Talk Half Listen To Half: Energy-Efficient Neighbor Discovery in Wireless Sensor Networks

By

Raudel Ravelo Suarez

Thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

In partial fulfillment of the requirements

For the MASTER OF SCIENCE degree in

Computer Science

Ottawa – Carleton Institute for Computer Science

School of Electrical Engineering and Computer Science

Faculty of Engineering

University of Ottawa

Abstract

Due to the combination of constrained power, low duty cycle, and high mobility, neighbor discovery is one of the most challenging problems in wireless sensor networks. Existing discovery designs can be divided into two types: pairwise-based and group-based. The former schemes suffer from high discovery delay, while the latter ones accelerate the discovery process but increase transmission package size or incur too much energy overhead, far from practical.

Guided by the *Talk More Listen Less* (TMLL) principle (published in 2016), in which beacons are not necessarily placed in the wakeup slots, we propose two different versions of a group-based protocol we called *Talk Half Listen Half* (THLH). For the first time, a group-based protocol uses the *Channel Occupancy Rate* (COR), one of the fundamental novel components of the TMLL model, for performance improvements, in the same way, Duty Cycle (DC) was used in previous group-based protocols. Both versions of the protocol use low transmission overhead in comparison with previous group-based discoveries.

After analyzing pros and cons of each approach, we arrived at the conclusion that both behave the best for networks where the average number of new neighbors per slot (β) is low, a metric that sets the bases for performance comparisons of any current/future work with variable COR usage. We also derived a formula that links this new metric with the worst case avg. COR usage of our proposed protocols. Finally, simulation results show that our protocol can improve the average discovery latency and worst case latency close to 50% given low β values.

Acknowledgment

First, I would like to express my sincere gratitude to Prof. Amiya Nayak for his continuous support, motivation, advice and immense knowledge throughout my thesis work, without him I would not have been able to complete this difficult process.

To my wife Gaby, for providing me with unfailing support and encouragement throughout the process of researching and writing my thesis. This accomplishment would not have been possible without her. Thanks, Gaby!

To my parents, Raudel and Leonor that did not stop worrying about it for a second and provided me with a handful of great advice. Thanks, mom, and dad.

To all my family in general, my siblings Diani and Emy for their many talks, love and kind words of support, my grandma Mami, my nephew Andre and my niece Anabel, my aunt Mercy, grandma Mini and cousins Caro, Patry, Sergo, and Marito.

To the other part of the family, my mother and father in law Any and Rudy, as well as all the others: Pipio, Tiali, Albertico, Bea, Luisi, Amaña, Abuela, and Abuelo.

To my great friends Rene and Maire for all we have experienced together during this process and their unconditional support and help any time I needed it.

To all my friends in general that in one way or another helped me: Randy, Hiram, Nacho, Otero, Valentín, Tania, Mercy, Mariana, Jean and to those I met on Canadian grounds: Rui, Coco, Thais, Felipe, Abdul, and Niteh. Thank you all guys!

To my favorite professors that contributed so much to the knowledge I acquired during this process: Lucia and Paola.

To my office neighbors that helped me with many different things: Laura, Silvie, and Veronique.

Finally, to all those who are not on this page because of some strange reason like I forgot :) and I will regret it later on,

Thank you.

Content

Introduction 1
1.1 Neighbor Discovery 2
1.2 Preliminaries 3
1.2.1 Assumptions 3
1.2.2 Time-Slotted Model (Listen-Listen Model) 3
1.2.3 Asynchronism 4
1.2.4 Talk-Listen (TMLL) Model 5
1.2.5 Channel Occupancy Rate (COR) 7
1.2.6 Evaluation Metrics 7
1.3 Motivation 8
1.4 Contributions 9
1.5 Thesis Organization 9
Background and Related Work 11
2.1 Pairwise-Based Protocols 11
2.1.1 Birthday 12
2.1.2 Quorum 13
2.1.3 Disco 14
2.1.4 U-Connect 15
2.1.5 Searchlight 16
2.1.6 G-Nihao 19
2.2 Group-Based Protocols 20
2.2.1 Acc - Generic On-Demand Accelerations for Neighbor Discovery in Mobile Apps 20
2.2.2 Group-based Neighbor Discovery in Low-duty-cycle Mobile Sensor Networks 21
2.2.3 Collective Neighbour Discovery in Wireless Sensor Network 22
Talk Half Listen-2-Half (THL2H) 24
3.1 Main Protocol 24
3.1.1 General Idea 25
3.1.2 Model 27
3.1.3 Beaconing on-demand 30

3.1.4 Asymmetry 32
3.2 Extended Protocol 33
3.2.1 Average number of new neighbors per slots (β) 33
3.2.2 General Idea 33
3.2.3 Model 34
3.2.4 Asymmetry 38
3.3 Impact of β on COR 38
3.3.1 Theoretical worst-case scenario 38
3.4 Transmission Overhead 40
3.4.1 Necessary information 41
Evaluation and Discussion 43
4.1 Simulation scenarios and platform 43
4.2 Symmetric Discovery 44
4.2.1 Group simulations 47
4.3 Asymmetric Discovery 48
4.3.1 Issues of asymmetry on group simulations 51
Conclusions and Future Work 52
5.1 Summary of contributions 52
5.1.1 General Results 53
5.2 Future work 55
References 56

List of Figures

Figure 1.1 Aligned slots for X and Y with different schedules 4
Figure 1.2 Slot un-alignment example 5
Figure 1.3 Talk-Listen model example 5
Figure 2.1 Matrix representation of a Quorum schedule with m = 5 13
Figure 2.2 Disco schedule with 5 and 7 as base primes 15
Figure 2.3 U-Connect schedule with p = 3 16
Figure 2.4 Matrix representation of a Searchlight schedule with t = 7 17
Figure 2.5 Matrix representation of the G-Nihao schedule with m = n = 4 19
Figure 3.1 Cumulative Latencies of the one-way discovery of the G-Nihao protocol 25
Figure 3.2 Matrix representation of the THL2H Fixed schedule for m = n = 4 27
Figure 3.3 Relative Offset between two nodes 28
Figure 3.4 Cumulative latencies of the first contact between two nodes 29
Figure 3.5 Time intersection between A and B with a given offset 31
Figure 3.6 Matrix representation of THL2H Extended Fixed schedule with m = n = 8 34
Figure 3.7 Possible time displacement when listening in Listen-Listen Model 35
Figure 3.8 Possible time displacement when listening in Talk-Listen Model 36
Figure 4.1 Fraction of Discoveries by latency (a) 44
Figure 4.2 Average Discovery Latency (b) 45
Figure 4.3 THL2H and THL2H Ext. vs GNihao with different COR usages 46
Figure 4.4 THL2H COR usage worst-case vs simulation 47
Figure 4.5 THL2H Extended COR usage by β 48
Figure 4.6 Fraction of Discoveries by latency (a) 49
Figure 4.7 Avg. Discovery Latency (b) 50

Chapter 1

Introduction

With the proliferation of portable computing platforms and small wireless devices, mobile ad hoc networks (MANETs) [1] and wireless sensor networks (WSNs) [2] have received more and more attentions as a means of data communication among devices regardless of their physical locations. However, due to the slow advance in battery technology, power remains a bottleneck that limits wide applications running in this type of environment where there is no power supplies available for recharging. Hence, wireless devices usually rely on portable power sources (batteries) to provide the necessary power. Among all extensive studies on minimizing power consumption, neighbor discovery is one of the fundamental components focusing on communication and access.

On one hand, MANETs consist of devices that are autonomously self-organizing, whereas most of the today's wireless communication depends on expensive, centrally deployed hub-and-spoke networks. Such a large degree of freedom and self-organizing capacities make them especially suitable for environments and situations in which pre-defined network infrastructure goes out of service or does not exist at all. One typical application scenario would be disaster areas where communication between rescue workers, search teams, and medical staff needs to be established in spite of the destruction of network infrastructure. On the other hand, WSNs are composed of low-power wireless sensors with data collection, processing, and transmission capacities. They are particularly attractive to those applications that need to collect and analyze environmental data (e.g., temperature, humidity, or concentration of carbon dioxide) in a large area. Traditional applications of WSNs include habitat monitoring, environmental monitoring, and sea monitoring. In addition, newly emerging applications such as hiker logging, object tracking, and social networking are penetrating into our daily life [3].

1.1 Neighbor Discovery

Discovery among neighboring nodes or neighbor discovery serves as a prerequisite for both types of networks in order to achieve connectivity. Only after an initial discovery can a node set up communication with others. However, the difficulty arises when trying to achieve the discovery among low-powered devices without any sort of previous knowledge. The latter refers to the assumption of nodes not having any information about their neighboring nodes such as position, power usage or further transmissions. The ultimate goal is to minimize the discovery latency, a term that refers to the time taken for making contact with neighboring nodes.

In order to reduce power consumption, nodes enter into a low-power sleep mode whenever possible. In this state, a node is neither able to transmit not to listen. On the contrary, nodes stay into what is called an active state whenever listening and/or transmitting. As the amount of energy used in the sleep mode is infinitesimal when compared with that of an active state, the former is not taken into account. The power usage is defined by the fraction of time consumed in active state over the total time. However, it is more common to see it as the percentage of time a node spends in active state, known as duty cycle (e.g. 5%). In practical applications, nodes turn the radio on/off from time to time to satisfy a given energy constraint, being the percentage of time that the radio is on, the duty cycle.

Among the extensive research literature on neighbor discovery duty cycle and discovery latency are usually of top concern. Generally, worst-case and average discovery latency have been adopted by most research work. In any case, it is desirable to have a low duty cycle and minimum discovery latency; having low average discovery latency also a point of interest. It is not difficult to observe a trade-off between duty cycle and discovery latency. A lower duty cycle usually leads to a higher discovery latency and vice versa. In fact, from the perspective of energy efficiency, duty cycle corresponds to the “energy” aspect, while discovery latency corresponds to the “efficiency” aspect. Therefore, how to balance these two conflicting metrics becomes the key to achieve energy efficiency.

As nodes may be assigned with tasks of different energy requirements or left with different energy budgets, limiting all nodes to run at a given duty cycle is too restrictive and inefficient [3]. That is why it is preferable to work with different duty cycles, a term known as asymmetry because of the asymmetric duty cycles. On the other hand, it is also not practical taking into account a perfect synchronization among all nodes; global synchronization among all nodes in a sensor network has been demonstrated to be difficult and energy-expensive [5]. That is why, it is also important to take the complexity of asynchronous clocks into consideration.

1.2 Preliminaries

Before delving into specific Neighbor Discovery Protocols there are several basic and common concepts and assumptions that are key to its understanding.

1.2.1 Assumptions

1) Bi-directional links: all nodes have the same transmission range. In this way, the protocol design is simplified as we only need to focus on how a node can discover others, and the inverse direction is assumed. Note that this may not hold in practical applications because *x* could take *y* as its neighbor, but *y* is unaware of *x*'s presence due to the shorter communication range of *x*.
2) No decoding failures: all transmitted messages are received without any type of errors (e.g collisions). Again, in real-world situations, there may be transmissions errors (i.e. collisions).
3) Perfect timing: We assume no warm-up delay from *idle* to *active* state and no clock drift. It is assumed that clocks of all nodes run at the exact same pace.

1.2.2 Time-Slotted Model (Listen-Listen Model)

A time-slotted model (also known as Listen-Listen model) is usually adopted for analysis simplicity. Continuous time is separated into discrete interval, called slot, whose length should be enough for basic communication (or for neighbor discovery at least). A node decides to be active or idle in any given slot. In an active slot, a node transmits and listen, in an idle slot it does nothing (stays in a power saving mode). The time slot is usually referred to as the value of a

fictional counter. The counter may start counting from 0 when the node is powered on (not to be confused with active/idle states), and increases by one every slot.

If we assume slot alignment as in Figure 1.1 Aligned slots for X and Y with different schedules (when slot boundaries are aligned), the formulation of neighbor discovery can be defined as follows:

Let x, y be nodes. x and y are within the transmission range of each other but have not yet discovered each other. If x is active at slot $k1$ and y at slot $k2$, and these two slots (fully) overlap, they discover each other. However, slot alignment is rarely among nodes and a solution to handle it is needed.

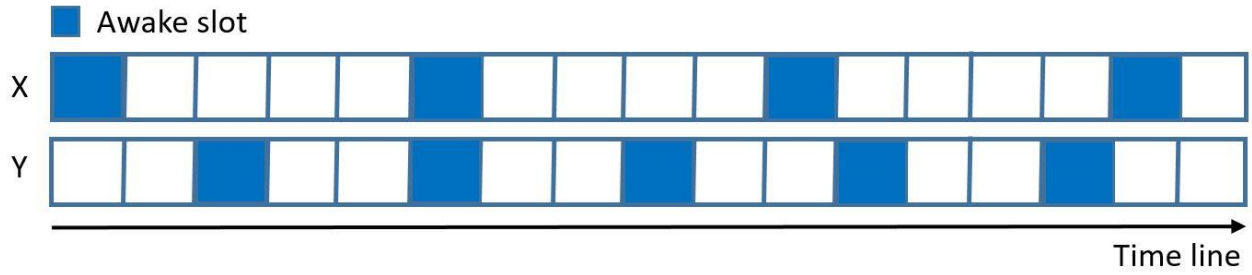

Figure 1.1 Aligned slots for X and Y with different schedules

1.2.3 Asynchronism

As nodes work independently and do not set up a global time reference, they may have to discover each other with a displacement among slot boundaries known as asynchronism. As a result, one node may not be able to discover the other even if their active slots overlap. As we can see in Figure 1.2 Slot un-alignment, when node x is active for the second time, node *y*'s state changes from idle to active in the middle of x's active slot. Hence, node *x* can successfully receive the beacon from node *y* and recognize *y* as its neighbor but *y* is unaware of *x's* presence as the beacon from *x* arrives at *y* when *y* is idle. The proposed solution for this problem was to transmit a beacon at both the beginning and end of a slot [6]. In this way, if there is an overlap between two active slots between neighboring nodes, one either listens to the beacon sent at the beginning of the active slot or the one sent at the end (depending on the relative positions of each other).

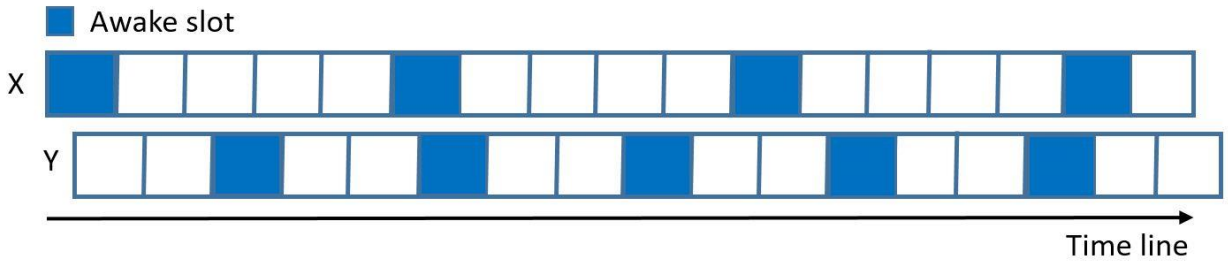

Figure 1.2 Slot un-alignment example

1.2.4 Talk-Listen (TMLL) Model

Most neighbor discovery protocols have been designed under the traditional model seen in Section 1.2.2. In this case, whenever a node is active, it transmits and listens within the same slot. Over the years, authors have proposed slight changes; for example, Blind Date [5] proposes a solution where beacons are not necessarily placed in the wakeup slot but next to it, and Lightning [28] defines a new type of slot where there is a beacon part and an idle part. However, it was in Talk More Listen Less: Energy-Efficient Neighbor Discovery in Wireless Sensor Networks [6] where nodes are allowed to beacon anywhere outside from the wake-up slots. The idea comes from the fact that a short beacon is more energy-efficient than an active slot, hence, idle-listening can significantly be reduced by decreasing wake-up slots and increasing the number of beacons sent. We can see this approach as a separation of the active slot into two: an active listening slot and an active transmitting slot. In this way, a node can decide whether to listen or to transmit (may be at the same time, see Figure 1.3 Talk-Listen model Example).

