Improving Neighbor Discovery by Operating at the Quantum Scale

Paper #52

ABSTRACT

Duty-cycling is generally adopted in existing sensor networks to reduce power consumption and these networks depend on neighbor discovery protocols to ensure that nodes will wake up and discover each other. The discovery latency is determined by two factors: the wake-sleep pattern and the slot size. To the best of our knowledge, previous works on neighbor discovery have thus far been focused on improving the wake-sleep pattern. In this paper, we investigate the extent to which we can improve discovery latency by reducing the slot size. We found that when listening times are reduced to a duration comparable to the beacon length with small time slots, the collisions between beacons and synchronization between nodes can lead to discovery failures that are not predicted by existing theoretical models. We show that we can mitigate these effects by reducing the number of beacons and by introducing randomization. We propose a new continuous-listening-based neighbor discovery algorithm called *Spotlight*. Our evaluations with both simulations and a practical sensor testbed suggest that Spotlight can achieve a 50% reduction in discovery latency over existing state-of-the-art neighbor discovery protocols without increasing power consumption in existing sensor networks.

1 INTRODUCTION

Power is a very important consideration for low-power mobile sensing devices, especially for sensors deployed in scenarios where it is difficult to replace batteries. Duty-cycling [2, 6] is commonly employed to save power, but duty-cycling then requires a neighbor discovery protocol to ensure nodes can actually detect other nodes. Existing neighbor discovery algorithms are based on wake-sleep patterns to ensure that nodes will wake up within the same time slot to discover each other [3, 4, 10, 12, 13, 19, 21, 22].

The key drawback of duty-cycling compared to "always on" operation is that the discovery latency is significantly higher, so the goal is to develop algorithms that have short discovery latencies and also guarantee discovery. Clearly, the discovery latency is determined by two factors: the wake-sleep pattern and the slot size. But interestingly, to the best of our knowledge, previous works on neighbor discovery have thus far been focused on the former, i.e., coming up with better wake-sleep patterns. In this paper, we focus on the latter and investigate the fundamental question: *exactly how far can we go in reducing discovery latency simply by reducing the slot size?*

Naively, we would expect the discovery latency to decrease proportionally to the decrease in slot size. However, our measurement study with a practical testbed showed that when the slot size was reduced to a size comparable to the beacon length, which we call the *quantum* regime, the neighbor discovery latency increased drastically. Existing worst-case latency bounds and measures such as the power-latency product were not sufficient to predict how algorithms, such as Searchlight or Nihao, would perform at small slot sizes [3, 18]. This demonstrated a gap in existing theoretical models and analysis when it comes to analyzing performance in this regime. In particular, we found in our measurement study that: (i) the collisions between beacons are relatively common, leading to discovery failures that are not predicted by existing theoretical models; (ii) the relative clock skew matters, and synchronization between nodes can lead to worst-case discovery latencies that are many times worse than those predicted by existing theoretical models.

We show that we can significantly reduce discovery failures from collisions and synchronization by (i) reducing the density of beacons and (ii) introducing some randomization in the wake-up time. But we can do even better. We discovered in our detailed analysis of discovery failures that Searchlight's approach of having probe slots [3] was harmful because it introduced new failure modes. We hence propose a new continuous listening and periodic beaconing pattern, called *Spotlight*. We describe a methodology for analyzing it and proving correctness called a *BL diagram*. We show that we can achieve optimal discovery latency for our new wake-sleep pattern with an $n \times 2n$ configuration.

To the best of our knowledge, our work is the first to systematically investigate how ultra-small slot sizes affect neighbor discovery latency for sensor networks. In addition, we make several contributions to the understanding of sensor performance in the *quantum* regime, detailed as follows:

- We completed a detailed measurement study of a practical duty-cycling sensor device and showed that existing models of sensor operation ignore practical radio phenomena, such as warm-up delays and radio state transitions. These have non-negligible effects in the *quantum* regime;
- We investigated the reasons for discovery failures in the *quantum* regime, and showed that (i) reducing the number of beacons and (ii) introducing randomization can effectively mitigate these failures, enabling us to operate at an effective slot size equal to one beacon length;
- We proposed a new continuous-listening-based neighbor discovery algorithm called *Spotlight* and showed with both simulations and experiments in a practical sensor testbed that Spotlight outperforms existing state-of-the-art neighbor discovery algorithms.

We show in our evaluations with both simulations and a practical sensor testbed that Nihao performed surprisingly well in the *quantum* regime, but even so, Spotlight can achieve a 50% reduction in discovery latency over Nihao given the same duty cycle setting. We have also shown in our testbed experiment that it is actually feasible to operate at the slot size of 1 ms, which is equal to the beacon length.

Traditional sensor applications typically adopt slot sizes between 10 ms [4, 18, 23] to 50 ms [19]. As sensor hardware continues to improve, we can expect to be able to operate at significantly smaller slot sizes. By adopting the techniques we have developed, we anticipate a reduction in neighbor discovery latencies by an order of magnitude with no increase in power consumption. Our simple randomization technique, which can prevent discovery failures arising from synchronization in the *quantum* regime, is generally applicable to existing neighbor discovery protocols, and is thus expected to be a general feature of such algorithms in the future.

The rest of the paper is organized as follows: in §2, we present an overview of existing neighbor discovery protocols. In §3, we describe the results of our measurement study, which reveal the challenges of working in the *quantum* regime, and two techniques to mitigate the observed problems. In §4, we describe Spotlight, our new continuous-listening-based neighbor discovery algorithm. Finally, we present our evaluation results obtained via simulation and a practical sensor testbed in §5, and conclude in §6.

2 RELATED WORK

Most neighbor discovery protocols are *slotted* protocols and divide time into identically-sized discrete slots [3, 4, 19, 22]. During *active* slots, the nodes wake up to transmit and listen for beacons from potential neighbors. The more recent proposals sought to minimize the theoretical *discovery latency*, i.e., the number of elapsed slots needed to guarantee that two nodes' active slots overlap [3, 18]. We have found that they typically ignored two important factors affecting the actual latency, *slot size* and *practical discovery failures* (failures that can occur when active slots overlap.)

Existing protocols can be classified by their adoption of either *probabilistic* or *deterministic* wake-sleep slot schedules to achieve an overlap [20]. Probabilistic protocols such as Birthday [13] can often achieve better average-case latencies, but are unable to provide worst-case bounds. Discovery guarantees are important for practical applications, so they have been the focus of recent research. The performance of deterministic protocols can be characterized by their *power-latency product*, $\Lambda = DC \cdot L$ [10], with a lower bound of $\Lambda \approx \sqrt{L}$ for an optimal mutual discovery protocol, where L denotes the worst-case latency in terms of slots [24] and DC denotes the duty cycle.

Disco [4] ($\Lambda = 2\sqrt{L}$) and U-Connect [10] ($\Lambda = 1.5\sqrt{L}$) employ prime-based active slot schedules that guaranteed discovery using the Chinese Remainder Theorem [15]. On the other hand, Quorum ($\Lambda = 2\sqrt{L}-1$) ensured overlaps by casting each period of n^2 slots as an $n \times n$ square, and designating an arbitrary row and column as active slots [21]. Subsequently, Searchlight [3] ($\Lambda = \sqrt{2L}$) introduced the practice of designating active slots as either stationary *anchor* or moving *probe* slots, with the probe slot iterating over possibilities until it achieved an overlap with another node's anchor slot; this concept was also adopted by BlindDate [22] ($\Lambda = \sqrt{\frac{9}{10}L}$). Sun et al. proposed a common unified framework, called *Hello*, that allowed these protocols (Quorum, Disco, U-Connect, Searchlight) to be compared.

To our knowledge, we are the first to comprehensively study the impact of small slot sizes. Most previous work either picked an arbitrary slot size that worked for their specific hardware platform [3, 4], or used the same slot size as previous work [18]. The slot sizes used include 10 ms [4], 25 ms [4], 50 ms [19] and 2 s [3]. Some reasons cited for not using smaller slot sizes included jitter [7], overflowing slots [19], and slow hardware state transitions [3]. We believe that some of these difficulties reflect the phenomena of slot misalignment exacerbated by clock drift and practical discovery failures, which we describe in §3 and §3.2. We have successfully achieved deterministic neighbor discovery at a slot size of 1 ms and 1% duty cycle using our custom sensor platform.

Motivated by the discovery failures observed when working with small time slots, we concluded that it was advantageous to adopt heterogeneous active slots (with different transmit-listen patterns) instead of the traditional homogeneous active slots (with the same transmit-listen pattern for all slots). Most of the previous protocols used the beacon-listen-beacon pattern (a listening period sandwiched by two beacons) proposed in Disco [4]. Our use of heterogeneous slots shares a similar motivation that was recently cited by U-Connect [10], WiFlock [17] and Nihao [18] ($\Lambda = \alpha\sqrt{L}, \alpha < 1$). U-Connect used small transmit-only slots and combined multiple slots into a single large slot for listening by relying on specialized hardware that supported Low Power Listening (leveraging periodic channel sampling) [16], while WiFlock used short listening slots and had long beacons spanning multiple contiguous slots. Similarly, Nihao asserted that slots should be used for transmitting or listening exclusively, and proposed that transmission slots consist of a single beacon followed by sleeping for the rest of the slot. Another strategy that we adopted, like other recent protocols, is to focus on achieving one-sided discovery, since this can be easily extended to mutual discovery via using the information received to predict and send a beacon during the other node's next listening period [9, 18]. We found that at some level one-sided discovery was relatively common once the slot sizes were comparable to the beacon length.

