On the Energy Efficiency and Performance of Neighbor Discovery Schemes for Low Duty Cycle IoT Devices

Junaid Ahmed Khan, Romain Pujol, Razvan Stanica, Fabrice Valois
Univ Lyon, INSA Lyon, Inria, CITI, F-69621 Villeurbanne, France
{junaid-ahmed.khan,romain.pujol,razvan.stanica,fabrice.valois}@insa-lyon.fr

ABSTRACT

Mobile sensing and proximity-based applications require smart devices to find other nodes in vicinity, though it is challenging for a device to find neighbors in an energy efficient manner while running on low duty cycles. Neighbor discovery schemes allow nodes to follow a schedule to become active and send beacons or listen for other active nodes in order to discover each other with a bounded latency. However, a trade-off exists between the energy consumption and the time a node takes to discover neighbors using a given activity schedule. Moreover, energy consumption is not the only bottleneck, as theoretically perfect schedules can result in discovery failures in a real environment.

In this paper, we provide an in-depth study on neighbor discovery, by first defining the relation between energy efficiency, discovery latency and the fraction of discovered neighbors. We evaluate existing mechanisms using extensive simulations for up to 100 nodes and testbed implementations for up to 15 nodes, with no synchronization between nodes and using duty cycles as low as 1% and 5%. Moreover, the literature assumes that multiple nodes active simultaneously always result in neighbor discovery, which is not true in practice as this can lead to collisions between the transmitted messages. Our findings reveal such scalability issues in existing schemes, where discovery fails because of collisions between beacons from multiple nodes active at the same time. Therefore, we show that energy efficient discovery schemes do not necessarily result in successful discovery of all neighbors, even when the activity schedules are computed in a deterministic manner.

KEYWORDS

Neighbor Discovery Schemes; Energy Efficiency; Low Duty Cycle; Wireless Sensor Networks; Internet of Things

1 INTRODUCTION

The proliferation of smart mobile devices results in an increasing demand for proximity-based networking applications. For example, proximity-based gaming applications on Sony PS Vita [1] require mobile devices in each others proximity to interact locally as an ad-hoc network. Similarly, in wireless sensor networks, nodes need to discover each other and cooperate for data collection purposes [2]. The first step for devices or nodes to connect is to efficiently discover each other once in communication range. Neighbor discovery for energy-constrained devices is a challenging task, particularly for battery-powered nodes opting for duty cycling to achieve energy efficiency [3]. Duty cycling enables a node to choose between sleep and active mode with the aim of conserving energy for a large fraction of time. We note that nodes can have a clock drift between their active periods, requiring a careful consideration for a neighbor discovery scheme to be robust with respect to time synchronization between nodes. Another issue is the existence of heterogeneous duty cycles in a neighborhood, i.e. the fraction of time a node is active can be different from other nodes, thus requiring a neighbor discovery scheme to cope with the diversity in the duty cycles of some neighbors.

Energy efficient asynchronous neighbor discovery schemes focus on minimizing the worst case latency between a pair of nodes, while ensuring discovery with low energy consumption. In these schemes, time is divided into slots, and each node becomes active on a limited number of slots as the schedule defined based on its respective duty cycle. Neighbor discovery is possible when two or more nodes are simultaneously active in the same slot. An overlapping wake up time is thereby considered as an opportunity for mutual discovery between nodes. However, it is possible that a fraction of nodes in a neighborhood fail to find each other using a neighbor discovery scheme. Such failures in discovery are due to *i)* non existence of an overlapping wake up time between nodes, and *ii)* collisions between nodes active in the same slot, due to multiple beacons transmitted simultaneously. This second phenomenon is rarely considered in the related literature. Therefore, we believe there is a lack of analysis in the field and that neighbor discovery schemes need to be evaluated under realistic settings.

To address this issue, in this paper, we evaluate the energy efficiency and latency of neighbor discovery schemes with respect to the ratio of discovered neighbors. We first define two metrics to study the relation between *i)* the energy consumption and the fraction of neighbors discovered, and *ii)* the discovery latency and the fraction of neighbors discovered. We consider this approach as energy efficiency and latency result in a trade-off, i.e. reducing energy consumption by lower duty cycling can lead to longer discovery latency. We perform extensive simulations as well as test-bed evaluations on an open large scale FIT (Future Internet of the Things) IoT-LAB platform [4], for an in-depth performance comparison of state of the art neighbor discovery mechanisms using our proposed metrics. Moreover, in order to deal with collisions between messages, we allow nodes to implement a collision avoidance mechanism (CSMA/CA), where nodes active choose a random back off period to wait within the wake-up slot before transmitting their beacons.

PE-WASUN'17, Miami, FL, USA
© YYYY Copyright held by the owner/author(s). 978-x-xxxx-xxxx-x/YY/MM. . . $15.00
DOI: 10.1145/nnnnnnn.nnnnnnn

We implement seven state of the art schemes using simulations, analyzing their scalability to discover up to 100 nodes in the communication range of each other (clique-like network structure), while operating on low duty cycles of 1% and 5%. Additionally, we perform real experiments using up to 15 nodes placed in a clique like structure as nodes are more prone to collisions. Our findings reveal that collisions greatly affect the neighbor discovery performance and a highly energy efficient scheme can sometimes perform poorly and miss substantial discovery opportunities when nodes in the communication range of each other (clique) tries to discover each other. Our contributions can be summarized as follows:

- we define two new metrics as a benchmark to evaluate the energy consumption and latency with respect to the fraction of discovered neighbors;
- Evaluate seven state of the art schemes under common simulation settings for use cases where we vary the number of nodes (up to 100) and their duty cycles;
- we analyze four state of the art neighbor discovery schemes using a testbed implementation for 15 nodes;
- we study the impact of collisions on neighbor discovery by adding a CSMA/CA mechanism on the nodes; we note that such an experiment was not previously performed for any of the considered solutions.

