Energy-Efficient Neighbor Discovery for the Internet of Things

Zhong Shen, *Member, IEEE*, Hai Jiang, *Senior Member, IEEE*, Qingkuan Dong, and Baocang Wang

***Abstract*—Internet-of-Things (IoT) networks are usually distributed in nature. And due to possible mobility of IoT devices, it is common and critical for each IoT device to keep discovering who are in its neighborhood, referred to as neighbor discovery. Due to the limited battery capacity of IoT devices, it is challenging to design a neighbor discovery protocol (NDP) that can achieve both low duty cycle and low discovery latency. In this paper, we build a model called Circle to characterize the process of neighbor discovery in IoT networks. Then, we give a necessary and sufficient condition for neighbor discovery and theoretically prove its correctness. This is the first time in the research community that a necessary and sufficient condition is given for neighbor discovery. According to the necessary and sufficient condition, we analytically derive a lower bound of the worst-case discovery latency and demonstrate when the lower bound can be achieved. The analytical model is generic as it can be used to analyze existing NDPs. Based on the Circle model and the analysis, we propose an NDP, which is also called Circle. We compare Circle with state-of-the-art NDPs in a real testbed, and experimental results show that Circle is superior to the existing state-of-the-art NDPs.**

***Index Terms*—Internet of Things, neighbor discovery protocols, duty cycle.**

I. Introduction

The Internet of Things (IoT) has been envisioned to seamlessly integrate heterogeneous IoT devices to the Internet via various wireless technologies (e.g., ZigBee, Wi-Fi, Bluetooth, etc) [1], [2]. Cisco predicts that there will be 50 billion IoT connected devices by 2020. IoT networks have been widely applied to smart cities/environments for intelligent detection, monitoring, coordination, and management [2]–[5].

IoT networks are *distributed* in nature. And due to possible mobility of the IoT devices, it is common and critical for each IoT device to keep discovering who are in its neighborhood [4], [6]–[8], referred to as neighbor discovery. Since IoT devices generally have limited battery capacities, it is not good for them to turn on their radios all the time to perform neighbor discovery. Instead, they can turn on radios periodically for a while for data communications, and then go to sleep to save energy. In other words, they work in duty-cycle mode. For an IoT device (called *a node* in the sequel), its duty cycle is the percentage of time that the device's radio is turned on. Low duty cycle prolongs lifetime of nodes, but may also increase discovery latency. Therefore, it is important to design an energy-efficient neighbor discovery protocol (NDP) that can achieve both low duty cycle and low discovery latency.

Several energy-efficient NDPs [9]–[18] have been proposed recently. Those NDPs can be classified to two categories: probabilistic and deterministic. Birthday protocol [9] is a representative of probabilistic NDPs, in which a node may select to transmit, receive, or sleep (each with a particular probability). Birthday has a low average discovery latency, but its worst-case latency is not bounded. To ensure a worst-case latency, deterministic NDPs such as Disco [11], U-Connect [14], Searchlight [12], Hello [18], and Nihao [15] have been proposed. In this paper, we focus on deterministic NDPs.

Existing deterministic NDPs are designed based on satisfying a condition that is *sufficient* to guarantee neighbor discovery. For example, Disco is designed based on the Chinese Remainder Theorem and uses primes. However, it is not clear whether the conditions in those protocols are necessary or not. In other words, can we have a weaker condition that can guarantee neighbor discovery? If yes, then it is very likely that better energy efficiency can be achieved due to the weaker condition used.

To fill this research gap, in this paper, we investigate *necessary and sufficient condition for neighbor discovery*. A necessary and sufficient condition will enable us to better understand the problem and then design more efficient NDPs. Further, a necessary and sufficient condition can provide guidelines for real applications. For example, in Find Me Profile [19], a typical application of Bluetooth Low Energy (BLE) [20]–[22], two Bluetooth devices try to find each other with one being advertiser and the other being scanner. The scanner listens to the channel for a duration of ω with a scanning interval a, while the advertiser sends advertising packets for a duration of τ with a advertising interval b. Although recommended values for these parameters are given for the low duty cycle discovery mode (e.g., b take values from 1s to 2.5s, ω =11.25ms, and a can be 1.28s or 2.56s), how to choose values of these parameters to ensure neighbor discovery is unknown. This practical problem can be solved once a necessary and sufficient condition for neighbor discovery is provided (see section IV-D for detail).

In this paper, we first build a model to characterize the process of neighbor discovery. Then, we give a necessary and sufficient condition for neighbor discovery and theoretically

Manuscript received May 1, 2019; revised September 6, 2019 and September 26, 2019; accepted October 23, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61671348, 61572390, and U1736111, by the National Key R&D Program of China under Grant No. 2017YFB0802000, the National Cryptography Development Fund under Grant No. MMJJ20180111. (Corresponding author: Zhong Shen.)

Z. Shen, Q. Dong, and B. Wang are with State Key Laboratory of Integrated Service Network, P.O. Box 108, Xidian University, Xi'an, Shaanxi 710071, P. R. China (e-mail: zhshen@mail.xidian.edu.cn, qkdong@mail.xidian.edu.cn, bcwang79@aliyun.com). B. Wang is also affiliated with School of Information Engineering, Xuchang University, Xuchang 461000, P. R. China. H. Jiang is with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB T6G 1H9, Canada (e-mail: hai1@ualberta.ca).

TABLE I
IMPORTANT SYMBOLS USED

Symbols	Meaning
$\gcd(a, b)$	the greatest common divisor of a and b
$a \bmod b$	a modulo b
$a \equiv b (\bmod m)$	a congruent to b modulo m
ω	time duration for receiving beacons
τ	time duration for sending a beacon
ϕ	the initial time offset between two nodes

prove its correctness. According to the necessary and sufficient condition, we derive a lower bound for the worst-case discovery latency and demonstrate when the bound can be achieved. Based on this analysis, we design an energy-efficient NDP called Circle. We compare Circle with state-of-the-art NDPs in a real testbed, and experimental results show that Circle outperforms existing NDPs.

In summary, our contributions are as follows:

1). Necessary and sufficient condition for neighbor discovery: We are the first to provide a necessary and sufficient condition for neighbor discovery.
2). Generic analytical model: Our worst-case discovery latency analysis is *generic*, i.e., it can be used to characterize our and other existing NDPs.
3). New NDP: Based on the necessary and sufficient condition for neighbor discovery and the analytical results, our proposed NDP is superior to existing NDPs in terms of discovery latency, as demonstrated by experiments using a real testbed.

The remainder of this paper is organized as follows. Existing relevant works in the literature are discussed in Section II. The problem statement is given in Section III. Section IV presents our model and theoretical analysis. The proposed NDP is detailed in Section V. Evaluation of the proposed NDP and comparison with existing works are provided in Section VI. Conclusion remarks are presented in Section VII. Table I defines important symbols that will be used in the sequel.

II. RELATED WORKS

The problem of neighbor discovery has received much attention recently. Although neighbor discovery may work in a synchronized manner, e.g., all the nodes get clock synchronization via GPS, it is expensive and energy intensive. Therefore, asynchronous neighbor discovery is of interest. Existing research efforts investigate neighbor discovery from different perspectives including initial neighbor discovery [9]–[18], continuous neighbor discovery [23], and collaborative neighbor discovery [24]–[26]. Our focus in this paper is on initial neighbor discovery.

To simplify design, most existing NDPs [9]–[18] adopt time-slotted model in which time is divided into fixed-length slots. The time slots are indexed from 0, and a node selects some slots as working slots (or active slots), while the other time slots are non-working slots (or sleeping slots). During an active slot, a node will turn on its radio and transmit/receive beacons.

Most NDPs have their own cycle lengths, which means that the time is partitioned into fixed-duration cycles. Each cycle consists of a number of slots: some are working slots, while others are non-working slots. And all the cycles have the same configuration of working slots and non-working slots. The percentage of active slots in a cycle is called duty cycle. For instance, if a node works at slots whose indices are multiples of 3, then its cycle length is 3 slots, and its duty cycle is $1/3$. Moreover, a hyper-cycle may have several basic periods, and each period has its own configuration of working slots and non-working slots. For example, a node works at slots whose indices are multiples of 3 or 5. In this example, the hyper-cycle has three periods, and each period has 5 slots. The hyper-cycle can be represented by a 3×5 matrix as

$$\begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

in which values 1 and 0 mean the corresponding time slot is a working slot and non-working slot, respectively. In other words, in each hyper-cycle, the node works at slot $0, 3, 5, 6, 9, 10,$ and 12, and its duty cycle is 7/15=46.7%.

In the literature, there are two categories of NDPs: probabilistic NDPs and deterministic NDPs. Birthday [9] is a representative of probabilistic NDPs. The state of a node at each slot will be transmitting, listening, or sleeping with probabilities. Birthday has low average discovery latency, but it suffers from unpredictable large latency due to its probabilistic nature. Conversely, deterministic NDPs have bounded worst-case discovery latency.

Based on the concept of quorum, Quorum protocol [10] has a hyper-cycle of n^2 slots arranged as an $n \times n$ matrix. From the matrix, a node randomly selects one column and one row of entries as active slots, which ensures that two nodes with the same cycle length must have at least two intersecting slots. Given a delay requirement, the problem of neighbor discovery is formulated as a block design problem in [13], and the problem is solved such that the minimum energy consumption is achieved. However, it is shown that the schemes proposed in [10], [13] can be applied to the scenario with *symmetric duty cycle* (i.e., when duty cycles of any two nodes are the same), but do not work in the scenario with *asymmetric duty cycle* (i.e., when the nodes have different duty cycles).

Since nodes may have the same or different duty cycles, neighbor discovery with both symmetric and asymmetric duty cycles should be supported. To support both symmetric and asymmetric duty cycles, existing deterministic NDPs are designed mainly based on coprime (e.g., Disco [11]), quorum (e.g., Searchlight [12] and Nihao [15]), or hybrid of both (e.g., U-Connect [14] and Hello [18]). These techniques ensure that the active slot sets of any two nodes have overlap, thus leading to discovery provided that each node is in communication range of the other node.

As a representative of coprime, Disco [11] selects two primes p_1 and p_2 for each node, and the node works at the time slots whose indices are multiples of p_1 or p_2. Based on the Chinese Remainder Theorem, Disco guarantees at least one

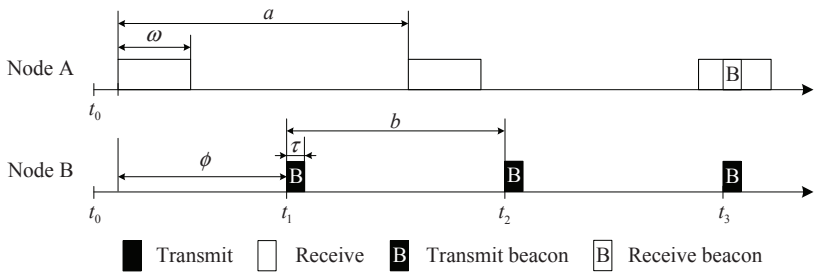


Fig. 1. A simple example of discovery procedure.

overlapping active slot between any two nodes, which means that neighbor discovery is guaranteed.

Instead of using coprime, Searchlight [12] adopts quorum technique. Each node is assigned a value t from set $\{c, 2c, 2^2c, 2^3c, ...\}$ (c being an integer constant), and has its hyper-cycle represented by a $\lfloor \frac{t}{2} \rfloor \times t$ matrix (here $\lfloor \cdot \rfloor$ means floor function). In the hyper-cycle matrix, the first slot of each row is an active slot (called the anchor slot of the row). And in row i ($i = 1, 2, ..., \lfloor \frac{t}{2} \rfloor$), the $(i+1)$th slot is also active. By this setting, Searchlight guarantees neighbor discovery.