Collaborative neighbor discovery schemes have been proposed to reduce neighbor discovery latency by deconflicting nodes' transmission schedules to mitigate the increased risk of beacon collision with a larger number of nodes [17, 23]. These schemes are largely orthogonal and will benefit from the faster one-sided discovery latency that our proposed method offers.

Driven by the popularity of low-power interval-based standards such as Bluetooth Low Energy (BLE) and ANT/ANT+, new *'slotless'* protocols have been proposed, such as PI-kM [11] and BLEnd [9]. These protocols ensure continuous mutual discovery and are constructed on top of the one-sided non-continuous discovery schemes provided by BLE and ANT/ANT+. PI-kM decoupled transmission from listening with different periodicities, and derived formulations for the transmission and listening interval parameters that represent local optima in the complex parameter landscape.

BLEnd extends BLE to construct a one-sided discovery protocol. Each BLEnd period begins with a single listening interval, followed by beacons repeated at regular intervals for the rest of the period's first half such that the last beacon falls just within the second half. Covering at least half the period is sufficient to ensure discovery within one period. Decoupled interval-based protocols, such as PI-kM, make latency analysis significantly more complex. On the other hand, the periodicity of protocols like BLEnd allows them to be approximated and analyzed as a slotted protocol. Given

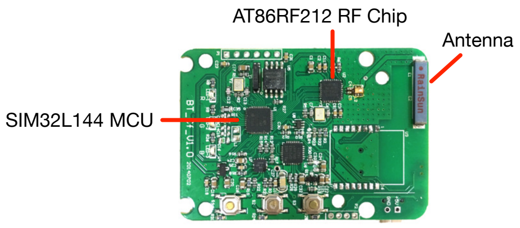


Figure 1: Custom hardware platform.

that Spotlight works with beacon-length-sized slots, it is somewhat equivalent in practice to a slotless protocol. We propose a methodology for proving worst-case discovery guarantees, called a *BL diagram.*

3 WHAT HAPPENS WITH SMALL SLOTS

The neighbor discovery latency L between two sensor nodes with duty cycling is expected to be $L = N \cdot S$, where N is the number of slots needed for neighbor discovery, and S is the slot size in terms of time. Most existing neighbor discovery protocols are focused on reducing N. In contrast, to the best of our knowledge, we are the first to investigate the extent to which we are able to improve latency by reducing the slot size S.

To this end, we use a custom hardware sensor platform (shown in Fig. 1) in our preliminary measurement study. The platform is equipped with a low-power transceiver AT86RF212 and a 32-bit ARM Cortex-M3 CPU. For our investigation, we used Searchlight [3], which is a state-of-the-art neighbor discovery protocol, at a duty cycle of 1%. While Nihao is a more recently-proposed protocol, its theoretical worst-case latency and parametrization suggested that Nihao would perform worse than Searchlight for small slot sizes [18].

We plot the neighbor discovery latency L against different slot sizes S in Fig. 2. While L was expected to decrease proportionally to the reduction in S, our testbed experiments showed that the neighbor discovery latency L increases drastically when the slot size is reduced beyond a certain point. Essentially, when operating at the *quantum scale*, where the slot size is comparable to the beacon length, the performance of Searchlight changed drastically.

In Fig. 2, we plot the results for two different pairs of sensor nodes with relative clock skews 0.6 ppm (parts per million) and 2.0 ppm, and compare them to Searchlight's theoretical worst-case discovery latency under 1% duty cycle [8]. We see that the average discovery latency is dependent on the relative clock skew, which was somewhat surprising.

In Fig. 3, we plot the probability of successful discovery within the first period. Our results suggest that the poor performance at small slot sizes was primarily caused by the dramatic increase in *discovery failures* in this regime. In the following sections, we describe what we have learned about these discovery failures and techniques for mitigating them.

3.1 Modeling a Real Active Slot

Existing deterministic neighbor discovery protocols [3, 4, 19, 22] generally adopt a beacon-listen-beacon configuration for the active slot shown in Fig. 4(a). In other words, during each active slot,

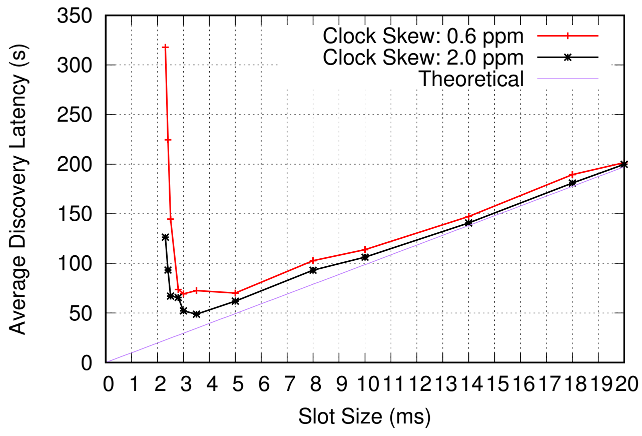


Figure 2: Average discovery latency for a pair of nodes.

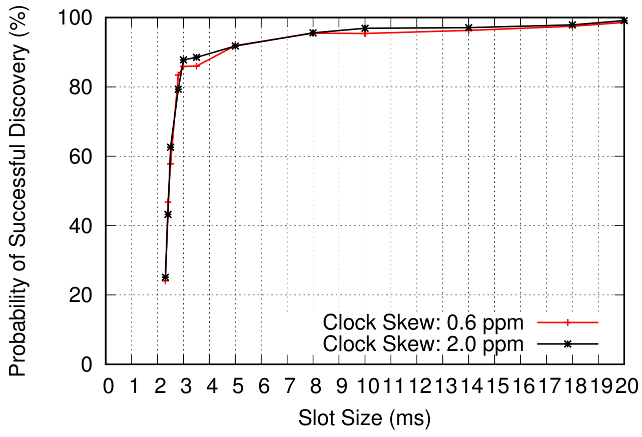
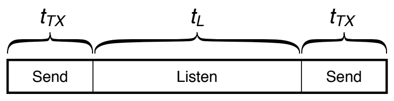
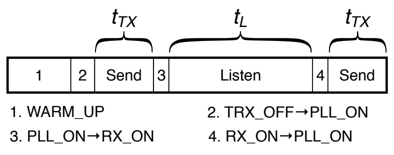


Figure 3: Probability of successful discovery within the first period.



(a) Model.



(b) Actual time components in practice.

Figure 4: Comparison of an ideal and a practical active slot.

a beacon is transmitted at the beginning and end, and listening is done between. This arrangement seems logical since it enables a pair of nodes to hear each other's beacon and achieve mutual discovery when their active slots overlap. In practice, radio devices experience non-negligible delays for warm-up and state transitions [5, 20]. Based on analysis of our own hardware platform [1], an active slot generally has four delays: a warm-up delay (Component 1 in Fig. 4(b)) and three state transition delays (Components 2, 3 and 4 in Fig. 4(b)). While these delays are negligible at large slot sizes, they become significant in the quantum regime.

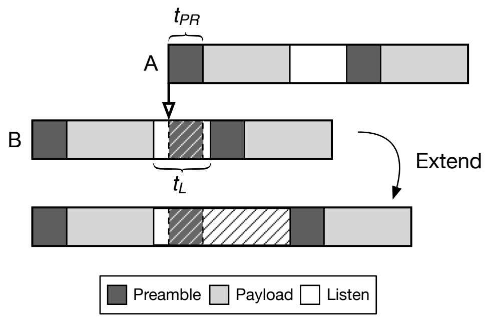


Figure 5: If B has received A's preamble, B extends its listening time if necessary in order to receive A's payload.

Warm-up Delay. In the sleep state, the radio transceiver is disabled. The radio transceiver cannot start transmitting until the internal oscillator and voltage regulator reach a stable state, leading to a *warm-up delay* [1]. The actual duration depends on the crystal oscillator and external capacitor setup, and is typically on the order of milliseconds [5]. For our hardware platform, we measured an average warm-up delay of 1.3 ms. To mitigate the impact of the warm-up delay, we can preemptively start the power-up process in the preceding sleep slots [3, 20].

State Transition Delays. There are three state transitions in each active slot, illustrated as components 2, 3 and 4 in Fig. 4(b). The manufacturer's theoretical state transition delays for components 2, 3 and 4 are 200 μs, 1 μs and 1 μs respectively [1]. However, the actual state transition delay should include the time consumed for the radio to switch states, e.g., writing commands to the register. For our hardware platform, the average state transition delays measured were 207 μs, 25 μs and 25 μs, respectively.

Sending Time. Sending time is a function of the beacon length and the physical data rate. The beacon length consists of a user payload in addition to headers of the MAC layer and physical (PHY) layer. For example, the widely-used 802.15.4-2006 protocol specifies a preamble length of 17 bytes (11 bytes in MAC layer and 6 bytes in PHY layer). The physical data rate is dependent on the hardware. If we denote l, l_{MAC}, and l_{PHY} as the length in bits of the user payload, MAC and PHY header respectively, the time required to send a beacon can be expressed as:

$$t_{\text{TX}}(l) = \frac{l + l_{\text{MAC}} + l_{\text{PHY}}}{\text{PhyRate}} \tag{1}$$

where t_{TX} shall be referred to as the *beacon length.*

Listening Time. Unlike the sending time, which is relatively fixed during operation, there are variations in the listening time. In practice, after detecting a valid synchronization header (SHR), the transceiver automatically enters the BUSY_RX state, during which we cannot change its state. Thus, it is possible to set a smaller listening time and to automatically extend the active slots once a valid preamble is received, as illustrated in Fig. 5. If we denote the time for receiving the payload as t_{RX} (inclusive of the possible idle period before receiving the preamble) and the time to receive a preamble as t_{PR}, then the total listening time $t_{\text{L}} = t_{\text{RX}} + t_{\text{PR}}$.

