TECHNISCHE UNIVERSITÄT MÜNCHEN

Lehrstuhl für Realzeit-Computersysteme

Optimal Neighbor Discovery Protocols

Philipp H. Kindt

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Die Dissertation wurde am 17.10.2018 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 10.9.2019 angenommen.

Acknowledgements

During this thesis, I received generous support from multiple sides, which I would briefly like to acknowledge below.

First of all, I would like to thank Prof. Samarjit Chakraborty for the really perfect supervision of this thesis and my research. Working with him has helped me sharpening my thinking, and it has truly been a very good experience for me to interact and discuss with and learn from him. Especially, given that his schedule has often been extremely busy, I really appreciate the great effort he invested to support my work.

Further, I would like to thank my parents Christine and Dr. Heinz Kindt for always having supported my studies and my PhD thesis in every possible way. Without them, becoming an electrical engineer would probably not have been possible. Thanks also for not running out of patience when i electrified my sister's doorhandle or built an electrical ropeway to my neighbor when i was a child! It is worth mentioning that my family has grown during the time I had been working on this thesis: At the beginning, I met the most wonderful person in my life, who I could fortunately convince to marry me only a moment after. My wife Carolin has always been on my side and supported me affectionately throughout the ups and downs of this thesis. Further, she has often covered my back whenever I could not sufficiently care about other matters due to high workloads. Thank you, Carolin, for being the foundation of everything in my life!

I would also like to thank Prof. Graziano Pravadelli for supporting me in various ways throughout my academic career at TU Munich. Further, I had the opportunity to collaborate with multiple partners from industry, which had always supported me and created a convenient working atmosphere. I very much enjoyed working with Bruno Ristok (C&S GmbH), Thomas Huber (Fertl EDV Systeme GmbH), Thomas Herbig (Reliatech GmbH) and Andreas Tobola (Fraunhofer IIS), whom I would like to thank for their cooperation and support.

Prof. Prabal Dutta, whose work on neighbor discovery has been an inspiration to my research, has accepted to examine this PhD thesis despite the effort and the travel to Europe this incurs. Therefore, i would like to thank him for joining my PhD committee, as well as Prof. Wolfgang Keller for serving as the committee chair. In addition, I would like to thank all my coauthors and all students who did their Bachelor's and Master's theses with me. Further, since this thesis is based on multiple IEEE and ACM publications, I would like to express my appreciations to the reviewers, editors and conference organizers that made these publications possible. Finally, I would like to thank the general public for providing the necessary funding for this research.

Munich, June 2018 / August 2019

Abstract

Within the last decades, wireless networks that do not require any fixed infrastructure (mobile ad-hoc networks, MANETs) have become widespread. Before two devices in a MANET can start communicating, they have to discover their mutual existence using a procedure called *neighbor discovery.* It is used e.g., for establishing a later connection of wireless gadgets to a smartphone, or for continuously tracking with whom a subject had been in contact with throughout the day.

Since all participants of a MANET are powered by batteries or energy harvesting, power-efficient communication is a crucial requirement. Low power protocols apply a common wakeup schedule among multiple network participants. In particular, two devices communicating with each other wake up at the same point in time, exchange data packets and then go into a sleep mode until the next scheduled wakeup occurs. With such a scheme, the devices can remain in a low-power mode during most of the time. However, such a wakeup-schedule has to be negotiated between the devices before becoming operative. In addition, the clocks of both devices need to be synchronized before such a synchronous operation can take place. This is realized using neighbor discovery protocols, which work as follows. A device repeatedly broadcasts packets on a certain set of channels, whereas other devices repeatedly switch on their receivers for short amounts of time. To save energy, the devices sleep in the meantime. Two devices have discovered each other successfully once both have received a first packet from their opposite. In general, longer and more frequent listening periods, as well as more frequent transmissions, result in higher duty-cycles and hence energy consumptions, but achieve shorter discovery latencies.

In the last two decades, a vast number of neighbor discovery protocols have been proposed, which attempt to guarantee an upper latency bound that is as short as possible, while at the same time minimizing the energy consumption of the radios. Whereas the proposed solutions became increasingly more efficient, the fundamental performance limits of such deterministic neighbor discovery protocols have not been clear prior to this work.

In this thesis, we present a theoretic analysis on the fundamental limits of neighbor discovery. Building upon this theory, we for the first time present safe and tight, duty-cycle-dependent bounds on the worst-case discovery latencies, which no neighbor discovery protocol can come below. In particular, we present bounds in two different metrics, viz., worst-case latencies vs. duty-cycles and worst-case-latencies vs. duty-cycles vs. maximum channel utilizations. Further, we present bounds for multiple special cases, e.g., for situations in which each device is obliged to operate on the same duty-cycle (viz.,

symmetric discovery), as well as for scenarios with asymmetric duty-cycles.

In the unidirectional case, in which one receiver discovers one sender, the latencies predicted by these bounds can be guaranteed for 100 % of all discovery attempts. For a pair of devices that both transmit and receive, as well as in networks with only few devices carrying out the discovery procedure simultaneously, the predicted latencies can be realized with a very high probability. For a small fraction of discovery attempts however, beacons from multiple devices collide, or packets are lost because the radio is unable to receive during the transition phases between reception and transmission. For increasing numbers of devices discovering each other simultaneously, this failure rate grows and deterministic discovery, in which practically all discovery attempts are successful within limited time, is no longer possible. Predicting the lowest possible latency that can be realized for a certain energy budget with a certain failure rate among a certain number of devices remains an open problem for future research.

We further analyze the performance of multiple popular protocols and compare them to the corresponding theoretical bounds. Most existing neighbor discovery protocols are referred to as *slotted protocols*. They subdivide time into multiple, equal-length intervals, called slots. In some slots, the device remains asleep, whereas other ones are active slots used for communication. At the beginning and/or the end of each active slot, a device sends a beacon. For the rest of the slot length, the device listens to the channel. The discovery procedure is complete once two active slots overlap in time. We conclude that such slotted protocols can actually reach the optimal performance only when jointly considering duty-cycles, channel utilization and worst-case latencies. However, whenever the channel utilization is unconstrained, which is typically the case in small networks with few numbers of devices being in discovery mode simultaneously, they cannot perform optimally.

To reach optimality also in this important metric, slotless discovery paradigms needs to be studied. We therefore consider slotless, periodic interval-based protocols, in which one device periodically broadcasts packets, whereas the other device periodically listens to the channel. Both periods are independent of each other and drawn over continuous time. Such protocols provide three degrees of freedom (viz., the intervals for broadcasting and listening and the duration of each listening phase). Though billions of existing Bluetooth Low Energy devices rely on this paradigm, neither their expected latencies nor beneficial configurations with good latency/duty-cycle relations were known prior to this work.

In this thesis, we present a mathematical theory to compute the neighbor discovery latencies of such periodic interval-based protocols for all possible parametrizations. Further, our theory shows that upper bounds on the latency can be guaranteed for all parametrizations, except for a finite number of singularities. Therefore, slotless, periodic interval-based protocols can be used in applications with deterministic latency demands, which have been reserved for slotted protocols until now.

Building on this, we propose a computationally light-weight optimization framework that translates any desired duty-cycle into a set of optimized parameters for periodic interval-based protocols that achieve low latencies and energy consumptions. We show that protocols parametrized using this framework actually achieve optimal latencies for every given duty-cycle. Further, we present a real-world implementation of a

slotless protocol parameterized using our proposed framework and also introduce several techniques to counteract the non-idealities exhibited by the wireless hardware. In addition, we provide a methodology for comparing the performances of slotless and slotted protocols, which has not been sufficiently dealt in the literature. Experiments performed show that the real-world implementation of a slotless protocol parameterized using our proposed optimization framework significantly outperforms the latency bounds of all known protocols, providing up to e.g. $4\times$ lower worst-case latencies compared to the popular U-Connect protocol.

Further, we propose a technique that additionally reduces the mean latencies of such optimal periodic interval-based protocols, without significantly affecting their worst-case latencies. In periodic interval-based protocols, the discovery latency of one device E discovering its opposite device F is independent from that of device F discovering device E. This leads to longer mean latencies than necessary for two-way discovery, for which both partial discovery procedures need to terminate successfully. In this thesis, we propose a cooperative approach to reduce this two-way discovery latency. In particular, each side broadcasts information on the time-period until its next reception phase takes place. The remote device adjusts its beacon schedule accordingly once a first packet is received. We call this technique *mutual assistance*. Compared to non-cooperative slotless protocols, this technique can reduce the two-way discovery latency by up to 43 %.

