2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)

A General Framework for Adjustable Neighbor Discovery in Wireless Sensor Networks

Zhaoquan Gu*, Yuexuan Wang†, Keke Tang*, Chao Li*, Mohan Li* and Lihua Yin*
*Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou, China
†College of Computer Science and Technology, Zhejiang University, Hangzhou, China

***Abstract*—Wireless sensor networks have been widely adopted in real-life applications. As one of the fundamental processes in constructing the network, neighbor discovery is to find out the existence of the nearby nodes. In order to enlarge the lifetime of each node, the nodes switch the radio OFF for most of the time, and only turn the radio ON for necessary communications. The fraction of time that the radio in ON is called duty cycle and low duty cycle schedule could help save energy. Most works focus on designing efficient discovery schedule under a pre-defined low duty cycle such that the neighboring nodes can discover each other in a short time. In this paper, we study a more practical problem where each node could adjust the duty cycle dynamically by the remaining energy or the subsequent tasks, which is referred to as adjustable neighbor discovery. We first propose a general framework to handle the problem, then present two distributed algorithms that can ensure the discovery between the neighboring nodes, no matter when they start and what duty cycle they choose. We also conduct simulations to evaluate the algorithms and the results corroborate our analyses.**

I. Introduction

With the realm of the Internet-of-Things [1], wireless sensor networks have been widely adopted in tracking and monitoring in various applications, such as health-care, smart buildings, agricultural management and assisted living. For example, the sensor network was deployed for agriculture information monitoring in [18], and the sensors are attached to the inventory items in a large warehouse for object identifying in [11].

As one fundamental process in constructing the wireless network, neighbor discovery, where the sensor could find the existence of the neighboring nodes within communication range, has drawn much attention in the last decade and has been extensively studied in wireless network literature [2], [4]–[6], [8]–[12], [14]–[17], [19], [20], [22], [23]. Since the sensor nodes are powered by limited battery and keeping consecutive wireless radio communication is quite costly, numerous works propose the idea of turning the sensor's radio "OFF" for most of the time, and switching the radio "ON" for sufficient communication only when necessary. The fraction of time that the radio is on is called *duty cycle* and most works focus on designing low duty cycle schedule to save energy.

There are two types of extant neighbor discovery protocols. Probability-based protocols select the state of radio as on or off with different probabilities, such as Birthday protocol [12]. These approaches can make the duty-cycle of the sensors quite small but they cannot guarantee the discovery between the neighboring sensors [17], [22]. Deterministic discovery protocols can ensure the discovery between the neighboring

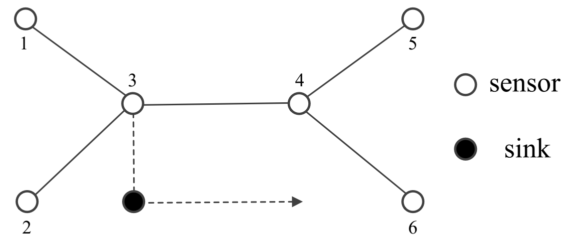

Fig. 1. A motivation example: the sensors will adjust the duty cycle dynamically

nodes within a determined time bound by designing a fixed discovery schedule. Quorum [16], Disco [6], U-Connect [9], SearchLight [2], BlindData [19], Hello [15], Hedis and Todis [4], Panda [11], TMLL [14] are some representative protocols and they can be adopted in many energy-efficient applications. However, most works design the schedule of the radio's state (on or off) to satisfy a pre-defined duty cycle for each sensor node, for example $1\%, 5\%$, they do not consider the scenario that each sensor may adjust its duty cycle dynamically as it may execute multiple energy-intensive tasks or it may reduce the communication when the interference is large. Therefore, we study the adjustable neighbor discovery problem in this paper, where the sensors could adjust the duty cycle locally on the basis of different energy status and different application requirements, and propose a general framework for the problem.