Model. Based on the above analysis, given a beacon length l, the slot size S is given by:

$$S = 2t_{\text{TX}}(l) + t_{\text{D}} + \underbrace{t_{\text{RX}} + t_{\text{PR}}}_{t_{\text{L}}} \tag{2}$$

where t_{D} is the sum of warm-up and state transition delays. It turns out that we can reduce the slot size by shifting the warm-up process to the preceding slot(s) [3] along with the first state transition delay (Component 2 in Fig. 4(b)). In other words, we compensate for this delay by making a node wake up slightly earlier. Since the last two state transition delays are negligible, we can simplify the slot size model to:

$$S = 2t_{\text{TX}}(l) + t_{\text{L}} \tag{3}$$

3.2 Practical Discovery Failures

While existing neighbor discovery protocols assume that discovery will always be successful whenever the active slots of two sensor nodes overlap [3, 4, 10], as illustrated in Fig. 6, there are actually three possible cases when two active slots overlap: *Discovery Failure*, *One-sided Discovery*, and *Mutual Discovery*. One-sided discovery, where only one node hears the beacon for the other node, is uncommon for large slot sizes and is thus typically ignored. In the quantum regime, one-sided discovery is relatively common because the listening time is comparable to the beacon length. In practice, one-sided discovery alone is not sufficient for practical applications. However, like others, we argue that achieving mutual discovery after one-sided discovery is primarily an engineering issue, and thus left it for future work [18].

Discovery failures occur when the beacons overlap and the two sensor nodes have to wait for the next overlapping active slots to try to discover each other again. This suggests that the actual neighbor discovery latency L_{actual} is as follows:

$$L_{\text{actual}} = N \cdot S + \Delta T \tag{4}$$

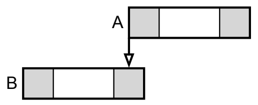
where ΔT is the additional latency caused by discovery failures. $\Delta T = 0$ when there is no discovery failure. The expected neighbor discovery latency $\bar{L}_{\text{actual}}$ can be expressed as:

$$\bar{L}_{\text{actual}} = N \cdot S + P_{\text{fail}} \Delta T \tag{5}$$

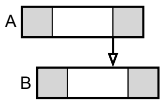
Quantifying P_{fail}. By analyzing the discovery failure modes, we found that P_{fail} was dependent on the relative slot offset of two active slots, the relative sizes of the beacon length (t_{TX}), and the listening time (t_{L}). Fig. 7 shows the detailed analysis of the relative slot offset with the corresponding discovery outcome. The results for the two possible cases: (i) $t_{\text{L}} \leq t_{\text{TX}}$ and (ii) $t_{\text{L}} > t_{\text{TX}}$, are summarized in Table 1.

Table 1: Duration of different discovery cases.

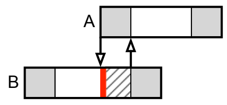
Condition	t_{fail}	$t_{\text{one-sided}}$	t_{mutual}
$t_{\text{L}} \leq t_{\text{TX}}$	$2t_{\text{TX}} - t_{\text{L}} + 2t_{\text{PR}}$	$t_{\text{L}} - t_{\text{PR}}$	$t_{\text{L}} - t_{\text{PR}}$
$t_{\text{L}} > t_{\text{TX}}$	$t_{\text{TX}} + 2t_{\text{PR}}$	$t_{\text{TX}} - t_{\text{PR}}$	$t_{\text{L}} - t_{\text{PR}}$



(a) Discovery Failure.

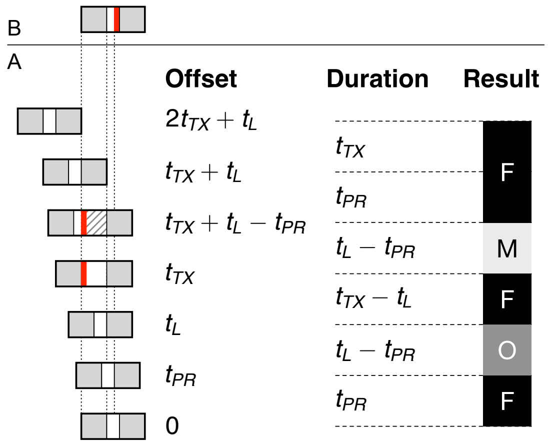


(b) One-sided Discovery.

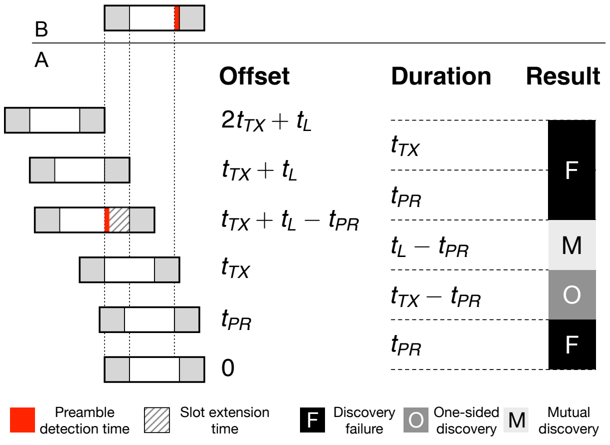


(c) Mutual Discovery.

Figure 6: Three possible discovery cases when two nodes' active slots overlap.



(a) Discovery cases with $t_L \le t_{TX}$



(b) Discovery cases with $t_L > t_{TX}$

Figure 7: Discovery results with different slot size and relative slot offset.

If we assume that the relative slot offset of two nodes is uniformly distributed in $[0, S)$, we can derive the following probabilities for the discovery failure:

$$P_{\text{fail}} = \begin{cases} \dfrac{2t_{\text{TX}} - t_{\text{L}} + 2t_{\text{PR}}}{2t_{\text{TX}} + t_{\text{L}}}, & t_{\text{L}} \le t_{\text{TX}} \\[2ex] \dfrac{t_{\text{TX}} + 2t_{\text{PR}}}{2t_{\text{TX}} + t_{\text{L}}}, & t_{\text{L}} > t_{\text{TX}} \end{cases} \tag{6}$$

Eq. (6) suggests that the probability of discovery failure increases when slot size decreases, which is consistent with the observations made in our measurement study.

Quantifying ΔT. Fig. 7 suggests that ΔT is related to both the slot size and the slot offset. To better understand their impact on the discovery latency, we conducted an experiment using a pair of sensor nodes with a relative clock skew of 1.7 ppm. The corresponding offset-ΔT graphs for different slot sizes are shown in Fig. 8.

The first observation from Fig. 8 is that ΔT is very sensitive to slot offset. ΔT can vary from near zero to more than 5 minutes (for the 2.3 ms case), which is much higher than the theoretical worst-case latency (~46 seconds). Such high variation is also present in larger slot sizes (see the graphs for 2.5 ms, 2.8 ms and 3.0 ms). Second, the offset-ΔT curve resembles a series of triangular patterns. For each triangle, the vertical ΔT surge is caused by the slot offset moving from the discoverable range to the failure range, and it is the local maximum as it requires the longest time for the slot offset to move to the next discoverable region.

The locations and the lengths of the bottom sides of the triangles are determined by slot size. As slot size decreases (in the order of 3.0 ms, 2.8 ms, 2.5 ms, 2.3 ms), two triangles start to appear in the middle. The bottom sides of these two triangles increase by the same amount as the decrease in the slot size, resulting in higher worst-case latency. This phenomenon is better illustrated in Fig. 9, which corresponds to the 2.3 ms case in Fig. 8. Note that for non-aligned slots, there are two overlaps in each period (Fig. 9(a)). The first overlap is as expected, and the discovery outcome follows the same pattern as in Fig. 7(a). The discovery outcome for the second overlap is a horizontal flip of the first overlap due to its complementary slot offset. The final discovery outcome within one period is a combination of the results of these two overlaps (Fig. 9(b)), which corresponds to the bottom sides of the triangles in the 2.3 ms case in Fig. 8. Note that the two failure regions in the middle have duration of $t_{\text{TX}} - t_{\text{L}}$. This explains why the bottom sides of these two triangles increase by an amount equal to the decrease in slot size (i.e., t_{L}).

The slope of the right sides of the triangles is determined by the clock skew. Smaller clock skew slows down the change of slot offset, resulting in higher worst-case latency. This is true even for large slot sizes. Let T be the the clock skew between the two nodes, and Δd be the distance between the current offset and the next discoverable offset. ΔT can be approximated by $\Delta d/T$ (this assumes that clock skew is small enough so that it does not skip the regions, which is true in the most common cases). In other words, the slope of the triangle is inversely proportional to the clock skew. This can be verified visually by the arrow in Fig. 8 which has a slope of $1/T$ ($T = 1.7\ ppm, i.e., -1.7 \times 10^{-6}$). In other words, as a rule of thumb, if clock skew is reduced by half, ΔT doubles. Also, contrary to our intuition, relative clock skew is not actually harmful because it is possible for ΔT to be extremely large for certain offsets between pairs of nodes, if the relative clock skew is small.