The remainder of the paper is organized as follows. Section 2 discusses the related work followed by the definition of key performance metrics in Section 3. In Section 4, we provide the theoretical basis of different neighbor discovery schemes we analyze. In Sections 5 and 6, the performance evaluation and important findings based on extensive simulations and experimentation are discussed respectively. Section 7 summarize our findings and finally, Section 8 concludes the paper along insights into future directions.

2 NEIGHBOR DISCOVERY IN A NUTSHELL

Asynchronous neighbor discovery mechanisms can be classified into two categories: *i)* direct, where nodes discover only the neighbors from which they receive a message [5] [6], and *ii)* indirect schemes, where nodes can learn the existence of neighbors from other nodes [7] [8] [9]. Indirect collaborative mechanisms are application-specific and difficult to compare quantitatively, since nodes in geographical proximity, but not direct neighbors in the network, can be discovered by such schemes. Therefore, in this paper we focus on direct schemes, which can be further classified into *i)* quorum-based [10], *ii)* prime number-based [11] [12], *iii)* dynamic listen slot [13] [14], *iv)* fixed listen slot [15] [16], and *v)* stochastic [17]. These schemes can work on one or multiple frequency channels [18] [19] [20], and they all follow a similar principle of dividing time into slots and letting the node to be active in a slot based on a schedule defined by the respective algorithm. One other underlying assumption of all these mechanisms is that nodes are not temporally synchronized, meaning that the beginning of a slot is different for all the nodes.

Quorum-based schemes [10] guarantee that two nodes have at least one activity slot in common in a period of N slots by being active in $\sqrt{N}$ slots. These mechanisms result in relatively high duty cycles and only function in homogeneous duty cycle conditions. The cyclic quorum design in heterogeneous duty cycle conditions is known as asymmetric design, and specific solutions were proposed to address this problem. Prime number-based asymmetric discovery schemes require a node to choose a single (e.g. U-Connect [12]) or a pair of prime numbers (e.g. Disco [11]) to derive its duty cycle. The activity slots of a node will be the multiples of the selected prime number(s). This approach can be extended, and differential codes can be built for each pair of nodes starting from relatively prime numbers [21]. Using results from number theory, it can be shown that any two nodes will finally wakeup on the same slot. The discovery latency in this case is the time slot corresponding to the product of the prime numbers used by the two nodes. The different strategies also take different approaches in the activity slots. Disco proposes to send two beacons in each activity slot, one at the beginning and one at the end, and listen for incoming beacons from potential neighbors in the rest of the slot. The slot of U-Connect comprises a single beacon, followed by a listen period.

However, the transmission and listen activities are independent and they can be conducted on different slots. In dynamic listen slot schemes, a large time period is divided into regular sized cycles, where each cycle is further composed of slots. Two types of slots exists, static transmission slots at fixed positions, either at the beginning or end of the cycle, and dynamic listen slots with a regular shift to either the left or right in consecutive cycles, up to the end of the period. Searchlight [14] is an example of such an approach, where a node has a static slot in the beginning of each cycle and an active slot shifted one slot to the right in each consecutive cycle. Similarly, Blinddate [13] uses one static slot in each cycle and two dynamic listen slots, one shifted to the right and one to the left in each consecutive cycle. A fixed schedule can also be used for listen slots. Nihao [16] takes the approach of *talk more listen less*, where more transmissions than listen slots exist in a given period. In the same context, Hello [15] is a highly parameterizable solution, where nodes listen more at the beginning of the period, and periodically wake up for transmissions. This scheme is shown to be a generalization of several other mechanisms, such as Disco, U-Connect and Searchlight. Finally, stochastic schemes such as Birthday [17] allow nodes to transmit beacons, listen for beacons from other nodes or sleep in a slot based on a probability distribution. Energy efficiency is ensured by choosing a lower probability for beacon transmission or for listening. Such schemes perform better on the average case compared with the deterministic approaches above, but they provide no bound on the worst case latency and they can lead to long tails in discovering the last fraction of nodes.

Existing schemes are only tested is small scenarios, where two nodes try to discover each other within a bounded delay. There is a need to analyze the scalability of different approaches in larger scenarios, particularly studying the influence of collisions on the fraction of discovered neighbors. We note that the proposed schemes function under the assumption of asynchronous clocks; indeed, if a transmission slot simultaneously begins on two nodes, they will both send beacons and collide. While synchronization is indeed difficult to achieve in an ad-hoc network and asynchronous solutions are clearly needed, we argue that counting on the clock drifts of the nodes to avoid collisions is a property just as difficult to obtain in practice.

3 ENERGY-LATENCY-DISCOVERY RELATION

We consider a set of nodes $N = \{n\}$ in the proximity (communication range) of each other, forming a clique like network structure where each node is with degree $k_n = |N| - 1$. To study co-located nodes discovery, we assume a relatively stable mobility, i.e. for a while, the nodes stay in the same neighborhood. A node operates on low duty cycles, i.e. alternates between sleep and active mode in order to save energy. The node becomes active for a small amount of time t_b to transmit a beacon, or during time t_l to listen to incoming beacons from other devices, in a relatively larger time period T, where $t_b < t_l << T$. The time T is divided into regular slots for a node to become active to transmit a beacon or listen for a slot duration. A node implementing a neighbor discovery scheme employs a given schedule to send beacons or listen to beacons from another nearby node in active slots. Thus, the goal is to opportunistically find a time when two or more nodes are simultaneously active, to guarantee a successful discovery. The energy consumption E_n of the node n to be active (send beacon or listen) by following a schedule defined by a neighbor discovery scheme as: $E_n = be_b + le_l$, where, b represents the number of transmitted beacons, e_b is the energy a node takes to transmit a beacon . Similarly, l are the number of listen slots each of with energy consumption e_l. The latency for the node n to discover its neighbors is L_n, characterizing the worst case latency obtained by using any neighbor discovery scheme. Generalizing for a total of N nodes in a neighborhood, a discovery scheme consumes on average $E_N = \frac{1}{N} \sum\limits_{n \in N} E_n$ with a delay of $L_N = \frac{1}{N} \sum\limits_{n \in N} L_n$.