Considering a motivation example in Fig. 1, there are 6 sensors and one sink node in the network. The sensors can construct the network topology after initialization (discovering the neighbors) and they can communicate to exchange the sensed information. Suppose the sink node tries to collect all information of the sensors, it can directly connect to node 3 and the other sensors would send the information to node 3. Therefore, node 3 would spend large time collecting and transmitting information. When the sink node moves along the arrow, it may collect data through node 4 later and node 3 can reduce its duty cycle to save energy, while node 4 has to increase the duty cycle for discovering the sink node in a short time and exchanging the sensed information. Therefore, we assume the sensors can adjust the duty cycle dynamically and the adjustable neighbor discovery has three main advantages:

1) Wireless sensor network suffers from the energy hole problem when some nodes die quickly due to the massive

978-1-7281-2980-8/19/$31.00 ©2019 IEEE
DOI 10.1109/iThings/GreenCom/CPSCom/SmartData.2019.00043

communication, such as node $3, 4$ in Fig. 1. If the sensors can adjust the duty cycle dynamically, they can reduce the duty cycle to enlarge the lifetime;
2) Collecting information through mobile sink nodes gains much attention [7], [21], it is reasonable that the sink nodes increase the duty cycle to collect sensitive information from nearby sensors quickly, while reduce the duty cycle when the number of sensors is small or the area's information is less important;
3) When new nodes are deployed, they should find the neighbors quickly to connect to the network by utilizing high duty cycle schedule, while remaining low duty cycle to maintain the neighbors' existence afterwards.

In solving the adjustable neighbor discovery problem, there are several challenges. First, each sensor can choose different duty cycles at different time, the algorithms have to guarantee rendezvous no matter what duty cycles they choose. Second, the sensors may start or be activated asynchronously (at different time), such as due to the delay of sensor deployment and different activation signals, they should discover the nearby nodes quickly. Third, all sensors are distributed arbitrarily in the monitoring area, there is no central controller informing them about the other sensors' information. Distributed algorithms are thus preferable. In this paper, we address all these issues and the contributions are threefold.

1) We initiate and formulate the adjustable neighbor discovery problem where the sensors can adjust the duty cycles dynamically;
2) We present a general framework for the adjustable neighbor discovery and propose two distributed algorithms: prime set based algorithm and quorum based algorithm. The algorithms can guarantee discovery latency in a bounded short time;
3) We conduct simulations to evaluate our algorithms and these results corroborate our analyses.

The remainder of the paper is organized as follows. We present the framework for the adjustable neighbor discovery and formulate the problem in Section II. We propose two different neighbor discovery algorithms in Section III and we show the simulations results in Section IV. Finally, we conclude the paper in Section V.

II. Adjustable Neighbor Discovery Framework

We present the framework for the adjustable neighbor discovery in Fig. 2. As shown in the figure, the control layer is in charge of the node's all activities. The energy unit monitors the energy supply level and can inform the control unit in case of emergencies. The application layer includes various sensor activities, such as sensing the environment, or moving if it has such capability. The network layer contains the detailed network operations. For example, initialization is to construct the communication link with nearby nodes, and routing is to forward the information in an efficient way.

The adjustable neighbor discovery works as follows. When the application layer increases (or decreases) its activity, or the

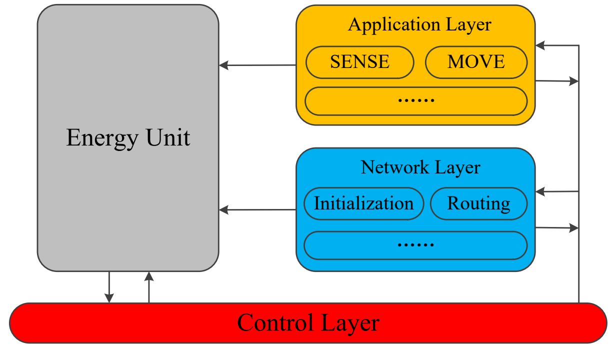

Fig. 2. Framework for adjustable neighbor discovery

energy unit needs to save (or spend) energy, the control unit would receive signal of changing the duty cycle. Then, the control layer adjusts the schedule of being awake and inform the network layer. Afterwards, the initialization is invoked and the node tries to discover the neighboring nodes. The neighbor discovery process and other activities lead to the change of energy, which is in turn monitored by the energy unit. The detailed process and the system model are described in the following parts of the section.

A. System Model

Considering a wireless sensor network which consists of N sensor nodes that are deployed in a monitoring area arbitrarily. Denote the nodes as $\{u_1, u_2, \ldots, u_N\}$ and they do not know the existence of others.