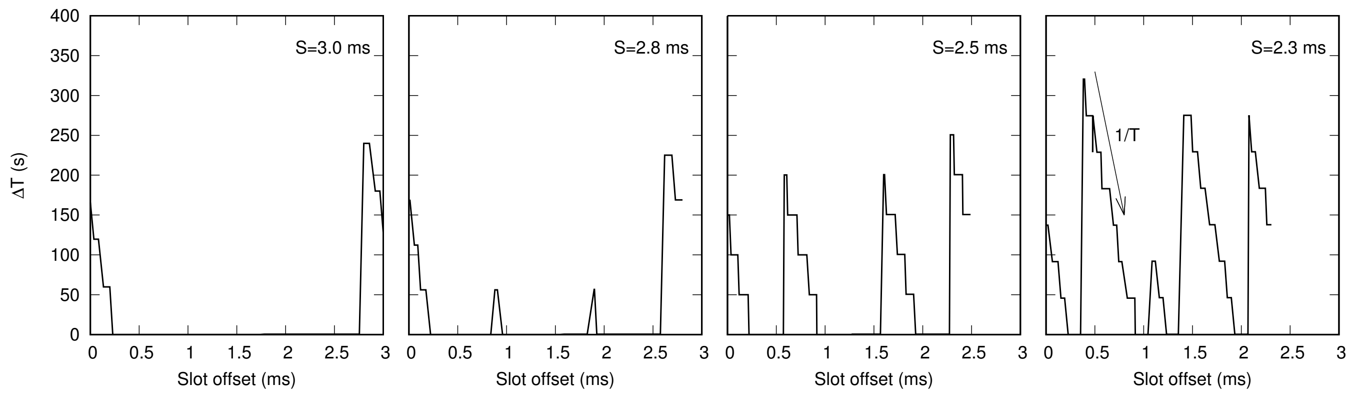
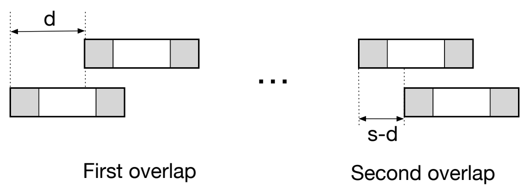
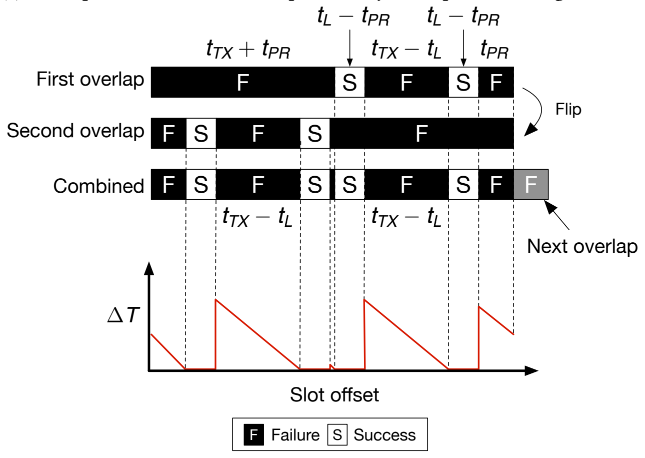


Figure 8: Plot of ΔT against slot offset for different slot sizes.



(a) In one period there are two complementary overlaps for non-aligned slots.



(b) ΔT is the result of the combined outcome of the two overlaps.

Figure 9: Discovery outcome within one period.

3.3 Discovering Multiple Neighbors

Most previous work [3, 4, 10, 12, 13, 19, 21, 22] on neighbor discovery considered neighbor discovery between just two nodes. However, for many applications, it is common for a node to be surrounded by multiple neighbors [17]. For instance, in the Tourist Tracking application at the Mogao Grottoes [14], the number of tourists in one cave often exceeds 20 at a time. For large slot sizes, the beacons are small relative to the listening time and thus collisions do not pose a problem. However, when the slot size is small, the issue of beacon collisions quickly becomes significant. This trend is alluded to by Fig. 10, which plots the probability that at least three nodes will be active (and likely causing collisions) given the presence of overlapping active slots.

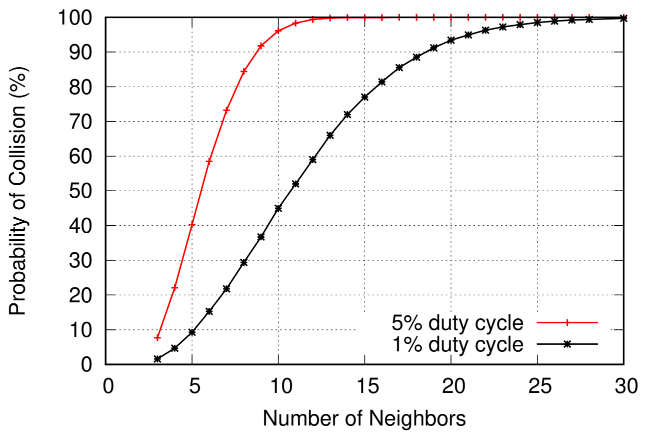


Figure 10: Probability of at least three nodes being active in an active slot overlap using Searchlight against number of nodes.

Impact on discovery latency. In Fig. 11, we plot the average pairwise neighbor discovery latency given different numbers of nodes for Searchlight at 1% duty cycle (with a clock skew of 0.6 ppm). Interestingly, there does not seem to be much impact. However, the CDFs as shown in Fig. 12 clearly illustrate that collisions caused by having multiple neighbors can cause latency to balloon to 2,000+ s in the worst case.

3.4 Mitigating Collisions & Synchronization

Our key findings in the previous sections show that there are two key challenges when operating in the quantum regime: (i) collisions between the beacons have a non-negligible effect when the beacon length is comparable to the listening time; and (ii) synchronization can cause ΔT to become very large when the relative clock skew between a pair of nodes is small.

Reducing Beacon Density. To reduce collisions, a natural approach is to reduce the number of beacons. We noticed that the active slot pattern for Searchlight consisted of an anchor slot and a probe slot. In each slot, there is a beacon-listen-beacon sequence. To reduce the density of beacons, we converted the probe slot into a listening slot, i.e., a node will only listen and not beacon during the probe slot. Next, we note that if we preserve the beacon-listen-beacon sequence for the anchor slot, the minimum slot size is twice

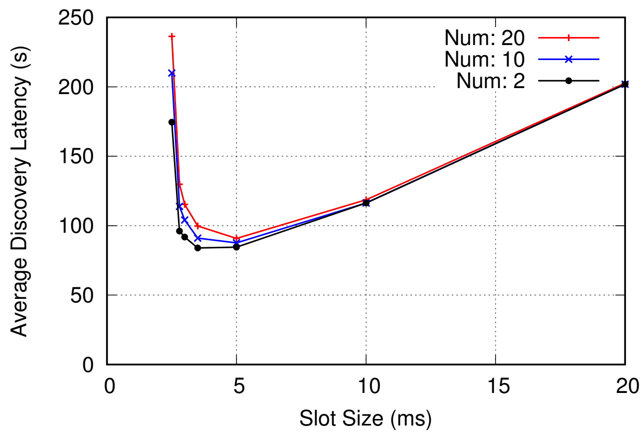


Figure 11: Average discovery latency between a pair of nodes when multiple nodes exist at 1% duty cycle (relative clock skew: 0.6 ppm).

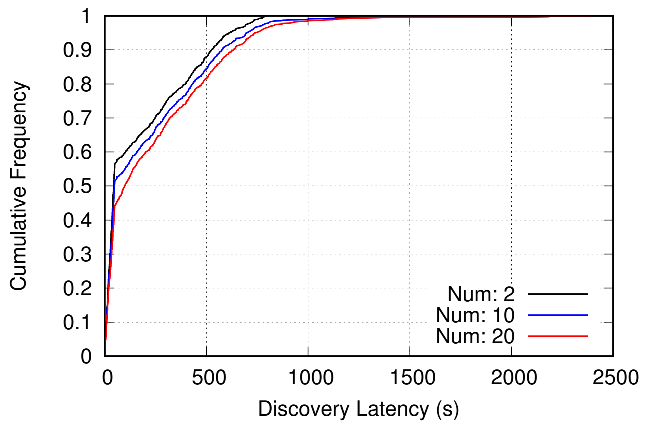


Figure 12: CDF of the discovery latency between a pair of nodes at the slot size 2.5 ms.

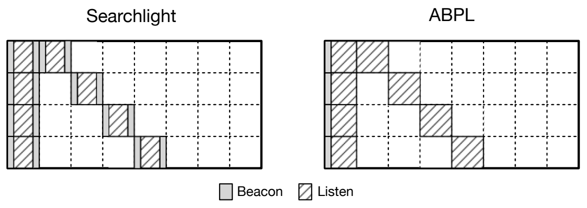


Figure 13: Comparing Searchlight and ABPL.

the beacon length. By reducing the number of beacons for the anchor slot to just one, we can not only reduce the density of beacons, but we can also further reduce the slot size. Others have shown that sending one beacon is sufficient for mutual discovery [18]. We call our new version of Searchlight, *Anchor Beacon Probe Listen (ABPL)*. The search pattern of ABPL is illustrated in Fig. 13.

Introducing Randomization to Mitigate Synchronization. In the rare occasion that two nodes have synchronized offsets, ΔT can become very large. Consequently, if relative clock skew is low, it can take a long time for the nodes to desynchronize. To address this, instead of following a rigid wake-sleep pattern, each sensor node jitters its wake-up time by a small amount to decouple the change in slot offset per period from clock skew. The amount of randomization introduced is proportional to the length of the beacon preamble. With randomization, we no longer have to wait for the slot offset to slowly drift to a discoverable region when a pair of nodes happen to be at a particular slot offset that causes discovery failure.