A node might fail to discover some neighbors due to the following reason, (i) in case it is in sleep mode or fails to become active to send or receive beacons from another node active at the same time, thus missing a discovery opportunity, (ii) in case beacons sent by multiple nodes arrive simultaneously at a node in listen mode can lead to collisions at the node due to interference, resulting in a discovery failure.

Thus, the number of neighbors discovered using a scheme can differ between nodes. We define below a measure to analyze different neighbor discovery schemes for such uneven number of neighbor discoveries among nodes.

Definition 1. **Average Discovered Neighbors** The number of neighbors discovered by a node n is defined as the cardinality of $D_n \subset N$, the set of neighbors discovered by n. Similarly, the average number of neighbors discovered for a set of N nodes is the cardinality of the set D_N represented as $D_N = \frac{1}{N} \sum\limits_{n \in N} D_n$, . where the unit of both D_n and D_N are measured as the number of nodes.

it is important to consider the worst case latency in discovering neighbors, though, in the case when there are only a fraction of neighbors discovered, we need to consider the joint relation of the latency and average neighbor discovery.

Definition 2. **Latency vs Discovery** The latency vs discovery relation for a neighbor discovery process considering uneven number of neighbors discovered for a set of nodes N is given as:

$$\theta_N = L_N \times (1 + \frac{D_N}{N})^{-1}, \tag{1}$$

Table 1: Neighbor Discovery Schemes Comparison

Scheme	Parameter(s)	Duty Cycle	No. of beacons	No. of listens	Latency(L_n)	Energy Consumption (E_n)
Birthday [17]	p_b, p_l, p_s	p_b, p_l	$p_b L_n$	$p_l L_n$	-	$L_n(p_b e_b + p_l e_l)$
Blinddate [13]	c, t, s	$5s \cdot \lceil \frac{s}{2} \rceil$	$2c$	c	$\frac{3}{5s}$	$c(2e_b + e_l)$
Disco [11]	p_1, p_2	$\frac{1}{p_1} + \frac{1}{p_2}$,	$2(p_1 + p_2)$	$p_1 + p_2$	$p_1 p_2$	$2(p_1 + p_2)e_b + (p_1 + p_2)e_l$.
Hello [15]	c, t, s	$\frac{\lfloor \frac{c}{2} \rfloor + t}{ct}$	c	c	$\frac{c^2}{2}$	$c(2e_b + e_l)$
Nihao [16]	c, t, s	$2/c$	s	c	$\frac{c^2}{2}$	$c(e_b + e_l)$
Searchlight [14]	c, t, s	$2/c$	$2c$	c	$\frac{c^2}{2}$	$c(2e_b + e_l)$
U-Connect [12]	p	$\frac{3}{2p}$	$\frac{3p}{2}$	$\frac{3p}{2}$	p^2	$\frac{3p}{2}(e_b + e_l)$

where L_N is the worst case latency for discovering D_N number of neighbors among N nodes. the term θ is measured as the number of time-slots.

The average neighbor discovery and latency vs discovery described above find the extent at which a scheme allows co-located nodes to successfully discover each other within a bounded delay L_N. However, we also need to consider a measure on the energy efficiency, as the discovery process should be less energy consuming and incur low delay. Therefore, we characterize jointly the energy consumption, the average number of neighbors discovered and the average delay in order to better analyze a neighbor discovery solution, using the relations below:

Definition 3. **Energy vs Discovery** We define, for a set of nodes N, the relation between the total fraction of neighbors discovered, the energy consumption and the latency needed for discovery as:

$$\delta_N = E_N \times \theta_N \tag{2}$$

where, E_N and L_N are the energy consumption and latency of N nodes discovering each other; the term $\frac{D_N}{N}$ measures the fraction of discovered nodes D_N among N nodes in a neighborhood. The relation δ is measured as energy times the number of time slots and provides a common benchmark, considering together the total fraction of discovered neighbors, latency and energy efficiency when using a given neighbor discovery scheme.

4 ANALYTICAL EVALUATION

In this section we discuss the analytical parameters of the considered neighbor discovery schemes used to evaluate their performance. In particular, we focus on the worst case latency and the energy consumption. We provide the analytical bound on the worst case latency L_n for a node to discover its neighbors using each scheme. Similarly, the energy consumption E_n by a node executing a neighbor discovery scheme can be derived from the number of beacons and listen periods associated with their respective schedule. Given L_n and E_n, we can compute the latency vs discovery and the energy vs discovery metrics for the average number of neighbors using Equations 1 and 2 respectively.

To ensure a fair comparison between each of the considered scheme for nodes running on a desired duty cycle, we theoretically derive key parameter for nodes to achieve the desired duty cycle. Specifically, we are interested in the nodes respective wake up schedule, the energy consumption on beacons and listen periods, and the latency for successful neighbor discovery when operating on a particular duty cycle. Table 1 summarizes such comparison for each neighbor discovery scheme. The first column shows the key parameters used to define a node wake up schedule in order to attain a respective duty cycle, where in the second column we provide the relation between these parameter to attain a desired duty cycle. For Birthday, p_b, p_l and p_s are the probabilities of sending beacon, listening or sleeping in a slot. Similarly, for Disco p_1, p_2 are the primes selected to derive a desired duty cycle, where for U-Connect the prime p is the key parameter to define its schedule. For Blinddate, Hello, Nihao and Searchlight, the cycle length c and t indicates the number of times the cycles is repeated within a worst case latency L_n.

The number of beacons and listen slots can be derived for each scheme are shown in the third and fourth column. Similarly, we can obtain the latency bound L_n for each mechanism (except Stochastic) as well as its energy consumption E_n in order to discover neighbors. It is to note that there is no bound on the worst case latency for discovery defined by Birthday due to its stochastic nature, however, we need to compare it with other benchmark deterministic schemes. Therefore, we consider $L_n = \frac{c^2}{2}$, as of Searchlight, Hello, Nihao to ensure fair evaluation as well as define a bound the discovery latency.

