



Article

A Low-Latency and Energy-Efficient Neighbor Discovery Algorithm for Wireless Sensor Networks †

Zhaoquan Gu ¹ , Zhen Cao ², Zhihong Tian ^{1,*} , Yuexuan Wang ^{3,*} and Xiaojiang Du ⁴ and Guizani Mohsen ⁵

¹ Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China; zqgu@gzhu.edu.cn

² Department of Computer Science, Rice University, Houston, TX 77025, USA; zhen.cao@rice.edu

³ College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China

⁴ Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122, USA; dxj@ieee.org

⁵ Computer Science and Engineering Department, Qatar University, Doha 2713, Qatar; mguizani@ieee.org

* Correspondence: tianzhihong@gzhu.edu.cn (Z.T.); amywang@zju.edu.cn (Y.W.)

† This paper is an extended version of our paper published in Cao, Z.; Gu, Z.; Wang, Y.; Cui, H. Panacea: A Low-Latency, Energy-Efficient Neighbor Discovery Protocol for Wireless Sensor Networks. In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 15–18 April 2018.

Received: 31 October 2019; Accepted: 21 January 2020; Published: 24 January 2020



Abstract: Wireless sensor networks have been widely adopted, and neighbor discovery is an essential step to construct the networks. Most existing studies on neighbor discovery are designed on the assumption that either all nodes are fully connected or only two nodes compose the network. However, networks are partially connected in reality: some nodes are within radio range of each other, while others are not. Low latency and energy efficiency are two common goals, which become even more challenging to achieve at the same time in partially connected networks. We find that the collision caused by simultaneous transmissions is the main obstruction of achieving the two goals. In this paper, we present an efficient algorithm called *Panacea* to address these challenges by alleviating collisions. To begin with, we design *Panacea*-NCD (*Panacea* no collision detection) for nodes that do not have a collision detection mechanism. When n is large, we show the discovery latency is bounded by $O(n \cdot \ln n)$ for any duty cycle (the percentage time to turn on the radio), where each node has n neighbors on average. For nodes that can detect collisions, we then present *Panacea*-WCD which also bounds the latency within $O(n \cdot \ln n)$ slots. Finally, we conduct extensive evaluations and the results also corroborate our analyses.

Keywords: neighbor discovery; wireless sensor networks; communication collision; discovery latency; duty cycle

1. Introduction

Wireless sensor networks have been widely adopted in a wide range of Internet-of-Things (IoT) applications such as smart cities [1], health-care systems [2,3], intelligent detection [4,5], and remote monitoring [6–8]. In general, a large number of sensor nodes would be dispatched or deployed in some monitoring area to collect or aggregate information for further intelligent decisions. With the fast development of sensor system and communication technology, these sensor nodes could be deployed in a distributed manner without any pre-defined network topology; they can compose a wireless network to exchange information with the help of advanced communication technologies, such as bluetooth, wifi, 5G, etc.

Neighbor discovery is a fundamental process of constructing the wireless network among the sensor nodes, where each sensor node could discover other sensors within its communication range. As a cornerstone and an essential step in configuring wireless sensor networks, neighbor discovery has been extensively studied in the past few years [9–22] and the main objective of these algorithms is to reduce the discovery latency for a specific node or the whole network.

The extant algorithms can be classified into two categories. The probabilistic methods exploit the probability theory to reduce the communication collision caused by multiple transmissions simultaneously. These methods could make sensor nodes discover the neighbors within an expected low latency, but they all assume the network is fully connected where any two nodes can communicate directly. However, a large number of sensor nodes would be deployed in a large area and these nodes cannot communicate directly when their distance is far away or some nodes are restricted due to extreme environment. The deterministic methods could guarantee the discovery process where the sensor nodes only turn on their radios for a short time, but these methods are applicable to two nodes since they do not consider the collisions that may exist among a large number of sensors. Hence, these methods cannot be adopted directly for a large number of sensors, as the collisions would affect the normal communication process.

In designing efficient neighbor discovery algorithms for wireless sensor networks in IoT applications, there are two main challenges. First of all, sensor nodes have limited battery since the size of the sensors is small and battery recharging would be costly in applications like agricultural monitoring or underwater detecting; the node could turn on its radio only for sufficient communication and keep the radio off to save energy. Second, some IoT applications need a large number of sensors that are distributed in a large area or in some hostile environment; the sensors can only compose a partially connected network in which some nodes are not within the radio range or some nodes can be detected with some probability due to the obstacle or the block in the environment. However, the extant works cannot solve the two challenges concurrently.

To save energy, sensor nodes switch their radios off for most of the time and only turn the radios on for necessary communication. The fraction of time a node turns its radio on is denoted as a *duty cycle*. Some existing algorithms transmit and listen with certain probabilities that are related to duty cycle [16,20,22], but they assume the network is fully connected, which is unrealistic in real IoT application. Some works focus on minimizing *discovery latency* [16,20], which is the time to discover all neighbors. However, they assume the nodes keep radios on all the time (duty cycle is 1) and these nodes would run out of energy quickly. Many deterministic protocols minimize discovery latency with a given duty cycle in [9,11,12,14,15,17,19]. Nevertheless, these methods only aim at two nodes, and they ignore the communication collisions among multiple nodes. By experiment, we find collisions caused by simultaneous transmissions result in a waste of time and energy. We considered a partially connected network with 1000 nodes and a probability of 0.5 for two nodes to be neighbors. We found that the collision slots of Hello [19], a deterministic protocol designed for two nodes, make up 99.8% of the time; while the collision slots of PND [18], a probabilistic protocol, make up 46.5% of the time. That means, because of collisions, existing works waste time and energy, and cannot achieve low latency and energy efficiency for the partially connected networks.

In this paper, we propose a low-latency and energy-efficient neighbor discovery algorithm called *Panacea* to address the above two challenges. Our algorithms take the sensors' duty cycle into consideration and adopt the collision detection mechanism to shorten the discovery latency. We summarize our contributions as follows:

- We propose the *Panacea* algorithm for a partially connected network which discovers all neighbors in $O(n \ln n)$ slots where n is the number of neighbors on average;
- With the collision detection hardware mechanism, we modify the algorithm, creating *Panacea-WCD*, which can alleviate the communication collisions;
- *Panacea* is suitable for both synchronous (i.e., all nodes are activated at the same time) and asynchronous (i.e., all nodes are activated at different time slots) scenarios.

We have conducted extensive simulations to evaluate the proposed algorithm. The results show that *Panacea* improves both latency and energy efficiency. For both synchronous and asynchronous scenarios, we analyze that the discovery latency can be bounded within $O(n \ln n)$ slots theoretically, and the simulation results also corroborate the analysis. Compared with Coupon [20] and PND [18], *Panacea* reduces the discovery latency by 60% and 92%, respectively, for synchronous scenarios, while reduces the discovery latency by 30% and 90% for asynchronous scenarios. *Panacea* could save 40% energy when it achieves similar discovery latency with Coupon. In addition, *Panacea* shows its best performance in regards to the trade-off between discovery latency and duty cycle, compared with Coupon, PND, Aloha-like [22], and Hello [19].

The rest of this paper is organized as follows. We review existing approaches in Section 2 and introduce the preliminaries in Section 3. Then, we propose the *Panacea* algorithm and analyze the efficiency in Section 4. With the collision detection mechanism, we propose the *Panacea*-WCD algorithm in Section 5 and present how the collision can be utilized in the algorithm. We implement *Panacea* and illustrate the comparison results in Section 6. Finally, we conclude this paper in Section 7.

2. Related Work

In recent years, wireless sensor networks have been extensively studied [1,3–8,23–28]; as a cornerstone of constructing wireless sensor networks, increasingly sophisticated protocols for neighbor discovery have been proposed. These neighbor-discovery protocols mainly fall into two categories. One category is the probabilistic methods, which is to exploit the probability theory to discover nodes within an expected time, statistically speaking. The other category is the deterministic methods, which is to utilize mathematical theorems to guarantee the discovery between two nodes.

In probabilistic algorithms, each node in the network transmits, listens, or sleeps with a certain probability at each time slot, and the sum of these three probabilities is 1. One of the traditional methods is Birthday protocols, seen in [16]. It utilizes random independent transmissions based on the birthday paradox (i.e., when there are 23 people, the probability that at least two have the same birthday exceeds 0.5), and saved a great deal of energy with a high expected proportion of discovered neighbors. This method was further studied as an Aloha-like algorithm in [20], based on the classical coupon collector's problem. This algorithm first introduced the reception feedback with a collision detection mechanism, and mathematically showed that discovery latency was bounded in both scenarios, when nodes in the network are with or without a collision detection mechanism. Later on, a detailed physical layer mechanism was proposed in [29] for how nodes in receive mode detect the channel status, and methods at higher layers were also described based on the reception status information at transmitters. However, these methods only focused on improving the proportion of discovered neighboring nodes, ignoring the significance of saving energy.