Kurzfassung in deutscher Sprache

In den letzten Dekaden haben sich drahtlose Netzwerke weit verbreitet, die ohne jegliche fest installierte Infrastruktur arbeiten. Bevor zwei Geräte in solchen mobilen ad-hoc Netzwerken (MANETs) miteinander kommunizieren können, müssen sie gegenseitig die Anwesenheit ihres Gegenübers feststellen. Dies geschieht mit einer Prozedur namens *Neighbor Discovery*. Sie wird beispielsweise dazu verwendet, um eine Verbindung von drahtlosen Geräten mit einem Smartphone aufzubauen, oder um zu tracken, mit wem eine Person über den Tag hinweg Kontakt hatte.

Da alle Teilnehmer in einem MANET durch Batterien oder Energy Harvesting betrieben werden, ist eine stromsparende Kommunikation essenziell. In stromsparenden Protokollen wird ein gemeinsamer Zeitplan für mehrere Geräte in einem Netzwerk verwendet, der wie folgt funktioniert: Zwei miteinander kommunizierende Geräte wachen zu den gleichen Zeitpunkten auf, tauschen Pakete aus und gehen dann zurück in einen Schlafzustand, bis der nächste geplante Aufwach-Zeitpunk eintritt. Mit diesem Schema können die Geräte für die meiste Zeit in einem Stromsparmodus bleiben.

Jedoch muss ein solcher Zeitplan zwischen den Geräten erst verhandelt werden, bevor er Gültigkeit erlangt. Zudem müssen die Takte (clocks) der Geräte synchronisiert werden. Dies erfolgt in Neighbor Discovery Protokollen wie folgt. Ein Gerät versendet wiederholt Pakete auf einer bestimmten Auswahl an Kanälen, während ein anderes Gerät wiederholt seinen Empfänger aktiviert, um nach Paketen zu suchen. Beide Geräte schlafen in der Zwischenzeit, um ihren Energieverbrauch zu minimieren. Zwei Geräte haben sich erfolgreich gefunden, sobald beide ein erstes Paket von der Gegenseite erhalten haben. Generell führen längere und häufigere Empfangsphasen sowie der häufigere Versand von Paketen zu einem erhöhten Energieverbrauch, aber auch zu kürzeren Latenzen für den Verbindungsaufbau.

In den letzten beiden Jahrzehnten wurden eine große Zahl von Neighbor Discovery Protokollen entwickelt, die eine möglichst niedrige obere Schranke für die Latenz garantieren und gleichzeitig den Energieverbrauch der drahtlosen Geräte minimieren. Während die vorgestellten Lösungen zunehmend effizienter wurden, blieben die grundlegenden Performance-Grenzen solcher Protokolle unbekannt.

In dieser Arbeit analysieren wir die grundlegenden Performance-Grenzen von Neighbor Discovery Prozeduren. Wir stellen eine Theorie auf, aus der erstmals eine sichere und erreichbare, Energie-abhängige untere Schranke für die Latenz von Neighbor Discovery Protokollen abgeleitet werden kann. Kein Neighbor Discovery Protokoll kann kürzere Latenzen als die von dieser Schranke vorgegebenen garantieren. Wir stellen mehrere solcher Schranken in zwei verschiedenen Metriken vor: Schranken

für die Maximallatenz bei gegebenem Energieverbrauch sowie für die Maximallatenz bei gegebenem Energieverbrauch und gegebener Kanalauslastung. Zudem stellen wir Schranken für unterschiedliche, relevante Szenarien vor. Ein Beispiel sind Szenarien, in denen alle Geräte gleich viel Energie verbrauchen (symmetrisches Neighbor Discovery). Ein anderes Beispiel sind Situationen mit unterschiedlichem Energieverbrauch verschiedener Netzwerkteilnehmer (asymmetrisches Neighbor Discovery).

Im Falle von unidirektionalem Neighbor Discovery, in welchem ein einzelner Empfänger mit einem einzelnen Sender kommuniziert, können die von den Schranken vorhergesagten Latenzen in 100 % aller Vorgänge erreicht werden. Wenn jedoch beide Geräte sowohl senden und empfangen, können die vorhergesagten Latenzen nur noch mit hoher Wahrscheinlichkeit erreicht werden. Ein bestimmter Anteil der Vorgänge schlägt fehl, weil die Pakete der beiden Geräte miteinander kollidieren, oder weil das Radio während dem Umschalten zwischen Empfangen und Senden eingehende Pakete verwerfen muss. In Netzwerken mit wenigen Geräten, die gleichzeitig die Neighbor Discovery Prozedur ausführen, lassen sich die vorhergesagten Latenzen ebenfalls mit hoher Wahrscheinlichkeit erreichen. Bei einer wachsender Anzahl von Geräten sinkt diese Wahrscheinlichkeit jedoch, und beschränkte Latenzen sind auch in der Praxis nicht mehr realisierbar. Welche Latenz mit welcher Fehlerwahrscheinlichkeit für eine bestimmte Zahl an Geräten erreicht werden kann, ist weiterhin ein offenes Forschungsproblem.

Des weiteren analysieren wir die Leistungsfähigkeit von mehreren bekannten Protokollen und vergleichen sie mit den zugehörigen theoretischen Grenzen. Die meisten existierenden Protokolle lassen sich als *Slot-basiert* kategorisieren. Sie teilen die Zeit in gleich lange Intervalle ein, welche *Slots* genannt werden. In einigen Slots schläft das Gerät, während andere aktiv zur Kommunikation genutzt werden. Zu Beginn und/oder am Ende jedes aktiven Slots sendet das Gerät ein Paket und horcht in der Zwischenzeit nach eingehenden Paketen. Die Prozedur ist erfolgreich, sobald sich zwei aktive Slots von unterschiedlichen Geräten zeitlich überlappen. Wir zeigen, dass mit Slot-basierten Protokollen tatsächlich die zugehörigen Leistungsschranken erreicht werden können, wenn man als Metrik die Maximallatenz bei gegebener Kanalnutzung und gegebenem Energieverbrauch betrachtet. Ist jedoch jede beliebige Kanalnutzung zulässig, was typischerweise in kleinen Netzwerken mit wenigen Teilnehmern der Fall ist, ist ihre Leistung nachweislich suboptimal.

Um auch in dieser wichtigen Metrik optimale Leistungen zu erzielen, müssen Paradigmen jenseits der Slot-basierten Techniken entwickelt werden. Wir betrachten daher slot-freie, intervallbasierte Protokolle, die wie folgt funktionieren. Ein Gerät sendet periodisch Pakete, das andere hört für eine bestimmte Dauer mit einer bestimmten Periode auf den Kanal. Beide Perioden sind voneinander unabhängig und können jeden beliebigen Wert annehmen. Solche Protokolle besitzen 3 Freiheitsgerade, nämlich die beiden Perioden und die Dauer der Empfangsphasen. Obwohl Milliarden von Bluetooth-Low-Energy-Geräten auf diesem Prinzip basieren, waren vor dieser Arbeit weder dessen Latenzen, noch Konfigurationen mit vorteilhaften Relationen zwischen Energieverbrauch und Latenzen bekannt.

In dieser Arbeit stellen wir daher eine mathematische Theorie vor, um die Latenzen intervallbasierter Protokolle für alle möglichen Parametrierungen zu berechnen. Unsere

Theorie zeigt, dass für alle Parameterwerte Maximallatenzen garantiert werden können, wenn man eine endliche Menge an Singularitäten mit unbeschränkten Latenzen ausnimmt. Daher können intervallbasierte Protokolle in Applikationen eingesetzt werden, in denen deterministische Latenzen notwendig sind. Für solche Applikationen waren zuvor nur Slot-basierte Protokolle geeignet.

Darauf aufbauend beschreiben wir ein mathematisches Optimierungsframework, welches jeden gewünschten Energieverbrauch (in Form eines Duty-Cycles) in ein Set an optimierten Parametern für intervallbasierte Protokolle übersetzt. Dieses Framework besitzt eine geringe Rechenkomplexität. Wir zeigen, dass intervallbasierte Protokolle, die mit diesem Framework parametrisiert werden, für jeden gewünschten Energieverbrauch die theoretisch optimalen Latenzen erzielen. Zudem stellen wir eine Implementierung eines solchen Protokolls vor und führen verschiedene Techniken ein, um das nicht-ideale Verhalten der drahtlosen Hardware zu kompensieren. Unsere Experimente zeigen, dass eine solche Implementierung die Leistungsfähigkeit aller bekannten Neighbor Discovery Protokolle übertrifft und beispielsweise $4\times$ geringere Maximallatenzen als das bekannte U-Connect Protokoll erreicht.