U-Connect [14] and Hello [18] take advantage of both coprime and quorum. In general, asymmetric neighbor discovery depends on the coprime whereas symmetric neighbor discovery relies on quorum. U-Connect uses a single prime p and its hyper-cycle is a $p \times p$ matrix. The first slot of each row is active. In the first row, the first $\frac{p+1}{2}$ slots are also active slots. Hello provides a generic framework in which the hyper-cycle is a $n \times c$ matrix where c is a prime, and n can be any number.[1] Hello works at the first $\lceil \frac{c}{2} \rceil$ slots of the first row and the first slots of other rows. It is shown that Quorum, Disco, U-Connect, and Searchlight are special cases of Hello [18].

All above slotted NDPs assume that overlapping active slots result in neighbor discovery. Due to limitation such as half-duplex transceiver and collisions, the overlapping duration may not be sufficient for bidirectional discovery. Taking this limitation into account, Nihao suggests "talk more listen less". The hyper-cycle of Nihao is an $n \times m$ matrix, where all slots in the first row are active for listening, and a beacon is sent at the beginning of the first slot of each row.

Although existing state-of-the-art NDPs are designed based on different techniques, none of them provides a necessary and sufficient condition for neighbor discovery. In this paper, we construct a model to characterize the process of neighbor discovery. Then, we provide a necessary and sufficient condition for neighbor discovery and theoretically prove its correctness. This is the first time in the research community that a necessary and sufficient condition is given for neighbor discovery.

III. Problem Statement

Here, we describe the neighbor discovery problem. Neighbor discovery is the process that two nodes, say node A and node B, receive beacons from each other. This process can be divided into two separate discoveries, i.e., node A discovers node B, and node B discovers node A. For presentation simplicity, in the sequel, we only discuss the case that node A discovers node B.

Assume at time instant t_0, node A and node B enter communication range of each other. Fig. 1 shows the snapshot that node A discovers node B. Suppose node A wakes up every a time units.[2] So the cycle length of node A is a. During each wake-up, node A turns on its radio, and listens to the channel for ω time units. After that, it turns off its radio and goes to sleep. Node B sends a beacon every b time units (so the cycle length of node B is b), and transmitting a beacon costs τ time units.

The discovery latency for node A to find node B is the delay from the time instant that both nodes go into communication range of each other to the time instant that node A first receives a beacon from node B. Suppose after time instant t_0, the beacons sent by node B are sequentially indexed from 1, and the moment at which the ith beacon is sent by node B is denoted by t_i. In Fig. 1, the third beacon sent by node B is received by node A, and therefore, the discovery latency is $t_3 - t_0 + \tau$.

Three fundamental questions naturally arise.

1). Is there a necessary and sufficient condition for node A to discover node B in finite time?
2). What is the lower bound for the worst-case discovery latency?
3). Can we design a protocol that achieves the lower bound for the worst-case discovery latency?

In Section IV and Section V, we will address these three fundamental questions.

IV. Model and Theoretical Analysis

A. Circle Model

We start by analyzing the time offset of node B's beacons to node A. The time offset of node B's ith beacon to node A is defined as the time difference between t_i and node A's wake-up time instant right before t_i. The time offset of node B's first beacon is called initial time offset, denoted by ϕ, as shown in Fig. 1. Note that node A's wake-up time instant that is right before t_1 could be earlier than t_0. It can be seen that $a > \phi \geq 0$.

Without loss of generality, given an arbitrary value of ϕ, assume there exists k such that the kth beacon from node B

[1] Such a Hello protocol is termed $Hello(c, n)$.

[2] Here, a time unit could be one second, or one millisecond, or one microsecond, etc.

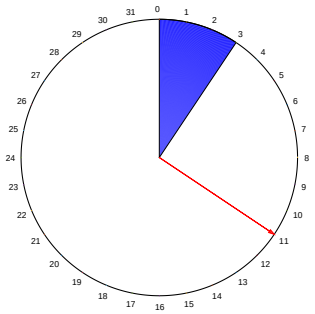


Fig. 2. An example of circle model where $a = 32$, $b = 18$, $\omega = 4$, $\tau = 1$, and $\phi = 11$. The shaded sector is the target area.

is the first beacon received by node A. For example, $k = 3$ in Fig. 1. If node A can never receive a beacon from node B, both k and t_k are infinite. The discovery latency $t_k - t_0$ can be divided into two delay components: one is from t_0 to t_1, and the other is from t_1 to t_k. It can be seen that the delay from t_0 to t_1 is deterministic and is no more than b, whereas the delay from t_1 to t_k is variable, and may even be infinite.

We restrict a, b, ω, ϕ, and τ to be integers because in real applications, the highest time resolution (e.g., millisecond or microsecond) supported by the system can be used, and therefore all parameters take integer values.

We draw a circle to characterize time offsets of node B's beacons. Like a clock face, the circle has numbers 0 to $a-1$ that are equally spaced by one time unit around the periphery of the circle with number '0' on the top, as shown in Fig. 2. There is also a "hand" indicating the time offset of the beacon of node B, so we call it beacon hand. Initially, the beacon hand points to ϕ. For the next beacon, the beacon hand clockwisely rotates b time units around the circle. The circular sector from 0 to $\omega - \tau$ represents the target area (the blue shaded sector in Fig. 2). Once the beacon hand falls into this area, then node A discovers node B. For example, in Fig. 2, $a = 32$, $b = 18$, $\omega = 4$, $\tau = 1$, and $\phi = 11$. The beacon hand initially points to 11, next moves to 29 (i.e., 11+18), which is called one hop. From 29, the next hop will be 15 that can be calculated as (29+18) mod 32. Then, the next hop will be 1, and thus discovery occurs. So, if node B has a initial time offset of 11, then it takes three hops (i.e., 54 time units) for node A to discover node B.

Based on the above circle model, neighbor discovery (i.e., node A discovers node B) can be stated as: given an arbitrary ϕ, there exists h ($h < \infty$) satisfying

$$(\phi + h \times b) \mod a \leq \omega - \tau. \tag{1}$$

Define $\hat{\omega} \triangleq \omega - \tau + 1$, and the inequality (1) becomes

$$(\phi + h \times b) \mod a < \hat{\omega}. \tag{2}$$

B. Necessary and Sufficient Condition for Neighbor Discovery and Analysis

In this subsection, we give a necessary and sufficient condition for neighbor discovery and analyze the worst-case discovery latency. Without loss of generality, we consider $a > 1$, $b > 1$, $a \geq \omega \geq \tau$.

Theorem 1: Given a, b, and $\hat{\omega}$, it follows:

(i) If and only if $\gcd(a, b) \leq \hat{\omega}$, there exists h satisfying $(\phi + h \times b) \bmod a < \hat{\omega}$ for an arbitrary integer $\phi \in \{0, 1, 2, ..., a-1\}$.

(ii) If $\gcd(a, b) \leq \hat{\omega}$, then $\max_{\phi} h_{\min}(\phi) \geq \left\lfloor \frac{a}{\hat{\omega}} \right\rfloor - 1$, where $h_{\min}(\phi) = \min\{h | (\phi + h \times b) \bmod a < \hat{\omega}\}$. Here $h_{\min}(\phi)$ is the minimum number of hops to achieve discovery for initial time offset ϕ.

(iii) If $\gcd(a, b) \leq \hat{\omega}$, then the worst-case discovery latency for node A to discover node B is at least $\left\lfloor \frac{a}{\hat{\omega}} \right\rfloor \times b$.

Proof: **Proof of (i)**:

Let $\gcd(a, b) = d$. Then a and b can be expressed as $a = a'd$ and $b = b'd$, respectively. It can be easily proven that $a' \times b \equiv 0 \bmod a$.

We have **Statement 1**: $\forall i, j$, $0 \leq i, j < a'$, if $i \times b \equiv j \times b \bmod a$, then $i = j$. The statement is proven as follows. Since $i \times b \equiv j \times b \bmod a$, it follows $i \times b' \equiv j \times b' \bmod a'$. Moreover, since a' and b' are coprime, we have $i \equiv j \bmod a'$. Noting that $0 \leq i, j < a'$, we have $i = j$.

Remark for Statement 1: Statement 1 means that the integers $0, 1 \times b, 2 \times b, \cdots, (a'-1) \times b$ are pairwise incongruent modulo a. Define $\mathbb{D} \triangleq \{0, d, 2d, \cdots, (a'-1)d\}$. Since a and b are both multiples of d, $(h \times b \bmod a)$ ($h = 0, 1, \cdots, a'-1$) is a multiple of d. Thus, for $h = 0, 1, \cdots, a'-1$, it can be seen that $h \times b \bmod a$ must be equal to a unique element in $\mathbb{D}$.

We first prove if $\gcd(a, b) \leq \hat{\omega}$, there exists h satisfying $(\phi + h \times b) \bmod a < \hat{\omega}$. If $\gcd(a, b) \leq \hat{\omega}$, for $\phi \in \{0, 1, 2, ..., a-1\}$, let $\phi = q \times d + r$, $r \in \{0, 1, 2, ..., d-1\}$.

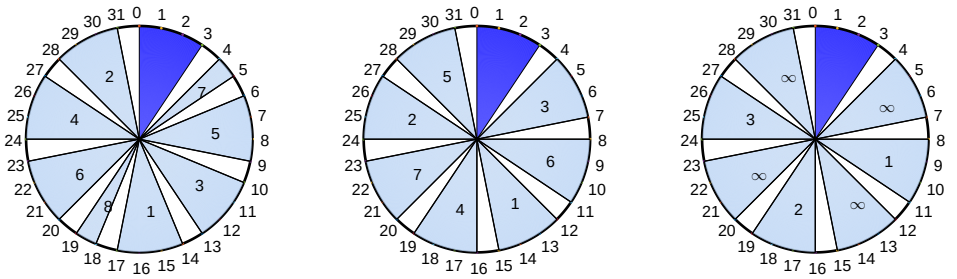
- If $q = 0$, then we have $(\phi + h \times b) \bmod a < \hat{\omega}$ when $h = 0$.
- Consider $q > 0$. Since $a - 1 \geq \phi \geq 0$, it follows $a' - 1 \geq q \geq 1$. Moreover, since the integers $0, 1 \times b, 2 \times b, \cdots, (a'-1) \times b$ are pairwise incongruent modulo a, there exists $h \in \{0, 1, 2, ..., a'-1\}$ such that $h \times b \bmod a = (a' - q) \times d$ (based on the Remark for Statement 1). Then, we have

$$\begin{aligned} & (\phi + h \times b) \bmod a \\ = \; & [q \times d + r + (a' - q) \times d] \bmod a \\ = \; & (a' \times d + r) \bmod a \\ = \; & r \bmod a \\ \leq \; & d - 1 \\ < \; & \hat{\omega}. \end{aligned}$$

We then prove when for arbitrary $\phi \in \{0, 1, 2, ..., a-1\}$, there exists h satisfying $(\phi + h \times b) \bmod a < \hat{\omega}$, then we have $\gcd(a, b) \leq \hat{\omega}$. Consider $\phi = \phi_0 \triangleq d - 1$. Suppose there exists h (denoted as h_0) such that $(\phi_0 + h_0 \times b) \bmod a < \hat{\omega}$. Since $(h_0 \times b \bmod a) \in \mathbb{D}$, it follows $(\phi_0 + h_0 \times b) \bmod a \in \{d - 1, 2d - 1, \cdots, a'd - 1\}$. Therefore, $(\phi_0 + h_0 \times b) \bmod a \geq d - 1$. Since $(\phi_0 + h_0 \times b) \bmod a < \hat{\omega}$, it follows $d - 1 < \hat{\omega}$, and thus, $\gcd(a, b) \leq \hat{\omega}$.

Proof of (ii):

Then, for initial time offset ϕ, $h_{\min}(\phi) = \min\{h | (\phi + h \times b) \bmod a < \hat{\omega}\}$, or briefly, we say it takes $h_{\min}(\phi)$ hops for ϕ.



(a) $a = 32\ b = 18\ \omega = 4\ \tau = 1$ (b) $a = 32\ b = 20\ \omega = 4\ \tau = 1$ (c) $a = 32\ b = 24\ \omega = 4\ \tau = 1$

Fig. 3. Examples of circle model, in which each sector contains one or more integers that represent initial time offset values, and the number inside the sector is the number of hops for all those time offsets to achieve neighbor discovery.