Similarly, Table 2 summarizes the numerical values of each parameters used to derive each neighbor discovery schedule for a node operating at 1% and 5% duty cycles within a delay L_n. For a node using Disco, the primes $p_1, p_2 = 191, 211$ can be used to numerically achieve a duty cycle of 1% where the worst case latency is $L_n = 40301$ slots. The energy consumption is given as $E_n = 804e_b + 402e_l$, where $2(p_1 + p_2) = 804$ are the number of beacons and $p_1 + p_2 = 402$ are the number of listen slots for the Disco schedule within the bounded delay. Similarly, $p_1, p_2 = 37, 43$ allow us to achieve 5% duty cycle within $L_n = 1591$ and the energy consumed as $E_n = 160e_b + 80e_l$ with 80 active time-slots.

U-Connect uses the prime $p = 151$ for 1% duty cycle to discover within $L_n = 22801$ time-slots of which 228 are active. It uses the prime $p = 31$ for a node on 5% duty cycle with 48 active time-slots. The energy consumption is $E_n = 228(e_b + e_l)$ within the respective latency. For a relative high duty cycle of 5%, $p = 31$ can generate 48 active slots for beacons/listens (48 each) within the bounded delay of $L_n = 961$ slots.

To achieve 1% duty cycle, Searchlight and Hello use $c = 200$, $t = s = 100$, where the number of beacons and listen periods are 400 and 200 respectively resulting in an energy consumption of $E_n = 200(2e_b + e_l)$ within the worst case latency $L_n = 20000$ slots. For 5% duty cycle, $c = 40$, $t = s = 20$ with a total of 80 beacons and 40 listen periods. The energy consumption is $E_n = 40(2e_b + e_l)$ within the worst case latency $L_n = 800$ slots. Nihao uses $c = 200$, $t = 100$ for 1% duty cycle, listen slot are the first $s = 100$ and at the beginning of each consecutive cycle, a beacon is sent upto $L_n = 20000$ as the worst case delay. The energy consumption is $E_n = 200(e_b + e_l)$. Similarly for 5% duty cycle, $c = 40$, $t = 20$ and $s = 20$ is used by the node for its wake up schedule upto $L_n = 800$ slots with an energy consumption of $E_n = 40(e_b + e_l)$.

For Blinddate, a duty cycle of 1% can be achieved using $c = 300$ repeated $t = 30$ times within $L_n = 9000$ slots, thus, a total of 90 active slots in the bounded latency. The cycle is divided into sub-slots of size $s = 60$ where there exists 5 sub-slots in a cycle. The energy consumption of Blinddate under 1% duty cycle is $E_n = 90(e_b + e_l)$, i.e. 90 beacons and 90 listen periods. Similarly, for 5% duty cycle, $c = 60$ repeated $t = 6$ times with a worst case latency of $L_n = 360$ slots. The cycle is divided into 5 sub-slots each of size $s = 12$ slots. The number of beacons and listen periods are 18 each leading to an energy consumption $E_n = 18(e_b + e_l)$. for a node under 5% duty cycle. Thus, theoretically Blinddate requires the least number of active slots for discovery. However there is no information on the energy consumption and the average number of discovered neighbors. Such analysis is discussed in the next section where we analyze each mechanism using extensive simulations.

Table 2: Neighbor Discovery Scheme Evaluation Parameters

	1%					5%				
Scheme	L_n	c	t	s	p_1,p_2/p	T	c	t	s	$p_1, p_2/p$
Birthday [17]	20000				200	800				40
Blinddate [13]	9000	300	30	60		360	60	6	12	
Disco [11]	40301			402	191, 211	1591			80	37, 43
Hello [15]	20000	200	100	100		800	40	20	20	
Nihao [16]	20000	200	100	100		800	40	20	20	
Searchlight [14]	20000	200	100	100		800	40	20	20	
U-Connect [12]	22801			228	151	961			48	31

Since there is no bound defined for stochastic schedule, we use $L_n = 20000$, and 800 for a node to active 200 and 40 slots for 1% and 5% duty cycle respectively. Thus, a node operating on 1% duty cycle uses $p_b = p_l = 200$, i.e. 200 beacons and 200 listens within $L_n = 20000$ slots. The energy consumption $E_n = 200(e_b + e_l)$. Similarly, for 5% duty cycle, $p_b = p_l = 40$ results in 40 beacons and listen periods respectively consuming $E_n = 40(e_b + e_l)$ within $L_n = 800$ slots.