For each node u_i, denote the energy supply at time t as $P_i(t)$ and it is bounded by P_{max} since the power is limited by the carrying battery. Suppose the node has four states $\{OFF, SENSE, DETECT, SEND\}$, where OFF means it turns its radio off to save energy, $SENSE$ means it senses the information such as the temperature or humidity, $DETECT$ means it tries to discover the neighboring nodes, and $SEND$ means it sends the sensed data to the neighbors or sink nodes.[1] We denote the state of node u_i at time t as $s_i(t)$. Denote the consumed energy for each state as p_f, p_s, p_d, p_{sd} respectively. p_f is very small when the radio is off, p_s is a fixed constant according to the sensing information, p_d and p_{sd} could be much larger and they are determined be the length of sending packages. Generally, we assume $p_d < p_{sd}$ since the sensor could send very short messages to detect the neighbors. In this paper, we mainly focus on the neighbor discovery process and we simplify four states to $\{OFF, ON\}$.[2]

Denote the communication range of the node is d_c and two nodes are neighbors if their distance is less than d_c. In fact, one node can communicate with another is determined by many factors, such as the environment noise, the sending

[1] In the framework, there may be more states, such as MOVE, ROUTE, we do not consider them in the paper.

[2] We can regard state $SEND$ and $DETECT$ as ON since the radio is turned on and the sensors can communicate through the radio; we can regard state $SENSE$ as OFF since the consumed energy is relatively small compared with $DETECT$ and $SEND$

TABLE I
NOTATIONS FOR ADJUSTABLE NEIGHBOR DISCOVERY

Notation	Description
N	Number of sensor nodes
u_i	Sensor node u_i
$P_i(t)$	The remaining energy for u_i at time t
P_{max}	The maximum energy for each sensor
$s_i(t)$	The state of u_i at time t
$\{ON, OFF\}$	The simplified states of the sensor node
d_c	Communication range of each sensor
t_0	The length of each time slot
S_i	Discovery schedule of sensor u_i
δ_i	The start time of sensor u_i
θ_i	Duty cycle of sensor u_i
$L(i, j)$	Discovery latency between u_i and u_j
Δ	The length of a discovery period
$\{\theta(1), \theta(2), \ldots, \theta(m)\}$	Different duty cycle levels

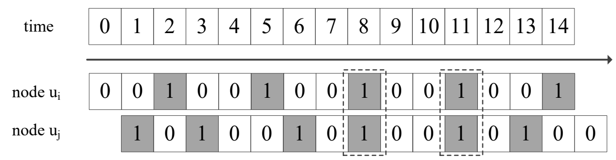

Fig. 3. Neighbor discovery example when $\Delta = 15$

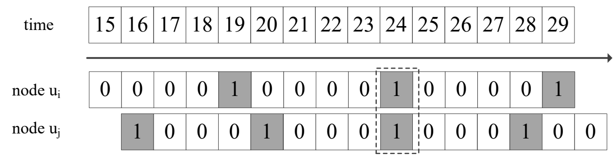

Fig. 4. Neighbor discovery example when $\Delta = 15$

power energy, and the path-loss exponent during communication, we simplify the process and suppose two sensors can communicate if they are within the range d_c. Suppose time is divided into slots of equal length t_0 and the sensor can set state $s_i(t)$ in time t. We summarize the notations in Table I.

B. Adjustable Neighbor Discovery

We first define the **neighbor discovery problem** as:

Problem 1: Design the neighbor discovery schedule $S_i = \{s_i(t)|t \geq 0\}$ for node u_i where:

$$s_i(t) = \begin{cases} 0 & if\ u_i\ chooses\ state\ OFF \\ 1 & if\ u_i\ chooses\ state\ ON \end{cases}$$

For two neighboring nodes u_i and u_j, denote the start time of u_i and u_j as δ_i and δ_j respectively, there exits T such that:

$$s_i(T) = s_j(T) = 1$$

Notice that, the nodes can start asynchronously which implies they could have different start time. In the problem definition, node u_i starts at time δ_i while u_j starts at slot δ_j. The discovery latency between them (denoted as $L(i, j)$) is defined as:

$$L(i, j) = T - \max\{\delta_i, \delta_j\} \tag{1}$$

which denotes the latency to discover each other once they have all already started.