Following that, similar and nicer probabilistic algorithms with a pre-defined duty cycle were proposed to save energy in [18,22]. However, these two methods only considered unrealistic fully connected networks, where every two nodes in the network are ideally connected. Besides, sophisticated hardware tools were used to reach a good tradeoff between latency and duty cycle in [30], but it required a complicated internal mechanism and increased the network cost. Recently, more methods targeted at collision problems are proposed in [31–34], but they introduced overhead to assist for packet collision indication, which adds to the complexity of networks.

In deterministic algorithms, radios are pre-scheduled to be "on" or "off" in each time slot based on some mathematical theorems to guarantee the discovery between every two neighbors. According to a survey in [35], three methods were usually adopted to guarantee the discovery, i.e., over-half occupation, quorum system, and co-prime. First, *over-half occupation* is to guarantee the overlap of active slots by keeping two nodes on for more than half of their slots. For example, given n slots in a period, if two nodes are active for at least $\frac{(n+1)}{2}$ slots, they must have some overlapping active slots. On the other hand, having more than half of the slots active means the energy usage is quite high. To reduce the excessive energy consumption, a more intelligent way is to divide a period into r cycles with

k slots each cycle and allocate active slots in each cycle. This incentive was adopted in SearchLight [9]. Second, *quorum system* considers m^2 slots as an $m \times m$ slot matrix, where each node takes one row and one column slots as active. This idea, implemented in [11,14,15] ensures discovery due to the indispensable intersected active slots. Among these algorithms, only Hedis [11] supports asymmetric duty cycle. Third, *co-prime* takes advantage of the Chinese remainder theorem [36]. For example, methods like Disco [12], U-Connect [15], and Todis [11] guarantees the intersection of active slots by making nodes active at the product of preset numbers that are co-prime to one another. However, most of these deterministic algorithms are designed for two nodes, despite being applied to multi-node scenarios without devising any approaches to deal with collisions.

In a more realistic scenario, when nodes are in a large, partially-connected network, probabilistic algorithms have an edge over deterministic ones in terms of reducing the number of collisions. Given the duty cycle, we can adjust the probabilities of transmission and listening in each slot to alleviate collisions. Ideally, the goal of designing neighbor discovery protocols is to guarantee discovery within a reasonable amount of time while minimizing the number of active slots for each node to save energy.

There are also some other algorithms that are designed for wireless sensor networks. Neighbor discovery algorithms are proposed with mobile sink nodes in [37,38] and a low latency protocol called Welcome is presented in [39] for mobile IoT devices. Some heuristic algorithms are also proposed with mobile sink nodes to improve the discovery performance in [40,41]. However, these algorithms are inapplicable since we do not consider mobile nodes in this paper.

3. Preliminaries

In this section, we introduce the system model and formulate the neighbor discovery problem for wireless sensor networks formally.

3.1. Sensor Node Model

Assuming that there are N sensor nodes in total that are deployed for a specific IoT application; we denote them as set $U = \{u_1, u_2, \dots, u_N\}$. Suppose each node u_i has a unique identifier (ID) i . Time is assumed to be divided into slots of equal length t_0 , which is sufficient to complete communications. All nodes can communicate through a specific channel for information exchanging. Suppose there are three states for each node: $\{Transmit, Listen, Sleep\}$, where *Transmit* means a node u_i broadcasts (sends packets) on the channel; *Listen* means node u_i listens on the channel and it can receive packets (message) from neighbors; and *Sleep* means node u_i turns its radio off and does nothing to save energy. In each slot, a node can choose to be in any state by turning on or off its radio. For simplicity, we assume state transitions do not consume any time or energy, and only nodes that are transmitting or listening consume their battery power.

Denote the activation time of node u_i as t_i^s , which implies the node does not work and stays in sleep mode until t_i^s . It is called a *synchronous scenario* if all deployed nodes are activated at the same time, i.e., $t_i^s = t_j^s, \forall u_i, u_j \in U$; otherwise it is called an *asynchronous scenario*. For a sufficiently long time from t_i^s to T , denote the number of slots that node u_i is in *Transmit*, *Listen*, or *Sleep* state as $T_i(T)$, $T_i(L)$ or $T_i(S)$, respectively; $T_i(T) + T_i(L) + T_i(S) = T - t_i^s$. We define the duty cycle of the node as

$$\theta_i = \frac{T_i(T) + T_i(L)}{T - t_i^s}. \quad (1)$$

For a specific IoT application, the duty cycle is normally defined in advance. In some hostile environment, it is costly to recharge the battery and thus the duty cycle would be very small, while the value could be large for some amicable applications.

3.2. Network Model

Most probabilistic works [16,20] only consider a fully connected network, where any two nodes are neighbors and they compose a fully connected network; this assumption is unrealistic. In this paper, we study the neighbor discovery process in a partially connected network, where some nodes are not neighbors, i.e., they are not connected directly for communication. We introduce two different network representations.

Considering a hostile environment where two nodes are neighbors with some probability, we use *neighboring matrix* $M_{N \times N}$ to represent the neighboring relations. If node u_i is a neighbor of u_j , we set M_{ij} and M_{ji} to be 1 (we assume the neighboring relation is undirected and u_j is also a neighbor of u_i), else the value of the entrance is 0. We assume u_i is a neighbor of u_j with probability p_n due to the obstacle or other blocking scenarios where $p_n \in (0, 1)$. Clearly, the network is a fully connected one when $p_n = 1$. In this paper, we consider a partially connected network where $p_n < 1$. Therefore, a node would have $n = p_n(N - 1)$ neighbors on average. For a large N , we can approximate $n \approx p_n N$ when it does not affect the analyses.

Considering another scenario where the nodes are deployed in a large field and the distance between two nodes may exceed the maximum radio range, we denote the distance of node u_i, u_j as $d(u_i, u_j)$ and two nodes are called neighbors ($M_{ij} = M_{ji} = 1$) if $d(u_i, u_j) \leq \Delta$, where Δ is the maximum radio range two nodes can communicate. Then the network topology would be determined after the deployment of the sensor nodes. In this paper, we suppose the nodes are deployed by the uniform distribution in a large area D . If the area D is covered by a $\Delta \times \Delta$ rectangle, all nodes are located within their radio range and they compose a fully connected network. We assume a large area D that cannot be covered by such a rectangle and the nodes compose a partially connected network.

3.3. Collision Model

Generally speaking, when one node u_i transmits through the channel, another node u_j who listens on the channel simultaneously could receive the transmitted packet/message and decode the message successfully. However, if two or more nodes transmit concurrently on the channel, *communication collision* occurs and u_j cannot decode the message correctly. Therefore, we assume a node u_j can discover its neighbor u_i only when u_j listens on the channel while u_i is u_j 's only neighbor who transmits.

In some design, communication collisions can be detected by hardware mechanism [18,20]. As described above, there are two situations for an unsuccessful discovery. The first scenario is no neighbor transmits while the second scenario is more than one neighbor transmit simultaneously. With the collision detection (CD) mechanism, a listening node can distinguish whether collisions occur or no neighbors is transmitting, apart from successful discovery. This CD mechanism enables the listening node to notify its transmitting neighbors of the transmission outcomes, and hence we can use this mechanism to design efficient algorithm.

3.4. Problem Definition

Neighbor discovery is not bidirectional, which means u_i discovering u_j is not equal to u_j discovering u_i . We first define discovery latency between two nodes as follows.

Definition 1. *Discovery latency $L(i, j)$ is defined as the duration from when the sensor node u_i starts to when u_i discovers its neighboring node u_j .*

Formally, suppose node u_i listens in time slot T while its neighbor node u_j transmits in slot T_j (the only neighbor who transmits), we say u_i discovers u_j and the discovery latency can be computed as

$$L(i, j) = T_j - t_i^s. \quad (2)$$

We define the discover problem for a specific node u_i as follows.

Problem 1. Considering an arbitrary node u_i and the set of neighbors $NS(u_i) = \{u_j | M_{ji} = 1\}$, design the algorithm for each node in each slot such that u_i can discover all nodes in $NS(u_i)$.