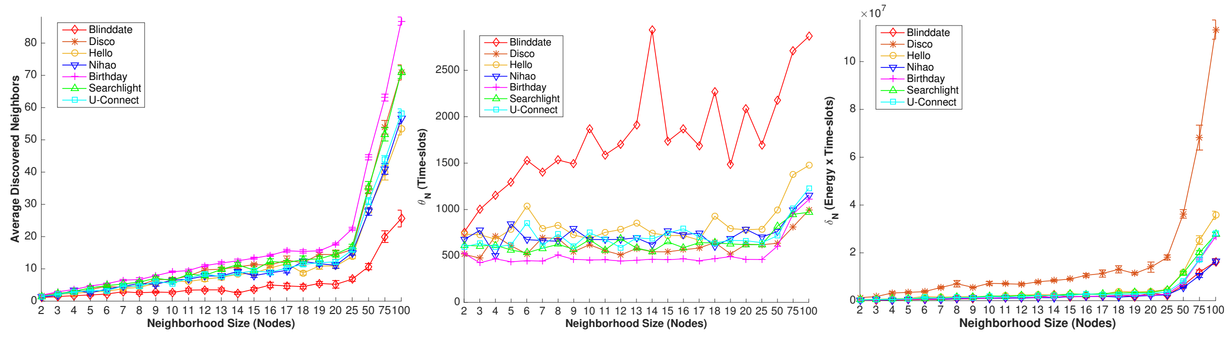
5 SIMULATION-BASED EVALUATION

5.1 Simulation Scenario

Simulations are performed by implementing each of the discussed neighbor discovery mechanism in NS-3. Neighborhood are formed by placing a set of nodes in the communication range (around 150m) of each other in an ad-hoc network corresponding to a clique based network. We assume nodes stay in the neighborhood for some time. i.e. the topology does not evolve during our analysis. We evaluated each mechanism by considering up to 100 co-existing nodes where the Friss propagation loss model is used to study the impact of fading in the wireless medium. We further classify a neighborhood to consider two possible use cases in order to perform a scalabe as well as fine-grained analysis. Since energy efficiency can be achieved by allowing a node to operate on low duty cycles, therefore, we consider low duty cycles of 1% and 5% for an individual node operation, thus dealing with asymmetric duty cycles for nodes with different clock drifts.

5.1.1 Use Case I - Small-scale neighborhood: refers to a scenario where few mobile nodes need to discover each other for applications such as mobile sensing or proximity-based gaming. We vary the number of co-existing neighbors from 2 nodes up to a total of 20 simultaneous nodes in a neighborhood.

5.1.2 Use Case II - Relatively large neighborhood: can be considered for a package shipment application where large number of energy constrained sensors on package need to discover each other autonomously for tracking purposes. We analyze the availability



(a) Average discovered neighbors **(b) Latency vs discovery** **(c) Energy vs discovery**

Figure 1: Simulation results

of each mechanism in analysis by varying number of co-existing nodes from 25 to 100 in a neighborhood.

For each individual node, the time T is divided into multiple uniform size slots where it is allowed to send beacons of 100 bytes at the beginning or end of a slot. Similarly, it can listen during a slot duration where the time 10 ms is the slot size. We consider an asynchronous discovery with the possibility of a clock drift between nodes as they are unaware of the time lag between each others active slots. Each node follows a schedule to become active in a slot using the respective parameters defined by each mechanism in Table 2 for 1% and 5% duty cycle. We implement (i) two prime-based DISCO, (ii) single prime based U-Connect, (iii) fixed-slot based Hello, and (iv) Balanced Nihao, (v) dynamic slot based Searchlight and (vi) Blinddate and (vii) Stochastic Birthday mechanism.

It is to recall that each of the above mentioned mechanisms assumes for two or more nodes be active simultaneously, thus finding an overlapping active slot between them in order to ensure a successful discovery. However, activating multiple nodes at the same time can lead to collisions, thus resulting in a discovery failure. We cater the issue by implementing CSMA/CA based back-off approach where a node finding to the medium as busy before transmitting a beacon choose a wait time randomly between its initial transmission time and the slot size (10 ms).

Moreover, finding an optimal schedule to activate nodes in order to find overlapping slots between nodes is an open problem [18]. Since each of the deterministic mechanisms discussed above ensures a successful discovery if the neighboring nodes is active during L_n slots. Therefore, we align the schedule of nodes in a neighborhood in such a way that there are at least L_n common slots between any two nodes.

The evaluation metrics are (i) Energy vs discovery relation (δ_N) from Equation 2to find the energy consumption of the node using each neighbor discovery mechanism for the set of discovered neighbors, (ii) Latency (L_n) vs discovery relation in Equation 1 to find the latency incurred by the node in applying the schedule of each mechanism to find the fraction of discovered neighbors. Similarly, we find (iii) for each use case, the average number of discovered neighbors D_N among N nodes in each others communication range. Since each mechanism incurs different worst case discovery bounds, each should be compared for the evaluation metrics within common time bounds. Our evaluation is based on the analysis of each mechanism till the maximum possible worst case discovery which is $T = 2 \times 1591$ slots for Disco. Thus, the schedule of mechanisms with $L_n < 2 \times 1591$ are restarted till the end of the Disco schedule. This allows the last active node in a neighborhood using Disco to complete its schedule.

5.2 Simulation Results

We performed simulations on both, nodes running on 1% and 5% duty cycles. However for brevity, we present below the results from simulations using 5% duty cycle to better analyze the behavior when nodes are more prone to collisions. The results are obtained using 10 simulation runs where the average is shown with 95% confidence intervals.

5.2.1 *Average discovered neighbors.*

We consider an uneven discovery between nodes where all the neighbors are not necessarily discovered by the node. We investigate such behavior by finding the average neighbors discovered in different neighborhood sizes. Figure 1a shows the average neighbors discovered for nodes operating on 5% duty cycle in a neighborhood. It is seen that the average number of neighbors are different for different mechanisms. Surprisingly, for both the small scale neighborhood as well as the large-scale neighborhood case, Blinddate yields the least number of neighbors. It discovered less than 50% neighbors for most neighborhood sizes, where for the large scale of 100 neighbors, it discovered around 20. One possible reason for Blinddate poor performance is due to the fact that it uses less active (overlapping) slots for discovery which reduce its chances to find other nodes. Moreover, its low worst case latency could also lead to failure in discoveries as its schedule wraps up earlier than other schemes, thus missing discovery opportunities.

It is also observed that the stochastic mechanism discovered more neighbors with a substantial difference with deterministic mechanisms. However, still it is clearly shown that the discovered neighbors can sometimes be less than anticipated, even for the well performing mechanisms. For instance, Birthday for the case of 100 nodes results in discovering around 85 nodes. Similar trend is noticed in all schemes, thus validates our previously discussed claim that the discovery failure can occur due to collisions or no due to non-existence of common active slots which impedes discovery opportunity between nodes.