We define the **duty cycle** of the discovery schedule as:

Definition 2.1: The duty cycle of schedule $S_i = \{s_i(t)| t \geq 0\}$ for node u_i is the percentage of time slots that u_i chooses state ON:

$$\theta_i = \frac{|\{s_i(t) = 1|0 \leq t < T_i\}|}{T_i}$$

In our framework, we suppose each node u_i would adjust its discovery schedule every Δ time slots according to the energy unit or application requirements, where Δ could be a large constant such as one day or one week. There are two main reasons. First, *neighbor discover process should be executed many times*. In many extant works, neighbor discovery is only conducted once to initialize the network topology and they assume such information will remain afterwards. However, some sensors may die when the energy is exhausted, or some new sensors may be deployed, they lead to the change of the original network. Thus, the nodes have to conduct the neighbor discovery process to re-construct the network. Second, *the nodes can adjust the duty cycle dynamically*. For example, the nodes can reduce the duty cycle to save energy as time goes on, thus we need to adjust the schedule to satisfy the duty cycle. In this paper, we assume the nodes could update the neighbor discovery schedule periodically to simplify the process.

We assume there are m different duty cycle levels as $\Theta = \{\theta(1), \theta(2), \ldots, \theta(m)\}$ where $\theta(i) > \theta(j)$ if $i > j$. At the beginning of each period of length Δ, each node chooses a corresponding duty cycle level by the remaining energy and the subsequent tasks. We define the **adjustable neighbor discovery problem** as:

Problem 2: Suppose node u_i starts at time δ_i, denote the chosen duty cycle at time $\delta_i(d) = \delta_i + d \cdot \Delta$ $(d \geq 0)$ as $\theta_i(d) \in \Theta$, design the neighbor discovery schedule $S_i(d) = \{s_i(t)|\delta_i(d) \leq t < \delta_i(d+1)\}$ for node u_i where:

$$s_i(t) = \begin{cases} 0 & if\ u_i\ chooses\ state\ OFF \\ 1 & if\ u_i\ chooses\ state\ ON \end{cases}$$

For any neighboring node u_j starting at time δ_j, there exists $\delta_i(d) \leq T < \delta_i(d+1)$ such that:

$$s_i(T) = s_j(T) = 1$$

As stated in the problem, each node u_i could choose an arbitrary duty cycle level $\theta_i(d)$ at the beginning of the d-th period, and it can discover the neighboring node u_j no matter what duty cycle it chooses. The ultimate goal is to minimize the discovery latency between every pair of neighboring sensor nodes once any node changes the duty cycle level, no matter when the node changes and which duty cycle level is chosen. In this paper, we assume the nodes change the duty cycle periodically for simplification.

C. Adjustable Neighbor Discovery Examples

Suppose node u_i starts at slot 0, node u_j starts at slot 1, and the period length is $\Delta = 15$. Suppose u_i chooses $\theta_i(0) = 33.3\%, \theta_i(1) = 20\%$ and u_j chooses $\theta_j(0) = 40\%, \theta_j(1) = 26.7\%$ for the first two periods. As depicted in Fig. 3, u_i chooses state ON every 3 time slots, while u_j chooses ON twice every 5 time slots. They discover each other at slot 8 in the first period and the discovery latency is $8-1 = 7$. The duty cycle in the first period can be computed as $\theta_i(0) = \frac{5}{15} \approx 33\%$ and $\theta_j(0) = \frac{6}{15} = 40\%$. As shown in Fig. 4, u_i chooses state ON every 5 slots while u_j chooses ON every 4 time slots. They can discover each other at time slot 24 in the second period and the discovery latency is $24 - 15 = 9$ if we regard time slot 15 as the starting slot of the next period. The duty cycles are $\theta_i = \frac{3}{15} \approx 20\%$ and $\theta_j = \frac{4}{15} \approx 26.7\%$ and the schedules can guarantee the discovery between them.

III. Adjustable Neighbor Discovery Algorithms

In this section, we present two adjustable neighbor discovery algorithms that can be adopted in our framework. The first one utilizes multiple primes to construct the discovery schedule, while the other one uses the quorum system for schedule construction.

A. Prime Set Based Algorithm

For node u_i, suppose it chooses duty cycle $\theta' \in \Theta$ at the beginning of each period, the method of generating the discovery schedule is described as in Alg. 1.