Since $\gcd(a, b) \leq \hat{\omega}$, from (i) we know that $h_{\min}(\phi) < \infty$. If $a \geq \hat{\omega} > \lfloor a/2 \rfloor$, then $\lfloor \frac{a}{\hat{\omega}} \rfloor = 1$. Clearly, $\max_{\phi} h_{\min}(\phi) \geq 0$.

Consider $\lfloor a/2 \rfloor \geq \hat{\omega} \geq 1$. We use proof by contradiction. Assume $\max_{\phi} h_{\min}(\phi) \leq \lfloor \frac{a}{\hat{\omega}} \rfloor - 2$. We consider $\lfloor \frac{a}{\hat{\omega}} \rfloor > 2$ (as the case when $\lfloor \frac{a}{\hat{\omega}} \rfloor = 2$ can be readily proven). Define $\Phi_i \triangleq \{\phi | h_{\min}(\phi) = i, a - 1 \geq \phi \geq 0\}$, $i \in \{0, 1, 2, \cdots, \lfloor \frac{a}{\hat{\omega}} \rfloor - 2\}$. It follows that $\forall i, j \in \{0, 1, 2, \cdots, \lfloor \frac{a}{\hat{\omega}} \rfloor - 2\}$, $i \neq j$, we have $\Phi_i \cap \Phi_j = \emptyset$, and $\bigcup_{i=0}^{\lfloor \frac{a}{\hat{\omega}} \rfloor - 2} \Phi_i = \{0, 1, 2, \cdots, a - 1\}$. Since $\bigcup_{i=0}^{\lfloor \frac{a}{\hat{\omega}} \rfloor - 2} \Phi_i$ has a elements, and is union of $\lfloor \frac{a}{\hat{\omega}} \rfloor - 1$ sets, there exists a set, say Φ_l ($l \in \{0, 1, 2, \cdots, \lfloor \frac{a}{\hat{\omega}} \rfloor - 2\}$), such that $|\Phi_l| \geq \frac{a}{\lfloor \frac{a}{\hat{\omega}} \rfloor - 1} > \hat{\omega}$. Here $|\cdot|$ means cardinality of a set. Based on definition of Φ_l and $h_{\min}(\phi)$, for any element (say ϕ) in Φ_l, we know that $(\phi + l \times b) \mod a < \hat{\omega}$. Since $|\Phi_l| > \hat{\omega}$, there exist two different elements in Φ_l, say ϕ' and ϕ'', such that $\phi' + l \times b \equiv \phi'' + l \times b \bmod a$. Then it follows $\phi' \equiv \phi'' \bmod a$. As $\phi' < a$ and $\phi'' < a$, we have $\phi' = \phi''$, which contradicts the fact that ϕ' and ϕ'' are different elements in Φ_l.

Proof of (iii):

By (ii), the maximal number of hops is at least $\lfloor \frac{a}{\hat{\omega}} \rfloor - 1$. Suppose the maximal number of hops happens at time offset ϕ ($a - 1 \geq \phi \geq 0$). As shown in Fig. 1, the worst-case latency from t_0 to t_1 is $b - \tau$, and the worst-case latency from t_1 to discovery time is at least $\left(\lfloor \frac{a}{\hat{\omega}} \rfloor - 1\right) \times b + \tau$. Therefore, the worst-case discovery latency is at least $b - \tau + \left(\lfloor \frac{a}{\hat{\omega}} \rfloor - 1\right) \times b + \tau = \lfloor \frac{a}{\hat{\omega}} \rfloor \times b$.

This completes the proof. ■

Theorem 1 shows that node A can discover node B if and only if $\gcd(a, b) \leq \hat{\omega}$. For initial time offset ϕ, it takes $h_{\min}(\phi)$ hops for neighbor discovery. Fig. 3 gives three simple examples. For each circle in Fig. 3, in addition to the target area $\{0, 1, ..., \omega - \tau\}$, the rest of area (i.e., $\{\omega - \tau + 1, \omega - \tau + 2, ..., a - 1\}$) is divided into sectors. Each sector contains one or more integers that represent initial time offset values,[3] and the number inside the sector is the number of hops for all those time offsets to achieve neighbor discovery. Clearly, the number inside the target area is 0, and thus, is omitted. For instance, in Fig. 3(a), when the initial time offset is 4 or 5 time units, it takes 7 hops for node A to discover node B. It can be seen that, $\gcd(a, b)$ is 2, 4, and 8 in Fig. 3(a), (b), and (c), respectively, while $\hat{\omega}$ is 4 in Fig. 3(a), (b), and (c). By Theorem 1, we know that we can guarantee neighbor discovery in Fig. 3(a) and (b) but cannot guarantee neighbor discovery in Fig. 3(c).

Given a, b, and $\hat{\omega}$, Theorem 1 not only gives a necessary and sufficient condition for node A to discover node B, but also provides a lower bound of worst-case discovery latency. The next question is under what conditions the lower bound can be achieved? The next theorem answers this question.

Theorem 2: If $\gcd(a, b) > 1$ and $\hat{\omega} = \gcd(a, b)$, then $\max_{\phi} h_{\min}(\phi) = \frac{a}{\hat{\omega}} - 1$, $a - 1 \geq \phi \geq 0$. The worst-case discovery latency is $\frac{a}{\hat{\omega}} \times b$.

Proof: Let $\Phi_0 = \{0, 1, \cdots, \hat{\omega} - 1\}$, and iteratively compute $\Phi_i = \{\phi | \phi = (\phi' - b) \bmod a, \phi' \in \Phi_{i-1}\}$, $i = 1, 2, \cdots, \frac{a}{\hat{\omega}} - 1$. Next we prove for $\forall i, j$, $i \neq j$, $\frac{a}{\hat{\omega}} - 1 \geq i, j \geq 0$, we have $\Phi_i \cap \Phi_j = \emptyset$. We use proof by contradiction. Assume $i \neq j$, and $\Phi_i \cap \Phi_j \neq \emptyset$. Without loss of generality, assume $i < j$ and $\phi \in \Phi_i \cap \Phi_j$. Since $\phi \in \Phi_i$, there exists $r \in \Phi_0$ such that $\phi = (r - i \times b) \bmod a$. Similarly, since $\phi \in \Phi_j$, there exists $r' \in \Phi_0$ such that $\phi = (r' - j \times b) \bmod a$. Therefore, we have $(r - i \times b) \equiv (r' - j \times b) \bmod a$, which leads to

$$(j - i) \times b \equiv (r' - r) \bmod a. \tag{3}$$

Since $\gcd(a, b) = \hat{\omega}$, it can be seen that $(j - i) \times b \bmod a$ is a multiple of $\hat{\omega}$, and thus, $(r' - r) \bmod a$ is a multiple of $\hat{\omega}$. Together with the fact that r and r' are both less than $\hat{\omega}$, we have $r' = r$. From (3) we have

$$(j - i) \times b \bmod a = 0. \tag{4}$$

As $\hat{\omega} = \gcd(a, b)$, we can denote a as $a'\hat{\omega}$ and denote b as $b'\hat{\omega}$, with $\gcd(a', b') = 1$. Then (4) becomes

$$(j - i) \times (b'\hat{\omega}) \bmod (a'\hat{\omega}) = 0. \tag{5}$$

As $\gcd(a', b') = 1$ and both i and j are less than $\frac{a}{\hat{\omega}} = a'$, from (5) we can conclude that $i = j$, which contradicts the assumption $i \neq j$.

Next we prove the following statement, referred to as **Statement 2**: if $\phi \in \Phi_i$, then $h_{\min}(\phi) = i$, $i = 0, 1, 2, \cdots, \frac{a}{\hat{\omega}} - 1$. We prove it by mathematical induction. When $i = 0$, we have $h_{\min}(\phi) = 0$ if $\phi \in \Phi_0$. Assume Statement 2 is true for $i = n < \frac{a}{\hat{\omega}}$, i.e., $h_{\min}(\phi) = n$ if $\phi \in \Phi_n$. Recall that $\Phi_{n+1} = \{\phi | \phi = (\phi' - b) \bmod a, \phi' \in \Phi_n\}$. For $\forall \phi \in \Phi_{n+1}$,

[3]As we only consider integer time offset values, there is a "white gap" between two neighboring sectors.

there exists $\phi' \in \Phi_n$ such that $\phi = (\phi' - b) \bmod a$. It follows $(\phi + b) \bmod a = \phi'$. Because $h_{\min}(\phi') = n$, we have $h_{\min}(\phi) = n + 1$. Thus, we have proven Statement 2.

Since $|\Phi_i| = \hat{\omega}$ and $\Phi_i \cap \Phi_j = \emptyset, \forall i, j, i \neq j$, $\frac{a}{\hat{\omega}} - 1 \geq i, j \geq 0$, we have $\bigcup_{k=0}^{\frac{a}{\hat{\omega}}-1} \Phi_k = \{0, 1, \cdots, a-1\}$. Thus, for any ϕ in $\{0, 1, \cdots, a-1\}$, if ϕ falls within set Φ_k, we have $h_{\min}(\phi) = k$. As the set $\Phi_{\frac{a}{\hat{\omega}}-1}$ is non-empty, we have $\max_{\phi} h_{\min}(\phi) = \frac{a}{\hat{\omega}} - 1$.

Finally, similar to the proof of statement (iii) of Theorem 1, the worst-case discovery latency is $\frac{a}{\hat{\omega}} \times b$.

This completes the proof. ■

Theorem 2 means that when $\gcd(a, b) > 1$ and $\hat{\omega} = \gcd(a, b)$, then the lower bound of the worst-case discovery latency is achieved. For example, $\hat{\omega} = \gcd(a, b)$ is satisfied in Fig. 3(b) but not in Fig. 3(a) or Fig. 3(c), while we have $\frac{a}{\hat{\omega}} - 1 = 7$ in Fig. 3(a), (b), and (c). It can be seen that $\max_{\phi} h_{\min}(\phi)$ is equal to 7 in Fig. 3(b), and is more than 7 in Fig. 3(a) and Fig. 3(c).

For the case when $\gcd(a, b) > 1$ and $\hat{\omega} = \gcd(a, b)$, the next theorem provides a method for calculating $h_{\min}(\phi)$ for a particular ϕ.

Theorem 3: Given a, b, and $\hat{\omega}$, if $\gcd(a, b) > 1$ and $\hat{\omega} = \gcd(a, b)$, it follows:

(i) The congruence

$$\left(\frac{b}{\hat{\omega}}\right) x \equiv -1 \bmod \left(\frac{a}{\hat{\omega}}\right) \tag{6}$$

has a unique solution k such that $\left(\frac{a}{\hat{\omega}}\right) > k > 0$.

(ii) For an arbitrary integer $\phi \in \{0, 1, 2, ..., a-1\}$, if $\phi \in \Phi_i = \{i\hat{\omega}, i\hat{\omega} + 1, \cdots, (i+1)\hat{\omega} - 1\}$, $i = 0, 1, \cdots, \left(\frac{a}{\hat{\omega}} - 1\right)$, then $h_{\min}(\phi) = i \times k \mod \left(\frac{a}{\hat{\omega}}\right)$.

Proof: **Proof of (i)**:

since $\hat{\omega} = \gcd(a, b)$, $\left(\frac{b}{\hat{\omega}}\right)$ and $\left(\frac{a}{\hat{\omega}}\right)$ are coprime, and thus congruence (6) has a unique solution modulo $\left(\frac{a}{\hat{\omega}}\right)$ [29]. This means, if congruence (6) has a solution $x = k$, then it follows that all integers x satisfying $x \equiv k \bmod \left(\frac{a}{\hat{\omega}}\right)$ are also solutions. It can be seen that we can restrict k such that $\left(\frac{a}{\hat{\omega}}\right) > k > 0$.

Proof of (ii):

We prove it by mathematical induction. Let $a' = \left(\frac{a}{\hat{\omega}}\right)$ and $b' = \left(\frac{b}{\hat{\omega}}\right)$. If $b'k \equiv -1 \bmod a'$, then $bk \equiv -\hat{\omega} \bmod a$ [30].