5.2.2 Latency vs Discovery. We analyze the worst case latency achieved by the nodes using each mechanism for the fraction of discovered neighbors as it is considered as a key evaluation metric. Figure 1b shows the latency vs discovery relation for different class of mechanisms running on nodes with 5% duty cycle. A lower value of latency vs discovery reflects a better performance in terms of latency. It is seen that the stochastic mechanism, Birthday resulted in the best performance i.e. quick discovery of fraction of neighbors in both, the small-scale and large-scale neighborhood size. Thus, It scales better and is relatively stable with respect to increase in the number of neighbors and is therefore, unaffected by the increase or decrease in the neighborhood size. Birthday is followed by other schemes with relatively similar performance with the exception of Blinddate, which resulted in the worst performance. The high latency vs discovery by Blinddate makes sense since with its least worst case latency L_n, it repeats its schedule till the finishing time of Disco's schedule. However, despite such a short schedule, it fails to discover substantial fraction of node during our simulation duration.

We see slight increase in latency discovery relation with the increase in neighborhood size. It is because due to the increase in number of nodes, collisions occur when multiple nodes transmit at the same time, thus leading to failure in transmitting node beacons at the receiver node. The beacon sending nodes fail to discover since their beacons collide as well as neither of the beacon is successfully recovered at the receiver node. This also results in large number of missing discovery opportunities between the nodes failing to transmit beacons in their respective slots. Thus, overall latency analysis is suggesting that probabilistic approaches quickly discover neighbors compared to deterministic approaches.

5.2.3 Energy vs Discovery. We analyze the energy vs discovery relation for each use case and for asynchronous node duty cycle implementing different neighbor discovery mechanisms. A low energy vs discovery relation infers better performance. Figure 1c shows the comparison results of such a relation using each neighbor discovery mechanism for nodes operating on 5% duty cycles. For the small scale neighborhood use case, we vary the set of co-located nodes from 2 to 20 with an increment of one node while the large scale neighborhood of size 25, 50 75 and 100 nodes are considered.

The static-slot based Nihao performs relatively better in terms of energy. Nodes consumes less energy to discover the fraction of neighbors using Nihao, however, increasing the neighborhood size impacts the results. The primed Disco resulted in a poor performance as it is seen as the outlier among them where a node becomes active to listen for large fraction of time. The reason Nihao performs better is due to its reduced number of idle listen slots, thus, resulting in spending less energy compared to other schemes while on the other hand the prime number dependency of Disco requires relatively more active slots compared to other schemes. The better performance of Nihao can also be justified by the "Talk more listen less" principle with more beacons than listen periods where the node energy consumption during the transmission of a beacon is less than listen periods i.e. an average 2.06 ms is observed compared to 10 ms, thus validating that nodes can benefit from the less listen periods in Nihao. Thus, the overall energy vs discovery analysis using simulations suggest that Nihao is an efficient asymmetric neighbor discovery scheme for nodes running on low duty cycles.

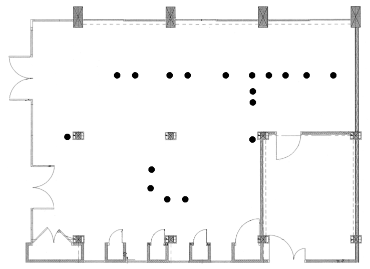


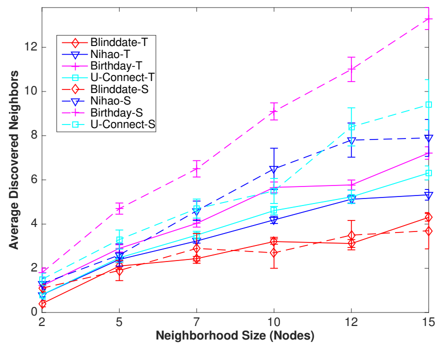
Figure 2: Layout of our test-bed nodes in FIT IoT Lab

6 TEST-BED BASED EVALUATION

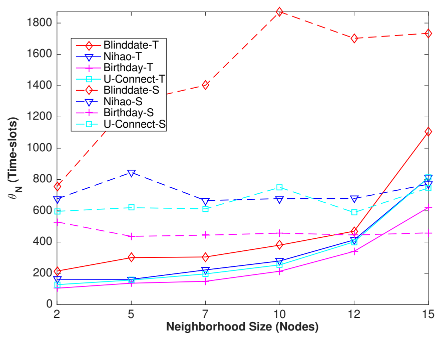
6.1 Test-bed Scenario

Test-bed experiments are performed by implementing each of the discussed neighbor discovery schemes on the large scale FIT (Future Internet of the Things) IoT-LAB platform [4]. It is a publically available platform for testing small wireless sensor devices, installed at Inria research centers in France. We perform experimentation on up to 15 ARM Cortex M3 (STM32) nodes with 72 Mhz processing and 64 kB RAM with FreeRTOS for development. The layout of the nodes placement is shown in Figure 2. Nodes in the communication range of each other are considered in a neighborhood where a clique based network is formed. We assume relatively stable neighborhood where topological changes does not occur during our analysis. We evaluated each mechanism by considering up to 15 co-existing nodes in a neighborhood. We consider low duty cycles of 1% and 5% for an individual node where results are shown for 5% duty cycles to study the impact of collisions as nodes are more prone to collisions when running on a higher duty cycle. Similar to the simulation scenario, the time T is divided into multiple uniform size slots where it is allowed to send beacons of 100 bytes at the beginning or end of a slot. Similarly, it can listen during a slot duration where we consider 20 ms as the duration of a single slot. Our test-bed takes into account an asynchronous discovery with the possibility of a clock drift between nodes as they are unaware of the time lag between each others active slots. We implement (i) dynamic slot based Blinddate, (ii) fixed slot based Balanced Nihao, the (iii) Stochastic Birthday and (iv) prime based U-Connect scheme as we the best performing neighbor discovery schemes in each category. Experiments are repeated 10 times and results are shown for 95% confidence intervals.