Algorithm 1 Prime Set Based Algorithm

1: Denote time slot $t := 0$, set $P := \emptyset$, schedule S_i;
2: **while** not terminated **do**
3:   **if** $t\%\Delta = 0$ **then**
4:     Node u_i chooses a duty cycle level θ';
5:     Invoke the generating prime set process on θ', and the output is set P;
6:   **end if**
7:   Compute time slot in the period $t' := t\%\Delta$;
8:   **if** there exists $p \in P$ such that $t'\%p = 0$ **then**
9:     Set $s_i(t) := 1$;
10:   **else**
11:     Set $s_i(t) := 0$;
12:   **end if**
13:   Update time slot $t := t + 1$;
14: **end while**

Generating prime set of θ'

1: Denote set $P := \emptyset$, a small constant ϵ, $f := 0$;
2: **while** $\theta' - f > \epsilon$ or $|P| < 2$ **do**
3:   Find the smallest prime $p \leq \Delta$ such that $\frac{1}{p} < \theta' - f$;
4:   Add p to set P as: $P := P \bigcup \{p\}$;
5:   Update $f := f + \frac{1}{p}$;
6: **end while**

After choosing the duty cycle, a set of primes $P = \{p_1, p_2, \ldots, p_k\}$ ($p_a < p_b$ if $a < b$) is generated such that:

$$0 < \theta' - (\frac{1}{p_1} + \frac{1}{p_2} + \cdots + \frac{1}{p_k}) < \epsilon$$

where ϵ is a very small constant. Afterwards, node u_i will turn the radio on every $p \in P$ time slots from Lines 8-12. Obviously, the duty cycle of node u_i can be computed as:

$$\theta_i \approx \sum_{p \in P} \frac{1}{p}$$

and it is very close to the chosen duty cycle θ'. Notice that, we use "$\approx$" because p may not be a divisor of Δ.

For example, suppose Δ is very large and $\epsilon = 0.000001$, when node u_i chooses duty cycle 5%, the generated prime set is: $P = \{23, 157, 6569\}$; when u_i chooses duty cycle 1%, the prime set is $\{101, 10103\}$; when u_i chooses duty cycle 0.5%, the prime set is $\{211, 3847\}$.

For any two neighboring nodes u_i, u_j, denote the duty cycles as θ'_i, θ'_j, and the corresponding prime sets as $P_i = \{p_i(1), p_i(2), \ldots, p_i(k_i)\}$, $P_j = \{p_j(1), p_j(2), \ldots, p_j(k_j)\}$ respectively. The discovery can be guaranteed as the following lemma. Since $p_i(1)$ cannot be equal to both $p_j(1)$ and $p_j(2)$, we may apply the Chinese Remainder Theorem [13] to guarantee that the nodes can discover each other in a short time.

Lemma 3.1: Two neighboring nodes u_i, u_j can discover each other within $L(i,j)$ time slots where:

$$L(i,j) \leq \min\{p_i(1) \cdot p_j(2), p_j(1) \cdot p_i(2)\}$$

Proof: Considering the two smallest primes $p_i(1), p_i(2)$ and $p_j(1), p_j(2)$ in each prime set, $p_i(1)$ cannot be equal to both $p_j(1)$ and $p_j(2)$. Suppose $p_i(1) \neq p_j(2)$, from Alg. 1, node u_i sets $s_i(t) = 1$ if there exists prime $p \in P_i$ such that $t\%p = 0$. Therefore, both nodes turn on the radios in time slot T (from u_j's local time view) if the following equalities hold:

$$\begin{cases} T + \delta_{ij} \equiv 0 \mod p_i(1) \\ T \equiv 0 \mod p_j(2) \end{cases}$$

According to the Chinese Remainder Theorem [13], such $T \in [0, p_i(1)p_j(2))$ exists. Thus, the discovery latency $L(i,j)$ is bounded by $p_i(1) \cdot p_j(2)$. Similarly, $L(i,j)$ is also bounded by $p_j(1) \cdot p_i(2)$ time slots. Thus, the lemma holds. ■

Therefore, we can conclude the theorem as:

Theorem 1: The prime set based algorithm can guarantee efficient neighbor discovery no matter what duty cycle the nodes choose in each period.