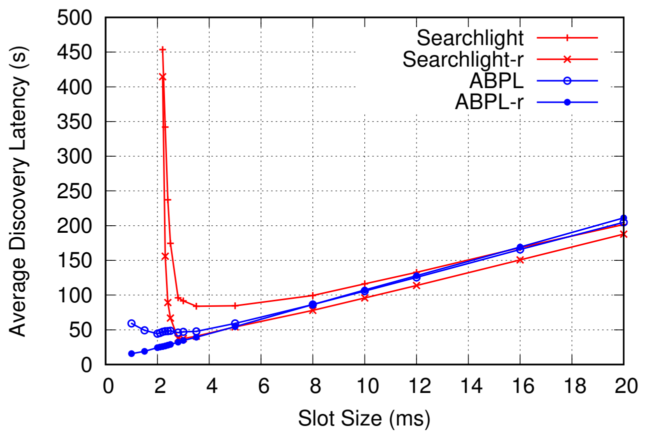


Figure 14: Improvement from ABPL strategy and randomization technique.

In Fig. 14, we compare the performance of vanilla Searchlight to Searchlight+randomization (Searchlight-r), ABPL and ABPL+randomization (ABPL-r) in terms of average discovery latency. We see that the impact of randomization on average latency is minor at large slot sizes. However, when we examine the impact on the long tail of the discovery latency, we see that as expected, randomization is able to reduce the worst-case latency significantly.

4 FOCUSED LISTENING WITH SPOTLIGHT

In ABPL, anchor slots still adhere to the beacon-listen pattern. The beacon length is fixed, so as we decrease the slot size, the listening period during the anchor slot is reduced correspondingly. Once the listening period is shorter than the beacon length, the probability that the anchor slot will successfully “catch” a beacon becomes extremely low, and in most cases, the overlaps are between beacons from the anchor slots and the listening period of probe slots.

In the limit, we are essentially left with two kinds of active slots: beacon (B) and listen (L). We refer to them as *heterogeneous slots* with different beacon-listen activities, in contrast to the conventional homogeneous active slots that use only the beacon-listen-beacon strategy [4]. This is illustrated in Fig. 16.

4.1 Probing considered harmful

Like Searchlight, ABPL distributes the listening time for a period over a series of probe slots. While Searchlight introduced this probing pattern to reduce average latency, we found that this probing pattern introduced discovery failures in the quantum regime that were similar to those discussed in §3.2. The key observation is that the effective listening time of a listening slot is smaller than its length because it would be unable to decode a packet if the preamble did not fall in its entirety within the listening slot. This failure mode is illustrated in Fig. 15, and is analogous to the modes

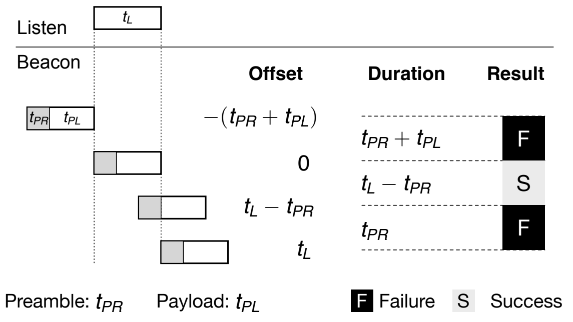


Figure 15: Discovery failure when preamble falls outside listening period. Note that slot extension still applies.

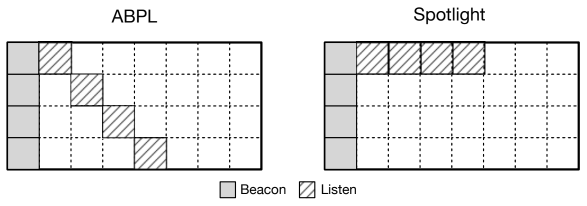


Figure 16: Comparing the active slots of ABPL to Spotlight. We are operating at the minimum slot size with heterogeneous active slots.

described earlier in Fig. 7. We observe from Fig. 15 that with a listening slot of length t_L, the effective listening time is only $t_\text{L} - t_\text{PR}$. Therefore, the probability of discovery failure can be expressed as:

$$P_\text{fail} = \frac{t_\text{PR}}{t_\text{L}} \tag{7}$$

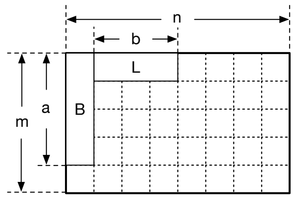
When operating in the quantum regime, the preamble occupies a substantial portion of a slot and this cause of failure can have a significant impact on latency. For example, on our hardware platform with a beacon of length 1 ms and a preamble length of 0.2 ms, the effective failure probability is approximately 20%.

Our key insight is that we can eliminate most occurrences of this failure mode by combining all probe slots into one long listening slot for each period, as shown in Fig. 16. Surprisingly, this does not affect worst-case latency (which we go on to prove in §4.2). While it may seem that the average latency would be adversely affected due to listening slots being less spread out, in practice the average latency is much better (see §5), because the probability of discovery failure due to missed preambles is greatly reduced. More specifically, by combining n probe slots into a single continuous listening slot, the failure probability becomes:

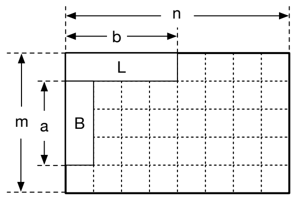
$$P_\text{fail} = \frac{t_\text{PR}}{n \cdot t_\text{L}} \tag{8}$$

In contrast to Searchlight, which spreads out its probe slots, we call our scheme, which has a long listening interval during each period, *Spotlight*, since our listening is focused at a fixed spot. Comparing Eqs. (7) and (8), we found that Spotlight has a much lower failure probability than Searchlight and ABPL. For example, compared to ABPL with 1% duty cycle, Spotlight reduces the failure rate from 20% to 0.2%, a two order of magnitude reduction.

Like all existing deterministic neighbor discovery protocols, discovery failure will occur if the wake-sleep schedules of two nodes are perfectly synchronized. As explained in §3.2, two synchronized nodes will discover each other eventually if the relative clock skew



(a) Variant 1: $(0, 0)$ is a beacon slot.



(b) Variant 2: $(0, 0)$ is a listening slot.

Figure 17: Two variants of BL diagrams.

between them causes them to drift apart. As described in §3.4, we can mitigate this synchronization issue by introducing randomization. Spotlight thus adopts the randomization technique used in ABPL-r. Since failure is caused by missing the preamble, we set the range of randomization to be the length of one preamble.

4.2 Achieving Discovery Guarantees

To analyze Spotlight, we consider a class of pattern matrices $\mathbb{M}(m, n, a, b)$, that we call *BL diagrams*. Each matrix represents a pattern of beacon, listening and sleep slots over a period with $m \cdot n$ slots. As illustrated in Fig. 17(a), the first a slots in the first column are all beacon slots, and the next b slots in the first row adjacent to the first beacon slot are listening slots. An alternative form, where the top left slot is a listen slot, is shown in Fig. 17(b). Spotlight is represented by Variant 1 of the BL diagram, with $\mathbb{M}(m, 2m, m, m)$, where m can be derived from the desired duty cycle setting.

Clearly, not all configurations of m, n, a, b guarantee discovery. To ensure discovery happens within one period, the BL diagram must satisfy two conditions:

(1) Either $a = m$ (Variant 1) or $b = n$ (Variant 2), i.e., either beacon slots must fill up the entire column, or the listening slots must fill up the entire row.
(2) If $a = m$, then $b \geq \lfloor \frac{n}{2} \rfloor$ (Variant 1). Otherwise, if $b = n$, then $a \geq \lfloor \frac{m}{2} \rfloor$ (Variant 2). In other words, with one filled row or column, the remaining row or column has to be longer than half the maximum length.

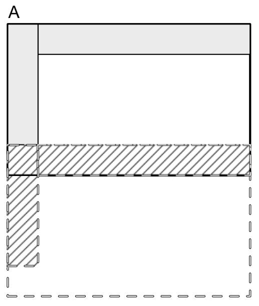
Proof of Condition 1: For Variant 1, b is always smaller than n. Suppose $a < m$. If we let slot $(0, 0)$ of node B overlap with slot $(m-1, 0)$ of node A, then there will not exist a beacon-listen overlap between A and B (see Fig. 18(a)) for any value of b. Therefore, $a = m$.

Similarly, for Variant 2, a is always smaller than m. Suppose $b < n$. If we let slot $(0, 0)$ of node B overlap with slot $(0, 1)$ of node A, then there will not exist a beacon-listen overlap between A and B (see Fig. 18(b)). Therefore, $b = n$. □

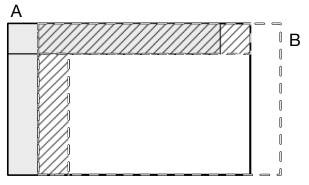
Proof of Condition 2: By symmetry, it suffices to consider Variant 1, i.e., the case $a = m$. Suppose $b < \lfloor n/2 \rfloor$. There will then be $b' = n - b - 1 > b$ empty columns at the right side of the matrix. If we let slot $(0, 0)$ of node B overlap with slot $(0, b + 1)$ of node A, then there will not exist a beacon-listen overlap between A and B (see Fig. 19). Therefore, $b \geq \lfloor n/2 \rfloor$. □

Claim: If Conditions 1 and 2 hold, discovery is guaranteed within one period.

Proof: Consider two nodes A and B with the same BL diagrams. Without loss of generality, we consider the case $a = m$ and let



(a) Variant 1.



(b) Variant 2.

Figure 18: Construction to prove Condition 1.