We define the discovery latency of node u_i as $L(i)$, which represents the time to discover all neighbors:

$$L(i) = \max_{u_j \in NS(u_i)} L(i, j) = \max_{u_j \in NS(u_i)} (T_j - t_i^s). \quad (3)$$

Since the sensor nodes may be activated in different time slots, we define the discovery latency of the network L_M as the maximum discovery latency of all nodes:

$$L_M = \max_{u_i \in U} L(i). \quad (4)$$

The objective is to design an efficient algorithm that can discover the neighbors in a short time (L_M) under the pre-defined duty cycle in a partially connected network. We list the notations in Table 1.

Table 1. Notations for neighbor discovery.

Notation	Description
U	The set of all sensor nodes $U = \{u_1, u_2, \dots, u_N\}$
u_i	Sensor node u_i with ID i
N	The number of nodes in the network is N
t_0	The length of a time slot is t_0
t_i^s	The activation time of sensor node u_i
θ_i	The duty cycle of sensor node u_i
M	Neighboring matrix, $M_{ij} = 1$ means u_i and u_j are neighbors
p_n	A node is the neighbor of another with probability p_n
$d(u_i, u_j)$	The distance between sensor nodes u_i and u_j
Δ	The maximum communication (radio) range
D	A large monitoring area
$L(i, j)$	The discovery latency that node u_i discovers node u_j
$NS(u_i)$	The set of node u_i 's neighboring nodes
$L(i)$	The discovery latency that node u_i discovers all neighbors
L_M	The discovery latency of the network

4. Panacea: An efficient Neighbor Discovery Algorithm

In this section, we describe *Panacea*, a low-latency, energy-efficient neighbor discovery algorithm for a partially connected network. To begin with, we assume the sensor nodes do not have a collision detection mechanism; we present the proposed *Panacea* algorithm and analyze the performance for both synchronous and asynchronous scenarios.

4.1. Algorithm Description

For any node u_i , suppose it has n neighbors on average and the pre-defined duty cycle of the node is θ_i . When the nodes cannot detect communication collision, we introduce the idea of designing the probabilistic algorithm called *Panacea*-NCD (*Panacea* no collision detection).

Without a collision detection mechanism, node u_i transmits with probability p_i^t , listens with probability p_i^l , and sleeps with probability p_i^s in each time slot t . For each state, node u_i takes the corresponding operations below.

- If u_i chooses state *Transmit*, it transmits a message containing its source ID on the channel;

- If u_i chooses state *Listen*, it listens on the channel and decodes the source ID of the received message if it receives a message successfully;
- If u_i chooses state *Sleep*, it does nothing.

We describe the algorithm formally as Algorithm 1. Each node u_i repeats the algorithm every time slot until it discovers all neighbors. In each time, node u_i generates a random value $r \in (0, 1)$ and takes corresponding actions when $r \leq p_i^t$, $p_i^t < r \leq \theta_i$, and $r > \theta_i$. The important part of the algorithm is to design the proper transmission probability p_i^t . It is obvious that $p_i^t + p_i^l + p_i^s = 1$. Due to different applications, each node's duty cycle is determined in advance and we denote it as θ_i for node u_i ; since the duty cycle denotes the fraction of time slots that u_i is transmitting or listening, hence $\theta_i = p_i^t + p_i^l$. Considering that nodes are comparatively well-distributed in the network, we suppose the probabilities of each node in each state are the same for simplicity. That is, $\forall i \in [1, N]$, $p_i^t = p_t$, $p_i^l = p_l$, $p_i^s = p_s$, and $\theta_i = \theta$. To minimize the probability of collisions, we derive an approximation of the optimal transmission probability that can alleviate communication collisions to help achieve low discovery latency:

$$p_i^t = \frac{1}{n}. \quad (5)$$

We will show the procedure of deriving the transmitting probability in the following parts. As described in the algorithm, node u_i selects the *Transmit* state with probability $p_i^t = \frac{1}{n}$ (Line 4–5), selects the *Listen* state with probability $\theta_i - p_i^t$ (Line 6–7), and selects the *Sleep* state with probability $1 - \theta_i$ (Line 8–9). We analyze how to determine the transmission probability and show the efficiency of the algorithm.

Algorithm 1 *Panacea*-NCD (u_i, θ_i, n).

```

1:  $p_i^t \leftarrow \frac{1}{n}$ 
2: while not terminate do
3:    $r \leftarrow \text{random}(0, 1)$ 
4:   if  $r \leq p_i^t$  then
5:     node  $u_i$  transmits on the channel
6:   else if  $p_i^t < r \leq \theta_i$  then
7:     node  $u_i$  listens
8:   else
9:     node  $u_i$  sleeps
10:  end if
11: end while

```

4.2. Algorithm Analysis of Synchronous Scenario

We first analyze the algorithm's efficiency of synchronous scenario where all nodes are activated simultaneously. We assume the network is represented by matrix $M_{N \times N}$ and any two nodes u_i, u_j are neighbors with probability $p_n \in (0, 1)$. We first derive the algorithm's discovery latency and derive the transmitting probability that minimizes the latency; then we show the upper bound of the discovery latency that can be utilized as a termination condition.

4.2.1. Expectation Analysis of Discovery Latency

According to the *Panacea*-NCD algorithm, we can easily derive that the probability that node u_i discovers a specific neighboring node u_j successfully in a given slot is:

$$p_{suc} = p_t(1 - p_t)^{n-1}(\theta - p_t). \quad (6)$$

This is because all nodes are activated at the same time and these nodes would select states independently. Node u_i can discover u_j only when u_j selects the *Transmit* state with probability p_t ,

node u_i selects the *Listen* state with probability $p_l = \theta - p_t$, while other $n - 1$ neighbors select the *Sleep* or *Listen* state with probability $p_l + p_s = 1 - p_t$.

By maximizing p_{suc} , we can alleviate the maximum amount of collisions. Thus, we derive the transmission probability p_t that maximizes p_{suc} with the derivative

$$p_t = \frac{\theta n + 2 - \sqrt{4 + (\theta n)^2 - 4\theta}}{2(1 + n)} \approx \frac{1}{n}. \quad (7)$$

Substituting this into the formulation, the probability that node u_i discovers a specific neighbor successfully in a given slot is $p_{suc} \approx \frac{\theta}{en}$. We approximate these equations when n is a large value and the approximation does not impact the discovery latency largely. When n is small, we can compute the exact value of p_t and derive the discovery probability.

We next show that maximizing p_{suc} helps to achieve the lowest expected latency. Let W be a random variable that denotes the time a node spends discovering all neighbors. Considering any node, we define the time spent in discovering a new neighbor (the j -th node) after it discovered $j - 1$ neighbors to be W_j , which follows a geometric distribution with parameter $p_{suc}(j)$: $p_{suc}(j) = (n - j + 1)p_{suc}$.

Hence, the expectation of W_j can be computed as:

$$E[W_j] = \frac{1}{p_{suc}(j)} = \frac{1}{(n - j + 1)p_{suc}}. \quad (8)$$

Clearly, $W = W_1 + W_2 + \dots + W_n$ which implies the time node u_i discovers all neighbors; the expectation of W can be formulated as

$$E[W] = \sum_{j=1}^n E[W_j] = \frac{1}{p_{suc}} H_n \quad (9)$$

where H_n is the n -th harmonic number, i.e., $H_n = \ln n + \Theta(1)$. By maximizing p_{suc} , the lowest expected discovery latency becomes:

$$E[W] \approx \frac{ne}{\theta} (\ln n + \Theta(1)) = \Theta\left(\frac{n}{\theta} \cdot \ln n\right). \quad (10)$$

When the duty cycle is a pre-defined parameter, the expected discovery latency can be bounded as $\Theta(n \cdot \ln n)$. Therefore, we can conclude the following theorem:

Theorem 1. *The Panacea-NCD algorithm ensures a node u_i can discover all its n neighboring nodes within $\Theta(n \cdot \ln n)$ time slots with a high probability for a synchronous scenario.*

4.2.2. Upper Bound Analysis of Discovery Latency

We show that the discovery latency is not likely to be much larger than its expectation, which can be utilized as the termination condition in Algorithm 1.

