lifetime for the new model with packet loss rate. All protocols are configured to attempt to achieve 90% probability of discovery. Random packet losses range from 0% of packets to 30% of packets (x-axis).



(a) Discovery probability with ψ_A

(d) Lifetime with ψ_A

(b) Discovery probability with ψ_L

(e) Lifetime with ψ_L

(c) Discovery probability with $\psi_A + \psi_L$

(f) Lifetime with $\psi_A + \psi_L$

Fig. 11. Results of discovery probability and lifetime for **Nod** in the presence of warmup interval. All protocols are configured to attempt to achieve 90% probability of discovery. Warm-up intervals range from 0 ms to 200 ms (x-axis).

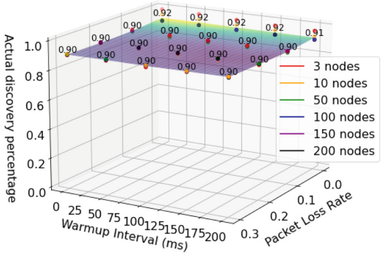
Warm-Up Interval. Figures 11(a)–(c) show how the performance of **NOD** (solid lines) compares with the performance of BLEnd (dashed lines; replicated from Figs. 3(g)–(i)). Notably, even with a large warm-up interval, **NOD** consistently achieves the target discovery probability, in contrast to BLEnd, which is dramatically impacted, especially by ψ_L.

In Figs. 11(d)–(f), we observe a slight decrease in the lifetime of nodes executing **NOD** schedules as the warm-up interval increases. This outcome can be attributed to a similar reason as discussed in Sect. 4.2. As the warm-up interval increases, the discovery probability within each epoch decreases, prompting the model to opt for a smaller E_r. Consequently, a larger portion of the node's time is spent in listening intervals, resulting in increased overall energy consumption and reducing the system's lifetime.

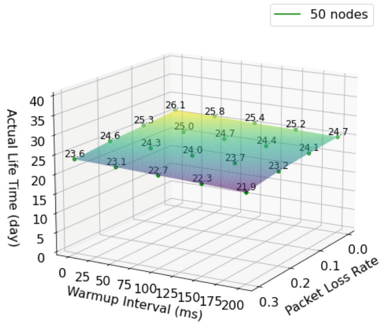
Combined Packet Loss and Warm-Up Interval. Figure 12(a) presents the results of **NOD** for different packet loss rates and scanning warmup intervals. The surface plot illustrates the discovery probability, with each data point corresponding to the average discovery for different numbers of nodes. Interestingly, we observe that the discovery probability is higher when the packet loss rate is zero. This outcome can be attributed to the configurations selected by 3 nodes under those conditions, which are inherently expected to yield higher discovery probabilities. These results highlight the effective performance of **NOD**, even in scenarios involving both scanning and warm-up intervals – in all cases, the discovery rates remain consistently above the target discovery probability of 90%.

As above, the improvement in discovery probability comes at a cost. In Fig. 12(b), we present the lifetime results for 50 nodes. The impact of packet loss and warmup on the lifetime is cumulative. If either of these issues becomes severe, the lifetime further decreases, necessitating a more aggressive scheduling approach to compensate for the lower discovery probability in each epoch.

Changing Application Constraints. Different applications have unique ρ_Λ and Λ requirements. Contact tracing often prefers a 30-second discovery latency with a high 0.9 discovery probability. In contrast, opportunistic data sharing apps vary these parameters based on data urgency. Critical real-time information demands low latency and high probability, while casual sharing, like "song of the day", can accept lower probability and longer latency. To showcase our models' adaptability, we simulated various scenarios with ρ_Λ values (0.7, 0.8, 0.9) and Λ durations (10 s, 30 s) for 50 nodes. Results are in Fig. 13. As the results show, **NOD** is resilient to diverse application conditions – in all cases the protocol achieves the target discovery latency. The tradeoff is, as always, energy consumption – as examples, the expected lifetime for a node with the worst case packet loss and warmup conditions when $\rho_\Lambda = 0.7$ and $\Lambda = 30$ (Fig. 13(d)) is 31.6, while the expected lifetime for a node in the same conditions with $\rho_\Lambda = 0.9$ and $\Lambda = 10$ is 21.9.



(a) Discovery Probability



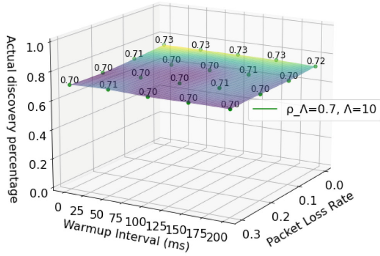
(b) Lifetime

Fig. 12. Discovery probability and lifetime results of the new model in the presence of packet loss rate and scanning warmup interval. All protocols are configured to attempt to achieve 90% probability of discovery. Random packet losses range from 0% of packets to 30% of packets. Scanning warm-up intervals range from 0 ms to 200 ms.