Clearly, it follows $h_{\min}(\phi) = 0$ for the base case, i.e., $i = 0$ and $\phi \in \Phi_0 = \{0, 1, \cdots, \hat{\omega} - 1\}$.

For $i = 1, \forall \phi \in \Phi_1 = \{\hat{\omega}, \hat{\omega}+1, \cdots, 2\hat{\omega}-1\}$, let $\phi = \hat{\omega}+r$, $r = 0, 1, \cdots, \hat{\omega} - 1$, and we have

$$\begin{aligned} &(\phi + (1 \times k \bmod a') \times b) \bmod a \\ &= (\phi + k \times b) \bmod a \\ &= (\hat{\omega} + r + k \times b) \bmod a \\ &= (\hat{\omega} + r - \hat{\omega}) \bmod a \\ &= r \\ &< \hat{\omega}, \end{aligned}$$

which means it takes no more than $(1 \times k \bmod a')$ hops for $\forall \phi \in \Phi_1 = \{\hat{\omega}, \hat{\omega}+1, \cdots, 2\hat{\omega}-1\}$. Next we prove $h_{\min}(\phi) = 1 \times k \mod a'$ for $\forall \phi \in \Phi_1 = \{\hat{\omega}, \hat{\omega}+1, \cdots, 2\hat{\omega}-1\}$. We prove it by contradiction. Let $h = 1 \times k \mod a'$. Suppose there exist ϕ' and h' such that $\phi' \in \Phi_1 = \{\hat{\omega}, \hat{\omega}+1, \cdots, 2\hat{\omega}-1\}$, $h' < h$, and $(\phi' + h' \times b) \bmod a < \hat{\omega}$. It can be seen that there exists ϕ, $\phi \in \Phi_1 = \{\hat{\omega}, \hat{\omega} + 1, \cdots, 2\hat{\omega} - 1\}$ such that $\phi' + h' \times b \equiv \phi + h \times b \bmod a$. It follows $\phi' - \phi \equiv (h - h') \times b \bmod a$. If $\phi' = \phi$, we have $(h - h') \times b \equiv 0 \bmod a$, and $(h - h') \times b' \equiv 0 \bmod a'$. By the **Statement 1** in the proof of Theorem 1, it follows $h' = h$, contradicted to the assumption $h' < h$. Otherwise, if $\phi' \neq \phi$, we have $\phi' - \phi \equiv (h - h') \times b \bmod a$, $\phi' - \phi \equiv (h - h') \times b \bmod \hat{\omega}$, and $\phi' - \phi \equiv 0 \bmod \hat{\omega}$, which cannot be true unless $\phi' = \phi$, contradicted to the assumption $\phi' \neq \phi$. Therefore, we have $h_{\min}(\phi) = 1 \times k \mod a'$ for $\forall \phi \in \Phi_1 = \{\hat{\omega}, \hat{\omega} + 1, \cdots, 2\hat{\omega} - 1\}$.

Assume $h_{\min}(\phi) = i \times k \mod a'$ when $(a' - 2) \geq i \geq 1$ and $\phi \in \Phi_i = \{i\hat{\omega}, i\hat{\omega} + 1, \cdots, (i+1)\hat{\omega} - 1\}$.

Now we consider the case of $i + 1$. Let $\forall \phi \in \Phi_{i+1} = \{(i+1)\hat{\omega}, (i+1)\hat{\omega}+1, \cdots, (i+2)\hat{\omega}-1\}$ and $\phi = (i+1)\hat{\omega}+r$, $r = 0, 1, \cdots, \hat{\omega} - 1$. We have

$$\begin{aligned} &(\phi + ((i+1) \times k \bmod a') \times b) \bmod a \\ &= (\phi + ((ik \bmod a' + k \bmod a') \bmod a') \times b) \bmod a, \end{aligned}$$

where there are two cases for the term $(ik \bmod a' + k \bmod a')$:

- Case 1: $(ik \bmod a' + k \bmod a') < a'$;
- Case 2: $2a' > (ik \bmod a' + k \bmod a') \geq a'$.

For the case 1, we have

$$\begin{aligned} &(\phi + ((i+1) \times k \bmod a') \times b) \bmod a \\ &= (\phi + ((ik \bmod a' + k \bmod a') \bmod a') \times b) \bmod a \\ &= (\phi + (ik \bmod a' + k \bmod a') \times b) \bmod a \\ &= ((i+1) \times \hat{\omega} + r + (ik \bmod a' + k \bmod a') \times b) \bmod a \\ &= (i\hat{\omega} + r + (ik \bmod a') \times b + \hat{\omega} + bk) \bmod a \\ &= (i\hat{\omega} + r + (ik \bmod a') \times b) \bmod a \\ &< \hat{\omega}, \end{aligned}$$

where the last inequality is based on the induction hypothesis.

For the case 2, it follows

$$\begin{aligned} &(\phi + ((i+1) \times k \bmod a') \times b) \bmod a \\ &= (\phi + ((ik \bmod a' + k \bmod a') \bmod a') \times b) \bmod a \\ &= (\phi + (ik \bmod a' + k \bmod a' - a') \times b) \bmod a \\ &= ((i+1) \times \hat{\omega} + r + (ik \bmod a' + k \bmod a' - a') \times b) \\ &\quad \bmod a \\ &= ((i+1) \times \hat{\omega} + r + (ik \bmod a') \times b + kb - a'b) \bmod a \\ &= (i\hat{\omega} + r + (ik \bmod a') \times b + \hat{\omega} + kb - a'b) \bmod a \\ &= (i\hat{\omega} + r + (ik \bmod a') \times b) \bmod a \\ &< \hat{\omega}, \end{aligned}$$

where the last inequality is based on the induction hypothesis.

It can be seen that for $\forall \phi \in \Phi_{i+1} = \{(i+1)\hat{\omega}, (i+1)\hat{\omega} + 1, \cdots, (i+2)\hat{\omega}-1\}$, it takes no more than $(i+1) \times k \bmod a'$ hops, and similarly, we can prove $h_{\min}(\phi) = (i+1) \times k \bmod a'$.

This completes the proof. ■

For example, in Fig. 3(b), $a = 32$, $b = 20$, and $\hat{\omega} = 4$. The congruence $\left(\frac{20}{4}\right) x \equiv -1 \bmod \left(\frac{32}{4}\right)$ has a solution $k = 3$. If $\phi = 17$, then $h_{\min}(\phi) = \left(\left\lfloor \frac{17}{4} \right\rfloor \times 3\right) \bmod 8 = 12 \bmod 8 = 4$.

C. A Generic Analytical Model

In this subsection, we will show that the Circle model is a generic analytical model, which can be used to explain and analyze existing well-known slotted NDPs such as U-Connect, Disco, Hello, and Nihao. In those protocols, the slot length is denoted by t_{slot}. An active slot contains transmission of beacons and listening to the channel. Assume it takes one time unit to transmit a beacon, then $\hat{\omega} = t_{slot}$. Moreover, it is assumed that overlapping active slots will result in neighbor discovery.

We first analyze neighbor discovery in these NDPs with asymmetric duty cycles.

Suppose U-Connect selects two different primes, say p_1 and p_2, for node A and node B, respectively. By circle model, $a = p_1 \times t_{slot}$ and $b = p_2 \times t_{slot}$. Since p_1 and p_2 are different primes, it follows $\gcd(p_1 \times t_{slot}, p_2 \times t_{slot}) = t_{slot} = \hat{\omega}$. By Theorem 1, node A can discover node B and vice versa. The worst-case discovery latency is $p_1 \times p_2 \times t_{slot}$ according to Theorem 2. Similar analysis can be applied to Hello. Consider nodes A and B respectively running $Hello(c_1, n_1)$ and $Hello(c_2, n_2)$, where c_1 and c_2 are primes. If $c_1 \neq c_2$, then $gcd(c_1 \times t_{slot}, c_2 \times t_{slot}) = t_{slot} = \hat{\omega}$. Thus, neighbor discovery is guaranteed according to Theorem 1, and the worst-case discovery latency is $c_1 \times c_2 \times t_{slot}$ according to Theorem 2.

For Disco, suppose the prime pairs of node A and node B are (p_{11}, p_{12}) and (p_{21}, p_{22}), respectively. It can be seen that there are four combinations of different primes used by the two nodes, i.e., (p_{11}, p_{21}), (p_{11}, p_{22}), (p_{12}, p_{21}), and (p_{12}, p_{22}), corresponding to four circle models. Any combination guarantees neighbor discovery according to Theorem 1. Given four combinations, by Theorem 2, the worst-case discovery latency will be $\min\{p_{11} \times p_{21} \times t_{slot}, p_{11} \times p_{22} \times t_{slot}, p_{12} \times p_{21} \times t_{slot}, p_{12} \times p_{22} \times t_{slot}\}$.

Nihao has two parameters m and n where n can be changed by nodes but m remains constant. Suppose the parameters for node A and node B are (m, n_1) and (m, n_2), respectively. It can be seen that $\hat{\omega} = m \times t_{slot}$. Since $gcd(m \times n_1 \times t_{slot}, m \times t_{slot}) = m \times t_{slot} = \hat{\omega}$, the worst-case discovery latency for node A to discover node B is $m \times n_1 \times t_{slot}$. Similarly, the worst-case discovery latency for node B to discover node A is $m \times n_2 \times t_{slot}$. Because Nihao adopts two separate one-way discoveries to achieve mutual discovery, the worst-case discovery latency is $m \times t_{slot} \times \max\{n_1, n_2\}$.

Next we analyze neighbor discovery in these NPDs with symmetric duty cycles.

Suppose for Disco, both node A and node B select (p_1, p_2). Since $\gcd(p_1 \times t_{slot}, p_2 \times t_{slot}) = t_{slot} = \hat{\omega}$, the worst-case discovery latency is $p_1 \times p_2 \times t_{slot}$. For Nihao, assume two nodes have the same parameters (m, n). Since $gcd(m \times n \times t_{slot}, m \times t_{slot}) = m \times t_{slot} = \hat{\omega}$, the worst-case discovery latency is $m \times n \times t_{slot}$.

However, for U-Connect, if two nodes select the same prime p, neighbor discovery cannot be guaranteed because $\gcd(p \times t_{slot}, p \times t_{slot}) = p \times t_{slot} > \hat{\omega}$. In other words, U-Connect cannot ensure discovery with symmetric duty cycle if nodes only works at a time slot for every p time slots. To address this problem, U-Connect adopts quorum technique. Specifically, U-Connect combines p basic cycles into a hyper-cycle, denoted by a $p \times p$ matrix. The first $\frac{p+1}{2}$ slots at the first row are active slots, and the first slot of row $2, 3, ..., p$ are also active. This technique guarantees overlapping active slots for two nodes with the same parameter because their time offset cannot be more than $\frac{p+1}{2}$ slots. Hello also adopts quorum technique to deal with discovery with symmetric duty cycles. Searchlight relies on quorum technique for discovery with both symmetric and asymmetric duty cycles. Each duty cycle in Searchlight should be a power-multiple of the smallest duty cycle, which actually limits the number of duty cycles that can be used.

In addition to analyzing existing NDPs, Circle also provides guidelines for designing new protocols. For example, according to Circle, new slotted NDPs can use coprime parameters instead of primes. For instance, there are 31 prime numbers from 13 to 151, but at least 36 numbers that are pairwise coprime.[4] In other words, using coprime parameters increases the number of feasible duty cycle values.

D. Discussion on Modeling Neighbor Discovery for BLE Networks

In this subsection, we show that Circle can also be used to model other more complex neighbor discovery problem. We take the neighbor discovery for BLE networks as an example.