Recall the definition of W_i in the above analysis; if W_i is given, the value of W_j will not be affected for $i < j$. That is, $i \neq j$, W_i , and W_j are independent and they satisfy $P(W_j = w_j | W_i = w_i) = P(W_j = w_j)$. Since W_j follows geometric distribution and $Var[W_j] = \frac{1-p_j}{p_j^2}$, the variance of W can be computed as:

$$Var[W] = \sum_{j=1}^n Var[W_j] \leq \frac{\pi^2}{6p_{suc}^2} - \frac{H_n}{p_{suc}}. \quad (11)$$

According to the *Chebyshev's inequality*, we can derive the probability that the discovery latency exceeds the expected time by two times, as

$$P[W \geq 2E[W]] \leq \frac{\text{Var}[W]}{E[W]^2} \leq \frac{\pi^2}{6H_n^2} - \frac{p_{suc}}{H_n}. \quad (12)$$

For a large n where $n \rightarrow \infty$, and $P[W \geq 2E[W]]$ is close to 0. That is, the time for a node to find all neighbors is very likely to be smaller than 2 times the expected latency. Therefore we derive $W = O(n \cdot \ln n)$. We can conclude the corollary as follows.

Corollary 1. *The Panacea-NCD ensures a node u_i can discover all its n neighboring nodes in $O(n \ln n)$ time slots with high probability for a synchronous scenario when $n \rightarrow \infty$.*

4.3. Algorithm Analysis of Asynchronous Scenario

The proposed *Panacea-NCD* algorithm is suitable for asynchronous scenario where the sensor nodes are activated in different time slots. For node u_i , denote the maximum activation time offset between u_i and any neighbor is $\delta = \max_{u_j \in NS(u_i)} (t_i^s - t_j^s)$. Denote the latency that u_i finds a new neighbor after it discovered $j - 1$ neighbors as W'_j , and the latency that u_i finds all neighbors as W' .

It is obvious that the start time offset δ will not affect the latency, which implies

$$E[W'_j] = E[W_j] + \delta. \quad (13)$$

Hence, we derive the expected latency for a node discovering all its neighbors as

$$E[W'] \approx \frac{e}{\theta} n \cdot \ln n + [\Theta(1) + \delta]n = \Theta(n \cdot \ln n). \quad (14)$$

We can conclude that the expected discovery latency for an asynchronous scenario is δn slots larger than that for a synchronous scenario. When δ is a constant value, the expected discovery latency is also $\Theta(n \cdot \ln n)$. Similarly, we can also derive the probability that the discovery latency is 2 times larger than the expectation, as $P[W' \geq 2E[W']]$ is still close to 0; hence the discovery latency can be bounded by $O(n \cdot \ln n)$ with high probability. Combining these, we conclude the following theorem:

Theorem 2. *The Panacea-NCD algorithm ensures a node u_i can discover all its n neighboring nodes within $\Theta(n \cdot \ln n)$ slots in expectation and in $O(n \cdot \ln n)$ slots with a high probability for an asynchronous scenario when $n \rightarrow \infty$.*

Remark 1. *We assume $p_n \in (0, 1)$ in our analyses for the following reasons. When $p_n = 0$, it implies all nodes are completely separated and no node is connected with even one neighboring node; then it is meaningless to analyze the algorithm performance. When $p_n = 1$, the network is fully connected which implies any node can communicate with all other nodes directly. The analyses can also be adopted to the situation. Since we define the network as a partially connected one, we assume $p_n < 1$.*

4.4. Analysis for Uniform Distribution

Uniform distribution is used in most deployments of wireless networks. For instance, to monitor an unknown area, many sensors are deployed uniformly to collect information, such as temperature and humidity [42]. The nodes are evenly deployed and the density function can be formulated as:

$$f(x, y) = \begin{cases} \frac{1}{A} & (x, y) \in D \\ 0 & (x, y) \notin D \end{cases} \quad (15)$$

where A is the area of D . For any node u_i , denote the range of u_i 's neighbors' positions as R_i and any neighboring node with coordinate $(x, y) \in R_i$ suits $(x - x_i)^2 + (y - y_i)^2 \leq \Delta^2$, where (x_i, y_i) is u_i 's position. Then, node u_i 's expected number of neighbors can be computed as

$$n = N \iint_{R_i} f(x, y) dx dy - 1 \simeq \frac{N\pi\Delta^2}{A}. \quad (16)$$

Combining this in the *Panacea*-NCD's design, we can still derive the upper bound of discovery latency to be $O(n \cdot \ln n)$ when n is large and compute the latency precisely.

We analyze the algorithm when the nodes have nearly the same number of neighbors for different network representations (including the uniform distribution). When the nodes obey some other distribution, the nodes may have a different number of neighbors and it would be an interesting future work. In addition, the analyses of the expectation and the upper bounds are derived when n is large. When n is small, these bounds may not hold but they show the increasing trend when n increases.

5. Panacea-WCD: Neighbor Discovery with Collision Detection

In this section, we propose *Panacea*-WCD algorithm, a novel random algorithm that achieves low latency for a given duty cycle when the nodes can detect collision.

5.1. Algorithm Description

If nodes could identify collisions, the transmitting node(s) could be notified whether transmissions are successful by their listening neighbors. We describe *Panacea*-WCD in Algorithm 2 and it works as follows:

Each time slot is divided into two sub-slots. In the first sub-slot, nodes transmit, listen, or sleep and act in response to one of the three states. In the second sub-slot, nodes notify their neighbor(s) and maintain *DiscoveredList* to record the discovered nodes.

Initially, each node u_i sets $k_i = 0$, and α is a constant. For each time slot t ,

- (1) In the first sub-slot, node u_i transmits a message containing its source node ID with probability $p_i^t = \frac{1}{n+\alpha k_i}$ (Line 4), listens with probability $p_i^l = \theta_i - p_i^t$, and sleeps with probability $p_i^s = 1 - \theta_i$;
- (2) In the second sub-slot, there are three sub-cases:
 - If u_i selects the *Listen* state in the first sub-slot: if u_i receives a message successfully, u_i decodes and records the source ID in the message. If the ID does not belong to u_i 's *DiscoveredList*, u_i adds the ID to *DiscoveredList*, and deterministically transmits a message on the channel (a bit is sufficient) in the second sub-slot;
 - If u_i selects the *Transmit* state in the first sub-slot: if u_i detects energy (a message or a collision by multiple messages), u_i is notified the successful transmission and sets $k_i := k_i + 1$, otherwise u_i regards its transmission as unsuccessful in the first sub-slot;
 - If u_i selects the *Sleep* state in the first sub-slot: u_i does nothing in the second sub-slot.

The core of *Panacea*-WCD is that once u_i discovers a new node u_j that does not belong to its *DiscoveredList*, u_i adds u_j 's ID to its *DiscoveredList* and notifies u_j of the successful discovery. In this successful scenario, u_i 's feedback in the second sub-slot can be 1-bit. Thus, the second sub-slot is much smaller than the first one, and it only introduces a small overhead.

Algorithm 2 Panacea-WCD(u_i, θ_i, α, n).

```

1:  $k_i \leftarrow 0, DiscoveredList \leftarrow \{\}$ 
2: while not terminate do
3:    $p_i^t \leftarrow \frac{1}{n+\alpha k_i}, r \leftarrow random(0,1)$ 
4:   if  $r < p_i^t$  then
5:     node  $u_i$  transmits in the first sub-slot
6:     if detects energy in the second sub-slot then
7:        $SuccessTransmission \leftarrow True, k_i \leftarrow k_i + 1$ 
8:     else
9:        $SuccessTransmission \leftarrow False$ 
10:    end if
11:  else if  $p_i^t < r < \theta_i$  then
12:    node  $u_i$  listens in the first sub-slot
13:    if Successful reception in the first sub-slot then
14:      node  $u_i$  decodes the message and the source ID (srcID)
15:      if srcID not in  $DiscoveredList$  then
16:         $DiscoveredList = DiscoveredList \cup \{srcID\}$ 
17:      end if
18:    end if
19:  else
20:    node  $u_i$  sleeps in the first sub-slot
21:  end if
22: end while

```

5.2. Algorithm Analysis of Synchronous Scenario

We also analyze the discovery latency from both expectation and upper bound aspects.