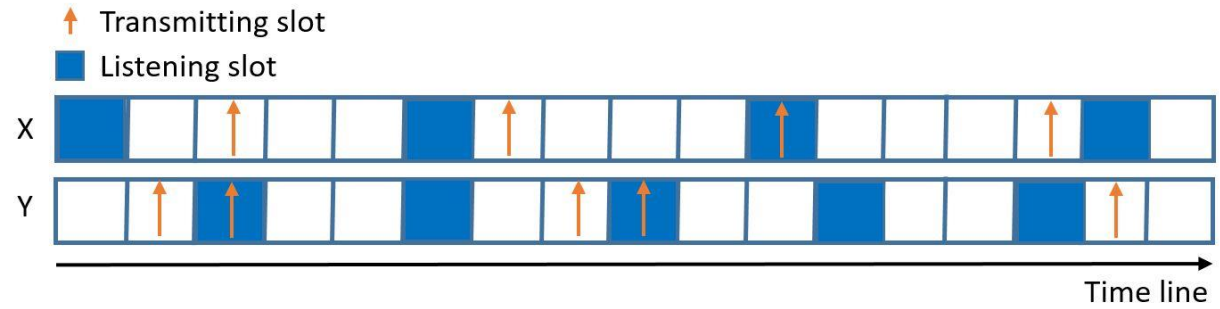

Figure 1.3 Talk-Listen model example

The discovery schedule of a node m in Talk-Listen model is defined as two binary functions: $\psi L(m,t)$ and $\psi B(m,t)$, representing the schedule of wakeup slots and beacons at time t, respectively [8].

$$\psi L(m,t) \begin{cases} 1, \text{listen for a slot} \\ 0, \text{sleep} \end{cases}$$

$$\psi B(m,t) \begin{cases} 1, \text{listen for a slot} \\ 0, \text{sleep} \end{cases}$$

In this way, neighbor discovery can be defined with $\psi L(m,t)$ and $\psi B(m,t)$. If node m1, m2 can directly communicate with each other, a unidirectional neighbor discovery meaning m1 discovers m2 is defined as:

$$\exists t |\, \psi L(m1,t) \;=\; \psi B(m2,t) \;=\; 1$$

Applying the analog reasoning, a bidirectional neighbor discovery meaning m1 and m2 discover each other is defined as:

$$iff: \exists t1, t2 |\, \psi L(m1,t1) \;=\; \psi B(m2,t1) \;=\; 1 \; and \; \psi B(m1,t2) \;=\; \psi L(m2,t2) \;=\; 1.$$

Since the concern in applications that involve neighbor discovery is energy-efficiency and discovery rate, the key metrics considered by existing discovery protocols are duty-cycle (DC) and worst-case discovery latency (L). When representing them in this model, the duty-cycle DC of a given periodic discovery schedule with period T is:

$$DC \;=\; \frac{1}{T}\Big(\sum_{t=0}^{T-1} \psi L(m,t) + \; \alpha\Big(\Big(\sum_{t=0}^{T-1} \psi B(m,t) - \; N_c\Big)\Big)$$

where Nc represents the number of common active slots that satisfy:

$$\psi L(m,t) \;=\; 1 \text{ and } \psi B(m,t) \;=\; 1$$

Subtracting Nc is to avoid double counting when wakeup and beacon occur in the same slot. α is the proportion between the time for transmitting a beacon and the duration of a wake-up slot.

None of the existing discovery protocols has considered α, since each beacon is placed inside the wakeup slot. However, α cannot be ignored when beacons are separated with wakeup slot in the Talk-Listen model. Although the beacon is a short packet that can be broadcasted in less than 1ms with an IEEE 802.15.4 compatible radio, it will dominate the duty-cycle especially when the TMLL principle is aggressively adopted [8].

1.2.5 Channel Occupancy Rate (COR)

COR measures the degree a discovery protocol occupies the channel or in other words, the fraction of the time that the channel is occupied. In a discovery schedule cycle, COR is defined as:

$$COR = \alpha * \frac{N_B}{T}$$

where $N_B = \sum_{t=0}^{T-1} \psi B(m,t)$ and since in each slot exists at most one beacon in Talk-Listen model, a simplified η is used to represent the COR:

$$\eta = \frac{COR}{\alpha} = \frac{N_B}{T}$$

1.2.6 Evaluation Metrics

Applications involving neighbor discovery concern about energy-efficiency and discovery rate. The main metrics considered by existing discovery protocols are duty-cycle (DC) and worst-case discovery latency (L).

Kandhalu et al. proposed a composite metric: Λ, called the power-latency product [8], which is the product of the average power consumption (i.e., duty cycle) with the worst-case discovery latency (L). In general, it serves as a good metric as when either factor is held constant, and the other needs to be minimized. The power-latency product is defined as:

$$\Lambda = DC \cdot L$$

This metric was widely used to compare discovery protocols on the Listen-Listen Model but it does not reflect the overall performance of those in Talk-Listen Model. Because of that, G-Nihao

[4] introduces another metric: A, as the product of duty-cycle, worst-case latency and COR. For a given periodic discovery schedule, A is defined as:

$$A = DC * L * \eta$$

We adopt the A metric to analyze the performance of our proposal. For the sake of clarity, we suppose the parameters are sufficiently large so that the constant terms and floors/ceilings in original expressions can be omitted. Finally, we should say that the main drawback of this metric is that it does not take into account the average latency performance as well as it does not support asymmetry.

1.3 Motivation

The Talk-Listen Model for neighbor discovery protocols serves as a base ground for the design of new neighbor discovery protocols. The new concept of Channel Occupancy Rate along with the beaconing-listening separation turns this model into a more powerful framework than the previous Listen-Listen Model.

We list the main points that drive the development of this research below:

- The G-Nihao protocol is a pure pairwise-based protocol, and it does not make use of any group-acquired information in such scenarios.
- Analysis of how G-Nihao achieves the first contact between a couple of nodes highlighted a significant percentage with high latencies.

Finally, even when G-Nihao improved all previously known neighbor discovery protocols, we believe that there still are many previous ideas that can be used, modified and adapted for its improvement, as well as new ones that may come up along this process.

1.4 Contributions

In this thesis, we study different neighbor discovery protocols and their approaches to develop better solutions for the neighbor discovery problem. The major contributions for its improvement are listed below:

- The addition of two new group-based protocols for the TMLL model where the COR usage is not fixed. Our protocols use the approach of mutual assistance while keeping the energy for listening constant. The first version we propose uses mutual assistance with a strategy for keeping the limit on the COR usage below twice the initial one.
- A second version approaches the problem with a different strategy: it does not restrict the number of possible extra beacons while rearranging the listening slots differently.
- A new metric for simulation networks (β) which sets the bases for comparison purposes in any future work.
- Two theorems presenting the upper bound of our protocol's average COR usage depending on the metric β.

1.5 Thesis Organization

This thesis is organized as follows:

- Chapter 1 introduces us to the problem of Neighbor Discovery in Wireless Sensor Networks and Ad-Hoc Wireless Networks. It also presents the preliminaries needed for a better understanding of the main neighbor discovery protocols that have been designed including the ones presented in this work.
- Chapter 2 provides the current state of the art of neighbor discovery protocols divided into two main approaches: pairwise-based and group-based. This chapter delves into the details of the different approaches there are as well as the main drawbacks of each of them.
- Chapter 3 introduces our main work: two new protocols called THL2H and THL2H Extended and the new metric β. It discusses the key elements used for the design of the proposed protocols as well as an analysis over their relationship with β.

- Chapter 4 presents the results obtained through simulations divided into symmetric scenario and asymmetric scenario. For each of them, we evaluate the performance of a single couple of nodes and also that of groups of nodes along with the impact on the COR usage of the presented protocols. This chapter also contains a small introduction to our simulation environment and the key metrics used for comparison.
- Chapter 5 gives the concluding comments of our work and presents our main findings as well as its possible future extensions.

Chapter 2

Background and Related Work

This chapter provides a background of neighbor discovery protocols for Ad-Hoc Wireless Networks and Wireless Sensor Networks. We explain some of the most revolutionary protocols divided into two main approaches: pairwise-based and group-based. Section 2.1 delves into the details of pairwise-based protocols while Section 2.2 discusses the approach of group-based protocols. The ideas behind all of them are the core principles of this thesis.

2.1 Pairwise-Based Protocols

The main focus of pairwise-based protocols is the discovery between a single couple of nodes. This does not mean these protocols are not supported in scenarios with groups of nodes, in fact, most of them behave the same for such scenarios. The theoretical analysis of the general performance of the protocols can be easily verified via simulations where all possible encounters are generated and the simulator records all the necessary data for each of them. Generally, pairwise protocols do not add extra information to default beacon, resulting in very little transmission overhead.

Through the years, there have been several proposals for asynchronous neighbor discovery that can work under both symmetric and asymmetric duty cycles. For example, a family of Birthday protocols [9] is representative of probabilistic designs where there is no latency bound guaranteed for the discovery. Because of this, latest attempts have been focused on deterministic protocols where there is a well-defined worst-case latency. For example, Zheng et al. [10] applied optimal block designs using different sets that are obtained by the Multiplier Theorem. Moreover, prime-based protocols as Disco [5] and U-Connect [8], provide a latency

bound by virtue of the Chinese Remainder Theorem. Another state-of-the-art deterministic protocol is Searchlight [11] which achieves high efficiency by leveraging constant offset between periodic active slots. Hello [12] provides a generic framework that incorporates existing deterministic protocols such as Quorum, Disco, U-Connect, and Searchlight and by exploring its parameter space finds optimal parameters for symmetric and asymmetric duty cycles. In other works such as Blind Date [6] and G-Nihao [7] beacons are not necessarily placed inside the active slots, however, it is the latter who proposes the complete independence of beacons and listening slots, leading to a redefinition of the listen-listen model. Finally, Lightning [28] applies mutual assistance to achieve two-way discovery, a concept described as: after one-way discovery, the bi-directional discovery can be achieved easily by the node receiving the packet proactively replying in the sender´s next available listening slot.

2.1.1 Birthday

The Birthday protocols [9] was one of the first protocols for neighbor discovery to come up. It is based on the Birthday Paradox [13] in which it is computed the probability that at least a pair out of a set of *n* randomly chosen people has the same birthday. While apparently, the probability reaches 100% when *n* reaches 367 by the pigeonhole principle, it is surprising that 99% can be reached with just 57 people and 50% with only 23 people. In the same way, the idea works as follows: two nodes independently and randomly select *k* slots over a period of *n* slots, one for transmitting beacons and the other for listening. They both remain idle in the remaining *n−k* slots. Under this scenario, the probability that the listening node can hear the transmitting one is given by

$$P(n,k) = 1 - \frac{\binom{n-k}{k}}{\binom{n}{k}}$$

The probability approaches 1 when the ratio $\frac{k}{n}$ is relatively small. For example, *P(1000,70) ≈ 0.995*. That is, a very high probability (99.5%) of discovery can be achieved in the presence of a low duty cycle (7%).

With such observation, the Birthday protocols work as follows. At the start of each slot, a node chooses with probability *pl*, *pt*, and *ps* whether the state for that slot is to be listening, transmitting, or sleeping (idle) respectively. For the purpose of saving energy during the deployment of nodes and maximizing the probability of discovery.

Given the probabilistic nature, it is natural to look at its average latency. For simplicity, we consider the case where nodes transmit and listen in an active slot with probability *p*. Then the probability that their first discovery occurs in the n^{th} slot is $(1-p^2)^{n-1}p^2$. The authors [9] state that the average latency expected is $\frac{1}{p^2}$ (e.g a duty cycle of 5% yields an average latency of 400 slots). In general, the Birthday protocol can achieve fast neighbor discovery in the average case because its probabilistic nature makes discovery independent of time displacement. Besides, the median latency would be shorter than the average latency due to the unbounded worst-case latency.

2.1.2 Quorum

Another very well known protocol and also one of the first to come up was Quorum [14]. The protocol arrange time in a sequence of time slot groups containing m^2 consecutive slots. In each group, we arrange the m^2 slots as an *m×m* matrix in a row-major manner. A node arbitrarily picks one column and one row of entries as active slots (called quorum intervals), while the remaining $m^2 - 2m + 1$ slots are idle slots. In Figure 2, it can be seen the schedule of a node that selects the 2nd row and 4th column, from the 5×5 matrix, to be awake.

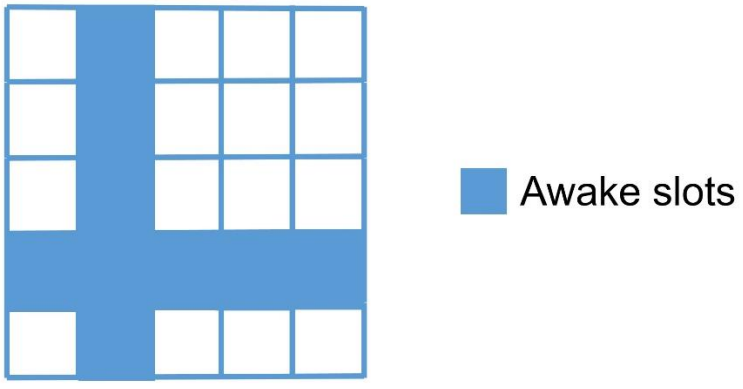

Figure 2.1 Matrix representation of a Quorum schedule with m = 5

Given two nodes that are perfectly time-aligned, their quorum intervals always have at least two intersecting slots because a column and a row in a matrix always have an intersection. The case becomes somewhat complicated when the nodes are not time-aligned. Still, the authors [14] proved that the Quorum protocol guarantees at least two overlapped active slots in every m^2 slots, no matter what the time displacement is. While the Quorum protocol provides a reasonable bound on the worst-case latency, it may perform much worse than the birthday protocols in the average case due to the redundant intersection. Besides, as *m* is a global parameter, it only supports symmetric operations, i.e., all nodes must operate at the same duty cycle.

Later, Lai et al. [15] improved the Quorum protocol so as to handle asymmetric cases in which two duty cycles are allowed. However, it can be said that it is too restrictive. Luckily, plenty of protocols with bounded latency and able to handle several different duty cycles have come up over time, Disco [5] being one of the first to achieve it.

2.1.3 Disco

In this prime based protocol, a pair of prime numbers $(p1, p2)$ is chosen to determine the neighbor discovery schedule. Whenever the slot number modulo any of them is zero, nodes wake up. The duty cycle can be set equal to the sum of the reciprocals of the prime numbers selected (i.e $DC = \frac{1}{p1} + \frac{1}{p2}$). Based on the Chinese Remainder Theorem, two different devices have a bounded discovery latency equals to the product of the primes $(p1 * p2)$ for the symmetric case and equals to $\min(p1, p2) * \min(p3, p4)$ for the asymmetric case (assuming $p1, p2, p3\ and\ p4$ are all different).

Figure 3 gives an example of a Disco schedule using 5 and 7 as parameters.