Zudem stellen wir eine Technik vor, die zusätzlich die Durchschnittslatenzen solch optimaler intervallbasierter Protokolle reduziert, ohne die Maximallatenzen signifikant zu erhöhen. In solchen Protokollen ist die Latenz, mit der ein Gerät E ein anderes Gerät F findet, unabhängig von der Latenz für das Auffinden von Gerät E durch Gerät F. Dies führt zu höheren Durchschnittslatenzen für das beidseitige Auffinden als nötig, weil beide partiellen Prozeduren erfolgreich abgeschlossen werden müssen. Wir stellen daher einen kooperativen Ansatz vor, der dazu geeignet ist, diese Latenz zu reduzieren. Dabei sendet jedes Gerät Informationen zum Zeitpunkt seiner nächsten Empfangsphase in seinen Paketen. Empfängt ein Gerät eine solche Information, passt es seinen Sendezeitplan entsprechend an. Wir nennen diese Technik *gegenseitiges Assistieren*. Verglichen mit nicht-kooperativen Ansätzen können die Durchschnittslatenzen damit um bis zu 43 % reduziert werden.

Zusammengefasst stellen wir zunächst mehrere fundamentale Schranken für Neighbor Discovery Protokolle auf. Zudem stellen wir eine umfassende Theorie vor, um intervallbasierte Protokolle zu modellieren. Darauf aufbauend präsentieren wir ein mathematisches Framework zum Entwurf von Protokollen, die diese Schranken in der Praxis erreichen. Nach unserem besten Wissen stellen wir das erste Protokoll vor, welches optimale Relationen zwischen seinem Energieverbrauch und seiner Maximallatenz erreicht. Zudem reduzieren wir die Durchschnittslatenzen durch *gegenseitiges Assistieren*.

Contents

1 Introduction

1.1 Towards Low-Power Wireless Networks

With the introduction of WiFi, wireless connectivity has become ubiquitous in our everyday lives. The publication of the IEEE 802.111-standard (which is commonly referred to as *WiFi*) in 1999 [49] has initiated a rapid development. For example, WiFi is frequently used to connect laptops and smartphones to digital subscriber line (DSL) base stations for providing Internet access. Almost every modern laptop and smartphone is equipped with WiFi. Besides private WiFi networks, public access points have become widespread. Further, beyond providing Internet access, novel applications of wireless networks, like smart homes, are further increasing the demand for wireless connectivity.

Typically, WiFi networks operate in a so-called *infrastructure mode*, with a fixed access point that coordinates the network. The most basic communication mode of the IEEE 802.11-standard follows a carrier-sense multiple access (CSMA)/collision avoidance (CA) paradigm [49] [48]. If one device has data to be transmitted, it senses the channel for ongoing transmissions. Only if the channel is found idle, the device transmits its beacon. If the channel is busy, the transmitter waits until all ongoing transmissions have terminated. In addition, a random backoff period prevents multiple waiting devices from transmitting their beacons at the same point in time and hence colliding.

The CSMA/CA scheme used in IEEE 802.111 allows for a reliable, high-throughput communication, which has been the main concern in the development of early wireless protocol suites. However, the fact that every device might initiate a transmission at any point in time implies that all network participants need to be ready for reception and hence awake during all times. This does not fit well the energy needs of mobile devices, which run on limited power sources, such as batteries or energy harvesting systems.

As stated in [62], “the best way to save power for wireless communication devices would be to switch them off”. Therefore, the main idea behind dedicated power-saving protocols is to allow mobile devices to remain asleep for large fractions of the time. Instead of being required to listen continuously, they only need to wakeup at certain predefined time instances. This, however, requires certain coordination schemes among multiple devices, which determine the common points in time at which multiple devices wakeup jointly. For this purpose, the 802.11-standard defines a technique called the *point coordination function* [49]. Here, a central device (i.e., the access point) defines periodic common wakeup times for all devices. If a network participant has no data to send, it only has to wakeup according to them. However, each device may transmit data during all times and the access point buffers incoming beacons. When the next regular, periodic-interval-based wakeup time occurs, the buffered beacons are dispatched. Since the AP is usually powered by the electricity grid, it can afford a higher energy-consumption than the mobile devices and therefore take over the buffering. This scheme allows smartphones and other battery-limited devices to be part of a WiFi network, without quickly draining their batteries.

With the ongoing trend towards wearable electronics and the Internet of Things (IoT), mobile ad-hoc networks (MANETs), which do not rely on any fixed infrastructure, are continuously gaining in importance. For example, wrist-worn fitness trackers, such as e.g., FitBit, send their data wirelessly to a mobile phone, without the need of any grid-powered base-stations. In MANETs, all participants of the network are mobile and hence need to maximize their battery lives. Further, novel applications of MANETs require extremely small form-factors an hence small batteries combined with long required battery runtimes. Examples for such size-constrained systems are wireless in-ear headphones or tags used for object tracking. Further, companies like Google and Apple have been working on smart wearables for health care, such as a wireless smart contact lens [64]. In such applications, due to the size constraints, no large batteries are possible and energy efficiency is a crucial design requirement. Another driving field of applications for MANETs are IoT scenarios, in which devices opportunistically form networks in changing environments. Here, ad-hoc communication is essential for the proper functioning of entire systems, since the availability of a fixed infrastructure changes with the environment a device is placed into.

Clearly, techniques that require an access point to buffer transmissions of the network participants are not applicable in such scenarios. While 802.11 also defines a power-management scheme for its (less frequently used) ad-hoc mode (called *independent basic service set*, *IBSS*), protocol-suites that are especially designed for low power communication in MANETs have been developed. The power demand of WiFi devices has been described to be around 500 mW [75], which is still significantly higher than the energy budgets of devices like e.g., wireless sensor nodes or heart rate monitors. Driven by the need for lower energy consumptions, dedicated MANET protocols, such as e.g., Bluetooth Low Energy (BLE) have been developed. BLE is designed to allow devices to operate on a coin cell for at least one year [75]. Other examples for similar low-power MANET protocols are IEEE 802.15.4/Zigbee [50] and ANT/ANT+ [46]. The number of such networks is growing rapidly, with 4.7 billion BLE devices being expected to be sold within 2018 [59], and additional ones using other protocols.

BLE adopts the paradigm of saving power by switching off devices for most of the time. This is achieved by an appropriate protocol, which forms a contract between the master M and the slave S. This contract specifies the points in time both devices have to be awake. At $t_{anchor} + k \cdot T_c, 0 \leq k \leq \infty$, both devices have to wake up. t_{anchor} is a point in time to which both devices have been synchronized, and T_c is a parameter referred to as the *connection interval*. After a pair of devices have woken up, M will start a *connection event* by sending a packet. S will then acknowledge it by sending another packet 150 µs later, which might be either an empty response, a payload packet or a packet performing control functions. If there is more data to be transmitted, more pairs of packets are exchanged in the same manner. Apart from necessary processing tasks, the devices may sleep during all other times. In addition, there is a parameter called the *slave latency* N_{sl}, which increases the sleep duration of S in case there is nothing to signal in a connection event. If M and S have agreed on a slave latency of N_{sl}, the slave might skip N_{sl} connection events without waking up. The connection interval T_c must range from 7.5 ms to 4.0 s and N_{sl} may be up to 500 events.

In general, the concept of prenegotiated wakeup schedules allows all devices to remain asleep for most of the time. It forms the technical foundation of most low power protocols, e.g., ANT/ANT+ [46]. The scheduling of such wireless links is a well-resolved problem. However, before such a wakeup schedule can be realized, two devices need to discover each other wirelessly, initiate a first contact, negotiate the connection interval and synchronize their clocks. The procedure of initializing a first contact is referred to as *ND*. It is not only necessary in BLE networks, but forms a task that is fundamental for the proper functioning of all MANETs. As of today, it is an active topic of research.

1.2 Neighbor Discovery

1.2.1 Applications of Neighbor Discovery

The purpose of ND is to allow all wireless devices finding the devices in range and, if necessary, to synchronize their clocks. Unlike in the connected communication schemes described above, there is no common anchor point and communication needs to be carried out asynchronously. For this purpose, dedicated wireless protocols (so-called *ND protocols*), are widely used. Some of these protocols are part of wireless protocol suites such as BLE, whereas other ones are standalone ND solutions proposed in the literature.

ND plays a key role in a host of applications. For example, using ND protocols, smartphones can connect to fitness gadgets, sensors and smartwatches. Similarly, location beacons are able to broadcast messages e.g., for pushing product details onto smartphones or for helping to track a lost item. Or, friend-finder applications on smartphones can detect the devices of other persons that are in range, without the need for a base-station or the Global Positioning System (GPS). In addition, the IoT requires mobile devices to be able to form ad-hoc-networks for realizing collaborative interactions between them. Many IoT devices are mobile and have to discover their surrounding devices continuously. For example, smart furniture will opportunistically connect to wearables of different users, without requiring any previous configuration procedures.