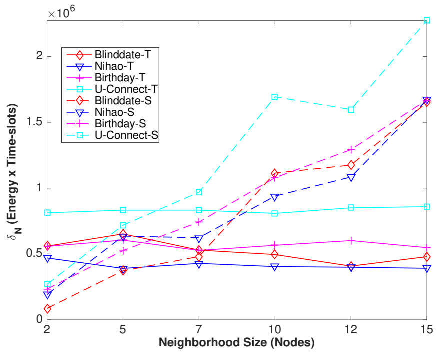
Since, multiple nodes active at the same time can result in discovery failure due to collisions. We implement a CSMA based approach where a node finding to the medium as busy does not



(a) Average discovered neighbors



(b) Latency vs discovery



(c) Energy vs discovery

Figure 3: Test-bed results

transmit its beacon. Moreover, our schedule ensure the existence of at least L_n common slots between any two nodes allowing two or more node to find discovery opportunity during the worst case latency bound defined by each scheme. Similar to the simulation study we use the evaluation metrics as (i) the average number of discovered neighbors D_N among N nodes in each others communication range (ii) Latency (L_n) vs discovery relation and (iii) Energy vs discovery relation (δ_N). The evaluation is based on the analysis of each mechanism till the maximum possible worst case discovery which is 1591 slots for Disco where the schedule of mechanisms with $L_n < 1591$ are restarted till the end of the Disco schedule.

6.2 Test-bed Results

The test-bed evaluation is performed on nodes running on both 1% and 5% duty cycles. Thought similar to the simulation results, we present here results for nodes running on 5% duty cycles as it is more prone to collisions.

6.2.1 Average discovered neighbors. Figure 3a shows the average neighbors (number of nodes) discovered for nodes operating on 5% duty cycle in a neighborhood of 2 to 15 nodes. The comparison results from both testbed and simulations (dotted line) are shown together, where for each compared scheme the suffix -'T' and 'S' denotes testbed and simulation respectively. We observe that the discovered neighboring nodes are less than the number of nodes in the neighborhood, irrespective of the neighborhood sizes. For a neighborhood of 15 nodes, around $6-8$ can be mutually discovered in a real scenario which is merely 60% of the deployed nodes. Blinddate discovered the least number of neighbors, for instance, in a neighborhood of 6 nodes, it discovered only 2 nodes, thus resulting in the poor performance. Nihao is slightly better with its more number of beacons increases the chance of discovery and thereby increasing the likeliness that a beacon is listened by a neighboring node. Both are outperformed by Birthday discovering up to 7 neighbors for the case of 15 neighboring nodes. We can infer from the overall average discovery analysis that despite low duty cycles, it is possible to discover around 50% nodes on average in a real scenario using existing neighbor discovery schemes.

6.2.2 Latency vs Discovery. In our test-bed analysis, we also study the worst case latency vs fraction of discovered neighbors relation for nodes implementing each scheme. Figure 3b shows the comparison results for such analysis for nodes with 5% duty cycle. As discussed before, a lower value of latency vs discovery reflects a better performance in terms of latency. Birthday seems to outperform other approaches. At the same time, it yields relatively stable results for increasing number of nodes in a neighborhood. We observe that Blinddate results in the poor performance among the compared schemes. Similar to the simulation results, we observe a high latency vs discovery while using Blinddate due to the repetition of its schedule in our experiments. It is to note that despite providing Blinddate sufficient time for discovery, it is unable to discover large number of nodes with a highest steep peak observed for the case of 15 nodes neighborhood. Collisions with increase in number of nodes degrades the latency vs discovery performance, where such trend is commonly seen for all schemes. Therefore, overall analysis suggests that fraction of nodes in a neighborhood fail to discover each other proportional to the neighborhood size.

6.2.3 Energy vs Discovery. Figure 3c shows the comparison results of the energy vs discovery relation for nodes implementing each neighbor discovery mechanism for nodes operating on 5% duty cycles. The relation δ is measured in energy $\times$ time-slots where we vary the set of co-located nodes in a neighborhood from 2 to 15. Similar to the simulation results, Nihao achieves relatively better performance in terms of energy vs discovery relation with a stable behavior with increase in number of co-located neighbors. Thus, it allows nodes to discover large fraction of neighbors in an energy efficient way. On the other hand, the remaining schemes achieves similar performance in terms of energy with the increase in neighborhood size. The better performance of Nihao is also due to its low listen slots, and therefore, low energy consumption compared to other schemes. U-Connect does not achieve better performance in terms of energy, validating the fact observed in simulations where prime-based schemes result in worst performance. Thus, the overall energy vs discovery analysis using real test-best based evaluation suggests Nihao as an efficient asymmetric neighbor discovery scheme for nodes running on low duty cycles.

7 SUMMARY OF FINDINGS

7.1 Latency/Energy vs Discovery trade-off

The above analysis suggest that, besides the natural trade-off between latency and energy, i.e. a neighbor discovery scheme consuming less energy not necessarily result in low latency. Another important findings reveal that there is a trade-off between the fraction of discovered neighbors by a scheme vs energy as well as latency. A low energy consuming mechanism not necessarily discover more neighbors, thus, resulting in uneven discovery among nodes. Such a behavior is noticed particularly for Blinddate where the schedule of Blinddate performed better on energy, however it yielded the worst performance with respect to average discovered neighbors. Thus, there is need for a schedule which increases the number of discovered neighbors with low energy consumption and latency.

7.2 Scalability issue: Collisions affect discovery

Our study also show that neighbor discovery mechanisms does not scale well, i.e. increasing number of co-located nodes in the same collision domain decreases the fraction of neighbors discovered. None of the existing schemes is suitable for large neighborhoods due to the large number of messages exchanged during the discovery process. There is a need for reduction in the number of beacons as well as idle listening slots where few nodes with few messages exchanged can improve discovery with minimum collisions.