Figure 2.2 Disco schedule with 5 and 7 as base primes

As different couples of primes can lead to approximately equal duty cycles, there may be many ways to pick the parameters for a given duty cycle. Depending on whether it is for symmetric or asymmetric discovery, the authors suggest different ways to do the selection in order to achieve the lowest possible latency. On one hand, they call *balanced primes* to a couple of primes where the difference between them is minimal for a given duty cycle, e.g (37, 43, for a duty cycle of 5%). On the other hand, they call *unbalanced primes* when this difference is maximal, e.g (23,157) for the same 5% duty cycle. Notice that $\frac{1}{37}+\frac{1}{43} \approx \frac{1}{23}+\frac{1}{157}$.

In order to get the best performance out of it, the authors analyzed the best parameter selection depending on the scenario and proposed the use of *balanced primes* for the symmetric case and *unbalance primes* for the asymmetric case. These differences in performance between them make Disco a weak candidate on those scenarios where nodes pick their duty cycle by their need and two different nodes may pick the same one leading to possible symmetric and asymmetric situations at the same time. Clearly, one of them will be affected by the parameter choosing ending up in higher latencies on both worst-case and average case.

2.1.4 U-Connect

It is easy to see that when nodes work at different duty cycles, the best solution would be to pick just one prime instead of two because in the end the worst-case latency is given by the product of them two. The only problem comes when two different nodes pick the same one. In this case, considering that their schedules may not be aligned, they will maintain a constant offset between every awake slot, leading to no discovery at all. Based on this observation, the authors of U-

Connect [8], another prime based protocol, designed a new schedule to be able to use just one prime number.

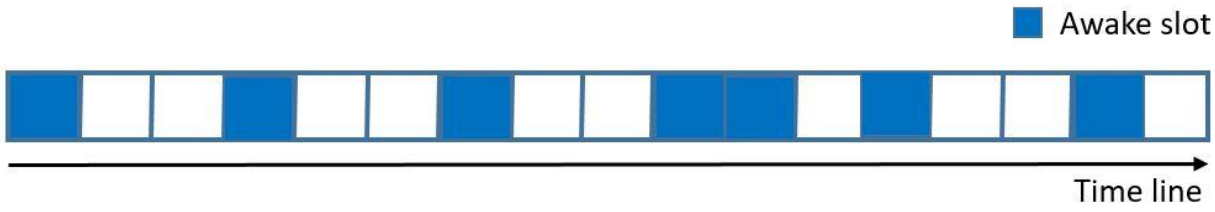

Figure 2.3 U-Connect schedule with p = 3

Nodes pick a prime number *p.* They stay active one slot every *p* slots and $\frac{p+1}{2}$ slots every p^2 slots. An example for p = 3 can be visualized in Figure 2.3. Basically, by adding this $\frac{p+1}{2}$ extra slots it is enough to have bounded discovery in p^2 slots for the symmetric case and $p1 * p2, (p1 \neq p2)$ for the asymmetric case. This gives U-Connect a duty cycle of $DC = \frac{3}{2p}$.

Even when U-Connect avoids Disco's parameter choosing and may improve its performance in a mixed scenario, it cannot beat Disco's worst case latency with unbalanced primes when only the symmetric scenario is considered. Moreover, both deterministic protocols (Disco and U-Connect) are considerably worse than the birthday protocol when comparing their average case latency.

2.1.5 Searchlight

The insight into the design of Searchlight [11] comes from the very simple observation that when two nodes wake up periodically with the same interval, the temporal distance between these awake slots always remains the same (assuming negligible clock drift). Searchlight leverages this constant temporal relationship to provide better worst-case bounds on discovery latency than previous existing protocols, and then improves the energy cost further by reducing the number of active slots required to ensure discovery by almost half by allowing active slots to "overflow".

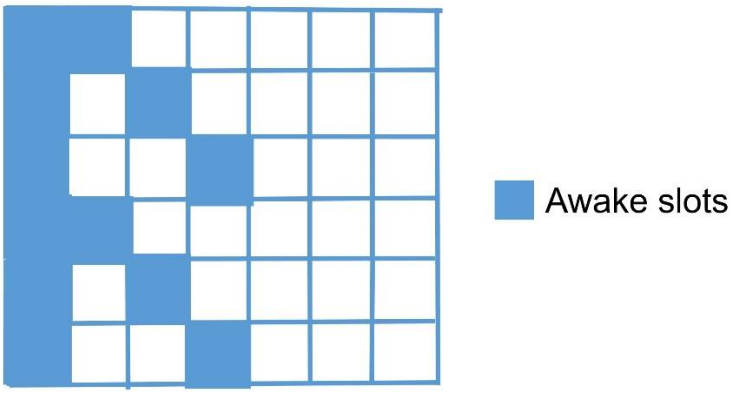

Figure 2.4 Matrix representation of a Searchlight schedule with t = 7

In Searchlight, each node has two active slots in every row. Originally, the authors used the word period and the letter *t* to refer to it, but as all the slots in *t* coincides with the span of a row in the matrix representation of a Searchlight's schedule (shown in Figure 2.4), we avoid the term period *t* as it can be confused with the more common used in the literature, period T.

The first slot in every row is called the anchor slot (it can be found as a *static slot* in the literature as well). In the symmetric case, any two *static slots* from two different nodes (one from each) only overlap if the difference between the start times is less than one time-slot. For all other offsets, the slots would never meet since the offset remains constant. As the relative position of the *static slot* of one node always remains the same with respect to that of the other node and is in the range [*1, t − 1*] a second active slot is used and it is called the *probe slot* (it can also be found as dynamic probe in the literature). This *dynamic probe* systematically searches for the anchor slot of the other node. The sense of movement is created when a node transmits it every *t+1* slots instead of every *t* (a span of a row).

A node transmits 2 times per row, one static and one dynamic and any overlap between *dynamic-static, dynamic-dynamic and static-static* result in discovery. However, the dynamic transmission should be restarted to position 1 relative to the period at some point. When to restart? Given that no two nodes can be offset by more than *t/2* slots, either one of them is in the first half relative to the other. Therefore, the dynamic probe only needs to go through the first half of the

t slots to guarantee an overlap with the static slot of the other node. Hence, it can be restarted to position 1 after *t/2* rows making the overall period $T = t\frac{t}{2}$.

Searchlight has a discovery latency for the symmetric case of:

$$L = t\frac{t}{2}$$

and a duty cycle of:

$$DC = \frac{2}{t}$$

Searchlight takes a novel approach to duty cycle asymmetry. It is possible to maintain the constant offset for the anchor slots of any two nodes if the larger valued period is an integer multiple of the smaller one. To see how this works, consider two nodes, node A with period *3t* and node B with period *t*. B's anchor slots always have the same relative distance from all of A's anchor slots. Since it only probes half of its slots, this ensures that the probe slot of A will eventually meet with any anchor slot of B that falls in the first half of A's cycle. In this example, B always probes the first $3 * \frac{t}{2}$ slots. Since A's anchor slots always align with slots 2 and 10 for B, discovery is guaranteed. This example can be generalized to A having any value $n * t$. While any multiple works for a given pair of nodes, to maintain the constant offset for all nodes, Searchlight restricts the duty cycle to a power-multiple of the smallest duty cycle (i.e. 2, 4, 8, 16 or 3, 9, 27, 81, etc.), guaranteeing that any two nodes' duty cycles are multiples of each other. This means that if t is the smallest duty cycle, the other nodes can select *2t, 4t, 8t*, etc, for any t.

The main drawback of Searchlight comes exactly from this approach for handling asymmetry, the requirement of duty cycles to be power-multiples of the smallest one results in only a small number of options available (e.g., at most four for a targeted span of duty cycles ranging from 1% to 10%). On the other hand, there are also redundant discovery opportunities within a latency bound making room for improvement.

2.1.6 G-Nihao

Based on the TMLL model (seen in 1.2.2) and born with it, this protocol [7] was designed to get the best of this approach as it is the first one to consider COR in neighbor discovery. With G-Nihao, the η can be flexibly adjusted to satisfy different requirements. G-Nihao's schedule is rather simple, it can be represented as a matrix of n rows and m columns where the first row is used for listening as shown in Figure 2.5. As the beaconing is set apart from the listening, there is one beacon every m slots, visually seen in the first column of its matrix representation. Because of that, the COR of G-Nihao is always:

$$\eta = \frac{n}{mn} = \frac{1}{m}.$$

As we can see, parameter n is more related with the duty cycle and m with the COR usage even when it also slightly affects the duty cycle. Both define G-Nihao's period $T = m * n$.

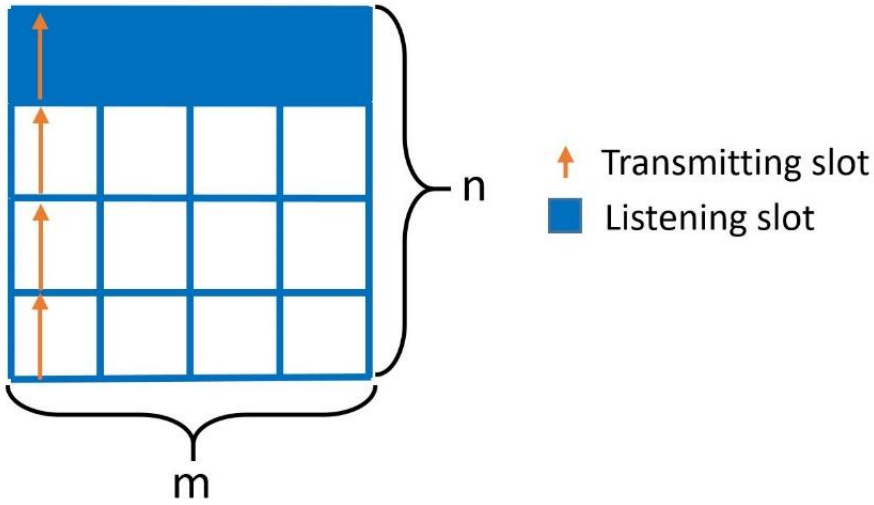

Figure 2.5 Matrix representation of the G-Nihao schedule with m = n = 4

The schedule of G-Nihao can be formally defined as:

$$\psi L(g,t) = \begin{cases} 1, \text{ if } [t]L < m \\ 0, \text{ otherwise} \end{cases}$$

$$\psi B(g,t) = \begin{cases} 1, [\mathrm{t}]L = m_i, \mathrm{i} = 0,1,\ldots,\mathrm{n}-1 \\ 0, \text{ otherwise} \end{cases}$$

The latency of G-Nihao is $L = m * n$, and the duty-cycle is:

$$DC = \frac{m + \alpha(n-1)}{mn} \approx \frac{m + \alpha n}{mn}$$

2.2 Group-Based Protocols

Group-based protocols do not approach the neighbor discovery problem in the same way pairwise protocols do. The main objective of the former is to incorporate any sort of data or information acquired by nodes while the discovery process is going on and use it for the improvement of the discovery latency of any neighboring node yet to discover. It is true that all nodes start this process without knowledge at all, however, the mere fact of discovering new neighbors is an acquired knowledge that can be broadcasted when beaconing, among many other information the nodes may want to share (schedules, list of current neighbors, duty cycle, etc.). Of course, beacons cannot longer contain its basic data packet as now they will be carrying all the extra information used by the protocols. This collaborative approach together and the use of extra information (shared while transmitting) is what set the main difference between group-based and pairwise protocols.

Many of this protocols are set on top of an underlying pairwise-based protocol. Because of their nature, the performance varies depending on the network properties and the specific protocol's approach. Generally, better results are obtained with a higher number of nodes [16] where more information is shared among them while pairwise scenarios are the most difficult to improve.

There are many different approaches on how to manipulate the information acquired. For example: coordinated synchronized listening [27], learning from spatial properties [17], formation of groups [18], etc. Below, we present the ones we consider the most revolutionary.

2.2.1 Acc - Generic On-Demand Accelerations for Neighbor Discovery in Mobile Apps

Acc is not a protocol, rather is a new layer that can be used on top of existing protocols. With Acc, an individual node can accelerate discovery in an on-demand and autonomous manner. Generally, protocols handle a need for a faster discovery by simply changing (increasing) the duty

cycle, with Acc, if a faster discovery is needed, an additional energy budget is used to set a number of new active slots. The key novelty of Acc is that it leverages knowledge in the neighbor table of known neighbors to maximize the utility of additional energy (i.e., the effectiveness of additional active slots) in order to accelerate the discovery of unknown neighbors, while also introducing no changes on any device except the discovering one. The authors also propose a novel concept of spatial-temporal coverage and formally define a model for quantifying the effectiveness of each active slot in the acceleration of neighbor discovery. The model is fully distributed and leverages only information in the neighbor tables of known neighbors. It does not make any assumptions regarding radio or location models. Based on the spatial-temporal coverage, a scheduling algorithm sets the additional active slots achieving improvements up to 50% [16].

The indirect discovery approach is basically the discovery of nodes through others. Hence, after the discovery process ends, some nodes may not have made contact. However, the idea is based on the known result that when nodes have equal transmission ranges, it is more probably for a neighbor of your neighbor to be also your neighbor.

The main idea can be seen clearly throughout an example. Suppose node X's active time slots are 0,7,14, node Y's active time slots are 0,2,4 and node Z's active time slots are 2,6,9. At global time 0, node X discovers node Y and node X includes the next active slot of node Y to its own schedule (as an additional active slot). So, node X would include slot 4 for possible indirect discovery. At global time 2, node Y discovers node Z, therefore, when node X become active in its additional time slot (4), node Y will forward node Z to node X.

The main drawback of this discovery process is that node X and Z may actually be out of range. When node X added Z as a new neighbor, X did it without actually listening to any of Z's transmissions, which is considered an indirect discovery.

2.2.2 Group-based Neighbor Discovery in Low-duty-cycle Mobile Sensor Networks

This work reduces discovery delay significantly by proactively referring wake-up schedules among a group of nodes. Since proactive references incur additional overhead, the authors introduce a

novel selective reference mechanism based on spatiotemporal properties of the neighborhood and the mobility of nodes. Quantitative analysis indicates that the discovery delay of their group-based mechanism is significantly smaller than that of the pairwise one. Experiments using 40 sensor nodes and extensive simulations showed up to one order of magnitude reduction in discovery delay compared with legacy pairwise methods in dense, uniformly distributed sensor networks with at most 8.8% increase in energy consumption [18].

This protocol uses a direct discovery approach. For example, suppose two nodes A and B discover one another during their active state. Then, they make a group and become aware of one another's active schedule. Let be C a third node whose schedule coincides with A much more earlier than with B; right after A and C discover each other, A sets up a new active slot in order to let node B know about node C. Finally, node B sets up a new active slot to discover C.

The main drawback of this approach comes from the fact that each node has to become active again in its existing neighbor's next active slot. So, this process adds redundant time slots by waking up to meet already discovered nodes.

2.2.3 Collective Neighbour Discovery in Wireless Sensor Network

In this work [19], collective neighbor discovery is proposed to reduce latency period and accomplish the discovery more efficiently. To achieve this purpose each node will be active during recommended neighbors' active time to attain rapid neighbor discovery. The original proposal sets the initial schedule randomly. However, this protocol can be combined with Searchlight and simulation showed how the combined protocol enhanced the performance and decreased the latency effectively.

The protocol defines a window of size n in each period, from which k slots are selected uniformly random as active slots for that period. When a node is in active state it can transmit and receive at the same time. When been active, each node broadcasts a message including its current neighbors and their next wake up time and receives messages from others simultaneously. For example, if the window size is 10, k = 3 and three nodes: A active at slots 1, 4, 9, B at slots 4, 6, 11 and C at slots 6, 10 and 13, Node A and B both are active at global time = 4 and they are in

transmission range of one another. So, they discover themselves and also retain record of each other next active schedule according to their own local time. At time = 6, node B discovers node C and recommends A (along with A's next active slot) to C. After that, C will be active at the proposed neighbor's next active slot number 9. Once they are both active, if they are in communication range then C adds A as neighbor. Otherwise, C includes node A to its out of range list so that in the future if any node recommends node A to C, the latter will ignore the recommendation.