In BLE networks, a node tries to announce its presence to other nodes by working in an advertising mode, thus referred to as an *advertiser*. A node tries to discover other nodes by working in a scanning mode, thus referred to as a *scanner*. Fig. 4 illustrates how these nodes work. Three predetermined advertising channels are assigned, which are channels 37, 38, and 39. Over each advertising channel, an advertiser announces its presence by periodically sending advertising packets (called Adv_PDUs) into the channel. The advertiser also tries to receive possible responses over the same channel immediately after each of its transmissions. A scanner tries to receive Adv_PDUs of other nodes by scanning (i.e., listening to) each of the three advertising channels periodically. The length of a scanning over each advertising channel is called a *scan-window*. If the scanner successfully gets an Adv_PDU in a scan-window over an advertising channel, we say that the advertiser of the Adv_PDU is successfully discovered by the scanner [20]–[22].

Assume that for the scanner, the interval from the beginning of a scanning event to the beginning of the next scanning event (noting that the two scanning events are over two different advertising channels) is a and scan-window length is ω; for the advertiser, the advertising interval is b, the duration for sending an Adv_PDU is τ, and the duration for receiving possible responses over each advertising channel is δ.

To characterize BLE neighbor discovery, we can build three Circle models, respectively for channels 37, 38, and 39. For presentation simplicity, we assume that at time instant t_0, the advertiser comes within the communication range of the scanner and sends the first Adv_PDU, as shown in Fig. 4.

[4]In Section V, we will give an example about how to find pairwise coprime numbers.

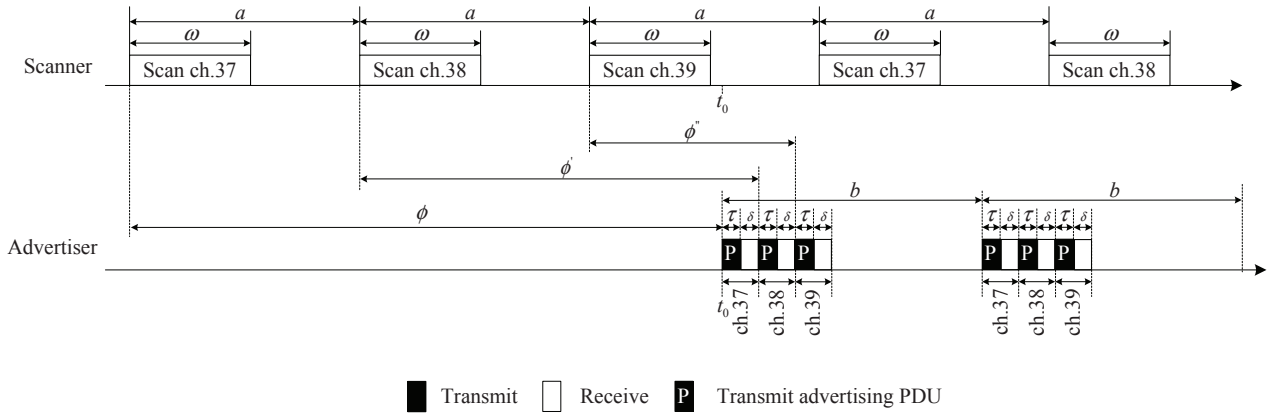


Fig. 4. The discovery procedure of BLE.

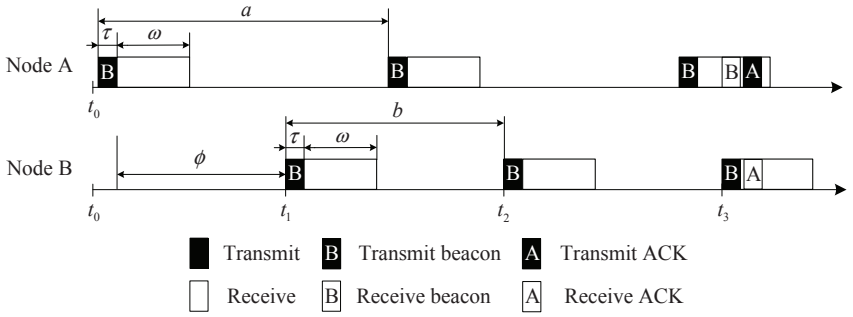


Fig. 5. An example of Circle protocol.

Assume the initial time offsets of the advertiser to the scanner on channel 37, 38, and 39 are ϕ, $\phi^{'}$, and $\phi^{''}$, respectively. Given ϕ, we can derive $\phi^{'}$ and $\phi^{''}$ as follows

$$\phi' = (\phi + 2a + \tau + \delta) \bmod 3a, \tag{7}$$

$$\phi'' = (\phi + a + 2\tau + 2\delta) \bmod 3a. \tag{8}$$

Based on the three Circle models, the BLE neighbor discovery can be stated as: given an arbitrary ϕ, there exists h ($h < \infty$) satisfying at least one of the following three inequalities:

$$(\phi + h \times b) \mod 3a < \hat{\omega}, \tag{9}$$

$$(\phi' + h \times b) \mod 3a < \hat{\omega}, \tag{10}$$

$$(\phi'' + h \times b) \mod 3a < \hat{\omega}, \tag{11}$$

where $\hat{\omega} \triangleq \omega - \tau + 1$.

By Theorem 1, if $\gcd(3a, b) < \hat{\omega}$, then neighbor discovery is guaranteed, that is, the scanner can discover the advertiser. Further, if $\gcd(3a, b) = \hat{\omega}$, the worst-case discovery latency is

$$\max_{\phi} \min(h_{\min}(\phi) \times b, h_{\min}(\phi') \times b, h_{\min}(\phi'') \times b), \tag{12}$$

where $\phi = 0, 1, 2, \cdots, 3a-1$. Given an arbitrary ϕ, $\phi^{'}$ and $\phi^{''}$ can be calculated by (7) and (8) respectively, and $h_{\min}(\phi)$, $h_{\min}(\phi')$, and $h_{\min}(\phi'')$ can be calculated according to Theorem 3. Similarly, the average discovery latency is

$$\frac{1}{3a} \sum_{\phi=0}^{3a-1} \min(h_{\min}(\phi) \times b, h_{\min}(\phi') \times b, h_{\min}(\phi'') \times b). \tag{13}$$

As can be seen from the above description, Circle is a generic analytical model, which can be used not only to analyze existing well-known slotted NDPs, but also to build a model for BLE neighbor discovery. In addition, Circle can also be used to design new NDPs. In the next section, we present a new unslotted NDP.

V. Circle Protocol Design

A. Working Mode Design

In this section, we present a new NDP which is designed based on Circle model. For simplicity, it is also called Circle. The basic working mode of Circle is very simple: each node periodically wakes up, sends a beacon, and listens to channel for a while. Once a beacon is received, an acknowledgement (ACK) is sent.[5] Fig. 5 shows an example, in which node A and node B periodically send a beacon with τ time units and then listen to channel for ω time units. The cycle lengths of node A and node B are a and b time units, respectively. In the example, node A receives a beacon from node B, and then it responds with an ACK. From this moment, node A and node B know each other.

By Theorem 2, Circle requires that the greatest common divisor of any two distinct cycle lengths should be $\hat{\omega}$. Let $L = \{\ell_1, \ell_2, \cdots, \ell_n\}$ be the set of distinct cycle lengths used by Circle. It follows $\ell_i = \hat{\omega} \times p_i$, $n \geq i \geq 1$, and the elements in the set $\{p_1, p_2, \cdots, p_n\}$ are pairwise coprime. We give an

[5]This working mode is similar to that of a receiver-initiated low-duty-cycle medium access control (MAC) Protocol [27], [28].

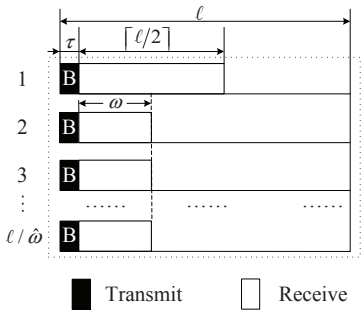


Fig. 6. The hyper-cycle of Circle.

example to show how to construct the set L. Suppose $\omega = 4$, $\tau = 1$, $\hat{\omega} = 4$, and we require that the cycle length should be in the range of $[100, 1000]$. Initially, $L = \emptyset$ (null set). We search cycle lengths in ascending order. Firstly, 100 is added into L. The next cycle length that can be added into L is 104 because $\gcd(104, 100) = 4$. Now L has two elements, and the next cycle length that can be added into L is 108 because it has the greatest common divisor of 4 with all elements already in L. Note that 112 cannot be added into L because $\gcd(104, 112) = 8$. Repeat this process until we have searched all integer values within $[100, 1000]$. Finally there are 49 elements in L.

If two nodes select the same cycle length, however, discovery cannot be guaranteed because the condition of Theorem 1 cannot be satisfied. To address this problem, inspired by the quorum technique used by U-connect and Hello, we combine multiple cycles into a hyper-cycle. Specifically, given cycle length ℓ, the hyper-cycle consists of $\ell/\hat{\omega}$ cycles, indexed by $1, 2, \cdots, \ell/\hat{\omega}$. During the first cycle, a node first sends a beacon for τ time units and then listens to the channel for $\lceil \ell/2 \rceil$ time units, and in all other cycles, the node first sends a beacon for τ time units and then listens to the channel for ω time units, as shown in Fig. 6. We denote Circle protocol with parameters ℓ and $\hat{\omega}$ by $Circle(\ell, \hat{\omega})$, and its duty cycle is given by

$$\begin{aligned} DC &= \frac{\frac{\ell}{\hat{\omega}} \times (\omega + \tau) + (\lceil \ell/2 \rceil - \omega)}{\frac{\ell}{\hat{\omega}} \times \ell} \\ &= \frac{\omega + \tau}{\ell} + \frac{(\lceil \ell/2 \rceil - \omega) \times \hat{\omega}}{\ell^2}. \end{aligned} \tag{14}$$

The first term of the equation (14) can be regarded as the basic duty cycle of Circle protocol, while the second term can be thought of as an extra duty cycle just in case the same cycle length is used by two nodes.

B. Worst-case Discovery Latency

Theorem 4: Given two nodes with parameter pairs $(\ell_1, \hat{\omega})$ and $(\ell_2, \hat{\omega})$, the worst-case discovery latency for Circle is $\frac{\ell_1 \times \ell_2}{\hat{\omega}}$.

Proof: Assume node A and node B adopt parameter pairs $(\ell_1, \hat{\omega})$ and $(\ell_2, \hat{\omega})$, respectively, with $\hat{\omega} = \gcd(\ell_1, \ell_2)$. First consider the case with $\ell_1 \neq \ell_2$. By Theorem 2, the worst-case discovery latency is $\frac{\ell_1 \times \ell_2}{\hat{\omega}}$.

Next we consider the case with $\ell_1 = \ell_2$. Since each node listens to the channel for $\lceil \ell/2 \rceil$ time units in its first cycle, either node A or node B would receive a beacon in its first cycle from the other node. The worst case happens, for example, when the two nodes come into communication range of each other at the time instance when one node just turns off its radio in the first cycle while the other node just finishes sending the beacon in the second cycle. It can be seen that it also takes $\frac{\ell_1 \times \ell_2}{\hat{\omega}}$ time units for the two nodes to discover each other. This completes the proof. ■

VI. Evaluation

In this section, we evaluate the performance of Circle protocol by comparing it with state-of-the-art NDPs including Disco, U-Connect, Hello, Hello-S, and Nihao in a testbed of TelosB motes.