Since all participants of a MANET are powered by batteries or energy harvesting, power-efficient ND is a crucial requirement in such networks. ND plays an important role in the energy-budgets of many mobile applications. For example, devices such as location beacons broadcast their presence wirelessly during their whole lifetimes. Hence, performing ND is their main mode of operation and the energy consumption is dominated by it. Consequently, the efficiency of this procedure determines whether the battery lifetime requirements can be fulfilled or the beacons stop operating preliminarily. Another example are scenarios in which multiple persons wear a sensor or smart watch for the purpose of contact tracking. Here, each device continuously advertises its presence and scans for other devices in range. For assessing which subjects are close to each other, all devices continuously perform ND and probably never establish any synchronized connection for exchanging larger volumes of data.

However, also on devices whose energy budgets are not dominated by ND, efficient protocols help in increasing the battery lifetime. An opinion poll in Germany in 2015 has revealed that the battery runtime is an important criteria for 48.04 % of the participants when buying a smartphone [66]. Since the heterogeneous components of a smartphone lead to different current draws in different usage scenarios, efficient ND protocols can reduce the energy consumption of the MANET interface and hence increase the battery runtime, whenever the surrounding is to be scanned continuously. E.g, efficient ND protocols can reduce the energy consumption for indoor localization on smartphones, where the packets of location beacons need to be scanned for in a continuous manner.

1.2.2 Properties of Neighbor Discovery Protocols

The analysis and design of efficient ND protocols, which is addressed in this thesis, is a fundamental problem of MANETs. The main goal is to design ND protocols that fulfill the following properties:

- Low Discovery Latencies: All neighbors should be discovered within the lowest possible amount of time.
- Low Energy Consumptions: The devices carrying out the ND procedure should consume as little energy as possible. Usually, the duty-cycle, which is the fraction of time a device is not in a sleep mode, is used as the main energy metric.
- Low Channel Utilizations: The fraction of time the channel is busy should be minimized. High channel utilizations lead to an increased number of beacon collisions. Hence, protocols with high channel utilizations do not scale well towards larger networks with a larger number of devices.

In other words, ND protocols should ensure fastest possible discoveries with lowest-possible duty-cycles and channel utilizations. Moreover, ND protocols should be practical in real-world applications, which imposes the need to support the following additional properties and features.

- Duty-Cycle Granularity: Ideally, protocols should support an arbitrarily fine granularity of duty-cycles that can be realized. In practice, only a certain set of duty-cycles are supported by most protocols. For example, Disco [15] in its basic form[1] can only realize the duty-cycles η that are given by $\eta = \frac{p_1 \cdot p_2}{p_1 + p_2}$, whereas p_1 and p_2 form a pair of coprimal numbers.

[1] A version with 3 different prime numbers has also been proposed in [15].

- Support for Asymmetry: Discovery protocols in which each device operates using the same duty-cycle are referred to as *symmetric* ND protocols. In contrast, some protocols, e.g., [15, 2], allow multiple devices to operate using different duty-cycles, while still achieving beneficial latency/duty-cycle relations. Such scenarios are referred to as *asymmetric*. In the best case, a ND protocol allows each device to adjust its duty-cycle autonomously during runtime, while still achieving low discovery latencies.
- Determinism: Deterministic ND protocols, e.g., [15, 23] guarantee deterministic discoveries within a certain, duty-cycle-dependent upper latency bound. Typically, this bound is only valid when the impact of beacon collisions is neglected or if the discovery is only performed unidirectionally.
- Robustness: The number of collisions is basically given by the channel utilization (i.e., the fraction of time a device occupies the wireless channel). However, different protocols potentially have different vulnerabilities to beacon collisions and hence, their discovery latencies are affected in different degrees whenever a collision occurs. This aspect has not gathered significant attention in the literature, yet.
- Gossip: Some protocols, e.g., [82], rely on piggy-backing information on already discovered devices onto their beacons. This can accelerate the discovery of larger numbers of neighbors. However, since every device has to be discovered by at least one other device, such schemes also make use of pairwise discovery for transmitting the data on already known neighbors.

Direct pairwise discovery is the procedure of two devices discovering each other independently by exchanging an initial pair of beacons. In contrast, indirect approaches transmit information on already known neighbors along with their beacons. Since pairwise ND is the underlying technique of gossip-based protocols, the optimization of pairwise discovery procedures is highly relevant for all ND protocols, including gossip-based ones. Therefore, finding optimal pairwise discovery protocols forms the fundamental problem of ND in MANETs. This thesis focuses on pairwise discovery. Furthermore, we consider only *asynchronous ND*, in which there is no global clock synchronization among the devices in discovery mode. Such a clock synchronization could be obtained e.g., using GPS, which is energy-expensive and not always available.

1.2.3 Basic Principle of Neighbor Discovery

In general, ND works as follows. A device broadcasts beacons on a certain set of channels, whereas other devices repeatedly switch on their receivers for short amounts of time. To save energy, the devices sleep in the meantime. Since any previous coordination between a pair of devices is not possible, the points in time at which each device wakes up are independent of each other. Two devices have discovered each other successfully once both have received a first beacon from their opposite. Therefore, the discovery of a device F by a device E is successful if, and only if F sends a beacon during a reception phase of E. Further, since the beacon from F must overlap *entirely* with a reception phase of E, the length of the reception phase is effectively reduced by one beacon length ω, as a beacon transmitted during the last ω time-units of a reception phase only partially overlaps with it. In general, longer and more frequent listening/broadcasting periods result in higher duty-cycles and hence energy-consumptions, but potentially achieve shorter discovery latencies. In other words, ND is similar to “Finding Needles of Connectivity in Haystacks of Time” [16], since relatively short reception phases need to be probed for by multiple short beacons.

It is worth mentioning that there are approaches such as [65, 38], which assume that each device has an additional low power receiver that continuously senses the channel for wakeup beacons. However, many wireless systems do not have such an additional receiver. Moreover, such a receiver can also be duty-cycled to further reduce its energy-consumption, which imposes the need for ND protocols even if such additional hardware is available.

Protocols that achieve beneficial trade-offs between discovery latencies, energy consumptions and channel utilizations are an active topic of research. We in the following present a brief overview on protocols presented in the literature.

1.2.4 Probabilistic Protocols

Probabilistic protocols attempt to discover a neighboring device with high probabilities, but cannot guarantee a successful discovery. For example, the main idea behind *Birthday Protocols* [54] is to transform the birthday paradox to the ND problem. As described in [54], if 23 persons are within one room, the probability that two persons in this room have the same birthday exceeds 50 %. From a protocol design perspective, an analogon can be formed as follows. Let time be subdivided into equal-length slots. Based on certain transition probabilities, each device determines randomly on a per-slot basis whether it goes to a sleep mode, starts receiving or transmits a beacon. For an average of k active slots out of n slots, the duty-cycle is given by $^k/_n$. For e.g., a duty-cycle of 7 %, a device running this protocol can discover a second device within $n = 1000$ slots in 99.5 % of all cases [54]. Since every device can independently chose its active slots, such protocols are especially well-suited for devices that are powered by energy harvesting [39]. Since the available energy varies over time, devices can adjust their active points in time without restrictions, leading to a maximum degree of flexibility. However, such protocols cannot guarantee any upper latency bounds, within which discovery can be assured. Since such upper latency bounds are desirable in many applications, the vast majority of proposed protocols focuses on deterministic ND, in which discovery is always successful within a certain amount of time, if beacon collisions are neglected. Next, we describe such deterministic protocols, which can be subdivided into slotted and slotless protocols.

1.2.5 Slotted Protocols

Most common ND protocols are referred to as *slotted protocols*. They subdivide time into multiple, equal-length partitions, called slots [15]. In some slots, the device remains asleep, whereas other ones are active slots used for communication. At the beginning and/or the end of each active slot, a device sends a beacon and listens for the rest of the sloth length I, as depicted in Figure 1.1 a). The discovery procedure is complete once two active slots overlap in time.

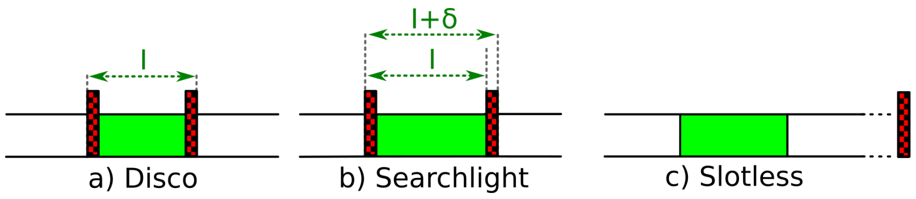


Figure 1.1: Slot design of a) Disco [15], b) Searchlight [2] and c) reception and transmission in slotless protocols. The hatched bars depict beacon transmissions, the colored rectangles depict reception phases.