5.2.1. Expectation Analysis of Discovery Latency

Similar to Panacea-NCD, we suppose the latency to find a new neighbor after discovered $j - 1$ neighbors is W_j . To simplify this analysis, we consider a node u_i discovering neighbors at an average level, where the number of discovered neighbors follows a normal distribution. That is, some nodes that discover neighbors faster and slower tend to compensate for the number of discovered neighbors of each other. In other words, we have a global transmission probability $p_t(j)$ in W_j , which is

$$p_t(j) = \frac{1}{n + \alpha(j - 1)}. \quad (17)$$

In a given time slot, the probability that u_i discovers a new neighbor after it discovered $j - 1$ neighbors is

$$p_{suc}(j) = p_t(j)(1 - p_t(j))^{n-1}(\theta - p_t(j))(n - j + 1). \quad (18)$$

As W_j follows geometric distribution with parameter $p_{suc}(j)$, we choose α close to 1, $\frac{1}{n+\alpha(j-1)} \ll \theta$, and the latency expectation that u_i discovers all neighbors $W = \sum_{i=1}^n W_i$ can be computed as:

$$E[W] = \sum_{j=1}^n E[W_j] = \sum_{j=1}^n \frac{1}{p_{suc}(j)}. \quad (19)$$

Because $\sum_{j=1}^n \frac{n+\alpha(j-1)}{n-j+1} = (\alpha + 1)nH_n - \alpha n$ where H_n is n -th harmonic number, we can obtain:

$$\frac{n}{\theta}[(\alpha + 1) \ln n + \Theta(1)] \leq E[W] \leq \frac{en}{\theta}[(\alpha + 1) \ln n + \Theta(1)]. \quad (20)$$

When θ is determined in advance, we can conclude the theorem as

Theorem 3. *The Panacea-WCD algorithm ensures a node u_i can discover its all n neighboring nodes within $\Theta(n \cdot \ln n)$ time slots in expectation for synchronous scenario.*

5.2.2. Upper Bound Analysis of Discovery Latency

We show that the latency is not likely to be much larger than its expectation. For $i \neq j$, W_i and W_j are independent by definition. As $\text{Var}[W_j] = \frac{1-p_{suc}(j)}{p_{suc}^2(j)}$ for geometric distribution with parameter $p_{suc}(j)$, we obtain:

$$\text{Var}[W] = \sum_{j=1}^n \text{Var}[W_j] = \sum_{j=1}^n \frac{1}{p_{suc}^2(j)} - \sum_{j=1}^n \frac{1}{p_{suc}(j)}. \quad (21)$$

We know that $\sum_{j=1}^n \frac{1}{p_{suc}^2(j)} \leq \frac{e^2}{\theta^2} [\alpha^2 n - 2\alpha n(1 + \alpha)H_n + \frac{\pi^2}{6}(1 + \alpha^2)n^2]$; according to *Chebyshev's inequality*, we derive:

$$P[W \geq 2E[W]] \leq \frac{e^2 \pi^2 (1 + \alpha^2) / 6}{[(1 + \alpha)H_n - \alpha]^2} - \frac{\theta / (en)}{[(\alpha + 1)H_n - \alpha]}. \quad (22)$$

The probability is close to 0 when n is large ($n \rightarrow \infty$). Hence, the latency is not likely to be 2 times larger than the expectation. Therefore, we conclude the corollary as:

Corollary 2. *The Panacea-WCD ensures a node u_i can discover its all n neighboring nodes in $O(n \ln n)$ time slots with high probability for synchronous scenario when $n \rightarrow \infty$.*

5.3. Algorithm Analysis of Asynchronous Scenario

For *Panacea-WCD*, we can also derive Equation (13) of latency expectation to discover a new neighbor after the discovered $j - 1$ neighbors. The expected latency for a node discovering all its neighbors is

$$\frac{(\alpha + 1)nH_n}{\theta} + (\delta - \alpha/\theta)n \leq E[W] \leq \frac{e}{\theta} [(\alpha + 1)nH_n] + (\delta - \alpha e/\theta)n. \quad (23)$$

We can derive $E[W'] = \Theta(n \cdot \ln n)$ for asynchronous scenario. Similarly to *Panacea-NCD*, the probability that the discovery latency is 2 times larger than the expectation is still close to 0. Hence, we conclude the following theorem:

Theorem 4. *The Panacea-WCD algorithm ensures a node u_i can discover its all n neighboring nodes within $\Theta(n \cdot \ln n)$ slots in expectation and in $O(n \cdot \ln n)$ slots with a high probability for an asynchronous scenario when $n \rightarrow \infty$.*

Notice that, when the proposed algorithm is deployed in realistic network technologies such as Zigbee and BLE Mesh, we have to adjust the algorithm if their collision detection mechanism is applicable. For example, Zigbee utilizes CSMA/CA to reduce collision, which is difficult to couple with *Panacea-WCD*, while BLE Mesh finishes events with a random delay, we may adjust the transmit and listen operations.

Remark 2. *The Panacea algorithm (including both Algorithm 1 and Algorithm 2) needs to know the average number of neighbors in advance. The introduced two network representations help identify this value. In practice, the sensors are deployed artificially or by some distribution (such as the uniform distribution in Section 4.4); it is reasonable to derive the number of neighbors beforehand. It would be an interesting future work to relax the limitation.*

6. Evaluation

We implemented *Panacea* in C++ and evaluated the algorithms in a cluster with four nodes, each with an Intel(R) Xeon(R) CPU E5-2640 v4 @ 2.40 GHz, 12 cores, 128 GB memory, and 2 GB spinning disk. We used two different initialization methods to generate the networks. There are N sensor nodes in total and we set the maximum activation offset $\delta = 1000$. The first method generates the neighboring matrix $M_{N \times N}$ with probability p_n , which implies node u_i is the neighbor of any other node u_j (i.e., $M_{ij} = M_{ji} = 1$) with probability p_n . The second method assumes all nodes are deployed in an area of size 100×100 m by the uniform distribution and the communication range is $\Delta = 10$ m. Any two nodes u_i, u_j satisfying $d(u_i, u_j) \leq \Delta$ are neighbors in the network. These two methods can generate the networks more complicated and realistic than that in [9,11,12,14–20,22].

We evaluate discovery latency of *Panacea*, Coupon [20], Aloha-like [22], and PND [18] in the generated partially-connected networks in both synchronous and asynchronous scenarios. Since Coupon [20] and PND [18] do not consider duty cycle, we evaluate the tradeoff of duty cycle and discovery latency among *Panacea*, Aloha-like [22], and Hello [19]. We also compare the evaluation results with the theoretical analyses. The expected discovery latency of *Panacea*-NCD is derived as Equation (10) while the expected latency of *Panacea*-WCD is bounded by Equation (20). We set the length of each time slot as $t_0 = 20$ ms and the nodes are activated randomly within time $[0, \delta]$ for asynchronous scenario; the results are based on 1000 separate runs.

6.1. Partially- and Fully-Connected Networks

To begin with, we evaluate the performance of the algorithms in different settings. Though *Panacea* is proposed for partially connected networks, it can be also adopted in a fully-connected network. As shown in Figures 1 and 2, when the network is partially connected ($p_n = 0.1, 0.5$), the algorithms use less time to discover the neighbors (y-axis, number of slots) compared to the fully connected setting when there are $N = 1000$ node in total. Among them, *Panacea* shows superior performance.

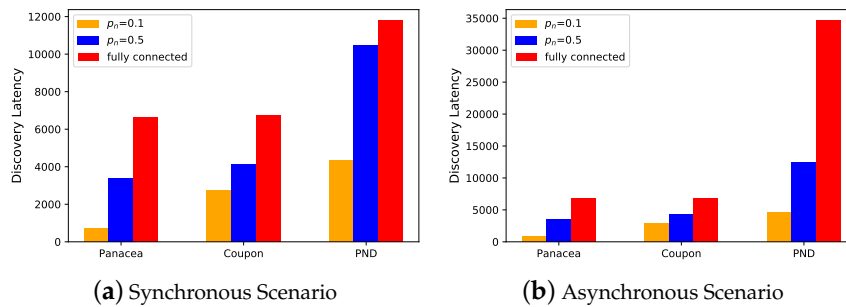


Figure 1. The neighbor discovery algorithms work better in partially connected networks and *Panacea*-NCD (*Panacea* no collision detection) outperforms the other algorithms.

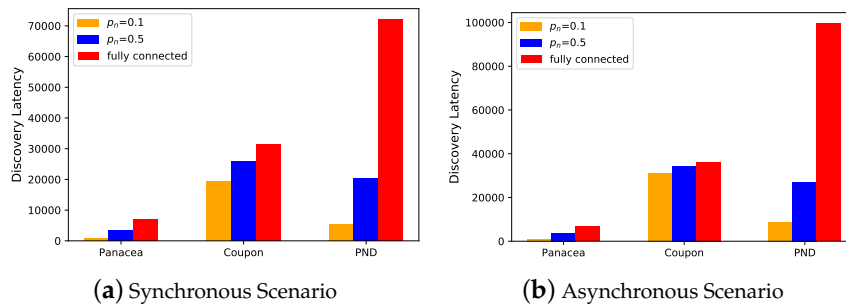


Figure 2. The neighbor discovery algorithms work better in partially connected networks and *Panacea*-WCD outperforms the other algorithms.