Chapter 3

Talk Half Listen-2-Half (THL2H)

In this chapter, we present the reasoning behind the design of THL2H protocol. We breakdown the analysis into four main sections. Section 3.1 introduces a series of analysis over the G-Nihao protocol, the general idea and the first version of the protocol: THL2H. Section 3.2 presents an extended version of the protocol and introduces the metric β for the first time. Both Section 3.1 and 3.2 also contain the key components of both proposed protocols such as their schedule, worst-case latency and COR usage. In Section 3.1 we present an analysis of the β impact on the COR usage of both versions, and finally in Section 3.4, we delve into the transmission overhead and compare it with that of other protocols.

3.1 Main Protocol

The TMLL model brings new possibilities to all protocols (either pairwise or group-based) designed within these rules. Now, nodes can use a given energy budget in two different ways: to listen more and/or to transmit more. Taking into account that the energy needed for beaconing is considered to be a fraction of that for listening, the use of a given budget may yield better results spent on transmissions than on listening.

In a group scenario, this led to the possible idea for nodes to variable increase the COR (i.e. send more transmissions per period, shown in Section 1.2.5) when necessary while keeping the same energy used for listening. On one hand, as previous group-based protocols have been based on the Listen-Listen model, they increase both the energy used for listening and transmissions as they cannot split them. On the other hand, pairwise scenarios may be ideal for not to increase the COR usage as each node only have to focus on a single neighboring node.

3.1.1 General Idea

When analyzing how discoveries between two nodes occur in G-Nihao, we focused on two main points:

1. The latency of the one-way discovery, in other words, the latency of the first node in listening to the other or latency of the first contact.
2. The latency between the one-way discovery and the two-way discovery which completes the discovery process (both nodes listened to each other) in a pairwise scenario.

When simulating all possible encounters between two nodes, we found first contact latencies beyond T/2 or more slots in 25% of them. The results, shown in Figure 3.1 belong to a couple of nodes running at 10% duty cycle (black), COR = 1/10 and T = 100 and another pair running at 5% duty cycle (grey), COR = 1/20 and T = 400. Notice how 75% of the discoveries occur with a difference of T/2 or less, leaving a margin of 25% with a difference of T/2 or more.

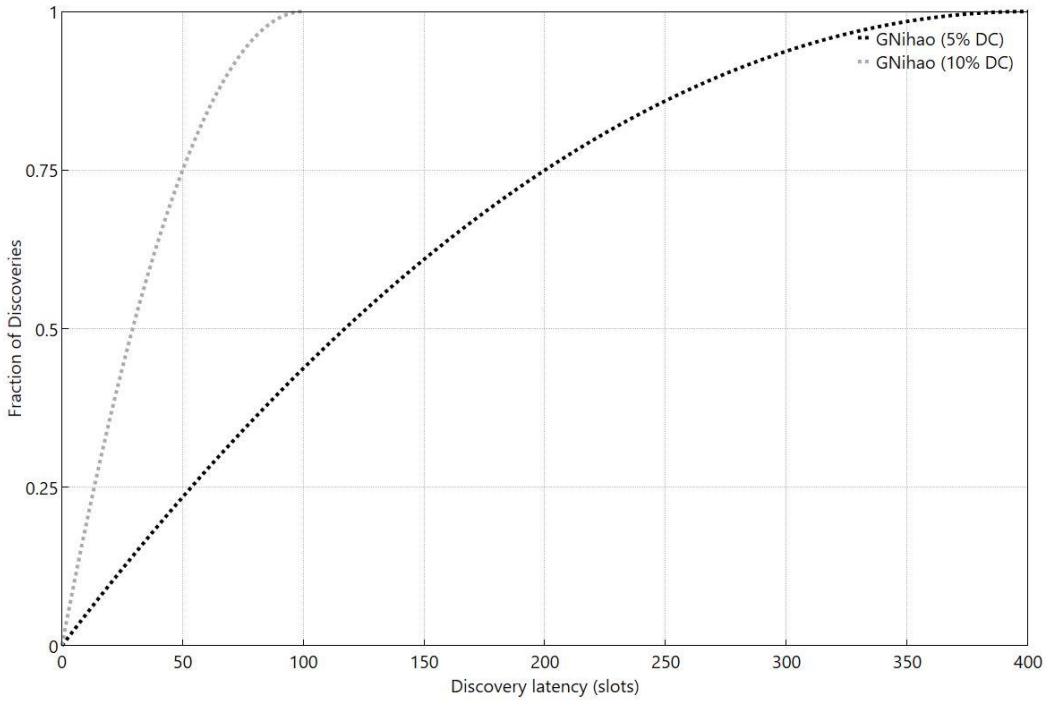

Figure 3.1 Cumulative Latencies of the one-way discovery of the G-Nihao protocol

We also recorded all the latencies between the one-way discovery and the two-way discovery and obtained the exact same chart. Meaning that in 25% of the discoveries, there is also a delay of at least T/2 or more slots between the time where node A listened to node B and the time when node B listened to node A.

After the previous analysis, we focused on a design that exploits those weak points of G-Niaho in order to improve the overall performance.

Retaking the idea of keeping the energy for listening untouched while only manipulating beacons, we consider the possibility of improving the latency of the two-way discovery, by an idea known as *mutual assistance* and that has been used for pairwise-based protocols for the listen-listen model [28]. Basically, an initial schedule that guarantees the one-way discovery is needed and then by sending an extra transmission the two-way discovery is achieved. To be able to send the reply, the transmitter only needs to know the receiver's next listening slot.

In order to differentiate between the transmissions belonging to the fixed schedule and the extra ones, the former are called *fixed* transmissions while the latter *dynamic* (or just extra) transmissions.

After the previous analysis, we have come to the conclusion that G-Nihao's schedule is not efficient in achieving a fast one-way discovery. Firstly, because as shown in Figure 3.1, the one-way discovery worst-case latency is as high as the two-way discovery latency. Secondly, as all listening slots are located in one row of the schedule (fig. 2.1.6), if we suppose the two-way discovery to occur at the last slot of the row, an extra transmission can only be sent at most m slots before. Moreover, as this happens after the one-way discovery occurs, it accounts for only 50% of the total discoveries.

So, in order to get substantial improvements, we present a different design that better exploits the aforementioned ideas: THL2H.

3.1.2 Model

Taking G-Nihao´s schedule as a starting point, we divide the listening row from G-Nihao into two listening halves. The first $m/2$ slots are kept untouched but the others are placed after $T/2 = m * n/2$ slots as seen in Figure 3.2. Then, we add a fraction of a slot (δ) after each half, big enough to be able to listen to a transmission. In this way, the fixed transmissions are kept untouched at the beginning of each row.

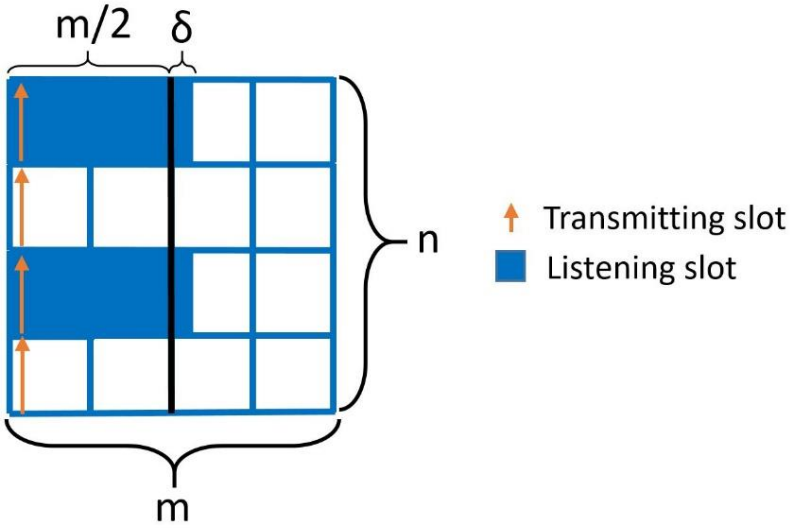

Figure 3.2 Matrix representation of the THL2H Fixed schedule for m = n = 4

From a node's point of view, all transmissions from any neighboring node with a relative offset greater than $\frac{m}{2} + \delta$ congruent m are missed. However, as the sum of the offsets is equal to m, then a node either listens to its neighbor or it is listened to by its neighbor. In this way, the one-way discovery is guaranteed between any pair of neighboring nodes.

Notice how the second half of every row is not covered by any listening slot (but for the small portion in δ at Figure 3.2). As the fixed transmissions are equally distributed, any neighboring node who's offset falls within the second half of any row cannot be listened. As this accounts for 50% the total number of possible offsets, it is easy to see that only one node in every pair is able to make contact with its neighbor.

This proof is similar to that in Searchlight. If we recall Searchlight (2.1.5). Any two Searchlight nodes with the same symmetric duty cycle satisfy that the sum of their relative offsets is equal to t (period in Searchlight), which implies at least one of the offsets is smaller or equal to $t/2$. In our model, the span of m coincides with the span of t in Searchlight's schedule. In addition, the fixed transmission slots coincide with the static slots in Searchlight. Hence, there will always be an overlap between one node's fixed transmission and the other's listening slot for all offsets but the case when both are equals to $T/2$.

As an example, Figure 3.3 shows how node A is able to listen to B's transmissions and B cannot. However, by moving B's offset one slot further, we fall into the exact opposite situation. It also shows how a node either listens to a fixed transmission or it is listened back making a one-way discovery between every pair of nodes always possible via their fixed schedules.

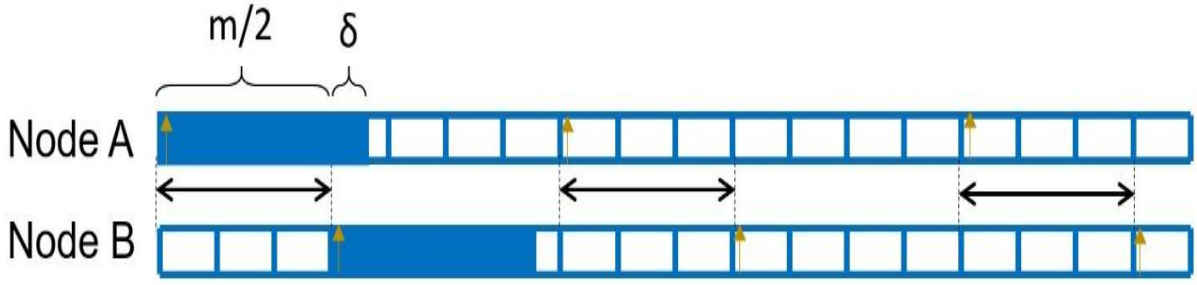

Figure 3.3 Relative Offset between two nodes

The addition of δ is only for handling the case when both nodes have the exact same offset equals to $m/2$ *mod.* m. Notice how if we remove it, both nodes would be just missing each other's transmissions. Another possible solution is adding a complete extra slot for listening but as we only need a fraction of a slot big enough as to listen to a single transmission, we proposed the former. We should say that for sufficiently large m, the use of an extra complete slot becomes infinitesimal when compared with the total power usage in a period (T). Finally, the expected power usage per period T is comparable with that of G-Nihao.

The presented schedule focuses on speeding up the one-way discovery. In order to analyze it, we recorded all one-way discovery latencies for all possible encounters between a couple of nodes and compare it with that of G-Nihao. The results are shown in Figure 3.4 for a couple of THL2H and a couple of G-Nihao nodes, both running at 10% duty cycle and a COR of $\eta = \frac{1}{10}$. We can see how the worst-case latency of our protocol is bounded by $\frac{T}{2}$ resulting in a lower average one-way discovery latency that settle the basis for the overall improvement we are looking for.

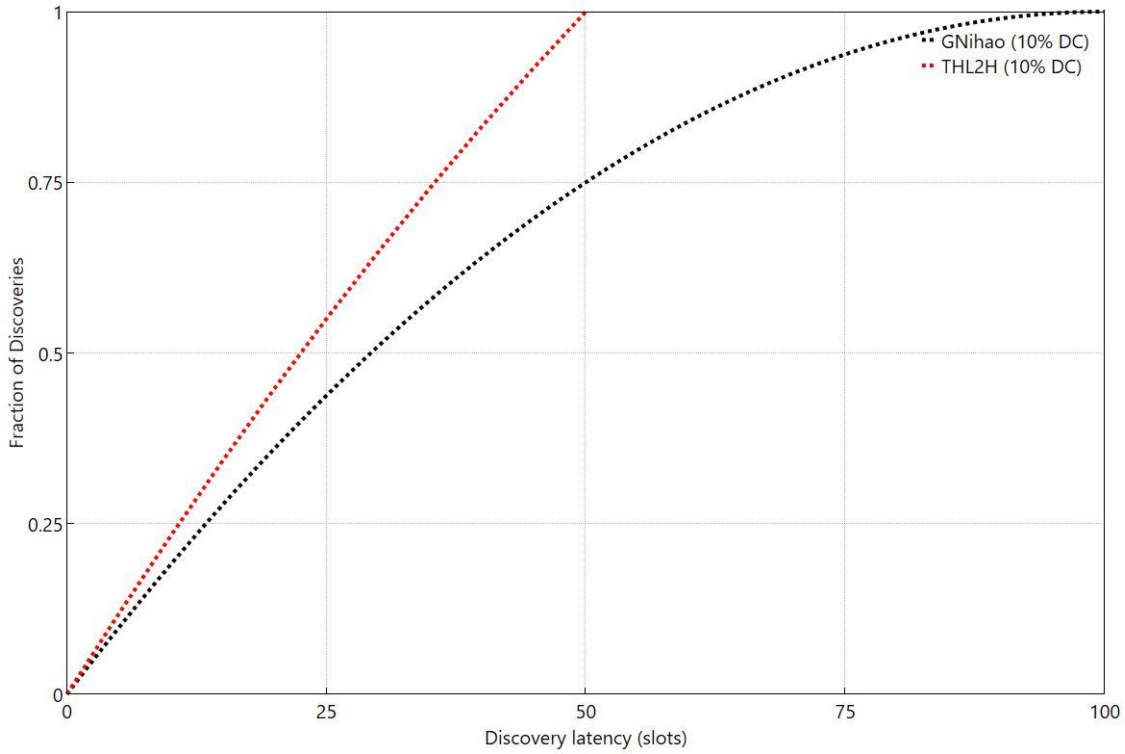

Figure 3.4 Cumulative latencies of the first contact between two nodes

Below, we present the formal definition of the presented schedule in the Talk-Listen model:

$$\psi L(g,t) = \begin{cases} 1, \text{ if } [t]T < \frac{m}{2} + 1 \text{ or } \frac{T}{2} \leq [t]T < \frac{T}{2} + \frac{m}{2} + 1 \\ 0, \text{ otherwise} \end{cases}$$

$$\psi B(g,t) \quad \begin{cases} 1, [\mathrm{t}]T = \mathrm{mi}, \mathrm{i} = 0,1,\ldots,\mathrm{n}-1 \\ 0, \text{ otherwise} \end{cases}$$

Here we use $[t]T$ notation to denote $t \ mod \ T$, which is the slot index in a schedule cycle.

The duty cycle is given by:

$$DC = \frac{m + \alpha n}{mn}$$

However, the previous schedule only guarantees the one-way discovery. Hence, a strategy for achieving the two-way discovery is still needed.