A. Implementation

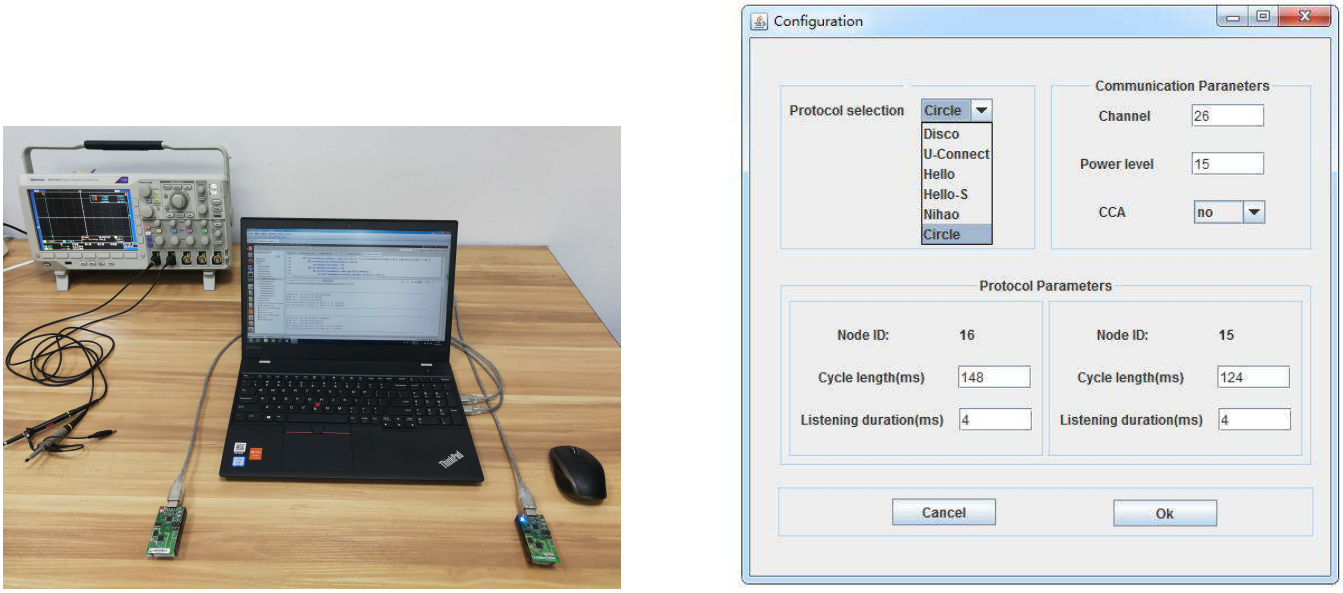
We have implemented Circle and other protocols for comparison on TinyOS 2.1.2.[6] All these protocols are implemented under the UPMA (Unified Radio Power Management Architecture) framework of TinyOS. Generally, nodes work in low duty cycle, and therefore we consider duty cycles from 1% to 10%. Disco, U-Connect, Hello, and Nihao are slotted protocols, while Circle is unslotted protocol. For these slotted protocols, the slot length is set to 10 ms as Disco does. Hello-S employs striped probing by increasing the length of active slot by 4 ms. For Circle, we empirically set $\hat{\omega}$ to 4 ms. The beaconing schemes taken by these protocols are different. Disco, U-Connect, and Hello send two beacons at each active slot, one at the beginning of the slot and the other at the end of the slot. Nihao and Circle both periodically send a beacon. Each beacon message contains the timestamp of transmitting, and its transmission time is about 1 ms.

To send a 1-ms beacon in a real system (such as the testbed), some extra radio-on time is needed. For periodically sending a beacon, the system should turn on the radio in advance, set the header and payload of the beacon, and send the beacon to underlying components for transmitting, which brings about extra radio-on time cost. Empirically, the extra radio-on time cost is 3 ms for Circle and Nihao. For Disco, U-Connect, and Hello, the extra radio-on time cost for an active slot is 2 ms, relatively smaller than Circle and Nihao as beaconing and listening are included in one slot in Disco, U-Connect, and Hello. The extra radio-on time cost is considered when we calculate the duty cycle of each protocol.

Fig. 7(a) shows the testbed consisting of a laptop and two TelosB motes, called node A and B. The two motes tried to discover each other, and the laptop was used to configure the two motes and collect results.

In order to fairly compare one NDP with other NDPs, we use the same configuration for each NDP, as a 4-tuple $(DC_A, DC_B, \varphi_A, \varphi_B)$. Specifically, DC_A and DC_B are desired duty cycles of nodes A and B, respectively; and φ_A and φ_B are initial phases of the cycles of node A and node B, respectively, when they enter communication range of each

[6]Note that TinyOS platform has also been used in [15] and [18] for performance evaluation.



(a) Experiment testbed

(b) Configuration interface of Java application

Fig. 7. Experiment testbed and the configuration interface of Java application.

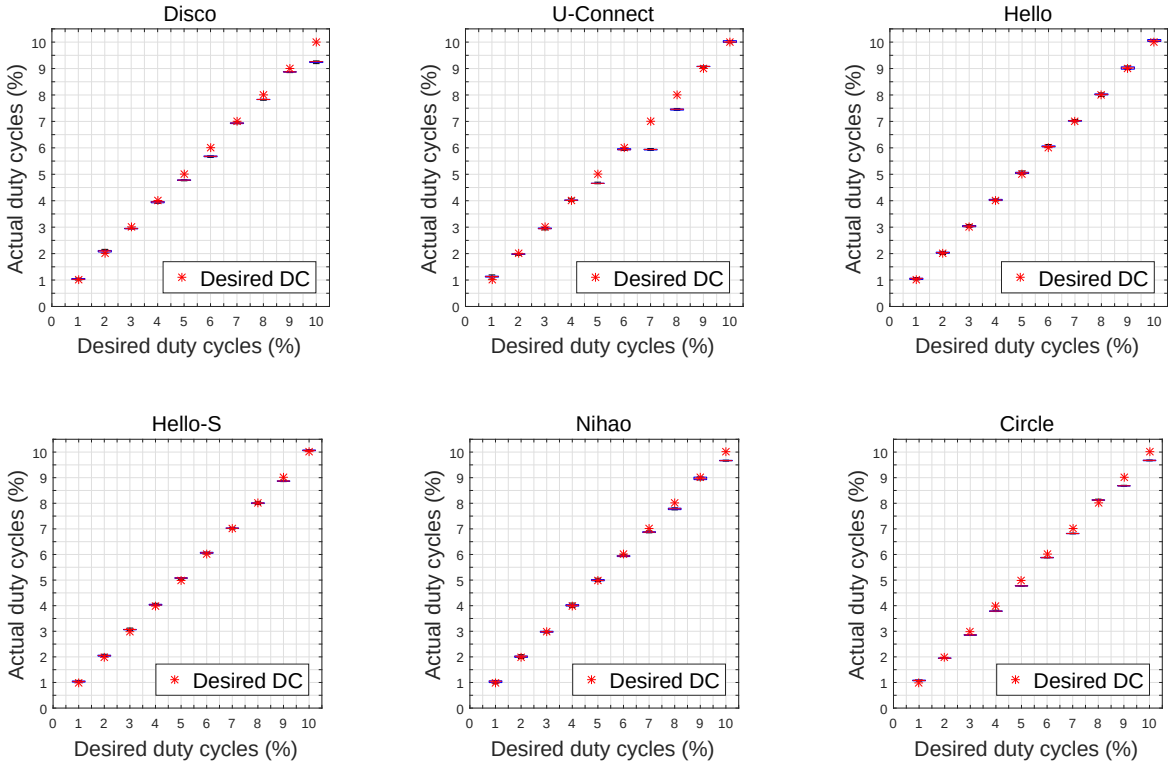


Fig. 8. The box plots of actual duty cycles (DCs) of NDPs.

other. DC_A and DC_B are chosen from $\{1\%, 2\%, \cdots, 10\%\}$, and therefore there are $\binom{2}{10} = 55$ pairs of duty cycles. For each pair of duty cycles, 100 pairs of initial phases (φ_A, φ_B) are randomly generated. For each configuration of $(DC_A, DC_B, \varphi_A, \varphi_B)$, we let the two motes implement each NDP and record the discovery latency.

We have implemented a Java application that enables interactions between the laptop and the two motes by serial communication. The interface Java application is shown in Fig. 7(b). The two motes are connected with the laptop by USB cables. To test an NDP, the Java application reads a configuration $(DC_A, DC_B, \varphi_A, \varphi_B)$ from the source file (which contains all configurations), and sends a configuration message to node A and node B respectively. For example, the message sent to node A contains working parameters corresponding to DC_A (e.g., the prime number p for U-Connect, or $(\ell, \hat{\omega})$ for Circle). In addition, the message also contains initial phase φ_A, based on which the mote will set its initial state. After configuration, the two motes run the NDP simultaneously with their own initial states. If node A discovers node B (i.e., node A receives a beacon from node B), node A sends a discovery message to the laptop containing the time-stamp of the beacon sent by node B, the moment when node A receives the beacon, and the discovery latency. Node B behaves similarly when it finds node A. After receiving two discovery messages, the Java application records the data, stops the two motes, reads the next data, and repeats the experiment.

For each desired duty cycle (i.e., $1\%, 2\%, ..., 10\%$), based on estimated extra time cost, we calculate working parameters for the NDPs. Then we use the interface CC2420Accounting provided by component CC2420Csma in the UPMA framework of TinyOS to accurately measure the actual duty cycles of the NDPs. Fig. 8 shows the box plots of actual duty cycles of different NDPs. It can be seen that in most cases, the actual duty cycles are close to desired ones, slightly lower or higher than desired values. However, one exception is that due to prime limitation, U-Connect selects the same prime for duty cycles of 6% and 7%. With the actual duty cycles approximately equal to the desired ones, we can make a fair comparison.

B. Experimental Results

Fig. 9 shows experimental cumulative distribution function (CDF) of discovery latency. It can be observed that Circle performs the best among the six protocols. Disco, Hello, and Hello-S have similar performance, while U-Connect is slightly better than these three protocols. For instance, 90 percent of discovery latencies are within 16s for Circle, 28s for Disco, 29s for U-Connect, 30s for Hello, 32s for Hello-S, and 85s for Nihao.

In our experiments, Circle and Nihao exhibit smaller worst-case latency. Specifically, the worst-case latencies in ascending order are 330s for Circle, 354s for Nihao, 581s for U-Connect, 712s for Hello-S, 791s for Disco, and 816s for Hello. In Circle and Nihao, there are dedicated time for receiving, which allows Circle and Nihao to successfully receive beacons from neighbors. In contrast, Disco, U-Connect, Hello, and Hello-S

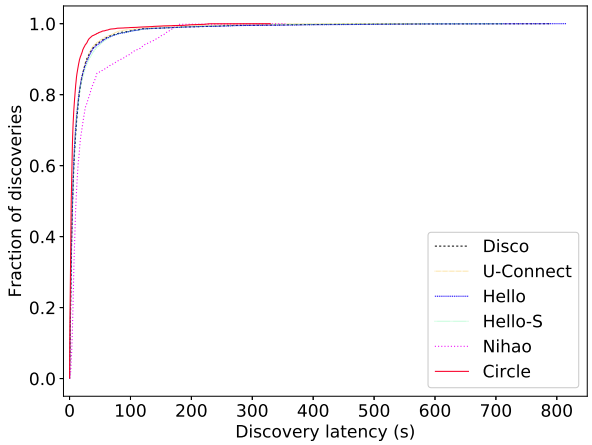


Fig. 9. Experimental CDF of discovery latency.

adopt the same time-slotted model in which sending beacon and listening to the channel happen in one slot. For Disco, U-Connect, Hello, and Hello-S, we debugged cases with large discovery latency, and found that they are mainly caused by half-duplex transceiver and short overlapping duration. Short overlapping duration is very likely not sufficient for the two nodes to receive beacons from each other.

Fig. 10 shows the average latency for all experimental duty cycle pairs, in which DC1 and DC2 are duty cycles of the two nodes. It can be seen that for each protocol, the largest average latency occurs at duty cycle pair (1%, 1%), and when one node increases its duty cycle, the discovery latency of all protocols except Nihao decrease significantly. To be more specific, Fig. 11(a) shows the average discovery latency for asymmetric duty cycle pairs (1%, 5%), (1%, 10%), and (5%, 10%). Interestingly, when only one node increases its duty cycle, the average latency of Nihao just decreases slightly. This is because in Nihao, the bidirectional discovery is accomplished by two separate one-way discoveries, and the discovery latency is the larger of the two one-way discovery latency values. Overall, Circle has the best average performance among the six protocols. For instance, compared with Disco, Circle reduces the average latency by 42%, 57%, and 37% at duty cycle pairs (1%, 5%), (1%, 10%), and (5%, 10%), respectively.