6.2. Comparison of Discovery Latency

We evaluate the performance of the proposed algorithms in different settings. In the first place, we generate the network with $N = 1000$ nodes and $p_n = 0.1$. We compare *Panacea* with Coupon and PND when the nodes always turn their radios on (i.e., the duty cycle is 1). As shown in Figure 3a where y -axis represents discovery latency, i.e., the number of slots, *Panacea*-NCD could achieve the lowest discovery latency when the nodes do not have a collision detection mechanism, reducing time by about 60% and 92% when compared to Coupon and PND, respectively. *Panacea*-WCD also has the best performance when the node can detect collision as illustrated in Figure 3b; it reduces latency by about 30% and 90% compared to Coupon and PND, respectively. Coupon can achieve good performance in a fully-connected network, but it works worse than *Panacea* in a partially-connected network. We do not compare the Aloha-like algorithm because it is the same with Coupon when the duty cycle is 1. By setting different duty cycles in our algorithm, *Panacea* could save energy by 40% when it achieves similar discovery latency with Coupon.

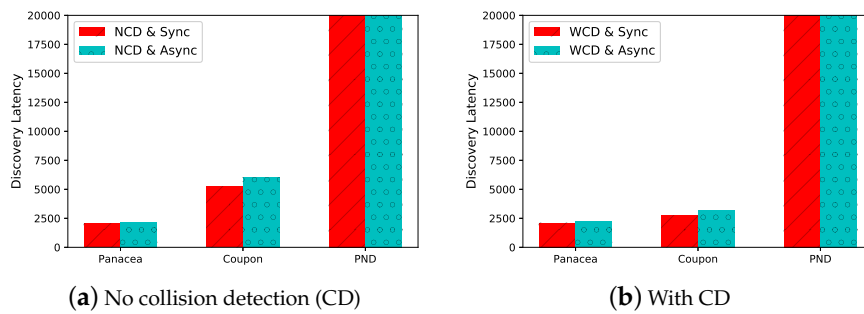


Figure 3. *Panacea*-NCD and *Panacea*-WCD have the lowest discovery latency under both synchronous and asynchronous scenarios when $p_n = 0.1, N = 1000$.

We also evaluate the algorithms when the number of nodes increases from 100 to 1000. As depicted in Figure 4, when the network size increases (x-axis), the discovery latency (y-axis, number of slots) increases for all algorithms. This is because each node has more neighbors to discover and the communication collision occurs more often. We compare *Panacea* to Coupon and PND for four different scenarios regarding whether the nodes are activated synchronously and whether they have a collision detection mechanism. The results show that our algorithms have the best performance, achieving lowest discovery latency when the number of nodes increases. Figure 4a,b also depicts the derived expected discovery latency of *Panacea*-NCD, and the results show that the evaluations corroborate the theoretical analyses. Regarding *Panacea*-WCD, Figure 4c,d shows the upper bound and lower bound of the expected discovery latency and the evaluation results also corroborate the analyses.

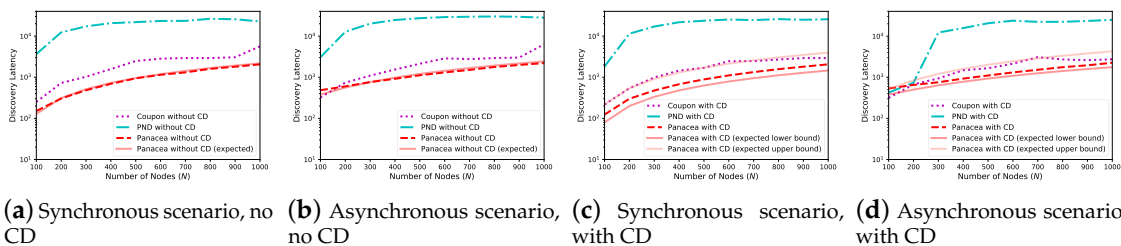


Figure 4. *Panacea*-NCD and *Panacea*-WCD achieve lowest discovery latency for both synchronous and asynchronous scenarios when $p_n = 0.1$ and N increases from 100 to 1000.

When the network is generated according to the nodes' positions, we compare the neighbor discovery algorithms in Figure 5. When the nodes are deployed according to the uniform distribution and the number of nodes N increases from 100 to 1000, we also evaluate their performance for four

different scenarios; the results show that the discovery latency increases when N increases and *Panacea* could achieve the lowest discovery latency among them. In addition, comparing with the expected discovery latency of *Panacea*, the evaluation results corroborate the theoretical analyses in Equation (10) and Equation (20).

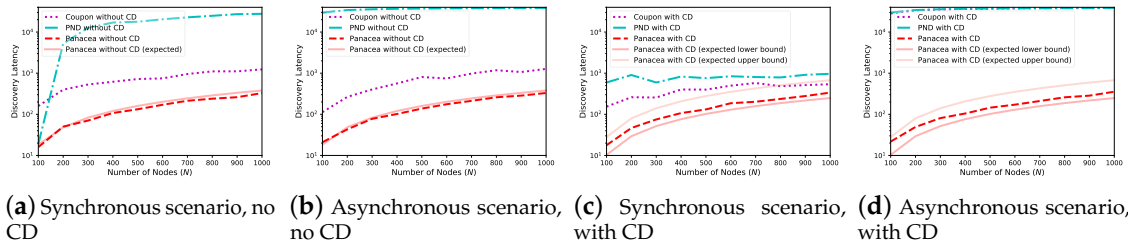


Figure 5. *Panacea*-NCD and *Panacea*-WCD achieve the lowest discovery latency for both synchronous and asynchronous scenarios under the uniform distribution.

As analyzed, *Panacea* could achieve the lowest discovery latency because Coupon and PND cannot handle communication collision efficiently, while *Panacea* could reduce discovery latency by handling the collision in an appropriate way.

6.3. Comparison of Discovery Rate

We evaluate the discovery rate of the algorithms which is defined as the percentage of discovered neighbors. We first evaluate *Panacea*-WCD for a dense network $p_n = 0.5$, $N = 1000$ and the duty cycle is $\theta = 0.5$. Figure 6 shows the discovery rate (y-axis) increases as time (x-axis) increases. In the synchronous scenario, *Panacea*-WCD could reach 100% discovery rate, while Coupon reaches 79.2% and PND reaches 84.6%, respectively. This is because Coupon and PND simply regard the case of no collision as a success, ignoring that some neighbors of transmitting nodes fail to discover in the *Sleep* state. The results show the similar phenomenon in the asynchronous scenario.

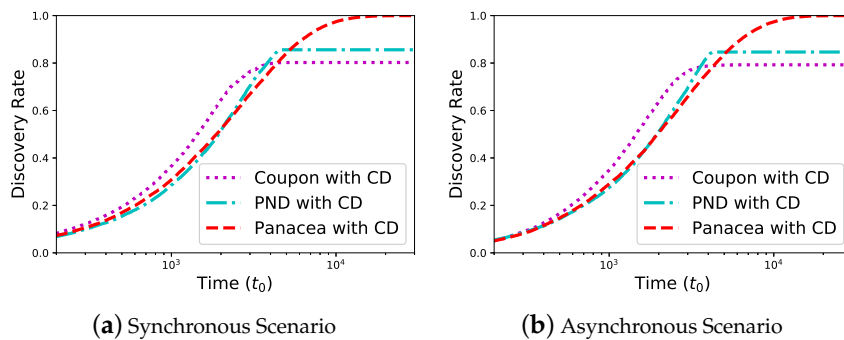


Figure 6. *Panacea*-WCD achieves higher discovery rate with CD.

We evaluate the discovery rate when $N = 1000$, $\theta = 0.5$ and all nodes are uniformly distributed. As shown in Figure 7, *Panacea* achieves higher discovery rate than others; both *Panacea*-NCD and *Panacea*-WCD could reach 100% discovery rate. Coupon could reach (nearly) 100% discovery rate in some scenarios when the network is sparser than that in Figure 6; this is because Coupon is more applicable to a sparse network.