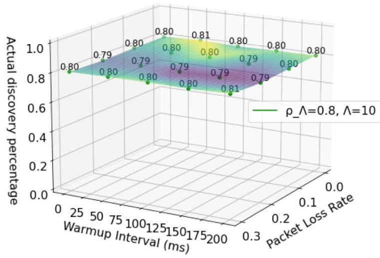
5.2 Nod in the Real World

Primarily as a proof-of-concept, we implemented **Nod** on three Android devices (also available in our repository): Google Pixel 3, Samsung S7, and Huawei N29. In our experiments, we designated the Samsung S7 device as the listener and observed its probability of detecting the other two devices. All three devices were randomly started asynchronously. We set the desired discovery probability to 0.9 and the discovery latency to 10 s, consistent with the settings used in our simulations. Based on empirical measurements with these devices (Sect. 2), we used an expected packet loss of 8%, advertising warmup as $\psi_A = 122$ and scanning warmup as $\psi_L = 147$.

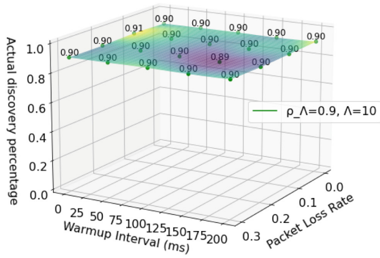
Table 1 shows varied scanning warmup interval across devices. For our tests, we used uniform parameters for three devices, considering the worst-case scenario. Alternatively, in real world deployments, **Nod** schedules can be customized to each device's scanning warmup. With inputs ($N = 3$, $A = 250$ ms, $\psi_L = 149$, $\psi_A = 135$, $\ell = 0.08$, $\rho_\Lambda = 0.9$, $\Lambda = 10000$ ms), the **Nod** model from Eq. 8 sets root epoch size E_r at 4951. The expected discovery probability based on the model is 0.903. Figure 14 shows the cumulative distribution of discovery latency. Points on the graph indicate the proportion of discoveries (y-axis) by a given time (x-axis). The pair of dashed lines indicate the **Nod** contract – the schedules are created to ensure that 90% of discoveries happen at or before 10,000 ms.



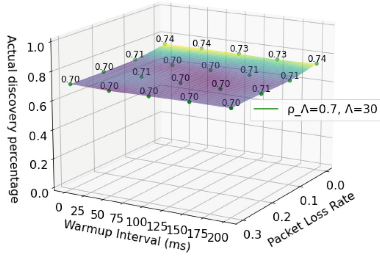
(a) $\rho_\Lambda = 0.7$ and $\Lambda = 10$



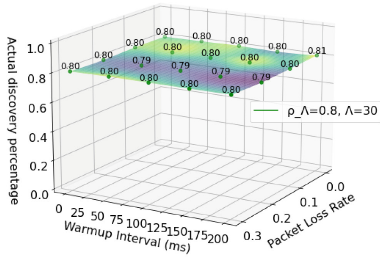
(b) $\rho_\Lambda = 0.8$ and $\Lambda = 10$



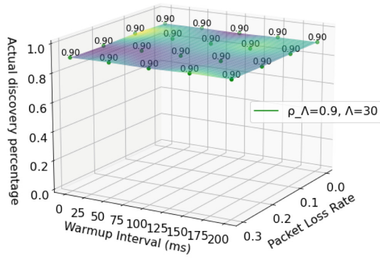
(c) $\rho_\Lambda = 0.9$ and $\Lambda = 10$



(d) $\rho_\Lambda = 0.7$ and $\Lambda = 30$



(e) $\rho_\Lambda = 0.8$ and $\Lambda = 30$



(f) $\rho_\Lambda = 0.9$ and $\Lambda = 30$

Fig. 13. Discovery probability for **NOD** in the presence of both packet loss and scanning warm-up intervals. **NOD** is configured to achieve target discovery probabilities of $\rho_\Lambda = 0.7$, 0.8, or 0.9, while Λ is set to either 10 s or 30 s. Random packet losses vary from 0% to 30%, and warm-up intervals from 0 ms to 200 ms.