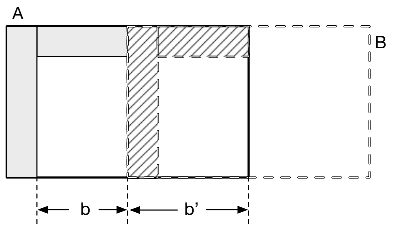
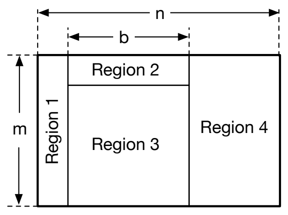
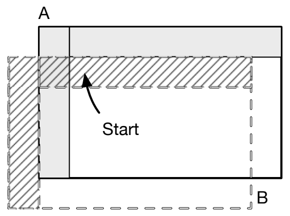


Figure 19: Construction to prove Condition 2.



(a) Enumeration of all possible overlap scenarios.



(b) Worst-case overlap scenario when $a = m$ and $b = n - 1$. Latency for any valid b given this offset is $m \cdot n - 1$ slots.

$B(0, 0)$ (a beacon slot) overlap with every slot $A(i, j)$, $(0 \le i < m, 0 \le j < n)$. We divide A's slots into four regions (see Fig. 20(a)), and consider the discovery outcome within each region exhaustively as follows:

- **Region 1**. All slots in Region 1 are beacon slots. As a result, beacon-beacon overlap always occurs, and discovery will fail. This will be repeated for all the beacon slots of A and B. In other words, Region 1 is an unavoidable "dead zone", as seen in other deterministic protocols. Spotlight incorporates randomization to avoid being trapped in this zone for a prolonged period.
- **Region 2**. All slots in Region 2 are listening slots, resulting in beacon-listen overlap. Discovery is successful with no delay.
- **Region 3**. Since Condition 1 is satisfied, there will exist at least one beacon in B that overlaps with a listening slot in Region 2 of A. Discovery will happen within one period.
- **Region 4**. Since Condition 2 is satisfied, the width of Region 4 is smaller than the width of Region 2. Therefore, one of B's subsequent listening slots $B(0, i), 0 < i \le b$ will overlap with A's beacon slots. Discovery will happen within one period.

We have shown that beacon-listen overlap is guaranteed no matter how A and B's slots overlap. As A and B fit on the same BL diagram, the position of the overlap between A, B is the same in each period. Therefore, given the current beacon-listen overlap, the next beacon-listen overlap will surely happen within $m \cdot n$ slots, i.e., discovery is guaranteed within one period. □

4.3 Worst-case Discovery Latency

Spotlight. Since Spotlight adopts the Variant 1 configuration, we have $a = m$. From Condition 2, we have $b \ge \lfloor n/2 \rfloor$. A smaller b translates to a lower duty cycle and less power used. However, counter-intuitively, choosing a larger $b > n/2$ does not improve worst-case latency. Consider the worst-case slot offset and starting position where $A(1, 1)$ overlaps with $B(0, 2)$, as illustrated in Fig. 20(b). The discovery would take $mn - 1$ slots for $\lfloor n/2 \rfloor \le b < n - 1$ and $mn - 2$ slots for $b = n - 1$.

Consequently, for a fixed set of constants m and n, setting $a = m$ and $b = \lfloor n/2 \rfloor$ would minimize energy consumption, while ensuring discovery within one period. It thus makes sense to first decide on the desired duty cycle C and then set m and n accordingly to minimize the worst-case discovery latency. While beaconing typically requires more energy than listening, the energy consumption for beaconing in modern sensor platforms is only slightly more [1, 10] ($\le 50\%$). For simplicity, we assume that energy consumption is the same, allowing us to express C as:

$$C = \frac{a + b}{mn} \tag{9}$$

(If so desired, the difference in energy consumption can be accounted for by adding an extra constant term to this analysis.)

Since $a = m$ and $b = \lfloor n/2 \rfloor$, we may assume without loss of generality that n is an even integer and substitute $a = m, b = n/2$ into Eq. (9) to obtain:

$$n = \frac{2m}{2Cm - 1} \tag{10}$$

Since the worst-case latency is one period (i.e., mn slots), we have:

$$L = mn = \frac{2m^2}{2Cm - 1} \tag{11}$$

From Eq. (11), L has one global minimum at $m = 1/C$. Assuming $1/C$ is an integer, we can substitute $m = 1/C$ into Eq. (10) to derive $n = 2/C$ and thus $n = 2m$. Hence, Spotlight's pattern achieves the best worst-case latency for a specified duty cycle.

Spotlight-Transposed (Spotlight-T). It turns out that for Variant 2, i.e., the case of $b = n$, we can repeat the same analysis and obtain an analogous conclusion that the minimum latency is achieved when $m = 2n$. Visually, we can transform Spotlight to this configuration, by transposing Spotlight's BL diagram, followed by swapping the beacon and listening slots. Hence, we call this alternative *Spotlight-Transposed* (or Spotlight-T). The worst-case discovery latency for Spotlight and Spotlight-T are equivalent, but we shall see in §5.1 that the average latencies for Spotlight-T are slightly worse. The key difference is that the beacons for Spotlight are uniformly distributed over time, while those for Spotlight-T are not distributed uniformly. Intuitively, it is not surprising that the former achieves better average-case latencies. We leave the detailed analysis as future work.

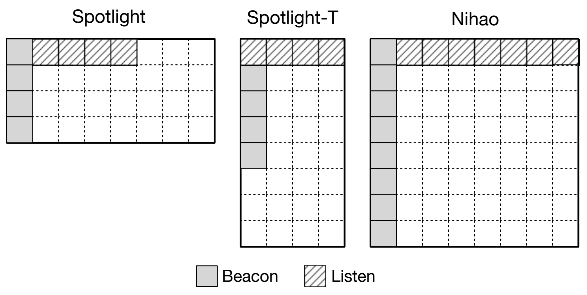


Figure 21: Comparing Spotlight, Spotlight-T and Balanced Nihao at a duty cycle of 25%.

Comparison with Nihao. We noticed that Nihao, when used with the minimum slot size (i.e., $\alpha = 1$), can also be described by our BL diagram, with a parametrization of $\mathbb{M}(m, n, m, n-1)$, where m and n are determined by the duty cycle. We illustrate the patterns for Spotlight, Spotlight-T and Balanced Nihao in Fig. 21. Since we proved that Spotlight and Spotlight-T have optimal worst-case latencies among all possible BL diagram configurations, it holds that Nihao's pattern is suboptimal and is bettered by Spotlight and Spotlight-T. In particular, for the same duty cycle, Balanced Nihao's worst-case latency is approximately twice that of Spotlight and Spotlight-T. Besides having a better beaconing pattern, a key advantage of Spotlight/Spotlight-T over Nihao is the use of randomization to avoid beacon synchronization.

5 PERFORMANCE EVALUATION

In this section, we present our evaluation results comparing Spotlight to existing state-of-the-art neighbor discovery protocols using both our custom simulator and a practical sensor testbed. We evaluated the algorithms at duty cycles of 1% and 5% because these are the most common in practice [4].

NDSim: High-Fidelity Neighbor Discovery Simulator. While we could have implemented and evaluated our algorithms on actual sensor nodes, the process of working with actual physical nodes is slow and laborious. To allow us to develop and evaluate our algorithms more efficiently, we developed *NDSim*, a high-fidelity Python simulator for neighbor discovery protocols.

NDSim is a discrete-event simulator that can efficiently simulate protocols at arbitrary time resolutions (we use a granularity of 1 picosecond for event times). This allows us to account for clock drift, by using slightly different slot sizes for each node, fine-grained failures (e.g., hearing an incomplete beacon, partial collisions between beacons), and the node state transitions derived from logic analyzer traces. The nodes are modeled as state machines for correctness and to correspond to actual protocol implementations in the sensor nodes. Discovery algorithms are implemented as schedulers in each node, which are executed when any node undergoes a state transition.

We validated the simulator against real hardware behavior at the micro-level by verifying state transitions and discovery outcomes using our signal sampler on multiple pairs of sensor nodes and different starting offsets. While a discrete-event simulation cannot perfectly model the behavior of a continuous time system, there were minimal differences and, upon further investigation, most discrepancies could be attributed to being boundary cases that were within a small margin of error. More importantly, at the macro-level, we ensured that the cumulative distributions for the discovery latencies matched those derived from actual hardware nodes over a large number of diverse scenarios. To evaluate average latency, we sampled 1,000 starting offsets for each parameter configuration, meaning that each line presented in the figures in this section is the product of 17,000 simulation runs.

Table 2: Parameters for evaluated algorithms.

Algorithm	Duty Cycle	Period (slots)	Active slots
ABPL-r	1%	20,000	100BL+100L
B-Nihao	1%	$10,000(1+\alpha)^2$	1BL+199B+199L
Searchlight	1%	20,000	200BLB
Spotlight	1%	20,000	100B+100L
Spotlight-T	1%	20,000	100B+100L
ABPL-r	5%	800	20BL+20L
B-Nihao	5%	$400(1+\alpha)^2$	1BL+19B+19L
Searchlight	5%	800	40BLB
Spotlight	5%	800	20B+20L

The simulator was extremely helpful in enabling the rapid prototyping of protocols and in debugging the latency long tails at small slot sizes. In the future, to explore other aspects of neighbor discovery, we may refine the simulator to consider node distances in transmission time, beacon loss due to interference, and to account for the actual resolution of the sensor node oscillators. We plan to release the source code for our simulator in the near future once we finish documenting it.