3.1.3 Beaconing on-demand

With the fixed schedule presented, there is such a small chance for a pair of nodes to listen to each other, that we will assume the one-way discovery is achieved by only one of them. Hence, this node has to send an extra transmission for the other to listen to it and complete the discovery process.

Instead of sending the reply at the first available listening slot, our solution consists in placing it at one of the slots within the middle column (Figure 3.5). The only question is, is there always a neighbors listening slot at that column? The answer is yes and the proof is rather simple. If a node listened to a fixed transmission, the transmitter is located with a relative offset smaller or equal to $\frac{m}{2} + \delta$. As fixed transmissions are all located in the first slots of every row, the first listening slot is located at the exact same offset (relative to m) as fixed transmissions are. As the listening slots span for half a row, then the middle slot of that row must be always covered by at least one of them.

Finally, the worst-case latency of THL2H is given by the latencies worst-case of the one-way discovery equals to $\frac{T}{2}$ plus the worst-case latency to reply. In this case, the worst-case placement for the reply, would be $\frac{n}{2}$ rows after the first contact, for a total of $\frac{T}{2}$ slots.

Hence, the worst-case latency in the symmetric environment is:

$$L = nm$$

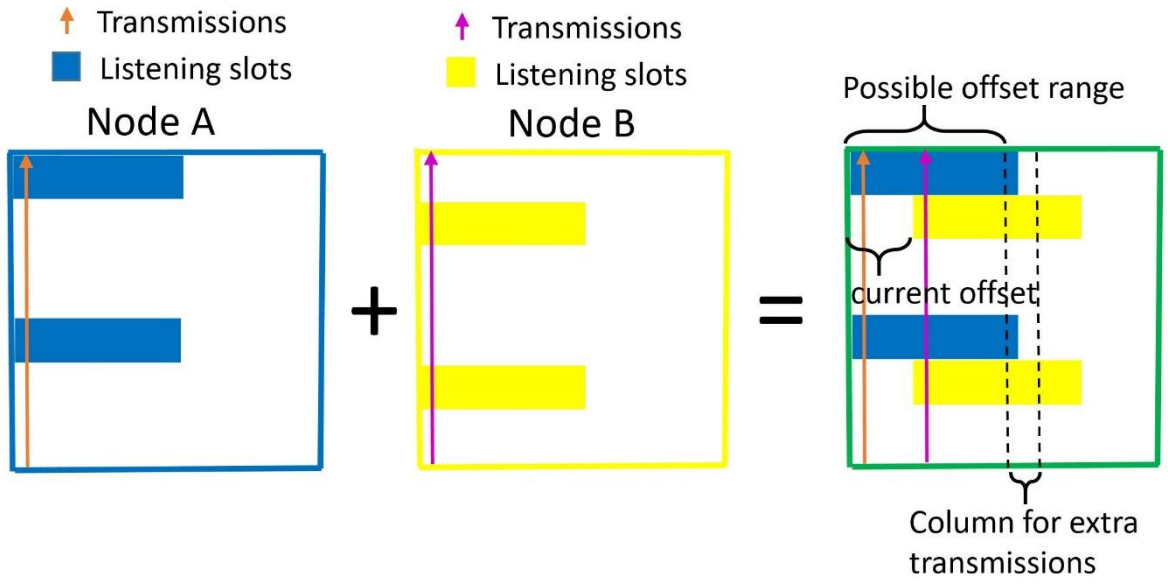

Figure 3.5 Time intersection between A and B with a given offset

Analysis on group environments

The main advantage of this approach comes in group environments where, all the replies are placed within a column, limiting the possible number of extra transmission per period T to a maximum of n (the number of initial transmissions in the fixed schedule). Therefore, doubling the COR usage in the worst-case. Moreover, as every time a new neighbor makes contact and the protocol schedules the reply, the probability of following neighbors to listen to one of the already placed replies increases. This is due to the fact that every extra transmission covers all possible neighbors with all the $m/2$ different offsets within the same row. As there are n different rows in a period T, the probability of an upcoming neighbor to listen to one of k replies is $\frac{k}{n}$.

The more replies are scheduled, the closer to a G-Nihao schedule with half the initial COR it gets. The transition goes from an initial COR of $\eta = \frac{1}{m}$ to a maximum of $\eta = \frac{1}{(m/2)} = \frac{2}{m}$. Hence, the impact of the maximum COR usage on the duty cycle is very little for small values of α (e.g. $\frac{1}{n}$).

Hence, the duty cycle varies (depending on the COR usage) in the range:

$$\frac{m\ +\ \alpha n}{mn} \leq\ DC \leq\ \frac{m\ +\ 2\alpha n}{mn}$$

and the performance metric A (defined in Section 1.2.6 as $the\ DC * L * \eta$ product) varies as well:

$$\left(\frac{m\ +\ \alpha n}{mn}\right)(nm)\left(\frac{1}{m}\right) \leq\ A \leq\ \left(\frac{m\ +\ 2\alpha n}{mn}\right)(nm)\left(\frac{2}{m}\right)$$

$$1 + \frac{\alpha n}{m}\ \leq\ A \leq 2 + \frac{4\alpha n}{m}$$

3.1.4 Asymmetry

Since heterogeneous devices are likely to have diverse energy budgets, it is reasonable that they operate with different duty-cycles. That is why supporting asymmetric scenarios is a desirable property to have on a protocol. Some protocols only focus on symmetric scenarios (Quorum [15]) while others handle it with restrictions (Searchlight [13]). In the case of G-Nihao, it supports the asymmetric case by simply adjusting the n value according to the required duty-cycle different nodes have. It guarantees discovery with different n_1 and n_2 as long as the m remains constant with a worst-case latency of $m * max\{n1, n2\}$ [8].

Our protocol handles asymmetry in the same way G-Nihao does: keeping m equal on all nodes. No matter the relative offsets between a pair of nodes, it always guarantees the first contact before half a node's period in the worst case. Hence, the reply adds up a number of slots equals to the other node's half.

Finally, the worst-case latency of the THL2H for the asymmetric case is:

$$L = m\ * (max\left\{\frac{n_1}{2}, \frac{n_2}{2}\right\} + min\left\{\frac{n_1}{2}, \frac{n_2}{2}\right\})$$

3.2 Extended Protocol

THL2H does not change its performance dramatically (in terms of the COR, hence, the performance metric A) as it is bounded by a maximum of up to double the number of transmissions in its fixed schedule. In this section, we present *THL2H Extended*, that does not limit the number of extra transmissions. This version is intended for pairwise scenarios and limited group scenarios where the number of neighboring devices is low. Part of this work is to be able to quantify the impact of the number of neighboring devices on the protocol's performance, specifically the COR usage.

3.2.1 Average number of new neighbors per slot (β)

In a real case scenario, the number of neighboring devices changes over time: some leave and new get within contact. So, the perception of how sparse/crowded the network is, comes along with a time variable.

First, we define as a new neighboring device, a non-discovered node that enters within range for any time > 0. Then, let be X the time (in number of slots) a node have been running in a given environment and K the number of new neighboring devices it experienced, we define β, the average number of new neighbors per slot as:

$$\beta = \frac{K}{X}$$

Notice that this metric is not about the number of neighbors a given node had (in a lifetime). It only focuses on how fast a node experienced the arrival of new neighbors. For example, for a network with $\beta = 0.1$, we can say that any node experiences one new neighbor every 10 slots.

3.2.2 General Idea

We can take the approach of sending extra transmissions to an extreme of having no boundaries on the number to be sent. Our extended protocol's main idea is to reply as soon as there is an available listening slot in the neighboring device´s schedule (mutual assistance [28]). This means there is no restriction on where to beacon (opposite to what we previously did).

3.2.3 Model

We noticed that the sparser the listening slots are, the more balanced is the distance between them, which in turn gives more opportunities all along the schedule to send the reply. The previous conclusion leads us to change the original fixed schedule of THL2H.

Ultimately, what we aim for is to re-accommodate THL2H´s fixed schedule without increasing the one-way discovery latency while spreading the listening slots more evenly. We present the new changes in Figure 3.6 for a node running at $1/n$ duty cycle and a COR of $\eta = \frac{1}{m}$, where $m = n$.

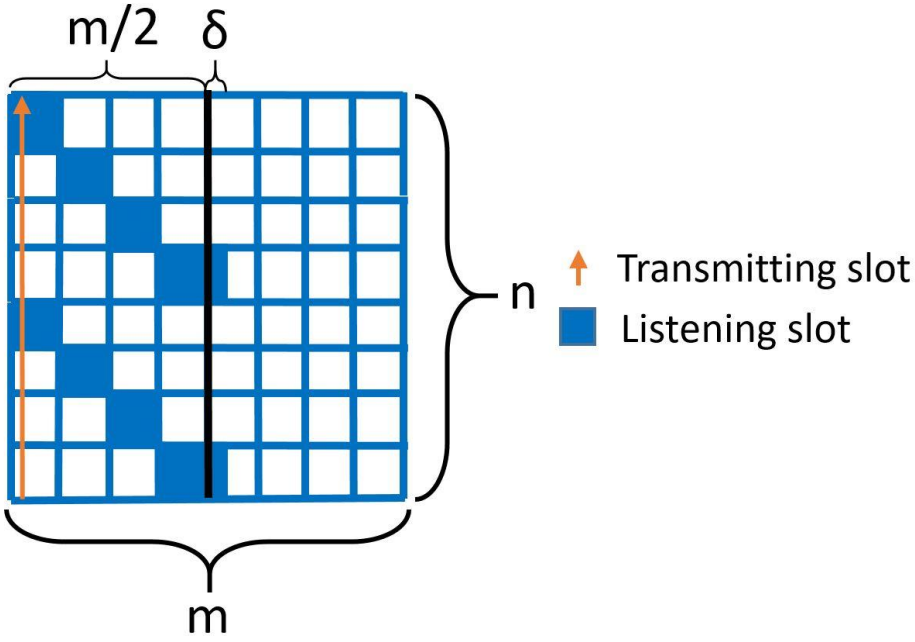

Figure 3.6 Matrix representation of THL2H Extended Fixed schedule with m = n = 8

The number of slots per column is still two, and also the columns covered by those slots are the same. The main difference this modification brings is that it has a higher number of rows where at least one listening slot lies. Because of this, the question that arises is: does the one-way latency gets affected by this modification? Luckily, it does not. In fact, these two fixed schedules are similar in terms of worst-case latency and average latency of the one-way discovery. The results obtained are the same than those in Figure 3.1. Therefore, the proposed schedule is only able to make contact with an average of half its neighboring nodes (same than THL2H). Hence,

the improvements we are considering now are only when taking into account the extra transmissions needed to handle the other half and complete the two-way discovery with lower latency.

Let us take the particular case of Figure 3.6. There is exactly one listening slot on each row and the maximum distance between one listening slot and the next one is m. Therefore, when a node calculates the next available slot of any of its neighbors, it always finds one in the following m slots. Hence, the discovery process ends at most m slots after the one-way discovery occurs. This example shows a worst case latency of $\frac{n^2}{2} + n$ as $m = n$. However, we do a full analysis further on.

One important point we have to address is the fact that this approach is based on being able to make contact with a single listening slot (in the worst case) of a neighboring node. Notice how in our previous design, this would only happen for $m = 2$, which was contemplated in the proof. In this case, we are interested in the particular case of a solitary slot that may not be aligned because of the known problem of asynchronism (Asynchronism, Section 1.2.3).

Handling asynchronism

The main problem we face is not related to knowing the exact amount of time displacement there is, is about the direction (left or right) of it. For example, in the listen-listen model, when a node listens to a transmission, it cannot figure out in which direction (left or right) the time displacement is. For a better understanding, in Figure 3.7, we show the two possible cases this can happen. Hence, the equivalent idea of sending only one transmission (on the Listen-Listen model) cannot be applied without any sort of previous information about the time displacement.

Figure 3.7 Possible time displacement when listening in Listen-Listen Model

In the case of listening to a transmission in the Talk-Listen model, there is only one possible direction for the time displacement. In Figure 3.8 we show how node B can only listen to node A if the time displacement relative to A is shifted to the left. Hence, if B wants to send a reply k slots after, it counts the slots relative to its own schedule and places the transmission at the end of its k^{th} slot as A's listening slot always covers this part.

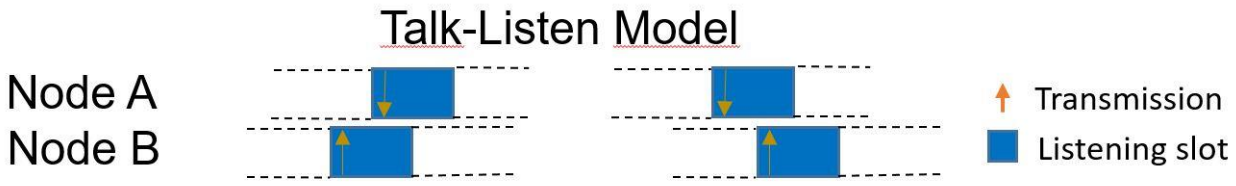

Figure 3.8 Possible time displacement when listening in Talk-Listen Model

The worst-case latency of THL2H Extended is given by the latency of the one-way discovery equals to $\frac{T}{2}$ plus the latency of the reply. As the m listening slots were equally distributed through the schedule, the distance between them is at most $\frac{T}{m}$.

Hence, the worst-case latency under a symmetric scenario is:

$$L = \frac{mn}{2} + \frac{mn}{m} = \frac{mn}{2} + n$$

As the duty cycle is slightly affected by the COR usage, there is a theoretically higher impact on the upper bound of the duty cycle for this version (in group environments). Clearly, an upper bound for the number of extra transmissions to be used is mn, in other words, one in every slot and the lower bound remains the same to that on a pairwise scenario.

Therefore, the DC of THL2H Extended lies within the range:

$$\frac{m + \alpha n}{mn} \leq DC < \frac{m + \alpha mn}{mn}$$

In the same way, the performance of the metric A varies as well:

$$\left(\frac{m\ +\ \alpha n}{mn}\right)\left(\frac{mn}{2}+n\right)\left(\frac{1}{m}\right) \leq\ A \leq\ \left(\frac{m\ +\alpha mn}{mn}\right)\left(\frac{mn}{2}+n\right) (1)$$

$$\frac{1}{2}+\frac{2+\alpha n}{2m}+\frac{\alpha n}{m^2}\ \leq\ A \leq\ \left(\frac{m\ +\alpha mn}{mn}\right)\left(\frac{mn}{2}+n\right)$$

Theoretical improvements (pairwise-scenario)

We present an analysis of the possible gains of this protocol in a pairwise scenario in comparison with the worst-case latency of G-Nihao $L = mn$. We are interested in the percentage of gains (G) defined as:

$$G = \left(1 - \frac{L_{THL2H_{Ext}}}{L_{Gnihao}}\right) * 100$$

$$= (1 - \frac{\frac{mn}{2}+n}{mn}) * 100$$

$$= \left(1 - \left(\frac{1}{2}+\frac{1}{m}\right)\right) * 100$$

$$= \left(\frac{1}{2}-\frac{1}{m}\right) * 100$$

As $\lim_{m\to\infty}(\frac{1}{m}) = 0$ the improvement is bounded by 50%

$$G \leq 50\%$$

The previous analysis shows how our approach takes advantage when nodes make use of low COR values as the higher the value of parameter m the closer to a 50% improvement is achieved. It is interesting to see how this improvement does not involve the duty cycle a node is working with.

3.2.4 Asymmetry

Our extended version is not different from the first one when it comes to handling asymmetry; it keeps m equal on all nodes and in this way it guarantees one-way discovery for two nodes working at different duty cycles. The worst-case scenario is also when the relative offsets are in such a way that the node running at lower duty cycle is the one that listens to the other running at a higher duty cycle. In such scenario, the latency of one-way discovery is dictated by the node with $max\ n$. However, the reply back is intended for the node with $min\ n$ achieving the two-way discovery within the next $\min(n_1, n_2)$ slots.