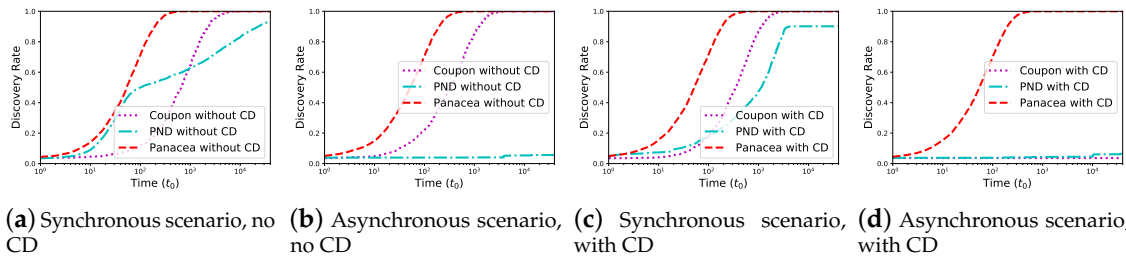


Figure 7. *Panacea*-NCD and *Panacea*-WCD achieve higher discovery rate for both synchronous and asynchronous scenarios under the uniform distribution.

These figures validate that *Panacea* could handle communication collision in the algorithm design and thus it achieves 100% discovery rate.

6.4. Tradeoff: Discovery Latency v.s. Duty Cycle

Since different applications could set different duty cycle values, we evaluate the tradeoff between discovery latency and duty cycle. When $N = 1000$, $p_n = 0.5$, Figure 8 demonstrates that *Panacea* has a better tradeoff between discovery latency (y-axis, number of slots) and duty cycle (x-axis); the discovery latency is proportional to $\frac{1}{\theta}$ for *Panacea*, which corroborates our analyses. The evaluated discovery latency of *Panacea*-NCD fits quite well with the expected discovery latency, while the evaluated latency of *Panacea*-WCD is also located within the lower bound and the upper bound of the expected latency. PND and Hello have higher latency, because PND's collision slots make up 46.5% of the time, and Hello's collision slots make up 99.8% of the time. The *power-latency product* [15] measurement is the product of the average power consumption with discovery latency. The power-latency product of *Panacea* is 9213, while that of Aloha-like is 20.3% larger, and PND is 2.69 times larger than *Panacea* in the synchronous scenario. The PND algorithm's performance is unstable when the duty cycle increases, this is because PND suffers from irregular collisions and the algorithm varies the sending probability according to different scenarios. In addition, we only measured the average discovery latency of 1000 separate runs.

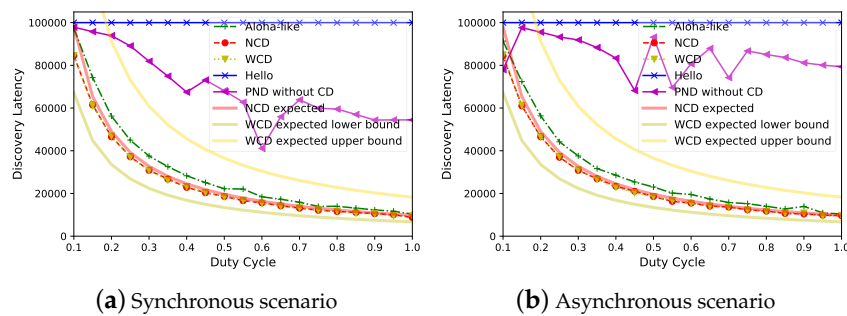


Figure 8. *Panacea* has better tradeoff between duty cycle and latency for both synchronous and asynchronous scenarios.

6.5. Network Density

Finally, we evaluate the performance of *Panacea* when the network becomes denser. We fix $N = 1000$ and increase p_n from 0.1 to 0.9; as shown in Figure 9, the discovery latency (y-axis, number of slots) increases for both scenarios (without or with collision detection). This is because collisions take place more frequently in denser networks. We depict three curves for different duty cycles (0.1, 0.3, and 0.5, respectively); the results show that higher duty cycle contributes to lower latency, since discovery latency is proportional to $\frac{1}{\theta}$. The results validate that the discovery latency of *Panacea* is

proportional to $\frac{1}{\theta}$ and that a larger p_n leads to larger latency due to the collisions. The evaluated results also corroborate the theoretical analyses of the algorithm.

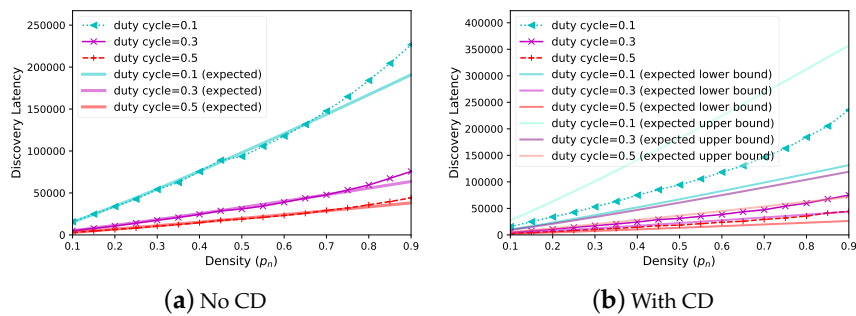


Figure 9. Discovery latency increases when the network becomes denser.

From these figures, the proposed *Panacea* algorithm can achieve a low discovery latency, a high discovery rate, and good tradeoff between discovery latency with duty cycle.

7. Conclusions

In this paper, we propose *Panacea* a low-latency and energy-efficient neighbor discovery algorithm for partially connected wireless sensor networks. First, we present *Panacea*-NCD for nodes without collision detection mechanism and we analyze that the discovery latency is bounded within $O(n \cdot \ln n)$ time slots for any pre-defined duty cycle when n is large. Then, we modify this method in *Panacea*-WCD when nodes can detect collision. The discovery latency of *Panacea*-WCD is also bounded within $O(n \cdot \ln n)$ slots for a large n . Our algorithms can be efficient in both synchronous and asynchronous scenarios because we alleviate collisions as much as possible. Furthermore, we verified our theoretical analyses by simulations and the results corroborate our analyses. Still, *Panacea* needs to know an average number of neighbors beforehand and that the nodes have the same number of neighbors, it would be important and interesting to relax such a priori knowledge in the future. In addition, we will evaluate the algorithms' performance in further practical applications.

Author Contributions: Conceptualization, Z.G. and G.M.; Formal analysis, Z.C.; Funding acquisition, Z.T.; Investigation, Z.C.; Methodology, Z.G. and Z.T.; Software, Y.W.; Supervision, Y.W.; Validation, Z.C.; Visualization, X.D.; Writing—original draft, Z.G.; Writing—review and editing, X.D. and G.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported in part by the National Key R&D Program of China 2018YFB1004003, China grants U1636215, 61572492, and Guangdong Province Universities and Colleges Pearl River Scholar Funded Scheme (2019).

Conflicts of Interest: The authors declare no conflict of interest.

References

- Zanella, A.; Bui, N.; Castellani, A.; Vangelista, L.; Zorzi, M. Internet of Things for Smart Cities. *IEEE Internet Things J.* **2014**, *1*, 22–32. [[CrossRef](#)]
- Wu, L.; Du, X.; Wang, W.; Lin, B.. An Out-of-band Authentication Scheme for Internet of Things Using Blockchain Technology. In Proceedings of the IEEE ICNC, Maui, HI, USA, 5–8 March 2018.
- Alemdar, H.; Ersoy, C. Wireless sensor networks for healthcare: A survey. *Comput. Netw.* **2010**, *54*, 2688–2710. [[CrossRef](#)]
- Liu, Y.; Mao, X.; He, Y.; Liu, K.; Gong, W.; Wang, J. CitySee: Not only a wireless sensor network. *IEEE Netw.* **2013**, *27*, 42–47. [[CrossRef](#)]
- Tian, Z.; Shi, W.; Wang, Y.; Zhu, C.; Du, X.; Su, S.; Sun, Y.; Guizani, N. Real Time Lateral Movement Detection based on Evidence Reasoning Network for Edge Computing Environment. *IEEE Trans. Ind. Inf.* **2019**, *15*, 4285–4294. [[CrossRef](#)]