Experimental Testbed. We have a sensor testbed consisting of 20 sensor nodes which are similar to those used in the micro-climate monitoring systems [14]. The sensor nodes (shown earlier in Fig. 1) have a hardware preamble length of 0.2 ms and our application payload is 0.8 ms, resulting in a beacon length of 1.0 ms. The sensor node does not run any sensor operating system and is programmed directly in C. This is an advantage for our measurement study since it has been shown that sensor operating systems such as TinyOS often introduce additional jitter [4, 23].

Furthermore, we developed a signal sampler to accurately measure the timings of events as well as the clock skew between nodes. Our signal sampler is similar to a logic analyzer but is instead equipped with an Oven-Controlled Crystal Oscillator (OCXO) as the reference clock with 1 ppb temperature stability. We also used the measured timings to calibrate the parameters of our custom simulator so that the results from our simulator are in good agreement with the testbed results.

5.1 Comparison to the State-of-the-Art

In Fig. 22, we compare Spotlight with the two state-of-the-art algorithms Nihao and Searchlight in simulation at 1% duty cycle.

As expected, Searchlight implodes once the slot size falls below 3 ms (3× beacon length). We observe that except for Searchlight, all the other algorithms evaluated could operate at the minimum slot size of 1 ms (i.e., one beacon length). This suggests at the beacon-listen-beacon pattern is not suitable for operating at the quantum scale due to high $P_{\text{fail}}\Delta T$.

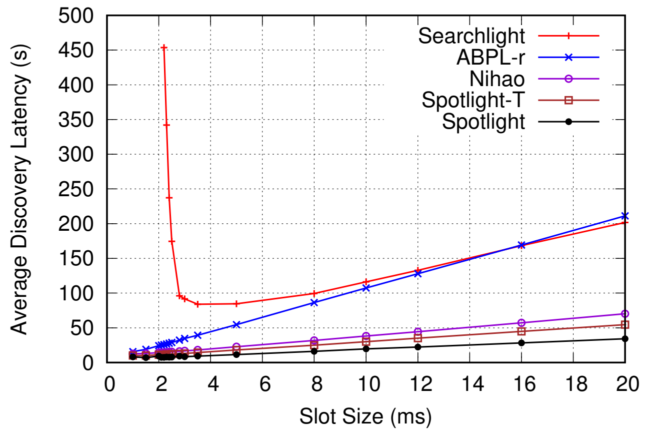


Figure 22: Average neighbor discovery latency at 1% duty cycle.

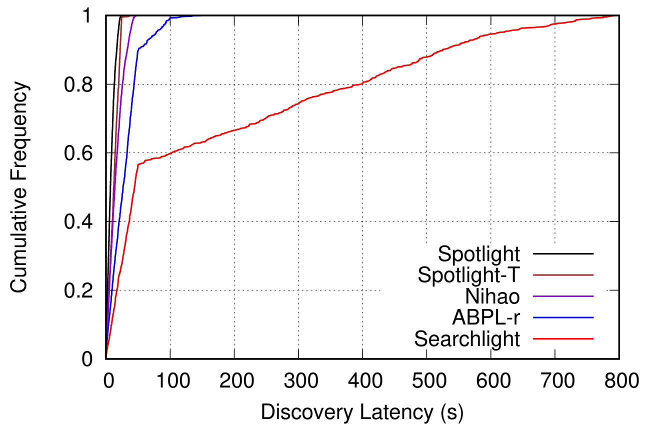


Figure 23: Discovery Latency CDF for 2.5 ms slot size at 1% duty cycle.

Even though Nihao has a similar worst-case latency and power-product latency Λ as other algorithms, it performed surprisingly well at small slot sizes relative to other algorithms. In a similar vein, Spotlight has the same $\Lambda = \sqrt{2L}$ as Searchlight but performs much better. This can be explained by how $\Lambda = DC \cdot L$ considers worst-case latency $N \cdot S$ but not the $P_{\text{fail}}\Delta T$ term in Eq. (5). This demonstrates the presence of gaps in existing theoretical models and analysis when it comes to predicting performance in the quantum regime and taking into account practical discovery failures.

It is plausible that Generic Nihao can be tuned to achieve somewhat better performance than the Balanced Nihao that we evaluated, but theoretically, it is not possible for Nihao to match Spotlight in worst-case performance as shown in §4.3. We also note that Spotlight-T performs slightly worse than Spotlight in the average case and that Spotlight achieves approximately 50% lower discovery latency relative to Balanced Nihao.

In Figs. 23 and 24, we plot the cumulative distributions for the discovery latencies at slot sizes of 2.5 ms and 1 ms, respectively. Unlike Searchlight and ABPL-r, Spotlight and B-Nihao do not exceed their theoretical worst-case discovery latencies of one period, which are 20 s (20,000 slots) and 40 s (40,000 slots) respectively, as shown in Table 2 (given 1 ms slot size and $\alpha = 1$ for Nihao.)

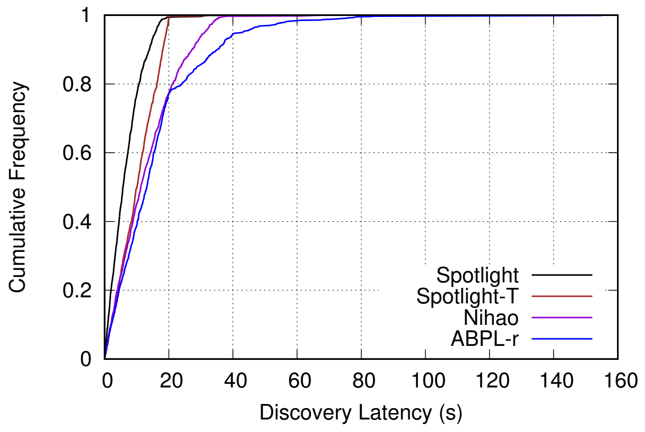


Figure 24: Discovery Latency CDF for 1 ms slot size at 1% duty cycle.

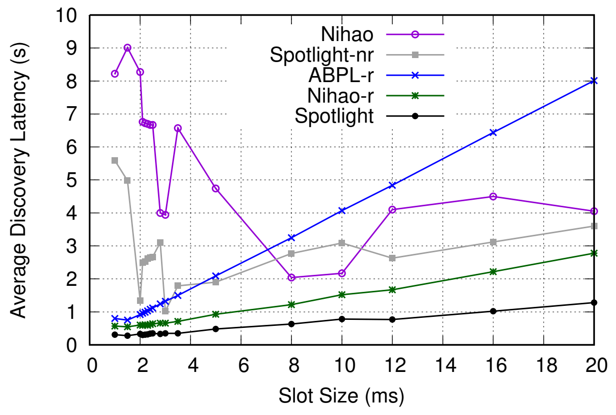


Figure 25: Average neighbor discovery latency at 5% duty cycle.

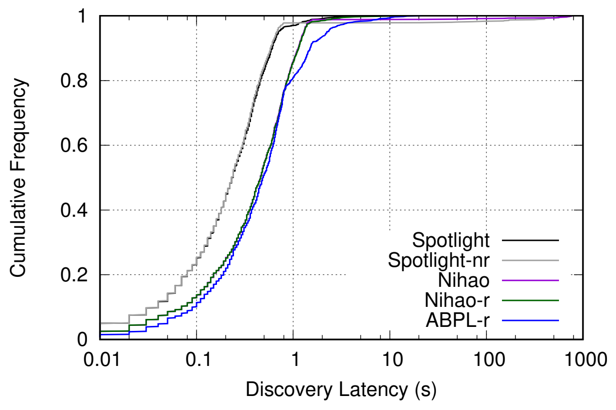


Figure 26: Discovery latency CDF for 1 ms slot size at 5% duty cycle.

5.2 Performance at Higher Duty Cycle

In Fig. 25, we plot the discovery latency against slot size at 5% duty cycle. We did not plot Searchlight in Fig. 25, because its discovery latencies were in the 14 s to 416 s range and much worse than the algorithms presented in the figure.

One key observation is that while Nihao adopts heterogeneous slots and contiguous listening to reduce the probability of discovery failure, at 5% duty cycle we can observe unusually high latency variations at small slot sizes, indicating that discovery failure has

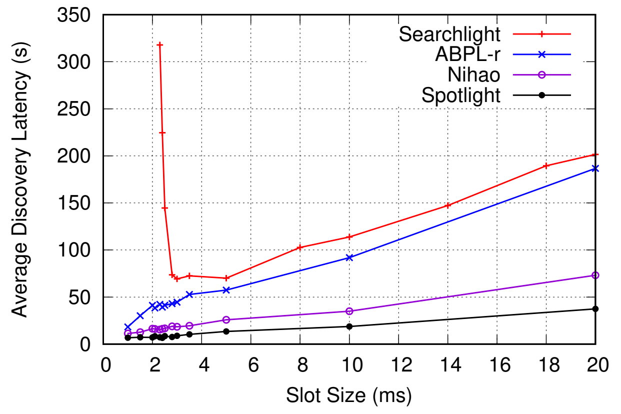


Figure 27: Comparing the neighbor discovery latencies on practical sensor testbed at 1% duty cycle.

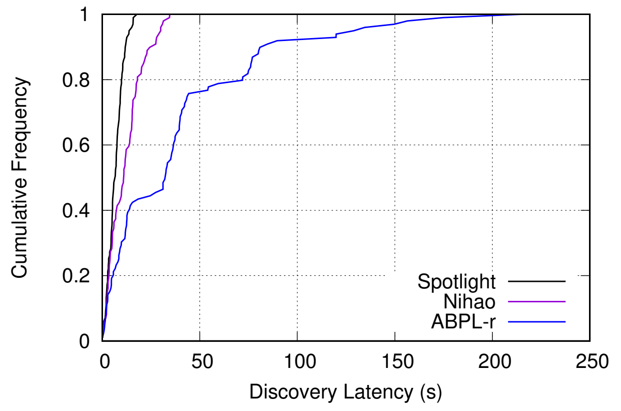


Figure 28: Discovery Latency CDF for 1 ms slot size at 1% duty cycle on practical sensor testbed.