The average discovery latency for symmetric duty cycle pairs (1%, 1%), (5%, 5%), and (10%, 10%) are shown in Fig.11(b). The discovery for (1%,1%) represents the worst case for all experiments. As the duty cycles increase, the average discovery latency of the protocols reduce significantly. It can be seen that Circle also outperforms other NDPs in neighbor discovery with symmetric duty cycles.

C. Further Discussion on Multiple-Node Case

Now we discuss the case when Circle is implemented in a network with multiple nodes. If two or more nodes transmit beacons at the same time (or their beacons overlap in time), this is a collision, which degrades the performance of Circle.

We consider a target node, say node A, that implements Circle with cycle length ℓ_A. Based on the cycle length ℓ_A, we

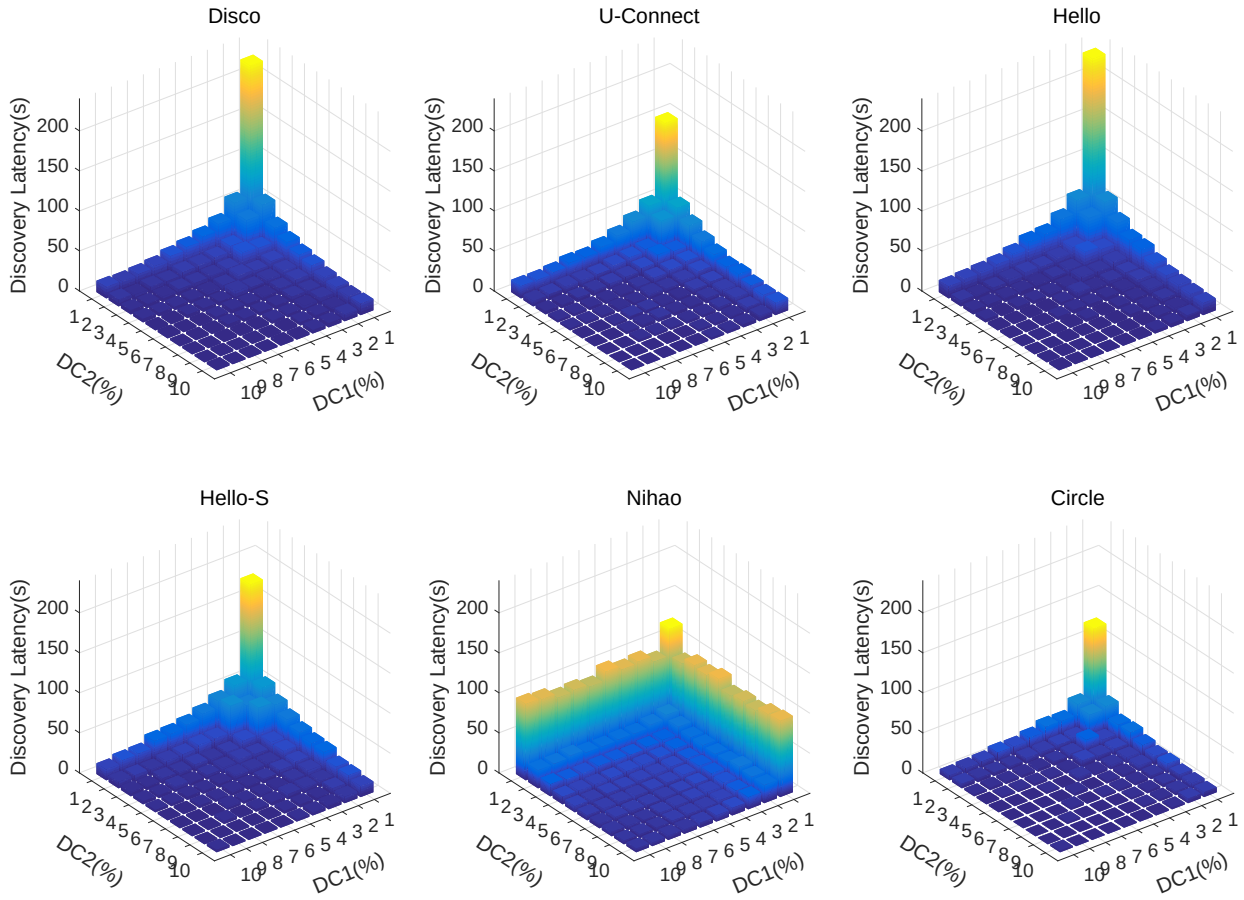
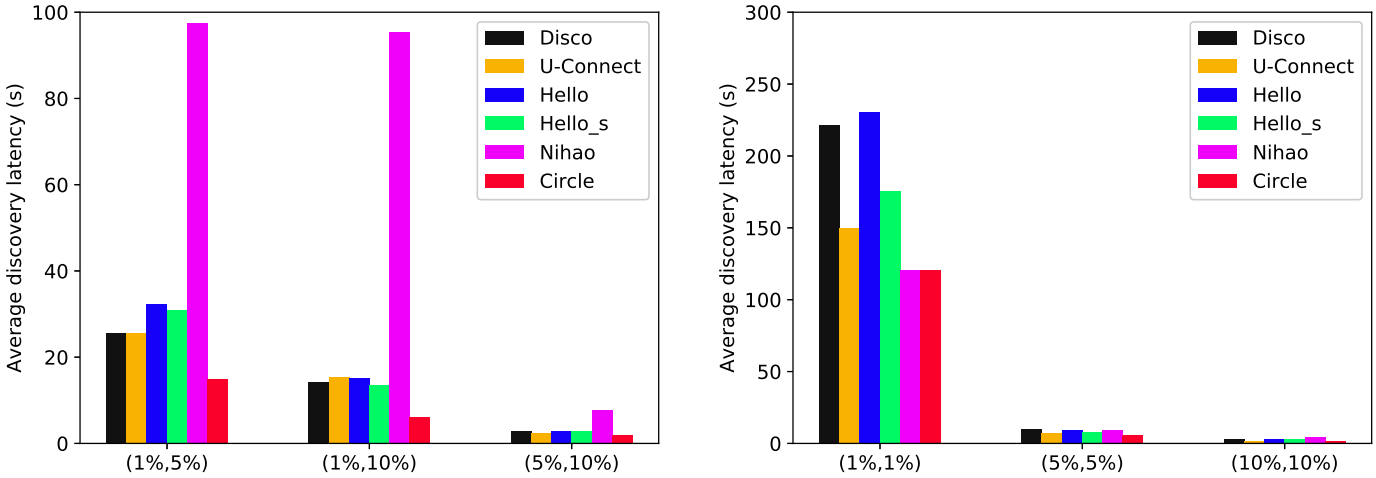


Fig. 10. Average discovery latency for all experimental duty cycle pairs.



(a) Average latency for asymmetric duty cycle pairs (1%, 5%), (1%, 10%), and (5%, 10%)

(b) Average latency for symmetric duty cycle pairs (1%, 1%), (5%, 5%), and (10%, 10%)

Fig. 11. Average latency for asymmetric and symmetric neighbor discovery.

know the duty cycle of node A, denoted as DC_A.[7] Consider another node, say node B, with cycle length ℓ_B. If $\ell_B \neq \ell_A$, then the greatest common divisor of the cycle lengths ℓ_A and ℓ_B is $\hat{\omega}$. In other words, one cycle length cannot be a multiple of the other cycle length. Therefore, even if some of node A's beacons may be collided by node B's beacons, other beacons of node A are not collided, referred to as *occasional collisions*. With occasional collisions, discovery of the target node (node A) still occurs (for example, when node A transmits a beacon while node B does not transmit).

The worst case of collisions happens when the two nodes have the same cycle length (i.e., $\ell_B = \ell_A$) and the time difference between their beacon transmissions in a cycle is smaller than the length of a beacon (τ), referred to as *persistent collisions* between node A and node B. Next we analyze the probability that the target node (node A) has persistent collisions with node B, denoted as $P_{\text{A\_by\_B}}$.

Denote φ_A and φ_B as the initial phases of node A's cycle and node B's cycle, respectively, when they enter communication range of each other. Then we have

$$P_{\text{A\_by\_B}} = \mathbb{P}\{\ell_B = \ell_A\} \cdot \mathbb{P}\{-\tau < \varphi_B - \varphi_A < \tau \big| \ell_B = \ell_A\}, \tag{15}$$

in which $\mathbb{P}\{\cdot\}$ represents probability of an event.

We have $\mathbb{P}\{\ell_B = \ell_A\} = \frac{1}{m}$, where m is the number of cycle lengths available for choosing. Next we calculate $\mathbb{P}\{-\tau < \varphi_B - \varphi_A < \tau \big| \ell_B = \ell_A\}$. If node B selects the same cycle length as that of node A, the initial phases of node A and node B are independent and uniformly distributed in $[0, \ell_A]$. Then, $\mathbb{P}\{-\tau < \varphi_B - \varphi_A < \tau \big| \ell_B = \ell_A\}$ can be calculated as the ratio of the area of the shaded region as shown in Fig. 12 to ℓ_A^2. It follows

$$\mathbb{P}\{-\tau < \varphi_B - \varphi_A < \tau \big| \ell_B = \ell_A\} = \frac{2\ell_A\tau - \tau^2}{\ell_A^2}. \tag{16}$$

Thus, we have

$$P_{\text{A\_by\_B}} = \frac{2\ell_A\tau - \tau^2}{m\ell_A^2}. \tag{17}$$

If node A has n neighbor nodes, then each of the n neighbor nodes persistently collides with node A with a probability equal to that in (17). Therefore, node A's overall probability of persistent collisions is given as

$$1 - (1 - P_{\text{A\_by\_B}})^n = 1 - \left(1 - \frac{2\ell_A\tau - \tau^2}{m\ell_A^2}\right)^n. \tag{18}$$

Fig. 13 shows the persistent collision probability of the target node (node A) when its duty cycle DC_A is 1%, 5%, or 10% and the number of neighbor nodes (n) varies from 1 to 200. Here we set τ as 1 ms and m (the number of cycle lengths available for choosing) as 10. The available duty cycles are 1%, 2%, ..., 10%. For each combination of (DC_A, n), the simulation result is averaged over 10^6 simulation runs. The analytical results in Fig. 13 are calculated using (18). It can be seen that the analytical and simulation results match with each other. When the target node (node A) has duty cycle of 1%, its probability of persistent collisions is low, for example,

[7]In Circle, each cycle length is corresponding to a duty cycle.

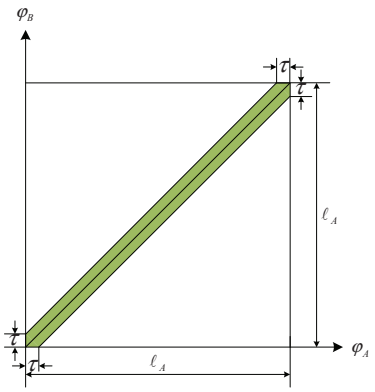


Fig. 12. The calculation of $\mathbb{P}\{-\tau < \varphi_B - \varphi_A < \tau \big| \ell_B = \ell_A\}$.

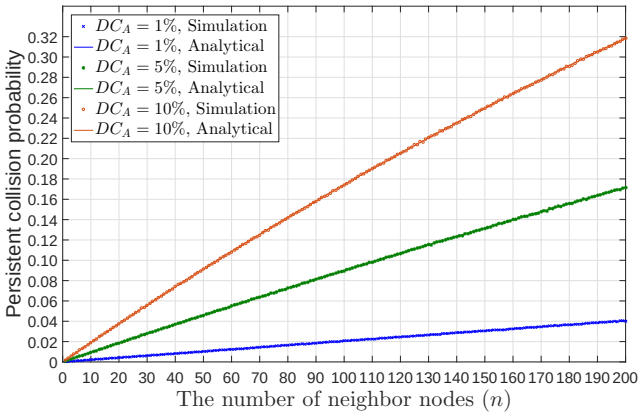


Fig. 13. Persistent collision probability of the target node (node A).