The main reason for the popularity of slotted protocols is their simplicity of analysis: Once two active slots overlap in time, discovery is guaranteed regardless of which fractions of these slots actually overlap. If two devices run the same pattern of active and passive slots, the temporal offset between these patterns on two devices determines the pair of overlapping slots. Since the pattern of active and passive slot repeats with a certain period, this offset is equivalent to the amount of time by which both periods are shifted against each other, which is usually random. The discovery latency of a ND protocol is measured from the time at which two devices come into their range of reception to the mutual reception of a pair of beacons. The global worst-case latency is determined by the offset that leads to the longest latency among all possible ones. In slotted protocols, all offsets that lead to an overlap of the same slots achieve the same latency. Hence, the amount of offsets that need to be considered when assessing the worst-case latency becomes finite, which greatly simplifies the design and analysis of such protocols.

Each slotted protocol known from the literature defines a unique discovery schedule, which determines the pattern of active and passive slots. Most known schedules are repetitive, which implies that the same pattern repeats after a certain number of slots. These schedules are designed such that the latency is bounded for all possible temporal correlations between two devices. We in the following present multiple popular deterministic schedules.

Quorum Schedules: One of the first slotted protocols has been proposed in [68], which is based on a quorum schedule to achieve determinism. The repetition period of the slot pattern is thereby organized in the form of a $m \times m$-matrix, and the active slots are determined by a column and a row of this matrix. Such a schedule is depicted in Figure 1.2. The upper part of the figure depicts a representation of the slot schedule, in which a period of m^2 slots is organized in a $m \times m$ matrix. From this matrix, device E chooses a column as its active slots, whereas device F chooses a row. The lower part depicts a linear representation of the first $3 \cdot m$ slots of this schedule, which is obtained from the matrix by concatenating its individual rows. As can be seen, with this scheme, a pair of overlapping slots exists in every period of m^2 slots and hence, the discovery latency becomes bounded.

Coprimal Intervals: Another popular approach, which achieves shorter worst-case latencies than those of the quorum protocol proposed in [68], is Disco [15]. Its schedule for

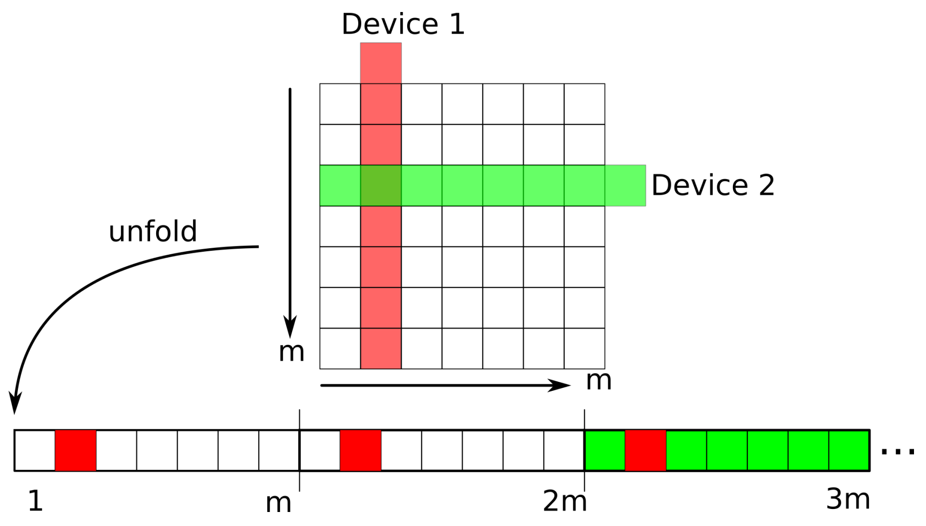


Figure 1.2: Design of a quorum schedule according to [68]. The upper part depicts the matrix representation of the schedule, the lower part its unfolded linear representation.

two devices that attempt to discover each other is shown in Figure 1.3 a). Disco assumes that each device chooses a period T, which consists of a prime-number of slots. The first slot in T is active, whereas the remaining ones are sleep slots. Each device can chose its prime period independently. The problem of finding points in time at which both devices are active simultaneously can be formulated as a set of congruences, to which the Chinese Remainder Theorem applies. This theorem states that the worst-case latency is the product of the coprimal periods of both devices. To prevent cases in which two devices chose the same period and therefore no deterministic discovery can occur, each device actually choses two different prime periods T_1 and T_2 in parallel for determining its active slots. Even in cases in which both devices choose the same pair of prime periods, discovery is guaranteed, since each primal interval is always paired with another coprimal interval on its opposite device.

Another popular protocol is U-Connect [23]. Its schedule, which also relies on coprimal interval lengths, is depicted in Figure 1.3 b). Like in Disco, each device choses a prime number of slots T as its period. In the example depicted in Figure 1.3 b), two devices have chosen the same period, but the protocol also permits different periods on different devices. The first slot in every period T is active. In addition, each device has a super-period of T^2 slots. The first $\lfloor \frac{T+1}{2} \rfloor$ slots of each super-period are also active. With this scheme, mutual discovery is guaranteed also in cases with identical periods on two devices. In symmetric settings (i.e., both devices have the same duty-cycle), for a given duty cycle, the discovery using U-Connect can be carried out within fewer slots than using Disco. Moreover, U-Connect supports shorter slot-lengths than most other protocols, which further reduces its discovery latencies. Unlike Disco (cf. Figure 1.1 a)), the implementation of U-Connect presented in [23] uses two different types of slots. During the regular slots of the interval T, the device is in reception mode (indicated by RC in Figure 1.3 b). Additionally, the device continuously transmits within the active slots of the hyper-period T^2 (indicated by TR in the Figure). This scheme allows for a reduction

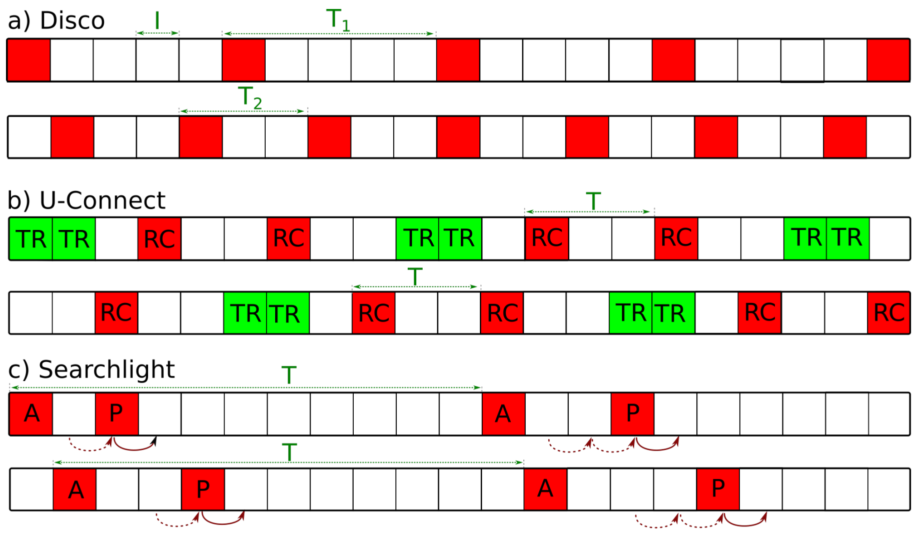


Figure 1.3: Discovery schedules of a) Disco [15], b) U-Connect [23] and c) Searchlight [2].

of the slot length, because in every RC slot, the device only needs to check whether there is power on the wireless channel, whereas other protocols need to define sufficiently long slot lengths for enabling a full beacon reception. Only if power is detected on the channel, a device listens for incoming beacons. However, this scheme is not directly comparable to many other protocols, including Disco, since the hardware has to support clear channel assessments. Further, noise might lead to extended listening durations and therefore increased duty-cycles.