B. Quorum Based Algorithm

First, we introduce the definition of quorum:

Definition 3.1: Given a finite set $U = \{0, 1, \ldots, n-1\}$, a *quorum system* S under U is a collection of non-empty subsets of U, which satisfies the intersection property:

$$p \bigcap q \neq \emptyset, \forall p, q \in S$$

Each element $p \in S$ is called a quorum.

Algorithm 2 Quorum Based Neighbor Discovery Algorithm

Denote time slot $t := 0$, schedule S_i, prime number p;
while not terminated **do**
  if $t\%\Delta = 0$ **then**
    Node u_i chooses a duty cycle level θ';
    Choose the smallest prime p such that $\frac{2p-1}{p^2} \le \theta'$;
    Choose value v as a random number in $[0, p)$;
  end if
  Compute the time slot in the period $t' := t\%\Delta$;
  if $0 \le t' < p$ or $(t' - v)\%p = 0$ **then**
    Set $s_i(t) := 1$;
  else
    Set $s_i(t) := 0$;
  end if
  Update time slot $t := t + 1$;
end while

0	1	...	v	...	p-1
p	p+1	...	p+v	...	2p-1
...	...	...	...	...	...
kp	kp+1	...	kp+v	...	2p-1
...	...	...	...	...	...
(p-1)p	..	...	(p-1)p+v	...	p^2-1

Fig. 5. An example of choosing the quorum

The quorum has been widely adopted in designing neighbor discovery algorithm [3], [4], and we show the method of extending it to the adjustable neighbor discovery problem.

As described in Alg. 2, node u_i chooses duty cycle θ' every Δ time slots, and it computes the corresponding prime p such that $\frac{2p-1}{p^2} \le \theta'$. This is because we choose the quorum as depicted in Fig. 5. Consider every p^2 time slots as a block, and we pick the first row and the v-th column such that the schedule of the corresponding time slots $s_i(t)$ is 1. Since the period length Δ is very large and node u_i repeats the block until the next period. Actually, we can compute the duty cycle value as:

$$\theta_i \approx \frac{2p-1}{p^2} \le \theta'$$

we use "$\approx$" since p^2 may not be a divisor of Δ.

For two neighboring nodes u_i and u_j, suppose the chosen primes are p_i, p_j and the start time are δ_i, δ_j ($\delta_i <= \delta_j$) respectively, the discovery can be guaranteed by the following lemmas. The main idea is: when $p_i \le p_j$, the discovery is guaranteed by the property of the quorum system. When $p_i > p_j$, we can apply the Chinese Remainder Theorem to guarantee discovery.

Lemma 3.2: Nodes u_i and u_j can discover each other within $L_1(i,j) \le p$ time slots if $p_i = p_j = p$.

Proof: When $p_i = p_j = p$, both users choose two quorums under set $\{0, 1, \ldots, p^2 - 1\}$, and they can discover each other within p^2 time slots by the property. Specifically, when u_j starts later, the schedule of the first p time slots are all 1, while there must exist one time slot t such that $s_i(t) = 1$ for u_i during any p consecutive time slots. Therefore, the discovery latency is bounded by p time slots. ∎

We can also deduce another two lemmas. Due to page limits, we omit the details.

Lemma 3.3: Nodes u_i and u_j can discover each other within $L_2(i,j) \le p_j$ time slots if $p_i < p_j$.

Lemma 3.4: Nodes u_i and u_j can discover each other within $L_3(i,j) \le p_j p_i$ time slots, if $p_i > p_j$.

Combining Lemmas 3.2-3.4, we conclude the theorem:

Theorem 2: The quorum based algorithm can guarantee efficient neighbor discovery no matter what duty cycle the nodes choose in each period.

Remark 3.1: Compared with the prime set based algorithm, the quorum based algorithm has better performance when $p_i \le p_j$. However, when $p_i > p_j$, the discovery latency is about four times that of the prime set based algorithm.