Hence, the worst-case latency for the asymmetric scenario is:

$$L = \ m \ * \max\left(\frac{n_1}{2}, \frac{n_2}{2}\right) + \min(n_1, n_2)$$

3.3 Impact of β on COR

THL2H Extended clearly outperforms both G-Nihao and THL2H in a pairwise scenario. However, this deliberate use of extra transmissions may be a drawback in group scenarios. In this section, we present an analysis of the impact of β over the COR usage for both THL2H Extended and THL2H. We should highlight that we are basing our analysis on the worst-case scenario in order to present an upper bound for both protocols while later on Chapter 4 we explore their actual behavior in simulations.

3.3.1 Theoretical worst-case scenario

The construction of a worst-case scenario for THL2H Extended is carried out with the adversary approach. We focus on a single period T of a node discovering (one-way) k new neighbors. The worst-case scenario would be for the next listening slot of each neighbor (waiting for a reply) to fall in different slots number of the sender's schedule. In this way, the sender has to transmit k extra beacons as none of them is able to make contact with more than one neighbor at a time. By the pigeon hole theorem, for a number of replies $k > T$ within a period T, at least two fall under the same slot. Hence, the worst-case scenario is applicable for $k \leq T$.

In the same way, we construct the worst-case for THL2H. However, we know by the pigeon hole theorem, that for $k > n$, the adversary would have to place at least two neighbors within the same row. As THL2H sends only one reply per row, covering all neighboring nodes with relative offsets within the first half of the row, the worst-case scenario is applicable only for $k \leq n$ new neighbors.

Theorem 1: Let N be a network of THL2H Extended devices with equal m and where the average number of new neighboring devices per slot is β. Then, the worst-case COR usage of THL2H Extended is:

$$\eta \leq \min(\frac{1}{m} + \frac{\beta}{2}, 1)$$

Proof: As the number of slots in a period T is nm, the average number of expected new neighbors per period T in the network N is $T\beta = nm\beta$. Moreover, for every pair of nodes, the underlying schedule guarantees the one-way discovery of at least one. Therefore, the expected number of one-way discoveries in a network is half the average neighboring devices per period (in the worst-case). Hence, only half of them need a reply, resulting in an increment of $\frac{mn\beta}{2}$ transmissions per period. Therefore, a COR equals to $\frac{n+\frac{mn\beta}{2}}{mn} = \frac{1}{m} + \frac{\beta}{2}$. Finally, as the COR usage of Talk-Listen model itself is bounded by 1 transmission per period maximum, for $\beta > T$ there is no increment.

We find this result quite interesting because our protocol has an initial COR usage of $\frac{1}{m}$, hence the formula give us the direct impact of β over the initial COR in a very clear result (Initial COR + $\frac{\beta}{2}$). Also notice how parameter n is not part of the necessary conditions, which means that for different values of n we obtain different duty cycles making this result also true for both Symmetric and Asymmetric scenarios. The latter is possible because parameter m is held constant, the only necessary condition for Asymmetry (seen in section 3.2.4) and off course, also constant in a Symmetric environment by definition.

The previous result is very useful for comparison and simulation purposes. For example, we could now calculate the theoretically maximum β the THL2H Extended can handle in order to maintain its COR with a value of no more than twice the initial one (as the THL2H does).

$$\frac{1}{m} + \frac{\beta}{2} \leq \frac{2}{m}$$
$$\beta \leq \frac{2}{m}$$

Again, we can see how this result is true for either Symmetric and Asymmetric environments. The obtained value can be interpreted as: in order to maintain an average COR under $\frac{2}{m}$, nodes can experience one new neighbor every $\frac{m}{2}$ slots on average.

Theorem 2: Let N be a network of THL2H devices with equal m and where the average number of new neighboring devices per slot is β. Then, the expected worst-case COR of THL2H is:

$$\eta \leq \frac{1}{m} + \min(\frac{\beta}{2}, \frac{1}{m})$$

Proof: We follow an analog reasoning of that in the proof of Theorem 1. The number of slots in a period T is nm. The average number of expected new neighbors per period T in the network N is $T\beta = nm\beta$ where half of them make a one-way contact without the need of replies. The total number of transmissions per period is $n + \frac{mn\beta}{2}$. Therefore, a COR equals to $\frac{n+\frac{mn\beta}{2}}{mn} = \frac{1}{m} + \frac{\beta}{2}$. However, as seen in Section 3.3.1, the worst-case scenario can only occur for a maximum number of $2n$ new neighbors in a period T, which makes $\beta \leq \frac{2n}{nm} = \frac{2}{m}$. Then, for $\beta = \frac{2}{m}$, the COR usage would be $\frac{2n}{mn} = \frac{2}{m}$, a value equals to the maximum COR usage of the protocol. Hence, for $\beta \geq \frac{2}{m}$ the COR usage does not increase beyond $\frac{2}{m}$.

3.4 Transmission Overhead

We call transmission overhead to any extra amount of information put on beacons as it increases the time the channel is occupied when transmitting. The first point we analyze is the necessary information for nodes to be able to reply to then see how much overhead it may cause.

3.4.1 Necessary information

This problem can be solved for both symmetric and asymmetric scenarios with ease. When nodes are running in the symmetric scenario (i.e. equal duty cycles) all schedules are equals. Therefore, nodes add their current internal state when sending their transmissions and the receiver can replicate the schedule within its own time representation. Furthermore, when working in asymmetry, the schedules may be different as the period T may vary from node to node. Hence, the current state by itself is not enough. Nevertheless, as parameter m remains constant for all nodes no matter the duty cycle, a node only needs the transmitter's n (parameter) in order to recreate its schedule.

Most group based protocols relay on heavy transmission overhead. Sending more information increases the size of the message resulting in longer listening slot's size and higher number of collisions. In fact, that is one of the weak points of group-based protocols in comparison with pairwise protocols. For example, let be GP a group-based protocol that improves the worst case latency of PP, a pairwise protocol, by 10%. As the worst case latency is measured in slots without taking into account the size of them, running PP with slot sizes of 10ms give us the same worst case latency (in time) than running GP with slots sizes of 11ms. In addition, if GP's transmissions carry to much information (neighbor's ids, schedules, etc.), the time needed to listen to one transmission increases. This, put together with a common scenario where there may be a need for listening to more than one transmission within the same slot, is the reason for setting a higher slot's size when running GP. Hence, it is clear that the smaller the transmission overhead, the wider the group-based protocol use cases. Only in real case scenarios where hardware constrains stop pairwise protocols to set slot sizes as small as they could be, this is not a concern. Even so, the expected number of collisions is always higher.

The transmission overhead of the proposed protocol is smaller than that of current group-based protocols for the listen-listen model. For example, in Acc [16], nodes add its complete neighbors table to transmissions. A better approach is taken in [19] where only a cache of them is held.

THL2H does not uses unknown neighbor's information as it gets it directly from the neighbor it is going to reply to. Hence, every transmission carries a particular node´s information. As explained in Section 3.1.3**Error! Reference source not found.** nodes only add an extra of two elements to their transmissions: internal state and duty cycle. It should be mentioned, that in a real case scenario, this information can be represented with only a few bits depending on the possible duty cycles and period length.

Chapter 4

Evaluation and Discussion

In this chapter, the performance of both versions is evaluated through simulation studies. We are mainly interested in the cumulative distribution function (CDF) of discovery latency, duty cycle, and COR usage. In order to avoid tying the results to particular hardware implementation platforms, the number of slots is used as the metric of the latency. Section 4.2 presents some comments on how the simulation was carried out. Section 4.2.1. shows the results for the symmetric scenario and Section 4.3 for the asymmetric scenario. The protocol chosen to compare with was G-Nihao as is the only slotted protocol that takes into account COR usage.

4.1 Simulation scenarios and platform

We divide our analysis into two different scenarios:

- The classic one vs one (pairwise) where a pair of nodes try to make contact with each other as fast as possible. As usual, nodes are put within range from the start of the simulation and without any previous information about each other. Off course, there is also no other neighboring node.
- The group scenario, where a number of nodes are randomly inserted in a constraint area and let discover each other for a given amount of time. In this more complex scenario, nodes may enter the area at any time after the start of the simulation, as well as leave it.

In order to control simulations, we implemented our own platform. With it, we have control over nodes specifications such as duty cycle, protocol, initial COR, etc. We also control the parameters for the simulation environment such as number of nodes, communication range, speed, etc. Table 1 shows the parameter used.

Table 1 Simulation parameters

Number of Nodes	**Deployment Area Size (m)**	**Communication Range (m)**	**Speed (m/s)**	**Time (seg.)**	**Slot size (ms)**	**α / Slot size ratio**	**Duty cycle**	**COR (η)**
50 - 100	100 x 100	20 - 100	1	40	10	0.05	5%	1/20

4.2 Symmetric Discovery

In this section, we evaluate the performance of THL2H and THL2H extended and compare it with that of G-Nihao for both pairwise and group simulations.

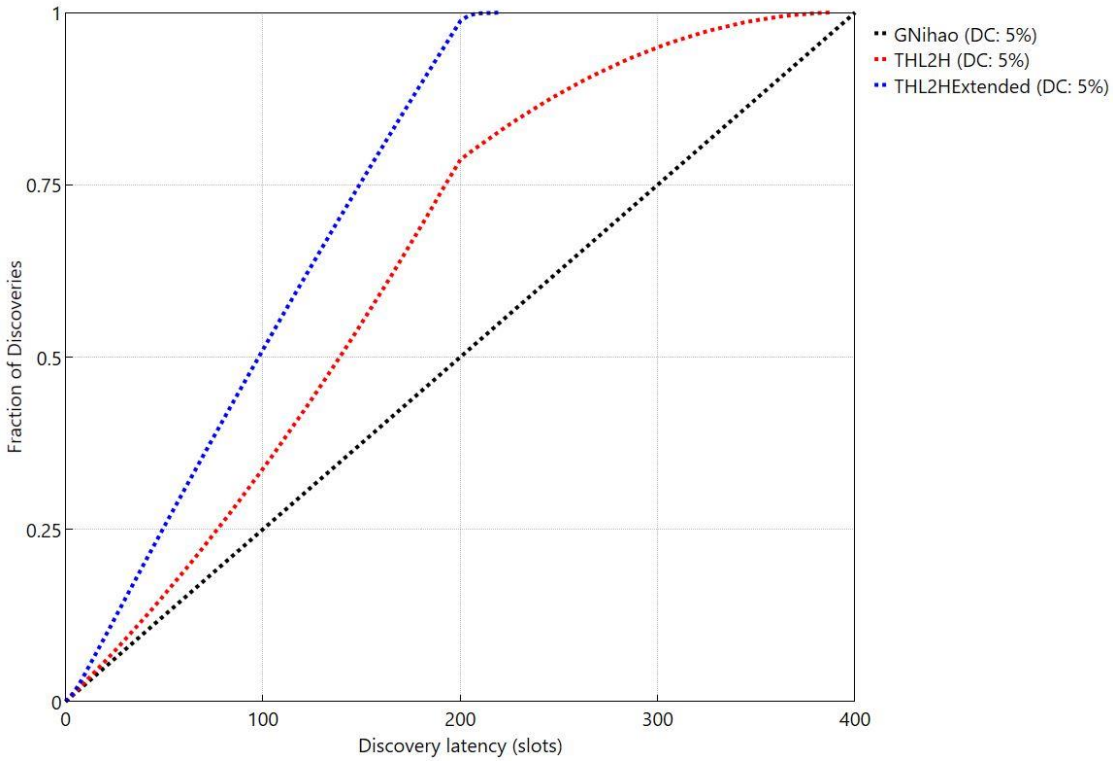

Figure 4.1 Fraction of Discoveries by latency

Pairwise

In this scenario, both versions presented only send one extra transmission. Hence, we consider the COR usage equal (ignoring the round error) to that of G-Nihao. The simulations were done on a pair of nodes running at 5% DC and COR of $\eta = 1/20$.

Figure 4.1 shows the CDF of discovery and Figure 4.2 the avg. discovery latency. We can see how both protocols presented are significantly faster than G-Nihao overall. The worst-case latencies of THL2H and THL2H Extended are 390 and 220 slots, respectively. These values are 2.5% and 45% faster than the 400 slots of G-Nihao.

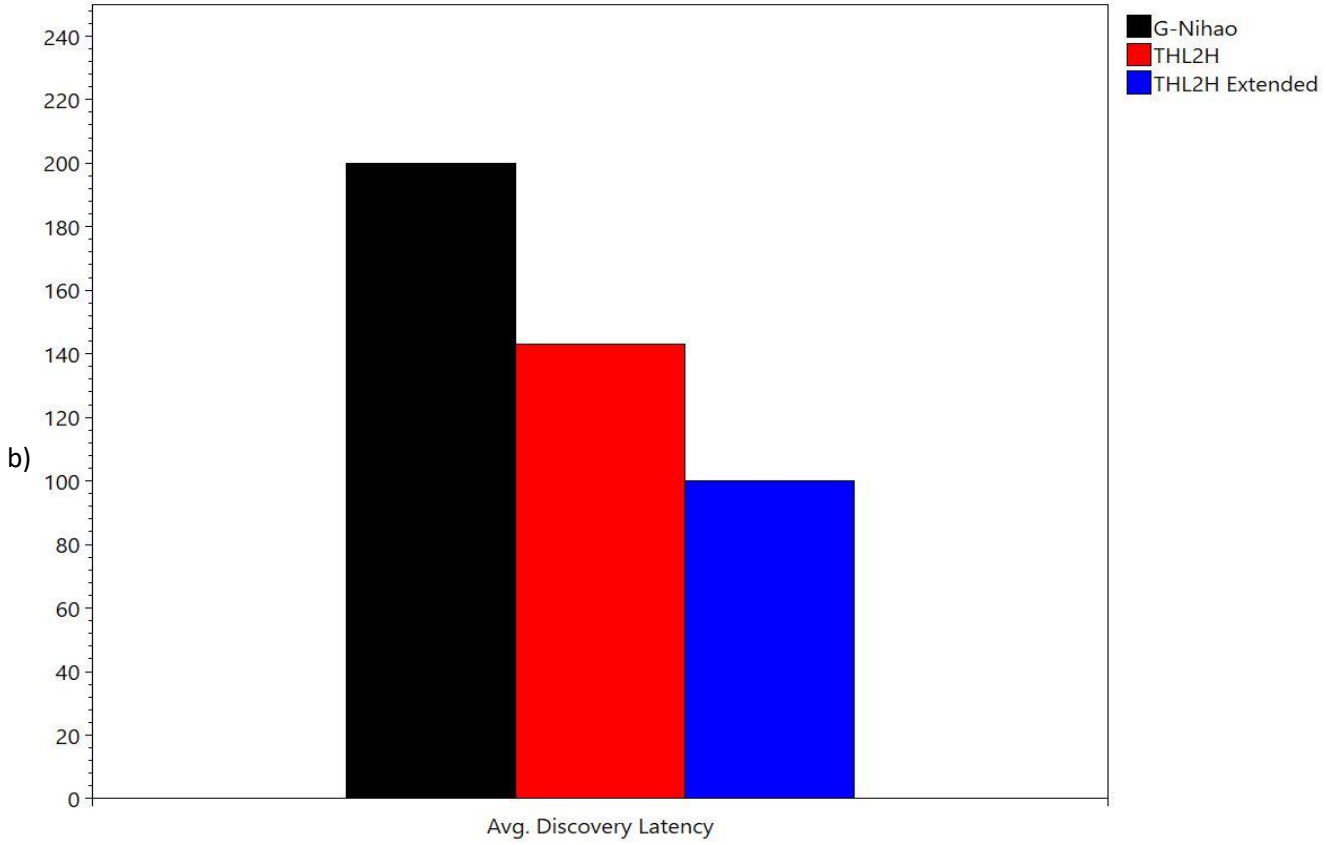

Figure 4.2 Average Discovery Latency

The result registered for the avg. case latency is considerably better. THL2H presents 143 slots and THL2H Extended 100 slots achieving a 28.5% and 50% improvement over G-Nihao's 200 slots, respectively. As expected, the extended version is the faster of them all.