Systematic Probing: The popular Searchlight [2] protocol is based on systematically changing the active slot within each period of T slots, such that at least half of the slots in T become active at some point in time. Like in Disco, each active slot consists of a listening window, which is preceded and terminated by sending a beacon beacon. However, as shown in Figure 1.1 b), active slots are by a certain duration δ longer than passive slots (so-called *over-length slots*). The schedule of Searchlight is shown in Figure 1.3 c). It is composed of a period of T slots and a hyper-period of (approximately) $1/2 \cdot T^2$ slots. The first slot of each period is referred to as an *anchor slot* A. In addition, there is another active slot per period, the so-called *probe slot* P. In the first period, the second slot is a probe-slot. In each succeeding period, the position of the probe-slot is increased by one until reaching half of the interval T. With this scheme, the probe-slot of one of the two devices is guaranteed to overlap with the anchor slot of its opposite device within the hyper-period of $1/2 \cdot T^2$ slots. Hence, an upper latency bound of approximately $1/2 \cdot T^2$ slot lengths is achieved. In addition, with the over-length slots shown in Figure 1.1 b), every second probe-slot can be skipped and therefore a maximum latency of about $1/4 \cdot T^2$ slots is guaranteed. This version of Searchlight is usually referred to as *Searchlight-Striped* or *Searchlight-S*.

Fundamental Bounds and Cyclic Difference Sets: A fundamental bound for slotted protocols has been presented in [87, 88]. This bound implies that for guaranteeing discovery within a period of T slots, no protocol may have less than $k = \sqrt{T}$ active slots per period. Further, schedules in which the active slots are defined by cyclic difference sets have been presented in [87, 88], which can actually achieve the latencies (in terms of slots) anticipated by this bound.

The underlying principle of such difference-set based protocols is as follows. A difference set (T, k, m) is defined [88] as a set of elements $X = \{x_1, x_2, x_3, ..., x_k\}$, in which there are exactly m ordered pairs (x_i, x_j), such that for every $d \in [1, T-1]$, it is:

$$d = (x_i - x_j) \mod T \tag{1.1}$$

Let us consider a protocol, in which the elements $x_1, .., x_k$ of a $(T, k, 1)$ - difference set determine the indices of the active slots, such that every x_i'th slot within each period T is active. Because of the following considerations, such schedules must be optimal.

- Since the pattern of slots repeats every T slots, all possible offsets between any two slots on two devices lie within $[0, T-1]$ slot lengths.
- As already explained, with an appropriate slot design, all possible offsets with unique discovery latencies can be described by an integer number.
- Let us first assume that two devices run the same schedule and that the slot boundaries of both devices are perfectly aligned in time. Then, two slots on two devices having an offset of d is equivalent to the difference of their slot positions $|x_i - x_j|$ having a value of $d \in \mathbb{N}_0$. Since the active slots are determined by a difference set, for every $d \in [0, T-1]$, there will be exactly $m = 1$ pairs of slots for which $x_i - x_j \mod T = d$. This means that a certain pair of two active slots (x_i, x_j) exists for all values of $d \in [1, T-1]$, which will be spaced by $x_i - x_j \mod T = d$ sloth lengths. If now the slot boundaries of two devices are no longer perfectly aligned, but shifted by an offset d that is a non-integer number, these slots will be shifted such that they partially overlap in time. The difference set ensures that a pair of (partially) overlapping slots will exist for every possible amount of shift d.
- From the points above, it can be concluded that discovery is guaranteed within T slots.
- Further, as there is exactly one pair of overlapping slots per offset, the number of active slots k per period T cannot be reduced any further, since deterministic discovery would become impossible otherwise. Hence, difference sets lead to the lowest possible duty-cycle for achieving a latency of T slots.

The vast majority of slotted protocols, such as e.g., Disco [15], Searchlight [2] or U-Connect [23], have been proposed after this bound and the difference set-based protocols had been known. Though they achieve higher worst-case latencies than the optimal, difference set-based protocols, they greatly improve the practicality. In particular, only a very limited number of cyclic difference sets are known [17], and hence the number of

Protocol	$L(\eta)$
Disco [15]	$\frac{4}{\eta^2}$
U-Connect [23]	$(\sqrt{\frac{1}{2\eta}+\frac{9}{16\eta^2}}+\frac{3}{4\eta})^2$
Searchlight Striped [2]	$\left\lceil \frac{\lfloor \frac{1}{\eta} \rfloor}{2} \right\rceil$
Bound for slotted protocols [87, 88]	$\frac{1}{\eta^2}$
Bound for overflowing slots [55, 56]	$\frac{1}{2\eta^2}$

Table 1.1: Worst-case numbers of slots within which discovery is guaranteed given a certain duty-cycle η.

duty-cycles that can actually be realized using them is small. In contrast, later protocols can operate using a larger number of different duty-cycles, though still not being able to realize a contiguous range of duty-cycles in an arbitrarily fine-grained manner.

The bound anticipated in [87, 88] has been refined in [55, 56] for protocols with overflowing slots (i.e., protocols that for each slot send a beacon slightly outside of the slot boundaries). The worst-case number of slots within which discovery is guaranteed using a certain duty-cycle can thereby be halved compared to non-overflowing slots. Though overflowing slots had been applied before in other protocols, e.g., Searchlight-Striped [2], none of them could actually reach this bound. However, with an overflowing slot design, difference set-based schedules [55, 56] can guarantee discovery within the refined bound. Table 1.1 summarizes the (approximate) worst-case number of slots achieved by multiple popular slotted protocols, and the known bounds for slotted ND.

However, both the performance bounds presented in [87, 88] and [55, 56] are given in terms of slots, whereas the latency in terms of time is relevant in practice. This latency depends on the slot length, and the smallest possible slot-length is currently not known. As a result, the latencies slotted protocols can achieve in terms of time, as we study in this thesis, have not been clear before.

Other Slotted Approaches: Whereas the slotted protocols above represent the main techniques for achieving determinism, a large number of further works have been carried out recently to improve slotted discovery protocols, of which we briefly present a selection below. In [67], a framework called HELLO to construct slotted discovery schedules has been proposed. Disco, U-Connect and Searchlight can be constructed as special cases of HELLO. As stated in [67], Searchlight is the optimum schedule for symmetric duty-cycles among the ones that can be constructed using this framework. Further, [82] proposes to apply additional slots in protocols like Disco to exchange information on already known neighbors. In [10], two new slotted protocols called HEDIS and TODIS have been proposed, which outperform Searchlight in cases with asymmetric duty-cycles. For symmetric discovery, Searchlight has been reported to perform better. Further, [55] combines overflowing slots (as described for Searchlight) with cyclic difference sets.

Whereas difference sets can only be realized for a few target duty-cycles, an algorithm to create approximations (with reduced performances) for every specified duty-cycle has been presented, which provide lower latency/duty-cycle products than Searchlight. In [10], a scheme to transform any schedule for non-overflowing slots, such as Disco or U-Connect, into an efficient one for overflowing slots has been proposed. Further, in [85], a scheme to adjust the slot length adaptively, based on the observed collision rate, has been presented. This can reduce the discovery latencies in busy networks.

Pseudo-Slotted Protocols: The assumption of fixed-length slots has been dismissed by some recent works. Though still assuming slotted time, in the recently proposed protocols Lightning [76] and Nihao [61], beaconing and reception have been decoupled from each other by using dedicated receive and transmit slots. In particular, Lightning partially abolishes the assumption of slotted time by introducing two slot-types for reception and transmission with different durations of activity. Though there is a universally defined slot length, only reception is carried out throughout the whole slot. In contrast, in each transmission slot, the device transmits a beacon and sleeps for the rest of the slot length. For determining the set of active and passive slots, Lightning applies a Searchlight-like schedule. Simulation results indicate better performances than Disco, U-Connect and Searchlight. However, the results rely entirely on simulations and therefore, it is not clear how Lightning performs in practical implementations. A similar approach with dedicated sleep- and reception slots is pursued by the Nihao-protocol [61]. Unlike most protocols described, Nihao concerns limiting the channel utilization for reducing the number of collisions. Similarly to Lightning, which is claimed to guarantee discovery within fewer slots than Disco and Searchlight [76], the results presented in [61] suggest that Nihao significantly outperforms the well-studied protocols Disco, U-Connect and Searchlight. These results suggest that temporally decoupling reception and transmission is a promising technique to decrease the worst-case latencies and duty-cycles. In fully slotless protocols, as studied in this thesis, reception and transmission are decoupled from each other completely, without assuming slotted time. They are described next. Note that also U-Connect [23] makes use of dedicated receive and transmit slots, but unlike in Lightning and Nihao, transmission and reception are carried out continuously during the whole slot.

1.2.6 Slotless Protocols

While the design of efficient slotted protocols has been extensively studied in the literature, many recent protocols follow a different paradigm, which may be characterized as “slotless”. Such slotless protocols have obtained less attention than slotted ones in the literature. Even though they are widely used in practice, the lack of related studies has lead to a limited understanding of their behavior.