TABLE II
DISCOVERY LATENCY COMPARISON

θ *Alg.*	Prime Set Based Alg.		Quorum Based Alg.	
	Average	Maximum	Average	Maximum
0.01	329352	909271	107.77	208
0.03	3662.77	11122	33.22	67
0.05	1104.5	2827	20.73	41
0.1	471.6	1131	11.77	23
0.15	323.44	295	8.47	17
0.2	35.64	115	5.52	11

IV. SIMULATIONS

We implement the proposed framework by tool NS3 and we present the comparison of the proposed algorithms between two neighboring nodes. We set the length of each time slot as $t_0 = 20ms$ and the length of each period $\Delta = 4320000$ which corresponds to one day. We set 20 levels of duty cycle as: $\Theta = \{\theta(k) = 0.01k | 1 \le k \le 20\}$ and each node chooses one duty cycle in each period. The start time of each node is generated randomly.

To begin with, we assume two nodes u_1, u_2 choose the same duty and we compute both the average and maximum latency of the algorithms after we repeat each period 100 times. As listed in Table II, when the duty cycle increases, the average and maximum discovery latency of both algorithms decrease. The quorum based algorithm has better performance than the prime set based algorithm, this is because the nodes choose the same prime set under the same duty cycle, and they are to discover each other in $p_1(1) \cdot p_2(2)$ time slots, which is much larger than the discovery latency of the quorum based algorithm.

We also show the comparison in Fig. 6 when two nodes can choose different duty cycles. We set node u_1's duty cycle as

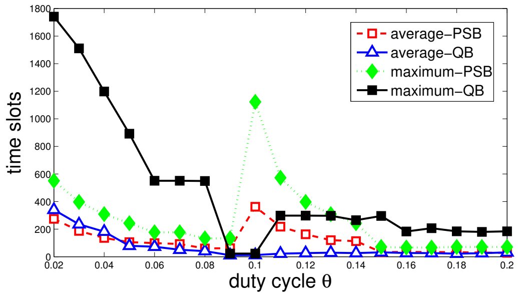

Fig. 6. The latency comparison of two algorithms when the nodes choose different duty cycles

$\theta_1 = \theta(10) = 10\%$ and node u_2 increases the duty cycle from $\theta(1) = 1\%$ to $\theta(20) = 20\%$. As shown in the figure (PSB is short for prime set based algorithm and QB is short for quorum based algorithm), when two nodes choose different duty cycles, the PSB algorithm has better performance than the QB algorithm. When node u_2 chooses the same duty cycle 0.1 as that of u_1, the discovery latency for the PSB algorithm is much larger. However, the discovery latency for the QB algorithm is smaller than the other situations. Regarding the maximum discovery latency, the QB algorithm is about 4 times of the PSB algorithm when the nodes choose different duty cycles. This is because the chosen prime number in the quorum system is about $p \approx \frac{2}{\theta}$, while the smallest prime in the prime set is about $p' \approx \frac{1}{\theta}$.

From the simulation results, the PSB algorithm has better performance when the nodes choose different duty cycles, while the QB algorithm performs better when the nodes have the same duty cycle. When the algorithms are applied to multiple nodes in the monitoring area, the difference with traditional neighbor discovery is that the interference occurs if multiple nodes send message simultaneously and this would lead to failure. We will handle this in our future works.

V. Conclusion

In this paper, we propose the adjustable neighbor discovery problem where the sensor nodes in the wireless network can adjust their duty cycle dynamically according to the remaining energy and the subsequent tasks. We propose a general framework to handle this problem where each node can change the duty cycle periodically and then initiate the neighbor discovery process to find the nearby nodes. Since all nodes are distributed arbitrarily in the monitoring area and they cannot know the others' information, such as the start time, and the duty cycle. We propose two distributed algorithms that can guarantee discovery between the neighboring nodes within a bounded latency, no matter when they start, and what duty cycle they choose. In the future, we are to implement the framework in real sensor networked applications and accommodate the extant neighbor discovery algorithms to the framework. In addition, we are to handle the scenario that multiple neighbors may communicate simultaneously and propose robust discovery algorithm in accordance with the interference.

Acknowledgement

This work is supported in part by the National Natural Science Foundation of China under Grant No. U1636215 and the National Key $R\&D$ Program of China 2018YEB1004003.