We find an interesting comparison when gradually increasing G-Nihao's COR up to where it matches the performance of the presented protocols. Figure 4.3 shows the series of G-Niaho plotted, with COR usages from 1/10 up to 1/20. We can see how THL2H Extended discovers more than 80% of the nodes at a rate matching that of G-Nihao with a COR of $\eta = 1/10$ and its worst-case latency falls right beside that of G-Nihao with a COR of $\eta = 1/11$. In the case of THL2H, it presents a rather complex chart where the 80% of the discoveries are done before 210 slots or less, matching G-Nihao with a COR of $\eta = 1/14$ and only 20% short in discoveries when compared with THL2H Extended. It is actually this 20% of discoveries left the ones accounting for its long tail that ends above G-Nihao with a COR of $\eta = 1/19$. Luckily, the first 80% is enough to place the average case of 143 slots below G-Nihao with a COR of $\eta = 1/15$.

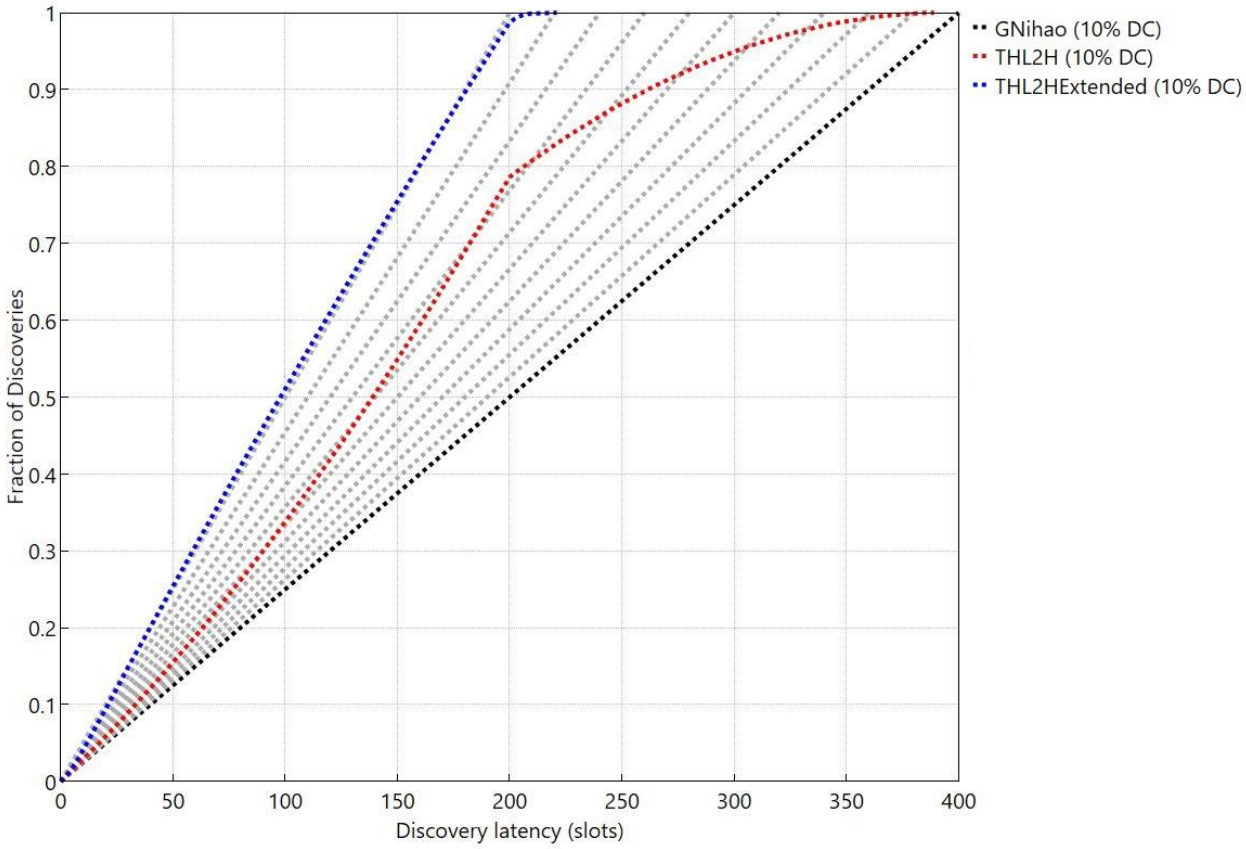

Figure 4.3 THL2H and THL2H Ext. vs GNihao with different COR usages

4.2.1 Group simulations

Moving on to a scenario with a group of nodes, we keep the nodes at 5% DC and initial COR usage of $\eta = 1/20$.

We set a maximum β (seen in Section 3.3) in order to keep the COR usage below twice the initial one and we obtained $\beta = 0.1$. Then, we ran simulations on several networks starting at $\beta = 0.01$ and increasing it with a step of 0.01 up to $\beta = 0.1$. This gave us networks where nodes faced an average of one new neighbor every 100 slots up to a maximum of one every 10 slots.

For every network, we recorded the actual COR usage and compare it with that calculated as the worst-case. Figure 4.4 contains the results for THL2H. The protocol's COR usage was up to a 33% smaller than that of the worst case scenario but most importantly, the simulations clearly show how this improvement gradually increases/decreases as β increases/decreases. A result that was expected as per our analysis (in Section 3.1.3) about how unlikely the worst case COR usage becomes the more extra transmissions are sent.

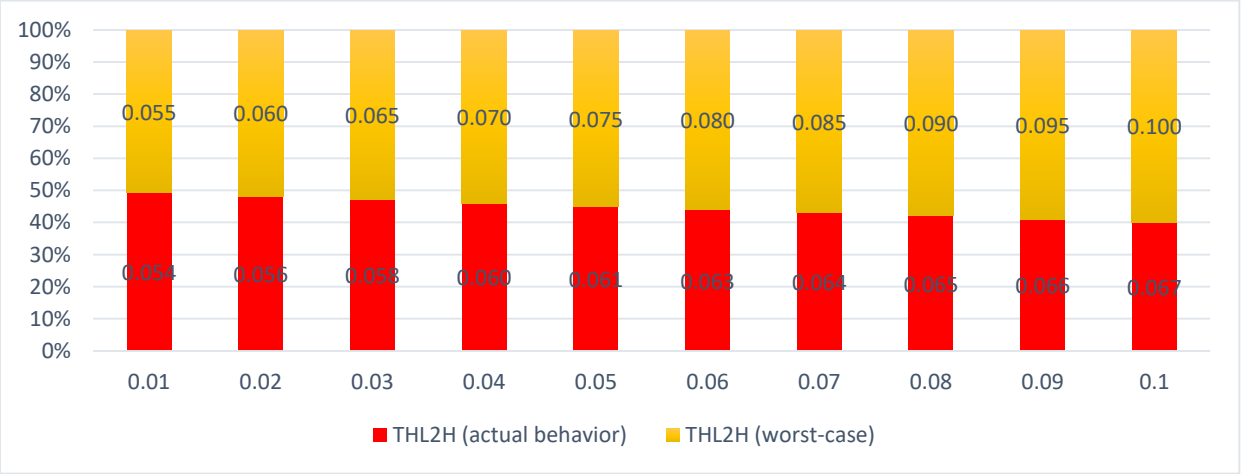

Figure 4.4 THL2H COR usage worst-case vs simulation

The results for THL2H Extended are shown in Figure 4.5. Notice how the average COR usage is clearly higher than that of THL2H but still smaller than the worst-case COR usage by a 13%.

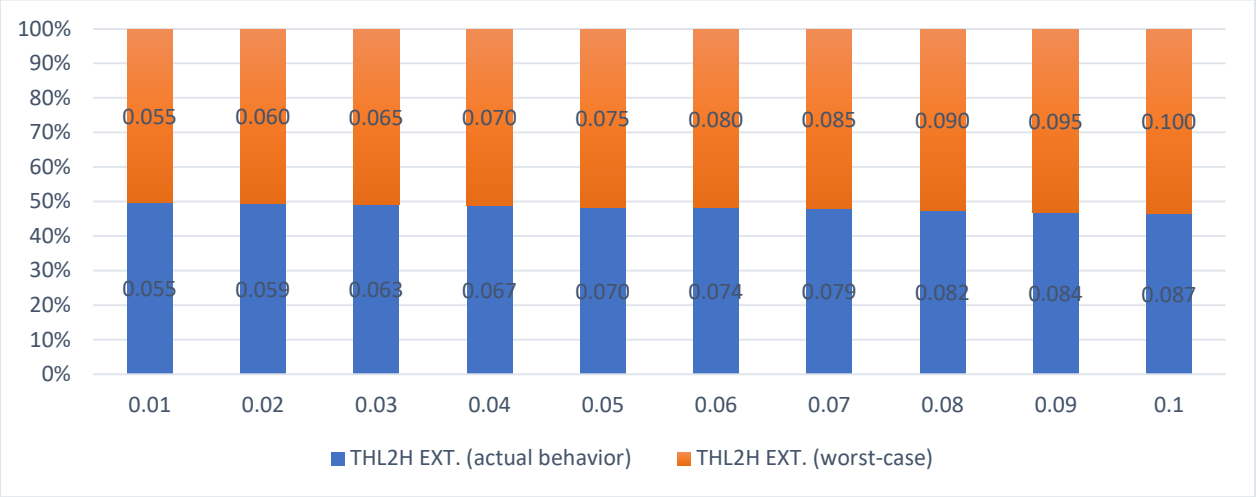

Figure 4.5 THL2H Extended COR usage by β

When running simulations on networks with $\beta = 1.5$ (a 50% higher than the previous maximum). The average COR usages obtained were $\eta = \frac{1}{14.14} = 0.070$ for THL2H and $\eta = \frac{1}{9.86} = 0.101$ for THL2H Extended. Result that clearly shows how THL2H performs much better (in terms of COR usage) and also highlights the unbounded impact of β on the extended version. Notice how the COR usage of the extended version surpassed the theoretical limit of THL2H of $\eta = 0.1$.

4.3 Asymmetric Discovery

The performance on the asymmetric scenario is again compared with that of G-Nihao. The results are shown in Figure 4.6(a) for the CDF of discovery and Figure 4.6(b) for the avg. case latency. Simulations were carried out on a pair of nodes running at duty cycles of 1% and 5% with a COR of $\eta = 1/20$.

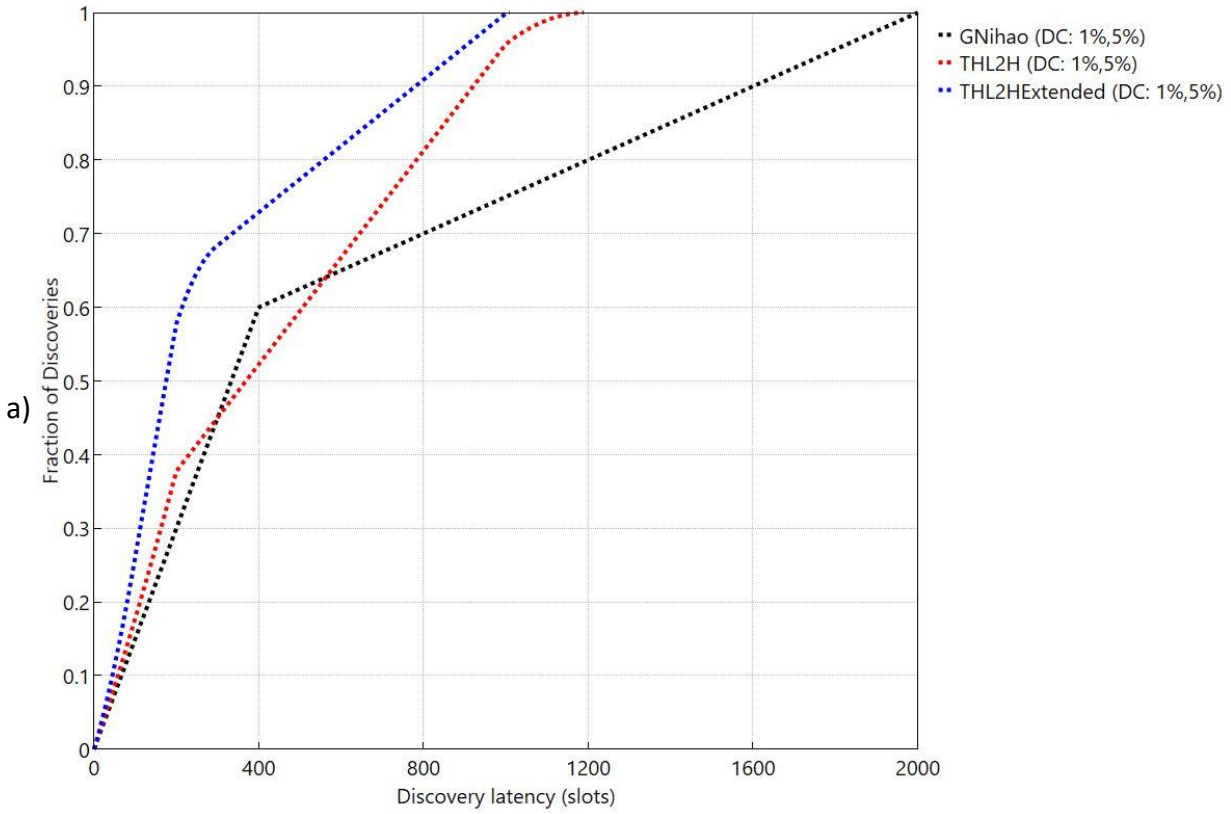

Figure 4.6 Fraction of Discoveries by latency (a)

We can see that our protocols are significantly faster than G-Nihao for this scenario as well. The worst-case latencies of THL2H and its extended version are 1190 and 1010 slots respectively. These values are a 40% and 50% improvement over the 2000 slots of G-Nihao, respectively. The average case latencies recorded were 433 and 293, around a 28% and 50% improvement over that of G-Nihao with 600.