8 CONCLUSIONS AND FUTURE DIRECTIONS

Energy efficient neighbor discovery mechanisms allow nodes in each others communication range (clique) to discover their neighbors while running on low duty cycles. For two or more nodes to discover each other, different neighbor discovery schemes define a schedule for nodes to become active to send or listen to beacons in a time slotted fashion where overlapping slot(s) with other nearby active nodes enable discovery. However such schemes do not scale well, beacons from multiple nodes can collide, thus leading to discover failure. In this paper, we defined a novel relation between the latency, energy efficiency and the amount of neighbors discovered. We evaluated seven such mechanisms through extensive simulations for up to 100 nodes operating at 1% and 5% duty cycles. We also implemented four of the such schemes on a real IoT based test-bed for up to 15 neighboring nodes. Our findings confirmed the impact of discovery failure on a neighbor discovery scheme performance where low latency energy efficient schemes such as Blinddate failed to discover large fraction of nodes.

Moreover, we believe there is room for improvement, particularly, catering collisions where neighbor discovery schemes end up finding less amount of nodes than expected. Moreover, the latency and energy vs discovery trade off requires careful consideration for novel neighbor discovery schemes. One important need is to suppress large number of beacons exchanged during a discovery process. Our future work also include the extension of this study on indirect schemes towards a Flock discovery with the possibility of nodes leaving or joining the neighborhood. The goal is to confirm the intuition that collaboration based neighbor discovery is indeed beneficial.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions. The work is partially funded by the French DGA CLOTHO project (2016-2018).

REFERENCES

[1] Sony ps vita, https://www.playstation.com/en-us/explore/psvita/.
[2] Wei Sun, Zheng Yang, Xinglin Zhang, and Yunhao Liu. Energy-efficient neighbor discovery in mobile ad hoc and wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials*, 16(3):1448–1459, 2014.
[3] Riccardo Pozza, Michele Nati, Stylianos Georgoulas, Klaus Moessner, and Alexander Gluhak. Neighbor discovery for opportunistic networking in internet of things scenarios: A survey. *IEEE Access*, 3:1101–1131, 2015.
[4] Cedric Adjih, Emmanuel Baccelli, Eric Fleury, Gaetan Harter, Nathalie Mitton, Thomas Noel, Roger Pissard-Gibollet, Frederic Saint-Marcel, Guillaume Schreiner, Julien Vandaele, et al. Fit iot-lab: A large scale open experimental iot testbed. In *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on*, pages 459–464. IEEE, 2015.
[5] Lin Chen and Kaigui Bian. Neighbor discovery in mobile sensing applications: A comprehensive survey. *Ad Hoc Networks*, 48:38–52, 2016.
[6] Ricardo C Carrano, Diego Passos, Luiz CS Magalhães, and Célio VN Albuquerque. A comprehensive analysis on the use of schedule-based asynchronous duty cycling in wireless sensor networks. *Ad Hoc Networks*, 16:142–164, 2014.
[7] Desheng Zhang, Tian He, Yunhuai Liu, Yu Gu, Fan Ye, Raghu K Ganti, and Hui Lei. Acc: generic on-demand accelerations for neighbor discovery in mobile applications. In *Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems*, pages 169–182. ACM, 2012.
[8] Desheng Zhang, Tian He, Fan Ye, Raghu Ganti, and Hui Lei. Neighbor discovery and rendezvous maintenance with extended quorum systems for mobile applications. *IEEE Transactions on Mobile Computing*, 2016.
[9] Liangyin Chen, Yuanchao Shu, Yu Gu, Shuo Guo, Tian He, Fan Zhang, and Jiming Chen. Group-based neighbor discovery in low-duty-cycle mobile sensor networks. *IEEE Transactions on Mobile Computing*, 15(8):1996–2009, 2016.
[10] Shouwen Lai, Binoy Ravindran, and Hyeonjoong Cho. Heterogenous quorum-based wake-up scheduling in wireless sensor networks. *IEEE Transactions on Computers*, 59(11):1562–1575, 2010.
[11] Prabal Dutta and David Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 71–84. ACM, 2008.
[12] Arvind Kandhalu, Karthik Lakshmanan, and Ragunathan Raj Rajkumar. U-connect: a low-latency energy-efficient asynchronous neighbor discovery protocol. In *Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks*, pages 350–361. ACM, 2010.
[13] Keyu Wang, Xufei Mao, and Yunhao Liu. Blinddate: A neighbor discovery protocol. *IEEE Transactions on Parallel and Distributed Systems*, 26(4):949–959, 2015.
[14] Mehedi Bakht, Matt Trower, and Robin Hilary Kravets. Searchlight: Won't you be my neighbor? In *Proceedings of the 18th annual international conference on Mobile computing and networking*, pages 185–196. ACM, 2012.
[15] Wei Sun, Zheng Yang, Keyu Wang, and Yunhao Liu. Hello: A generic flexible protocol for neighbor discovery. In *INFOCOM, 2014 Proceedings IEEE*, pages 540–548. IEEE, 2014.
[16] Ying Qiu, Shining Li, Xiangsen Xu, and Zhigang Li. Talk more listen less: Energy-efficient neighbor discovery in wireless sensor networks. In *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*, pages 1–9. IEEE, 2016.
[17] Michael J McGlynn and Steven A Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, pages 137–145. ACM, 2001.
[18] Lin Chen, Kaigui Bian, and Meng Zheng. Never live without neighbors: From single-to multi-channel neighbor discovery for mobile sensing applications. *IEEE/ACM Transactions on Networking*, 24(5):3148–3161, 2016.
[19] Tong Meng, Fan Wu, Aijing Li, Guihai Chen, and Nitin H Vaidya. On robust neighbor discovery in mobile wireless networks. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, page 38. ACM, 2015.
[20] Tingpei Huang, Haiming Chen, Yuqing Zhang, and Li Cui. Easind: Effective neighbor discovery algorithms for asynchronous and asymmetric-duty-cycle multi-channel mobile wsns. *Wireless Personal Communications*, 84(4):3031–3055, 2015.
[21] Tong Meng, Fan Wu, and Guihai Chen. On designing neighbor discovery protocols: A code-based approach. In *INFOCOM, 2014 Proceedings IEEE*, pages 1689–1697. IEEE, 2014.