6. Tian, Z.; Luo, C.; Qiu, J.; Du, X.; Guizani, M.. A Distributed Deep Learning System for Web Attack Detection on Edge Devices. *IEEE Trans. Ind. Inf.* **2019**. [[CrossRef](#)]
7. Wang, Y.; Wang, Y.; Qi, X.; Xu, L.; Chen, J.; Wang, G. L3SN: A Level-Based, Large-Scale, Longevous Sensor Network System for Agriculture Information Monitoring. *Wirel. Sens. Netw.* **2010**, *2*, 655. [[CrossRef](#)]
8. Zhai, Z.; Ortega, J.-F.M.; Castillejo, P.; Beltran, V. A Triangular Similarity Measure for Case Retrieval in CBR and Its Application to an Agricultural Decision Support System. *Sensors* **2019**, *19*, 4605. [[CrossRef](#)]
9. Bakht, M.; Trower, M.; Kravets, R.H. Searchlight: Won't you be my neighbor? In Proceedings of the Annual International Conference on Mobile Computing and Networking (MobiCom), Istanbul, Turkey, 22–26 August 2012.
10. Cao, Z.; Gu, Z.; Wang, Y.; Cui, H. Panacea: A Low-Latency, Energy-Efficient Neighbor Discovery Protocol for Wireless Sensor Networks. In Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC), Barcelona, Spain, 15–18 April 2018.
11. Chen, L.; Fan, R.; Bian, K.; Gerla, M.; Wang, T.; Li, X. On heterogeneous neighbor discovery in wireless sensor networks. In Proceedings of the INFOCOM, Hong Kong, China, 26 April–1 May 2015.
12. Dutta, P.; Culler, D. Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications. In Proceedings of the Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys), Raleigh, NC, USA, 5–7 November 2008.
13. Gu, Z.; Wang, Y.; Shi, W.; Tian, Z.; Ren, K.; Lau, F.C.M. A Practical Neighbor Discovery Framework for Wireless Sensor Networks. *Sensors* **2019**, *19*, 1887. [[CrossRef](#)]
14. Jiang, J.-R.; Tseng, Y.-C.; Hsu, C.-S.; Lai, T.-H. Quorum-based Asynchronous Power-Saving Protocols for IEEE 802.11 Ad Hoc Networks. *Mob. Netw. Appl.* **2005**, *10*, 169–181. [[CrossRef](#)]
15. Kandhalu, A.; Lakshmanan, K.; Rajkumar, R.R. U-Connect: A Lower Latency Energy-Efficient Asynchronous Neighbor Discovery Protocol. In Proceedings of the IPSN, Stockholm, Sweden, 12–16 April 2010.
16. McGlynn, M.J.; Borbash, S.A. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In Proceedings of the MobiHoc, Long Beach, CA, USA, 4–5 October 2001.
17. Qiu, Y.; Li, S.; Xu, X.; Li, Z. Talk More Listen Less: Energy-Efficient Neighbor Discovery in Wireless Sensor Networks. In Proceedings of the INFOCOM, San Francisco, CA, USA, 10–15 April 2016.
18. Song, T.; Park, H.; Pack, S. A probabilistic neighbor discovery algorithm in wireless ad hoc networks. In Proceedings of the IEEE 79th VTC, Seoul, South Korea, 18–21 May 2014.
19. Sun, W.; Yang, Z.; Zhang, X.; Liu, Y. Hello: A Generic Flexible Protocol for Neighbor Discovery. In Proceedings of the INFOCOM, Toronto, ON, Canada, 27 April–2 May 2014.
20. Vasudevan, S.; Towsley, D.; Goeckel, D.; Khalili, R. Neighbor Discovery in Wireless Networks and the Coupon Collector's Problem. In Proceedings of the MobiCom, Beijing, China, 20–25 September 2009.
21. Xiao, Y.; Rayi, V.; Sun, B.; Du, X.; Hu, F.; Galloway, M. A Survey of Key Management Schemes in Wireless Sensor Networks. *J. Comput. Commun.* **2007**, *30*, 2314–2341. [[CrossRef](#)]
22. You, L.; Yuan, Z.; Yang, P.; Chen, G. ALOHA-like neighbor discovery in low-duty-cycle wireless sensor networks. In Proceedings of the WCNC, Cancun, Quintana Roo, Mexico, 28–31 March 2011.
23. Du, X.; Xiao, Y.; Guizani, M.; Chen, H.H. An Effective Key Management Scheme for Heterogeneous Sensor Networks. *Ad Hoc Netw.* **2007**, *5*, 24–34. [[CrossRef](#)]
24. Du, X.; Chen, H.H. Security in Wireless Sensor Networks. *IEEE Wirel. Commun. Mag.* **2008**, *15*, 60–66.
25. Du, X.; Guizani, M.; Xiao, Y.; Chen, H.H. A Routing-Driven Elliptic Curve Cryptography based Key Management Scheme for Heterogeneous Sensor Network. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 1223–1229. [[CrossRef](#)]
26. Shen, T.; Wang, Y.; Gu, Z.; Li, D.; Cao, Z.; Cui, H.; Lau, F.C.M. Alano: An Efficient Neighbor Discovery Algorithm in an Energy-Restricted Large-Scale Network. In Proceedings of the MASS, Chengdu, China, 9–12 October 2018.
27. Tian, Z.; Su, S.; Shi, W.; Du, X.; Guizani, M.; Yu, X. A Data-driven Method for Future Internet Route Decision Modeling. *Future Gener. Comput. Syst.* **2019**, *95*, 212–220. [[CrossRef](#)]
28. Wei, L.; Sun, W.; Chen, H.; Yuan, P.; Yin, F.; Luo, Q.; Chen, Y.; Chen, L. A fast neighbor discovery algorithm in WSNs. *Sensors* **2018**, *18*, 3319.
29. Khalili, R.; Goeckel, D.; Towsley, D.; Swami, A. Neighbor discovery with reception status feedback to transmitters. In Proceedings of the INFOCOM, San Diego, CA, USA, 15–19 March 2010.

30. Julien, C.; Liu, C.; Murphy, A.L.; Picco, G.P. BLEnd: practical continuous neighbor discovery for Bluetooth low energy. In Proceedings of the IPSN, Pittsburgh, PA, USA, 18–21 April 2017.
31. Wang, Y.; Yu, Z.; Huang, J.; Choi, C. A novel energy-efficient neighbor discovery procedure in a wireless self-organization network. *Inf. Sci.* **2019**, *476*, 429–438. [[CrossRef](#)]
32. Kaleem, Z.; Li, Y.; Chang, K. Public safety users priority-based energy and time-efficient device discovery scheme with contention resolution for ProSe in third generation partnership project long-term evolution-advanced systems. *IET Commun.* **2016**, *13*, 1873–1883. [[CrossRef](#)]
33. Wang, Z.; Sun, E.; Zhang, Y. Delay and collision optimization for clustered machine type communications in random access procedures. In Proceedings of the 2016 6th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 17–19 June 2016.
34. Zhang, Z.; Luo, L.; Wang, L. D2D multicast retransmission algorithm in mobile cloud based on SINR constraint. *China Commun.* **2016**, *13*, 41–52. [[CrossRef](#)]
35. Sun, W.; Yang, Z.; Zhang, X.; Liu, Y. Energy-efficient neighbor discovery in mobile ad hoc and wireless sensor networks: A survey. *Commun. Surv. Tutor.* **2014**, *16*, 1448–1459. [[CrossRef](#)]
36. Nathanson, M.B. *Elementary Methods in Number Theory*; Springer: New York, NY, USA, 2000; Volume 195.
37. Montero, S.; Gozalvez, J.; Sepulcre, M. Neighbor discovery for industrial wireless sensor networks with mobile nodes. *Comput. Commun.* **2017**, *111*, 41–55. [[CrossRef](#)]
38. Zhang, Y.; Wei, L.; Guo, M.; Wang, W.; Sun, Y.; Wang, J.; Chen, L. VN-NDP: A Neighbor Discovery Protocol Based on Virtual Nodes in Mobile WSNs. *Sensors* **2019**, *19*, 4739. [[CrossRef](#)] [[PubMed](#)]
39. Harmassi, M.; Khan, J.A.; Ghamridoudane, Y.; Faucher, C. Welcome: Low Latency and Energy Efficient Neighbor Discovery for Mobile and IoT Devices. In Proceedings of the 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), Limassol, Cyprus, 15–17 October 2018.
40. Xiao, L.; Li, Y.; Huang, X.; Du, X. Cloud-based Malware Detection Game for Mobile Devices with Offloading. *IEEE Trans. Mob. Comput.* **2017**, *16*, 2742–2750. [[CrossRef](#)]
41. Wang, J.; Cao, J.; Sherratt, R.S.; Park, J.H. An improved ant colony optimization-based approach with mobile sink for wireless sensor networks. *J. Supercomput.* **2018**, *74*, 6633–6645. [[CrossRef](#)]
42. Flammini, A.; Marioli, D.; Sisinni, E.; Taroni, A. A real-time wireless sensor network for temperature monitoring. In Proceedings of the IEEE International Symposium on Industrial Electronics, Vigo, Spain, 4–7 June 2007.



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).