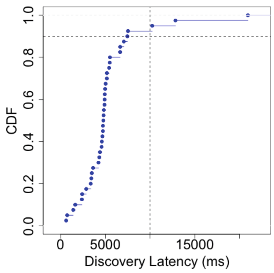


Fig. 14. Cumulative distribution of discovery times for real devices.

6 Conclusions

In this paper, we tackled two practical issues –packet loss and warmup intervals – in the context of continuous neighbor discovery protocols for real-world deployments. Our study quantified and revealed the significant impacts of these factors on the discovery performance, which are almost entirely overlooked by existing models. To overcome these challenges, we proposed a new protocol called **Nod**. This innovative protocol introduces the concept of a random extra epoch, enabling nodes initially trapped in non-functional areas to be discovered in subsequent epochs. As a result, **Nod** substantially improves the overall discovery probability. To determine optimal schedule parameters of **Nod** considering packet loss and warmup intervals, we developed analytical models that account for these practical limitations. In our evaluation, we assessed the discovery probability and energy cost associated with implementing **Nod**. While the protocol demonstrated improved resiliency, it does come at the cost of a slightly reduced node lifetime. Nonetheless, **Nod** presents a promising solution for continuous neighbor discovery, offering reliable performance on mobile devices and making it a feasible choice for real-world deployments.

References

1. Bakht, M., et al.: Searchlight: won't you be my neighbor? In: Proceedings of MobiCom, pp. 185–196 (2012)
2. Cohen, R., Kapchits, B.: Continuous neighbor discovery in asynchronous sensor networks. IEEE/ACM Trans. Netw. **19**(1), 69–79 (2010)
3. Dodangeh, M., Oude Alink, M.S., Nauta, B.: Quantifying the trade-off between latency and power consumption in bluetooth low energy and its mitigation by using a wake-up receiver. In: 2022 IEEE 8th World Forum on Internet of Things (WF-IoT), pp. 1–8 (2022). https://doi.org/10.1109/WF-IoT54382.2022.10152115
4. Dutta, P., Culler, D.: Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In: Proceedings of SenSys, pp. 71–84 (2008)
5. Girolami, M., et al.: Sensing social interactions through BLE beacons and commercial mobile devices. Pervasive Mob. Comput. **67** (2020)
6. Hatke, G.F., et al.: Using bluetooth low energy (BLE) signal strength estimation to facilitate contact tracing for COVID-19. arXiv preprint arXiv:2006.15711 (2020)

7. Herrera, J.L., Chen, H.Y., Berrocal, J., Murillo, J.M., Julien, C.: Context-aware privacy-preserving access control for mobile computing. Pervasive Mob. Comput. **87**, 101725 (2022). https://doi.org/10.1016/j.pmcj.2022.101725, https://www.sciencedirect.com/science/article/pii/S1574119222001389
8. Julien, C., Liu, C., Murphy, A.L., Picco, G.P.: Blend: practical continuous neighbor discovery for bluetooth low energy. In: Proceedings of IPSN, pp. 105–116 (2017)
9. Kindt, P.H., Chakraborty, S.: Performance limits of neighbor discovery in wireless networks. arXiv preprint arXiv:2103.05026 (2021)
10. Kindt, P.H., et al.: Griassdi: mutually assisted slotless neighbor discovery. In: Proceedings of IPSN, pp. 93–104 (2017)
11. Kindt, P.H., et al.: Optimizing BLE-like neighbor discovery. IEEE Trans. Mob. Comput. (2020)
12. Kohvakka, M., Suhonen, J., Kuorilehto, M., Kaseva, V., Hännikäinen, M., Hämäläinen, T.D.: Energy-efficient neighbor discovery protocol for mobile wireless sensor networks. Ad Hoc Netw. **7**(1), 24–41 (2009)
13. McGlynn, M.J., Borbash, S.A.: Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In: Proceedings of MobiCom, pp. 137–145 (2001)
14. Qiu, Y., Li, S., Xu, X., Li, Z.: Talk more listen less: energy-efficient neighbor discovery in wireless sensor networks. In: Proceedings of the 35th Annual IEEE International Conference on Computer Communications, pp. 1–9 (2016)
15. Radhakrishnan, M., et al.: Smartphones and BLE services: empirical insights. In: Proceedings of MASS, pp. 226–234 (2015)
16. Saloni, M., Julien, C., Murphy, A.L., Picco, G.P.: LASSO: a device-to-device group monitoring service for smart cities. In: Proceedings of the International Smart Cities Conference, pp. 1–6 (2017)
17. Tseng, Y.C., et al.: Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks. Comput. Netw. **43**(3), 317–337 (2003)
18. Woolley, M.: Bluetooth core specification version 5.2. Bluetooth SIG (2020)