Though a wide variety of different protocols that achieve deterministic latencies without relying on slots appear to be feasible, to the best of our knowledge, the only known deterministic slotless protocols are periodic interval (PI)-based ones. In such protocols, the transmitting device sends beacons with periodic intervals. The transmission duration is determined by the size of a beacon. The other device switches on its receiver also with periodic intervals and remains on for a specified duration during each period. The lengths of these periodic intervals, as well as the on-duration of the receiver, are independent of each other and drawn over continuous time. In commercially available slotless protocols, these two periods are chosen freely from a prescribed range of values. In addition, the reception window length is also chosen freely by the device. Typically, for a group of devices being able to discover each other, compatible parameter values need to be universally defined in their firmware. Once a beacon is sent within a reception window of the remote device, the ND procedure is successful. This scheme, which has been proposed first in [63], can be seen as an asynchronous form of the cyclic connected communication scheme used in BLE. Actually, the BLE specification [43] has adopted a PI-based ND scheme, which is described next.

ND in BLE Networks: BLE defines two different roles a connecting device can play. In particular, for connection setup, one device needs to be in the *advertising mode* and its opposite one needs to be in the *scanning mode*. Therefore, ND in BLE is commonly referred to as *advertising and scanning*. Advertising and scanning take place as shown in Figure 1.4: An advertising device periodically sends out advertising beacons. A group of consecutive beacons form an *advertising event*. In each of these advertising events, an advertising beacon is sent on at least one of three dedicated adverting channels. Multiple beacons in the same event are separated by d_{ch} time-units. Which of the three channels are to be used is determined by the application. Advertising events occur periodically with the *advertising interval* T_a. Independently of this, a scanner periodically switches on its receiver for a duration of d_s time units, called the *scan window*. This is repeated at every interval T_s, called the *scan interval*. In the next period, the scanner hops to the next advertising channel and again listens for advertising beacons. The BLE specification [41] requires the scanner to use all three advertising channels. If the scanner receives an advertising beacon, it may send a response to the advertiser within the same advertising event. The advertiser expects a response on the same advertising channel $d_{IFS} = 150\,\mu\text{s}$ after the end of the advertising beacon. d_{IFS} is called the *interframe-space*. Further, the advertising interval is composed of a static interval $T_{a,0}$ and a random part ρ, i.e.,

$$T_a = T_{a,0} + \rho, \tag{1.2}$$

where ρ is a random amount of time between $0ms$ and $10ms$. The random offset ρ is added to $T_{a,0}$ mainly to mitigate the effects of collisions. If two devices chose the same $T_{a,0}$, if one beacon collides, all subsequent ones would also collide without the random delay ρ. With the random delay, the probability of such repeated collisions is reduced.

The popular belief is that PI-based protocols cannot guarantee any upper bounds on the discovery latency and are therefore classified as random (similar to, e.g., birthday protocols [54]). Their discovery latencies for different parametrizations have still been unclear. Due to the potential to achieve high performances by temporally decoupling reception and transmission, we study their latency behavior in-depth in this thesis.

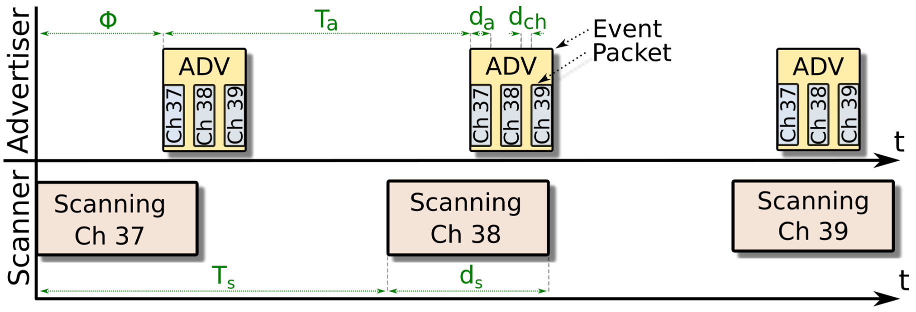


Figure 1.4: Neighbor discovery by advertising and scanning in BLE.

1.2.7 Outlook on Further Approaches

Beyond the representative selection of ND protocols we have presented above, there is a vast number of further approaches. Though all of them can be classified into one of the previously introduced categories (viz., slotted, slotless and probabilistic protocols), the intended purposes or the underlying assumptions of these protocols differ. For example, there are dedicated approaches for ND on radios with directional antennas [70, 20, 86]. Other notable approaches, which we do not describe in detail in this thesis, are e.g., [54, 39, 80, 71, 63, 68, 15, 23, 2, 55, 56, 10, 82, 67, 61, 22, 76, 87, 88, 11, 24, 32, 26, 69, 60, 25, 81, 4, 73, 85, 18, 83, 6, 7, 9, 31, 78, 72, 19, 84, 33, 14, 77, 74].

1.3 Scope, Contributions and Organization

This thesis concerns optimal ND protocols. First, we present discovery latencies that, for a given duty-cycle, no ND protocol can beat. Next, we model the discovery latencies of PI-based protocols. Based on this model, we derive optimal parametrizations using which PI-based protocols actually reach the anticipated performance bounds. Finally, we study trading reductions of the mean discovery latencies against slightly higher worst-case latencies.

This thesis organized in 6 chapters. An overview on low-power MANETs and ND is given in this chapter, along with the contributions made in this thesis. Further, in this section, we describe the problems addressed and contributions made in the remaining chapters. In addition, we list the previous publications on which parts of this thesis are built upon. In the remaining chapters, the following problems are addressed and the following contributions are made.

Chapter 2: In Chapter 2, we study the fundamental limits of ND. Our analysis is based on the fact that if two wireless devices run a certain ND protocol, due to their lack of synchronization, their sequences of beacons and reception phases are shifted against each other by a certain, random time-offset Φ. Because it represents the point in time both devices come into their range of reception for the first time, this offset solely determines the discovery latency. We introduce a property that we call the *coverage* Λ, which is a metric for the amount of different offsets Φ for which a sequence of beacons and reception windows lead to a successful discovery. Using this theory, we can infer a minimum number of beacons needed for guaranteeing a successful discovery in conjunction with a certain sequence of reception windows. Using analytical methods, we derive multiple fundamental bounds on ND from this number of beacons, which are in particular:

- The lowest worst-case latency that can be guaranteed in a unidirectional scenario, i.e., between a sender with duty-cycle β and a receiver with duty-cycle γ.
- The lowest worst-case latency any bi-directional, symmetric ND protocol can guarantee for a pair of devices with given duty-cycle η
- The lowest worst-case latency any bi-directional, asymmetric ND protocol can guarantee for any given pair of duty-cycles (η_E, η_F) of two devices E and F
- The lowest worst-case latency any ND protocol can guarantee without exceeding a certain channel utilization and hence collision rate

For the unidirectional case (i.e., one receiver attempts to discover one sender), the discovery latencies anticipated by the corresponding bound can be realized in 100 % of all cases, as long as no external disturbance occurs. For bi-directional discovery, where every device both sends beacons and listens for incoming transmissions, some discovery attempts fail and the predicted worst-case latency is exceeded. Such failures are caused by colliding beacons or by the radio being unable to receive any incoming beacons while switching between reception and transmission. However, a large fraction of all discovery attempts (e.g., more than 99.8 % for the protocol presented in Section 4) are successful within the anticipated latencies. For networks with small numbers of devices carrying out the discovery procedure simultaneously, the failure probability also remains negligible. However, for larger numbers of devices discovering each other simultaneously, collisions become more frequent and the bounds are no longer meaningful. A feasible performance metric for such scenarios would be the latency L that can be reached with a given probability for a given number of devices with given energy budgets. The optimal performance in this metric remains unknown and is left open for further research.

We also study whether existing protocols can actually reach the optimal performances given by these bounds. Towards this, we determine a theoretical lower limit on the slot length in slotted protocols. Using this slot length, we show that they can reach the latency/duty-cycle relations anticipated by our bounds in theory. However, due to hardware constraints, only larger slot lengths than optimal are feasible in practice, which in conclusion leads to slotted protocols always performing sub-optimally in the latency/duty-cycle metric. However, slotted protocols can reach optimal performances in the latency/duty-cycle/channel utilization metric, in particular for low channel utilizations and/or duty-cycles. In other words, for certain ranges of duty-cycles and channel utilizations, slotted protocols perform optimally in scenarios in which the channel utilization is constrained. This is typically the case in large networks with a high number of nodes being in discovery mode simultaneously.

Since slotted protocols cannot reach optimal latency/duty-cycle relations in scenarios without any constraints on the channel utilization, we next study slotless, PI-based protocols in detail. Such unconstrained scenarios are relevant for small networks with few devices carrying out the ND procedure in parallel. As already described, PI-based protocols provide 3 tunable parameters (viz., the intervals for transmission and reception and the reception window length). For determining whether such protocols can achieve higher performances than slotted ones, the relation between their parameter values and the corresponding latencies needs to be understood. This is addressed in Chapter 3.