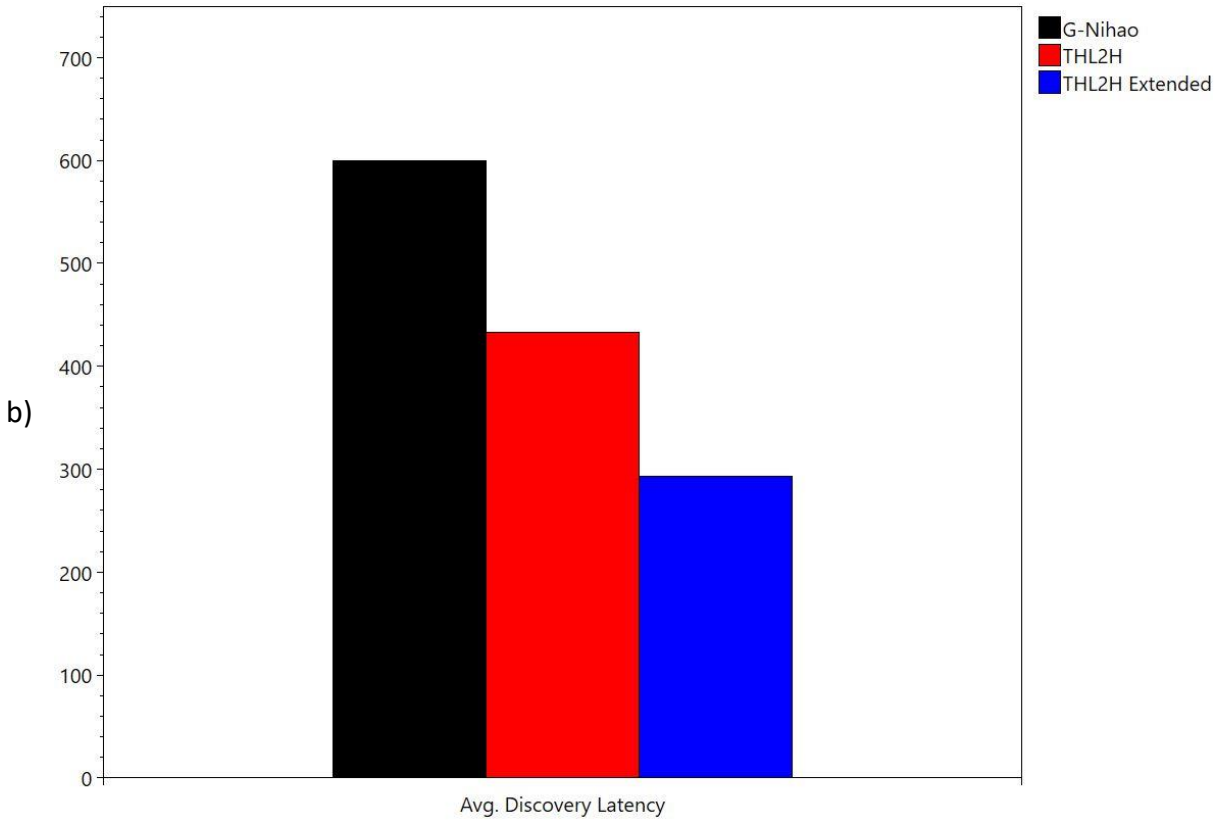

Figure 4.7 Avg. Discovery Latency (b)

Finally, we are interested in the COR usage increment needed by G-Nihao in order to match a similar performance of both our protocols. When gradually increasing G-Nihao's COR we found that THL2H worst-case latency of 1190 slots is equivalent to that of G-Niaho with a COR of $\eta = 1/12$, a 40% increment of the COR usage. The average case latency is closer to that of G-Nihao with a COR of $\eta = 1/14$, a 30% increment of the COR usage. Overall, we can say THL2H performed better in this scenario (asymmetric). THL2H Extended is again the best of them all, with a worst-case and average case latency comparable with G-Nihao with COR of $\eta = 1/10$. Result that shows a 50% increment of G-Nihao's COR usage to be able to match its performance.

4.3.1 Issues of asymmetry on group simulations

When simulating a real case scenario with a group of nodes running at any desired DC many asymmetric scenarios arises. In the same way, many symmetric scenarios may arise as well. This happens naturally as not all neighboring nodes may be running at different DC. Because of this, any environment permitting asymmetry tends to be dictated by a worst-case scenario coming from the symmetric situation of a neighboring couple running at the lowest possible DC. This is why we do not compare the results of a pairwise scenario with nodes running at 1% DC and 5% DC with that of a group of nodes randomly running at either 1% or 5%. We considered the possible solution of manually ignoring all symmetric cases that may arise, however, this *pure* asymmetric simulation may be rather synthetic and not practical. Hence, we decided not to analyze it.

Chapter 5

Conclusions and Future Work

This chapter summarizes the thesis, discusses its main findings and contributions, and outlines areas for future research and development. The ideas prevalent in our work on neighbor discovery protocols for Ad-Hoc Wireless Networks and Wireless Sensor Networks were proved useful. However, there still are clear drawbacks that may be taken into account for future works. The chapter is divided into two main sections. Section 5.1 gives a summary of the thesis, and Section 5.2 discusses the future work.

5.1 Summary of contributions

In this work, we have introduced two new protocols for the Talk-Listen model: THL2H and THL2H Extended. From a pairwise point of view, the proposed protocols use the mutual assistance approach which basically focuses on achieving one-way discovery to then complete the 2-way discovery by sending one extra transmission. We proved the THL2H Extended approach could achieve a theoretical maximum improvement of up to 50% in a symmetric environment over the worst-case discovery latency of the G-Nihao protocol, for sufficiently large m.

We introduced the concept of average new neighbors per slot (β) for group environments. This metric serves as a practical measurement for simulation scenarios focused on slotted models such as the Listen-Listen and Talk-Listen. Finally, we presented two theorems (3.3) regarding the β impact on our protocols COR usage.

5.1.1 General Results

Our protocols achieve the one-way discovery up to 50% faster than the worst-case scenario of G-Nihao for the symmetric scenario. However, in order to complete the two-way discovery, extra transmissions are sent on-demand. With our first version, we achieved the same worst-case discovery latency than the G-Nihao protocol but with lower average case latency. Then, our second version improves both the worst-case and the average-case latency of G-Nihao up to a 50%. The difference in performance we obtained were given by the way the extra transmissions are managed. The first protocol was designed to maintain a COR usage below twice as much the initial one while the second version may increase it deliberately. Group simulations under networks with $\beta < \frac{1}{m}$ proved THL2H Extended to perform better than THL2H and G-Nihao for the performance metric A and also average-case latency. In the case of the performance of the asymmetric scenario, the results were a significant improvement over that of G-Nihao as both protocols performed better in terms of worst-case and average-case latency. Moreover, we also found that the smaller the ratio of asymmetry the more improvements over G-Nihao's worst case latency.

Results on pairwise simulations

On simulations with symmetric nodes at 5% duty cycle and COR $\eta = 1/20$, THL2H Extended proved to have a significantly faster worst-case bound than our first version and G-Niaho by around a 45%. However, is the avg. discovery latency the major improvement achieved over G-Nihao, reaching up to 28% for THL2H and 50% for the extended version. As for the asymmetric case, the results showed very strong improvements overall. THL2H achieved a 40% and a 28% improvement over G-Nihao's worst-case and average-case latency, respectively. In the case of THL2H Extended, we achieved a 50% improvement on both worst-case and average-case latency.

Results on group simulations

Based on our metric β and simulations on groups of nodes in a symmetric scenario, we found improvements over the G-Nihao protocol on networks up to $\beta < \frac{1}{m}$. We also found the impact

of β on the COR usage to be lower than the theorethical expected and again, it was THL2H Extended the best of them all.

Below, we present two tables that summarize our main findings:

Table 2 Worst-case latency for symmetry and asymmetry.

Protocol Name	**Protocol Parameter**	**L (symmetry worst-case latency)**	**L (asymmetry worst-case latency)**
G-Nihao	m, n	mn	$m * \max(n_1, n_2)$
THL2H	m, n	mn	$m * (max\left\{\frac{n_1}{2}, \frac{n_2}{2}\right\} + min\left\{\frac{n_1}{2}, \frac{n_2}{2}\right\})$
THL2H Ext.	m, n	$\frac{mn}{2} + n$	$m * \max\left(\frac{n_1}{2}, \frac{n_2}{2}\right) + \min(n_1, n_2)$

Table 3 COR usage for pairwise and group scenario

Protocol Name	**Protocol Parameter**	**Network Parameter**	**COR usage (pairwise)**	**COR usage (group)**
G-Nihao	m, n $m \geq 2, n \geq 1$	β	$\frac{1}{m}$	$\frac{1}{m}$
THL2H	m, n $m \geq 2, n \geq 1$	β	$\frac{1}{m}$	$min(\frac{1}{m} + \frac{\beta}{2}, \frac{2}{m})$
THL2H Ext.	m, n $m \geq 2, n \geq 1$	β	$\frac{1}{m}$	$min(\frac{1}{m} + \frac{\beta}{2}, 1)$

5.2 Future work

While this thesis has demonstrated the potential of improvements for neighbor discovery on the Talk-Listen model, many opportunities for extending the scope of this research remain. Below, we present those we think are the most important ones:

- Group-based protocols are yet to be explored on this model but we believe that most of the existing ideas based on the Listen-Listen model can be re-think or modified to fit this newer model.
- Group-based protocols that also take into account the addition of listening slots as part of the dynamic 'moves' that can be done.
- Neighbor discovery based on multi-hop neighbors. Such neighbors have the potential to benefit a number of network services, such as routing [29], connectivity [30] and localization [31][32], at present, no work has focused on multi-hop neighbor discovery.

References

[1] S. Basagni, M. Conti, S. Giordano and I. Stojmenovic, Mobile Ad Hoc Networking, John Wiley & Sons, pp. 85-98, 2004.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A Survey On Sensor Networks," *IEEE Commun. Mag.,* vol. 40(8), pp. 102–114, 2002.

[3] W. Sun, Z. Yang, X. Zhang and Y. Liu, "Energy-Efficient Neighbor Discovery in Mobile Ad Hoc and Wireless Sensor Networks: A Survey," *IEEE Communications Surveys & Tutorials,* vol. 16(3), pp. 1448-1459, 2014.

[4] A. Kandhalu, K. Lakshmanan and R. Rajkumar, "U-Connect: A Low-Latency Energy-Efficient Asynchronous Neighbor Discovery Protocol," in 9th ACM/IEEE International Conference on Information Processing in Sensor Networks, pp. 350-361, 2010.

[5] B. Sundararaman, U. Buy and A. D. Kshemkalyani, "Clock synchronization for wireless sensor networks: a survey," *Ad Hoc Networks,* vol. 3, pp. 281-323, 2005.

[6] P. Dutta and D. Culler, "Practical Asynchronous Neighbor Discovery and Rendezvous for Mobile Sensing Applications," in 6th ACM Conference on Embedded Network Sensor Systems, pp. 71-84, 2008.

[7] K. Wang, X. Mao and Y. Liu, "Blind Date: A Neighbor Discovery Protocol," in 42nd Int. Conference on Parallel Processing, pp. 120-129, 2013.

[8] Q. Ying, L. Shining, X. Xiangsen and L. Zhigang, "Talk More Listen Less: Energy-Efficient Neighbor Discovery in Wireless Sensor Networks," in 35th Annual IEEE Int. Conference on Computer Communications (INFOCOM), pp. 1-9, 2016.

[9] S. Lai, B. Ravindran and H. Cho, "Heterogenous quorum-based wakeup scheduling in wireless sensor networks," *IEEE Trans. Comput.,* vol. 59, pp. 1562-1575, 2010.

[10] M. J. McGlynn and S. A. Borbash, "Birthday Protocols for Low Energy Deployment and Flexible Neighbor Discovery in Ad Hoc Wireless Networks," in 2nd ACM Int. Symposium on Mobile Ad Hoc Networking & Computing, pp. 137-145, 2001.

[11] R. Zheng, J. Hou and L. C. Sha, "Asynchronous Wakeup for Ad Hoc Networks," in 4th ACM Int. Symposium on Mobile Ad Hoc Networking & Computing, pp. 35-45, 2003.

[12] I. Anderson, Combinatorial designs and tournaments, Oxford University Press, pp. 135-142, 1998.

[13] M. Bakht, M. Trower and R. Kravets, "Searchlight: Helping Mobile Devices find their Neighbors," in 3rd ACM SOSP Workshop on Networking, Systems, and Applications on Mobile Handhelds, pp. 132-144, 2011.

[14] "Wikipedia, “Birthday paradox”," [Online]. Available: http://en.wikipedia.org/wiki/Birthday problem.

[15] T. Yu-Chee, H. Chih-Shun and H. Ten-Yueng, "Power-Saving Protocols for IEEE 802.11-Based Multi-Hop Ad Hoc Networks," in 21st Annual Joint Conference of the IEEE Computer and Communications Societies , pp. 277-289, 2002.

[16] D. Zhang, T. He, Y. Liu, Y. Gu, F. Ye, R. K. Ganti and H. Lei, "Acc: Generic On-Demand Accelerations for Neighbor Discovery in Mobile Applications," in 10th ACM Conference on Embedded Network Sensor Systems, pp. 1345-1357, 2012.

[17] L. Chen, Y. Shu, Y. Gu, S. Guo, T. He, F. Zhang and J. Chen, "Group-based Neighbor Discovery in Low-duty-cycle Mobile Sensor Networks," *IEEE Trans. on Mobile Computing,* vol. 15(8), pp. 1996-2009, 2014.

[18] S. B. Manir, "Collective Neighbour Discovery in Wireless Sensor Network," *International Journal of Computer Applications,* vol. 131(11), pp. 378-390, 2015.

[19] Q. Niu, W. Bao and S. Xia, "An Improved Group-Based Neighbor Discovery Algorithm for Mobile Sensor Networks," *International Journal of Distributed Sensor Networks,* pp. 514-524, 2014.

[20] H. Wang, J. Ma, Y. Liu, W. Liu and L. Wang, "Bi-directional Probing for Neighbor Discovery," in IEEE 17th Int. Conference on Computational Science and Engineering, pp. 223-231, 2014.

[21] L. Chen, R. Fan, K. Bian, M. Gerla, T. Wang, and X. Li, "On heterogeneous neighbor discovery in wireless sensor networks," in Proc. of the 2015 IEEE International Conference on Computer Communications, pp. 693–701, 2015.

[22] L. M. Feeney and M. Nilsson, "Investigating the energy consumption of a wireless network interface in an ad hoc networking environment," in IEEE Conference on Computer Communications (INFOCOM), pp. 67-79, 2001.

[23] W. Sun, Z. Yang, K. Wang and Y. Liu, "Hello: A Generic Flexible Protocol for Neighbor Discovery," in IEEE Conference on Computer Communications (INFOCOM), pp. 234-246, 2014.

[24] I. Niven and H. S. Zuckerman, An Introduction to the Theory of Numbers, John Wiley and Sons (WIE), pp. 150-152, 1991.

[25] "Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Twin_prime.

[26] P. Chen, Y. Chen, S. Gao, N. Qiang and J. Gu, "Efficient group-based discovery for wireless sensor networks," *International Journal of Distributed Sensor Networks,* vol. 13, pp. 107-115, 2017.

[27] A. Purohit, N. Priyantha, and J. Liu. "Wiflock: Collaborative group discovery and maintenance in mobile sensor networks", In 10th ACM/IEEE Int Conf. on Information Processing in Sensor Networks, pp. 45-56, 2011.

[28] L.Wei, B.Zhou, X.Ma, D.Chen, J.Zhang, J.Peng, Q.Luo, L.Sun, D.Li, and L. Chen. “Lightning: A High-efficient Neighbor Discovery Protocol for Low Duty Cycle WSNs”. IEEE Communications Letters, vol. 20, no. 5, pp. 966-969, 2016.

[29] J. N. Al-Karaki and A. E.Kamal, “Routing techniques in wireless sensor networks: a survey,” Wireless Commun., vol. 11, no. 6, pp. 6–28, 2004.

[30] A. Ghosh and S. K. Das, “Coverage and connectivity issues in wireless sensor networks: a survey,” Pervasive and Mobile Computing, vol. 4, no. 3, pp. 303 – 334, 2008.

[31] Z. Yang and Y. Liu, “Quality of trilateration: confidence-based iterative localization,” IEEE Trans. Parallel Distrib. Syst., vol. 21, no. 5, pp. 631–640, 2010.

[32] Y. Liu, Z. Yang, X. Wang, and L. Jian, “Location, localization, and localizability,” J. of Computer Science and Technol., vol. 25, no. 2, pp. 274–297, 2010.