4% even with 200 neighbor nodes. When node A's duty cycle changes to 5% and 10%, the probability of persistent collisions increases to about 17% and 32%, respectively. Note that here we only adopt 10 cycle lengths. In fact, the number of cycle lengths could be much larger than 10. For example, if we use the set of cycle lengths L constructed in Section V-A, then we have $m = 49$, and node A's persistent collision probability for $(DC_A = 5\%, n = 200)$ and $(DC_A = 10\%, n = 200)$ will be reduced to no more than 4% and 8%, respectively. As observed, the target node's probability of persistent collisions increases with its duty cycle and the number of neighbor nodes. In case the persistent collision probability is above an acceptable level, the target node may dynamically reduce its duty cycle when it is aware that a large number of neighbor nodes are around. Further, as persistent collision is also an issue in almost all existing NDPs, the target node in Circle can apply the methods used in existing NDPs to deal with persistent collisions, such as performing a clear channel assessment (CCA) before transmitting as suggested in [14], or cooperating with underlying MAC protocols to avoid collisions as suggested in [8].

VII. Conclusion

In this paper, we present Circle model to characterize the process of neighbor discovery for the Internet of Things. Based on this model, we give a necessary and sufficient condition

for neighbor discovery and analyze the worst-case discovery latency. Circle model is a generic analytical model because it can be used to explain and analyze existing well-known NDPs. Further, based on this model, we propose an NDP also called Circle. The basic working mode of Circle is very simple: each node periodically wakes up, sends a beacon, and listens to channel for a while. Once a beacon is received, an ACK is sent. Experimental results show that Circle is superior to existing state-of-the-art NDPs.

Limitation of Circle: Compared to existing NDPs, one limitation of Circle is that its parameters, such as τ (length of a beacon), ω (length of a listening period), and ℓ (cycle length), should be integer values (i.e., each should be an integer number of time units).

Extendability of Circle: Circle can be extendable to some new wireless technologies such as WiFi Neighbor Awareness Networking (NAN) [31]. NAN enables a wireless device to continuously find, in an energy-efficient fashion, available devices and available services in its neighborhood. The NAN stack consists of the discovery engine (DE) and the NAN MAC, and the main task of DE can be implemented by Circle.

Applicability of Circle: Circle can be applied to discover IoT devices (e.g., sensors, actuators, smartphones, tablets, etc.) with the same wireless interface (e.g., ZigBee, Wi-Fi, or BLE). For IoT devices with different wireless interfaces, Circle is still applicable when IoT devices are equipped with multiple wireless interfaces and one wireless interface is in common for all IoT devices. For example, future smartphones will not only have Wi-Fi and BLE, but also ZigBee [32]. If it is impossible to have a common wireless interface among all involved IoT devices, a feasible solution is to partition the devices into different groups, and each group has a common wireless interface. Circle can be implemented in each group. In each group, a group leader is elected, which has multiple wireless interfaces. So the group leader can use other wireless interfaces to communicate with leaders of other groups to exchange node discovery information.

Future work: Future research topics include cooperative neighbor discovery, where multiple IoT devices are organized to discover new neighbors in a collaborative manner. Compared to pairwise neighbor discovery, cooperative neighbor discovery expects better performance in terms of shorter discovery latency as well as lower duty cycle. Moreover, cooperative neighbor discovery naturally fits with the cluster structure adopted by some new wireless technologies such as NAN, in which the devices within a cluster work cooperatively to discover new devices to be included into the cluster.

VIII. Acknowledgements

The authors would like to thank Dr. Mingwu Yao, Mr. Jun Li, Mr. Yaozhong Ma, Mr. Wen Li, and Mr. Chengcheng Gu at Xidian University, Xi'an, China for their valuable discussions and suggestion.

References

[1] J. Lin, W. Yu, N. Zhang, X. Yang, H. Zhang, and W. Zhao, "A survey on Internet of things: Architecture, enabling technologies, security and privacy, and applications," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1125–1142, Oct. 2017.

[2] Q. Ye and W. Zhuang, "Distributed and adaptive medium access control for Internet-of-things-enabled mobile networks," *IEEE Internet Things J.*, vol. 4, no. 2, pp. 446–460, Apr. 2017.

[3] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, Feb. 2014.

[4] A. Kamilaris and A. Pitsillides, "Mobile phone computing and the Internet of things: A survey," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 885–898, Dec. 2016.

[5] X. Hu, T. H. S. Chu, V. C. M. Leung, E. C.-H. Ngai, P. Kruchten, and H. C. B. Chan, "A survey on mobile social networks: Applications, platforms, system architectures, and future research directions," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1557–1581, 3rd Quart., 2015.

[6] R. Pozza, M. Nati, S. Georgoulas, K. Moessner, and A. Gluhak, "Neighbor discovery for opportunistic networking in Internet of things scenarios: A survey," *IEEE Access*, vol. 3, pp. 1101–1131, 2015.

[7] V. Galluzzi and T. Herman, "Survey: Discovery in wireless sensor networks," *Int. J. Distrib. Sens. Networks*, vol. 8, no. 1, Article ID 271860, Jan. 2012.

[8] W. Sun, Z. Yang, X. Zhang, and Y. Liu, "Energy-efficient neighbor discovery in mobile ad hoc and wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 3, pp. 1448–1459, 3rd Quart., 2014.

[9] M. J. McGlynn and S. a. Borbash, "Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks," in *Proc. 2nd ACM Int. Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, pp. 137–145, 2001.

[10] Y. Tseng, C. Hsu, and T. Hsieh, "Power-saving protocols for IEEE 802.11-based multi-hop ad hoc networks," in *Proc. IEEE INFOCOM*, pp. 200–209, 2002.

[11] P. Dutta and D. Culler, "Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications," in *Proc. 6th Int. Conf. Embedded Networked Sensor Systems (SenSys)*, pp. 71–83, 2008.

[12] M. Bakht, M. Trower, and R. H. Kravets, "Searchlight: Won't you be my neighbor?," in *Proc. ACM MobiCom 2012*, pp. 185–196.

[13] R. Zheng, J. C. Hou, and L. Sha, "Optimal block design for asynchronous wake-up schedules and its applications in multihop wireless networks," *IEEE Trans. Mobile Comput.*, vol. 5, no. 9, pp. 1228–1241, Sep. 2006.

[14] A. Kandhalu, K. Lakshmanan, and R. R. Rajkumar, "U-Connect: A low-latency energy-efficient asynchronous neighbor discovery protocol," in *Proc. Int. Conf. Information Processing in Sensor Networks (IPSN) 2010*, pp. 350–361.

[15] Y. Qiu, S. Li, X. Xu, and Z. Li, "Talk more listen less: Energy-efficient neighbor discovery in wireless sensor networks," in *Proc. IEEE INFOCOM*, pp. 1–9, 2016.

[16] L. Chen, R. Fan, K. Bian, M. Gerla, T. Wang, and X. Li, "On heterogeneous neighbor discovery in wireless sensor networks," in *Proc. IEEE INFOCOM*, pp. 693–701, 2015.

[17] L. Wei, B. Zhou, X. Ma, D. Chen, J. Zhang, J. Peng, Q. Luo, L. Sun, D. Li, and L. Chen, "Lightning: A high-efficient neighbor discovery protocol for low duty cycle WSNs," *IEEE Commun. Lett.*, vol. 20, no. 5, pp. 966–969, May 2016.

[18] W. Sun, Z. Yang, K. Wang, and Y. Liu, "Hello: A generic flexible protocol for neighbor discovery," in *Proc. IEEE INFOCOM*, pp. 540–548, 2014.

[19] Bluetooth find me profile specification. (2011). [Online]. Available at: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=239389

[20] Bluetooth SIG. Specification of the Bluetooth System. 02 December 2014. Core Version 4.2. [Online]. (Available: https://www.bluetooth.com/specifications/bluetooth-core-specification/legacy-specifications)

[21] J. Liu, C. Chen, and Y. Ma, "Modeling neighbor discovery in Bluetooth low energy networks," *IEEE Commun. Lett.*, vol. 16, no. 9, pp. 1439–1441, Sep. 2012.

[22] W. S. Jeon, M. H. Dwijaksara, and D. G. Jeong, "Performance analysis of neighbor discovery process in Bluetooth low-energy networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1865–1871, Feb. 2017.

[23] R. Cohen and B. Kapchits, "Continuous neighbor discovery in asynchronous sensor networks," *IEEE/ACM Trans. Netw.*, vol. 19, no. 1, pp. 69–79, Feb. 2011.

[24] B. Han, J. Li, and A. Srinivasan, "On the energy efficiency of device discovery in mobile opportunistic networks: A systematic approach," *IEEE Trans. Mobile Comput.*, vol. 14, no. 4, pp. 786–799, Apr. 2015.

[25] D. Zhang, T. He, Y. Liu, Y. Gu, F. Ye, R. K. Ganti, and H. Lei, "Acc: Generic on-demand accelerations for neighbor discovery in mobile

applications," in *Proc. 10th Int. Conf. Embedded Networked Sensor Systems (SenSys)*, pp. 169–182, 2012.
[26] D. Zhang, T. He, F. Ye, R. K. Ganti, and H. Lei, "Neighbor discovery and rendezvous maintenance with extended quorum systems for mobile applications," *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 1967–1980, Jul. 2017.
[27] Y. Sun, O. Gurewitz, and D. B. Johnson, "RI-MAC: A receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks," in *Proc. 6th Int. Conf. Embedded Networked Sensor Systems (SenSys)*, pp. 1–14, 2008.
[28] L. Tang, Y. Sun, O. Gurewitz, and D. B. Johnson, "PW-MAC: An energy-efficient predictive-wakeup MAC protocol for wireless sensor networks," in *Proc. IEEE INFOCOM*, pp. 1305–1313, 2011.
[29] M. B. Nathanson, *Elementary Methods in Number Theory*. Springer-Verlag New York, 2000.
[30] I. Niven, H. S. Zuckerman, and H. L. Montgomery, *An Introduction to the Theory Of Numbers*. Wiley, 5th Edition, 1991.
[31] D. Camps-Mur, E. Garcia-Villegas, E. Lopez-Aguilera, P. Loureiro, P. Lambert, and A. Raissinia, "Enabling always on service discovery: Wifi neighbor awareness networking," *IEEE Wireless Commun.*, vol. 22, no. 2, pp. 118–125, Apr. 2015.
[32] H. Qin and W. Zhang, "ZigBee-assisted power saving management for mobile devices," *IEEE Trans. Mobile Comput.*, vol. 13, no. 12, pp. 2933–2947, Dec. 2014.

Zhong Shen (M'11) received the B.E. degree from the University of Shanghai for Science and Technology, Shanghai, China, in 1992, and the M.E. degree in computer science and the Ph.D. degree in information and communication engineering from Xidian University, Xi'an, China, in 2002 and 2006, respectively. He is currently an Associate Professor with the School of Telecommunications Engineering, Xidian University. His research interests include wireless sensor networks and mobile computing.

Hai Jiang (SM'15) received the Ph.D. degree in electrical engineering from the University of Waterloo, Waterloo, Ontario, Canada, in 2006. Since July 2007, he has been a faculty member with the University of Alberta, Edmonton, Alberta, Canada, where he is currently a Professor at the Department of Electrical and Computer Engineering. His research interests include radio resource management, cognitive radio networking, and cooperative communications.

Qingkuan Dong received the B.S.degree in communications engineering from Xidian University in 1998. After that, He studied for the M.S. degree in cryptology in Xidian University. In 2000, he began to work for the Ph.D. in cryptology in Xidian University and received the Ph.D.degree in 2004. From 2004 to 2005, he worked in the post-doctoral research center of Institute of Software Chinese Academy of Sciences. He is currently an associate Professor with the state key laboratory of integrated services networks, Xidian University. His research interests include Internet of Things security, cryptology and trusted networks.

Baocang Wang received the B.S. and M.S. degrees in mathematics, and the Ph.D. degree in cryptography from Xidian University in 2006, 2004, and 2001, respectively. He is currently a professor with the School of Telecommunications Engineering in Xidian University, Cryptographic Research Center in Xidian University and School of Information Engineering in Xuchang University respectively. His main research interests include public key cryptography, wireless network security, fully homomorphic encryption and data mining.