References

[1] L. Atzori, A. Lera, and G. Morabito. The Internet of Things: A Survey. *Computer Networks*, 54(15), pp. 2787-2805, 2010.
[2] M. Bakht, M. Trower, and R. H. Kravets. Searchlight: Won't you be my neighbor? In *MobiCom*, 2012.
[3] K. Bian, J.-M. Park, and R. Chen. A Quorum-Based Framework for Establishing Control Channels in Dynamic Spectrum Access Networks. In *Mobicom*, 2009.
[4] L. Chen, R. Fan, K. Bian, L. Chen, M. Gerla, T. Wang, and X. Li. On Heterogeneous Neighbor Discovery in Wireless Sensor Networks. In *INFOCOM*, 2015.
[5] J. Cui, L. Ren, X. Wang, and L. Zhang. Pairwise Comparison Learning based Bearing Health Quantitative Modeling and Its Application in Service Life Prediction. *Future Generation Computer Systems*, vol. 97, pp. 578-586, 2019.
[6] P. Dutta, and D. Culler. Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications. In *SenSys*, pp. 71-84, 2008.
[7] Z. Gu, Q.-S. Hua, Y. Wang, and F. C.M. Lau. Reducing Information Gathering Latency through Mobile Aerial Sensor Network. In *INFOCOM*, 2013.
[8] J.-R. Jiang, Y.-C. Tseng, C.-S. Hsu, and T.-H. Lai. Quorum-based Asynchronous Power-Saving Protocols for IEEE 802.11 Ad Hoc Networks, *Mobile Networks and Applications*, 10(1-2): 169-181, 2005.
[9] A. Kandhalu, K. Lakshmanan, and R. R. Rajkumar. U-Connect: A Lower Latency Energy-Efficient Asynchronous Neighbor Discovery Protocol. In *IPSN*, 2010.
[10] S. Lai, B. Ravindran, and H. Cho. Heterogeneous Quorum-based Wake-up Scheduling in Wireless Sensor Networks. *IEEE Transaction on Computers*, 59(11): 1562-1575, 2010.
[11] R. Margolies, G. Grebla, T. Chen, D. Rubenstein, and G. Zussman. Panda: Neighbor Discovery on a Power Harvesting Budget. In *INFOCOM*, 2016.
[12] M. J. McGlynn, and S. A. Borbash. Birthday Protocols for Low Energy Deployment and Flexible Neighbor Discovery in Ad Hoc Wireless Networks. In *MobiHoc*, 2001.
[13] M. B. Nathanson. Elementary Methods in Number Theory. Vol. 195, Springer, 2000.
[14] Talk More Listen Less: Energy-Efficient Neighbor Discovery in Wireless Sensor Networks. In *INFOCOM*, 2016.
[15] W. Sun, Z. Yang, X. Zhang, and Y. Liu. Hello: A Generic Flexible Protocol for Neighbor Discovery. In *INFOCOM*, 2014.
[16] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-Saving Protocols for IEEE 802.11-based Multi-Hop Ad Hoc Networks. In *Computer Networks*, 43(3), pp. 317-337, 2003.
[17] S. Vasudevan, D. Towsley, D. Goeckel, and R. Khalili. Neighbor Discovery in Wireless Networks and the Coupon Collector's Problem. In *MobiCom*, 2009.
[18] Y. Wang, Y. Wang, X. Qi, L. Xu, J. Chen, and G. Wang. L3SN: A Level-Based, Large-Scale, Longevous Sensor Network System for Agriculture Information Monitoring. In *Wireless Sensor Networks*, 2010.
[19] K. Wang, X. Mao, and Y. Liu. Blinddata: A Neighbor Discovery Protocol. In *TPDS*, 26(4), pp. 949-959, 2015.
[20] X. Wang, L. T. Yang, H. Liu, and M. Jamal Deen. A Big Data-as-a-Service Framework: State-of-the-Art and Perspectives. *IEEE Transactions on Big Data*, Vol. 4, No. 3, pp. 325-340, 2018.
[21] X. Xu, J. Luo, and Q. Zhang. Delay Tolerance Event Collections in Sensor Networks with Mobile Sink. In *INFOCOM*, 2010.
[22] W. Zeng, S. Vasudevan, X. Chen, B. Wang, A. Russel, and W. Wei. Neighbor Discovery in Wireless Networks with Multipacket Reception. In *MobiHoc*, 2011.
[23] H. Zhu, T. Xie, Y. Guan, L. Deng, and X. Wang. Hypergraph Matching with an Entropy Barrier Function. *IEEE Access*, Vol. 7, pp. 16638-16647.