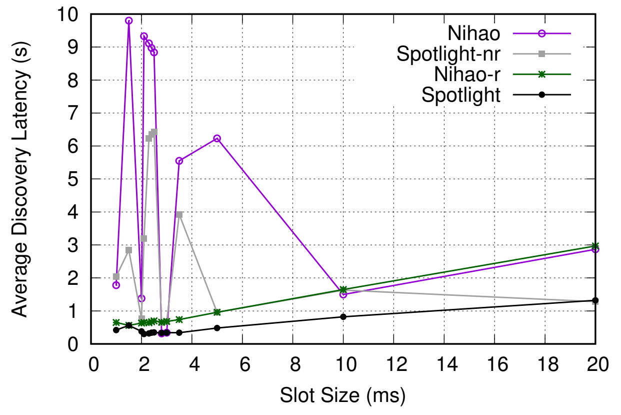


Figure 29: Comparing the neighbor discovery latencies on practical sensor testbed at 5% duty cycle.

substantial effect on the practical discovery latency. This is because at 5% duty cycle, the length of one period becomes much shorter (more than an order of magnitude difference between 1% and 5% duty cycle), so that the equivalent clock drift in one period becomes much smaller. As a result, latency becomes more vulnerable to clock skew.

To verify our reasoning, we evaluated a modified version of Nihao with slot randomization ("Nihao-r"). For comparison, we evaluated a version of Spotlight that did not incorporate randomization ("Spotlight-nr"). Nihao-r seemed to achieve good performance while Spotlight-nr performed poorly. This suggests that our randomization technique is critical in allowing Spotlight to achieve good latencies, and that it can be generally applied to existing neighbor discovery protocols.

We can see from Fig. 26 that the cumulative distributions for the randomized and non-randomized versions of the various algorithms are very similar in shape. The key difference is that the randomized versions are able to better guarantee the theoretical worst-case latencies, while simulated discovery latencies for the non-randomized algorithms can exceed the theoretical worst-case latency by two orders of magnitude.

5.3 Experiments on Sensor Testbed

In Figs. 27 and 29, we plot the results obtained from our sensor testbed for 1% and 5% duty cycles, respectively. If we compare the results with those from our simulations in Figs. 22 and 25, we find that the results are in general agreement for both the trends and the values. The CDF for 1% duty cycle at a slot size of 1 ms is presented in Fig. 28. Searchlight is not included because it cannot operate at a slot size of 1 ms. This validates the results from our simulations.

What is perhaps more significant about the results attained with our sensor testbed is that we have shown that it is feasible to operate at a slot size of 1 ms. Ordinarily, if Searchlight were employed with a slot size of 10 ms, the average discovery latency would have been ~120 s. Spotlight can achieve an average discovery latency of 5 s. This realizes an almost 2-orders-of-magnitude improvement. Even if Searchlight were to be employed with a small slot size of 3 ms, the achieved latency would have been ~70 s, meaning that Spotlight is still able to deliver an order-of-magnitude improvement. Nihao does surprisingly well at small slot sizes, but even then Spotlight still achieves latencies that are half that of Nihao.

6 CONCLUSION

Previous neighbor discovery algorithms focused on exploring the trade-off between power consumption and discovery latency. Our investigation of the impact of clock skew and beacon collisions when operating at the quantum scale suggests that there was a gap in the existing theoretical models for modeling worst-case and average-case discovery latency. To this end, we developed techniques to enable neighbor discovery to operate at effective slot sizes comparable to the beacon length. We present and evaluate *Spotlight*, which incorporates randomization and a provably optimal beaconing strategy. While slot offset randomization may be a simple idea, we show that it is an important and generally applicable mechanism in the quantum regime for higher duty cycles. To the best of our knowledge, we are the first to demonstrate in a practical sensor network that neighbor discovery can be achieved at a slot size of 1 ms. We believe that our approach will lead to significantly lower discovery latencies over time as the sensor hardware continues to improve and support smaller beacons.

REFERENCES

[1] Manual of AT86RF212, http://www.atmel.com/images/doc8168.pdf. (2010).

[2] Giuseppe Anastasi, Marco Conti, Mario Di Francesco, and Andrea Passarella. 2009. Energy Conservation in Wireless Sensor Networks: A Survey. *Ad Hoc Networks* 7, 3 (2009), 537–568.

[3] Mehedi Bakht, Matt Trower, and Robin Hilary Kravets. 2012. Searchlight: Won't You Be My Neighbor?. In *Proceedings of MobiCom '12*.

[4] Prabal Dutta and David Culler. 2008. Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications. In *Proceedings of SenSys '08*.

[5] Valerie Galluzzi and Ted Herman. 2012. Survey: Discovery in Wireless Sensor Networks. *International Journal of Distributed Sensor Networks* (2012).

[6] Kai Han, Jun Luo, Yang Liu, and Athanasios V Vasilakos. 2013. Algorithm Design for Data Communications in Duty-cycled Wireless Sensor Networks: A Survey. *IEEE Communications Magazine* 51, 7 (2013), 107–113.

[7] Jason Hill, Robert Szewczyk, Alec Woo, Seth Hollar, David Culler, and Kristofer Pister. 2000. System Architecture Directions for Networked Sensors. *ACM SIGOPS Operating Systems Review* 34, 5 (2000), 93–104.

[8] Shuaizhao Jin, Zixiao Wang, Wai Kay Leong, Ben Leong, Yabo Dong, and Dongming Lu. 2015. Improving Neighbor Discovery with Slot Index Synchronization. In *Proceeding of MASS '15*.

[9] Christine Julien, Chenguang Liu, Amy L Murphy, and Gian Pietro Picco. 2017. BLEnd: Practical Continuous Neighbor Discovery for Bluetooth Low Energy. In *Proceedings of IPSN '17*. ACM.

[10] Arvind Kandhalu, Karthik Lakshmanan, and Ragunathan Raj Rajkumar. 2010. U-connect: A Low-latency Energy-efficient Asynchronous Neighbor Discovery Protocol. In *Proceedings of IPSN '10*.

[11] Philipp H Kindt, Marco Saur, and Samarjit Chakraborty. 2016. Slotless Protocols for Fast and Energy-Efficient Neighbor Discovery. *arXiv preprint arXiv:1605.05614* (2016).

[12] Shouwen Lai, Binoy Ravindran, and Hyeonjoong Cho. 2010. Heterogenous Quorum-based Wake-up Scheduling in Wireless Sensor Networks. *IEEE Trans. Comput.* 59, 11 (2010), 1562–1575.

[13] Michael J McGlynn and Steven A Borbash. 2001. Birthday Protocols for Low Energy Deployment and Flexible Neighbor Discovery in Ad Hoc Wireless Networks. In *Proceedings of MobiHoc '01*.

[14] Xia Ming, Dong Yabo, Lu Dongming, Xue Ping, and Liu Gang. 2008. A Wireless Sensor System for Long-term Microclimate Monitoring in Wildland Cultural Heritage Sites. In *Proceedings of IPSA '08*.

[15] Ivan Niven, Herbert S Zuckerman, and Hugh L Montgomery. 2008. *An Introduction to The Theory of Numbers*. John Wiley & Sons.

[16] Joseph Polastre, Jason Hill, and David Culler. 2004. Versatile Low Power Media Access For Wireless Sensor Networks. In *Proceedings of IPSN '04*.

[17] Aveek Purohit, Bodhi Priyantha, and Jie Liu. 2011. WiFlock: Collaborative group discovery and maintenance in mobile sensor networks. In *Proceedings of IPSN '11*.

[18] Ying Qiu, Shining Li, Xiangsen Xu, and Zhigang Li. 2016. Talk More Listen Less: Energy-Efficient Neighbor Discovery in Wireless Sensor Networks. In *Proceedings of INFOCOM '16*.

[19] Wei Sun, Zheng Yang, Keyu Wang, and Yunhao Liu. 2014. Hello: A Generic Flexible Protocol for Neighbor Discovery. In *Proceedings of INFOCOM '14*.

[20] Wei Sun, Zheng Yang, Xinglin Zhang, and Yunhao Liu. 2014. Energy-efficient Neighbor Discovery in Mobile Ad Hoc and Wireless Sensor Networks: A Survey. *Communications Surveys and Tutorials, IEEE* 16, 3 (2014), 1448–1459.

[21] Yu-Chee Tseng, Chih-Shun Hsu, and Ten-Yueng Hsieh. 2003. Power-saving Protocols for IEEE 802.11-based Multi-hop Ad Hoc Networks. *Computer Networks* 43, 3 (2003), 317–337.

[22] Keyu Wang, Xufei Mao, and Yunhao Liu. 2013. BlindDate: A Neighbor Discovery Protocol. In *Proceedings of ICPP '13*.

[23] Desheng Zhang, Tian He, Yunhuai Liu, Yu Gu, Fan Ye, Raghu K Ganti, and Hui Lei. 2012. ACC: Generic On-demand Accelerations for Neighbor Discovery in Mobile Applications. In *Proceedings of SenSys '12*.

[24] Rong Zheng, Jennifer C Hou, and Lui Sha. 2003. Asynchronous Wakeup for Ad Hoc Networks. In *Proceedings of MobiHoc '03*.