Chapter 3: In Chapter 3, we study the discovery latencies of PI-based protocols in depth, thereby making the following contributions:

- For PI-based protocols, though being used in billions of BLE devices, the discovery latencies that can be achieved using a given set of parameters (i.e., the interval for beacon transmissions, the interval for reception and the length of the reception windows) have not been understood previously to this work. In Chapter 3, we present a model that can compute both the worst-case latencies and the mean latencies for all possible parameter values.
- Furthermore, we present an implementation of this model, which has low computational complexity.
- Using measurements on a large number of discovery procedures, we evaluate the validity of this model.

However, due to the large design space spanned by the 3 continuous-valued parameters, performing an exhaustive search over the whole design space for finding parameter values with minimal latencies is not feasible. Further, since the functions of this model cannot be simply "inverted" due to their recursive structure, using analytical methods for finding optimal valuations is aggravated. The problem of identifying beneficial parameter values is addressed in the next Chapter.

Chapter 4: In Chapter 4, we address the computation of optimal parameter values in the latency/duty-cycle metric. The main goal is to derive a set of parameters (viz., the interval lengths for advertising and scanning and the scan window length) for every specified duty-cycle, using which low discovery latencies are achieved. Since optimality in the latency/duty-cycle metric requires that collisions must not wield a major impact on the discovery procedure, PI-based protocols applying such configurations are well-suited for networks in which only a few devices are in discovery mode simultaneously. Related to this, we make the following contributions:

- We present two different parametrization schemes that translate any desired target-duty-cycle into a set of optimized parameter values. For one of these schemes, we formally prove that it can achieve the optimal latency/duty-cycle relations anticipated by the bounds presented in Chapter 2. Since the other scheme provides only marginally reduced discovery latencies, it is also essentially optimal.

- Further, we describe an accordingly parametrized PI-based protocol that minimizes collisions between a pair of nodes and minimizes the impact of hardware non-idealities.

- The performance achieved with this protocol is compared to those of multiple slotted protocols for slot lengths that lead to similar probabilities of failed discoveries. We show that our protocol achieves the lowest worst-case latencies among all known ND protocols for all duty-cycles. In addition, we demonstrate its proper functioning using measurements on a real-world implementation.

Whereas we focus on achieving the lowest-possible worst-case latencies in Chapter 4, there is further potential to reduce mean latencies of such protocols. The mean latencies are the delays experienced on the average in practice and therefore also relevant. We address reducing the mean latencies in Chapter 5.

Chapter 5: As already described, two-way discoveries in most slotted protocols take place nearly at the same points in time as the first of both one-way discoveries do, since both encounters happen within the same pair of slots. In contrast, in the slotless protocols considered in this thesis, both partial discoveries occur independently from each other. Because of this, two-way discovery can be accelerated in many constellations by exploiting mutual assistance, which works as follows. Every device broadcasts the point in time of its next wakeup phase with its beacons. If such a beacon is received, the receiving device schedules an additional beacon accordingly, which accelerates the second one-way discovery. In Chapter 5, we study slotless protocols that exploit this technique. In particular, we present the following contributions:

- Following the insight that different parametrizations allow for different latency improvements caused by mutual assistance, we propose a parametrization scheme for PI-based protocols that, when accepting a small degradation of the worst-case latencies from their optimal values, achieves a significant improvement of the mean latencies. In particular, this scheme reduces the mean latencies by up to 43 % compared to parameterizations that are optimized for worst-case latencies, only.

- We compare the mean- and worst-case latencies achieved by protocols built on such a parametrization scheme to those of multiple previously known protocols.

- Using real-world measurements, we evaluate the proper functioning of such a mutually assisted protocol in practice.

Finally, in Chapter 6, we conclude our work and give an outlook on potential future research topics. For an improved readability, we give a complete table of symbols for the Chapters 2 to 5 in Appendix A. In addition, to ease the readability of parts with high numbers of uncommon symbols, we present multiple selections of relevant symbols whenever appropriate throughout this thesis.

1.4 Bibliographic Notes

1.4.1 Publications Directly Related to this Thesis

Parts of this thesis have appeared in the following publications.

1. Chapter 2, which describes the fundamental bounds of ND, is mainly based on the following publication:

   ```
   Philipp H. Kindt, Samarjit Chakraborty:
   On Optimal Neighbor Discovery.
   C. of the ACM Special Interest Group on Data Communication (SIGCOMM), 2019
   ```

2. The latency model for PI-based protocols presented in Chapter 3 has appeared in the following journal article:

   ```
   Philipp H. Kindt, Marco Saur, Michael Balszun and Samarjit Chakraborty.
   Neighbor Discovery Latency in BLE-Like Protocols.
   IEEE Transactions on Mobile Computing (TMC),
   vol. 17, no. 3, pp. 617-631, 2018
   ```

   Preliminary versions of this paper are available on arXiv.org under the title “Neighbor discovery latency in BLE-like duty-cycled protocols”, which can be accessed online via `https://arxiv.org/abs/1509.04366`.

3. The optimized parametrizations for PI-based protocols presented in Chapter 4 have appeared in the following manuscript.

   ```
   Philipp H. Kindt, Marco Saur, Samarjit Chakraborty.
   Optimizing BLE-Like Neighbor Discovery
   In submission, 2019
   ```

   Preliminary versions of this paper are available on arXiv.org under the title “Slotless Protocols for Fast and Energy-Efficient Neighbor Discovery”, which can be accessed online via `https://arxiv.org/abs/1605.05614`.

4. The techniques for exploiting mutual assistance that are addressed in Chapter 5 have been proposed in:

   ```
   Philipp H. Kindt, Daniel Yunge, Gerhard Reinerth, Samarjit Chakraborty.
   Griassdi: Mutually Assisted Slotless Neighbor Discovery.
   ACM/IEEE International Conference on Information Processing
   in Sensor Networks (IPSN), 2017
   ```

   ```
   Philipp H. Kindt, Nils Heitmann, Gerhard Reinerth, Samarjit Chakraborty.
   Demo: Understanding Slotless Neighbor Discovery.
   ACM/IEEE International Conference on Information Processing
   in Sensor Networks (IPSN), 2017
   ```

5. Further, parts of Chapter 1 are related to the German Research Foundation (DFG)-funded project titled “Slotless Protocols for Fast and Energy Efficient Neighbor Discovery”, which is currently running under grant number *CH 918/5-1*. Additional parts of Chapter 1 are further based on the following manuscript:

Philipp H. Kindt, Daniel Yunge, Robert Dimer, Samarjit Chakraborty.
Energy Modeling for the Bluetooth Low Energy Protocol.
In submission, 2019

Preliminary versions of this paper are available on arXiv.org under the title "Precise Energy Modeling for the Bluetooth Low Energy Protocol", which can be accessed online via https://arxiv.org/abs/1403.2919.

1.4.2 Other Related Publications

Though not being included in this thesis, the following publications are broadly related to the topic.

Philipp H. Kindt, Samarjit Chakraborty.
Power-aware Reliable Communication for the IoT
IEEE Int. Symposium on On-Line Testing and Robust System Design (IOLTS), 2019 (Invited)

Philipp H. Kindt, Sangyoung Park, Samarjit Chakraborty.
Fast Collision Simulation for Cyclic Wireless Protocols.
arXiv preprint arXiv:1807.01645, 2018

Daniel Yunge, Sangyoung Park, Philipp H. Kindt, Samarjit Chakraborty.
Dynamic Alternation of Huffman Codebooks for Sensor Data Compression.
Embedded Systems Letters (ESL), vol. 9, no. 3, pp. 81-84, 2017

Nils Heitmann, Philipp H. Kindt, Thomas Rosner, Kapil Sikka,
Amit Chirom, Dinesh Kalyanasundaram, Samarjit Chakraborty.
Sound4All: Towards Affordable Large-Scale Hearing Screening.
International Conference on Design & Technology
of Integrated Systems in Nanoscale Era (DTIS), 2017

Daniel Yunge, Sangyoung Park, Philipp H. Kindt, Samarjit Chakraborty.
Dynamic Service Synthesis and Switching for Medical IoT and
Ambient Assisted Living.
IEEE International High-Level Design Validation
and Test Workshop (HLDVT), 2016

Philipp H. Kindt, Daniel Yunge, Andreas Tobola,
Georg Fischer, Samarjit Chakraborty.
Dynamic Service Switching for the Medical Internet of Things.
Annual IEEE International Symposium on Personal,
Indoor and Mobile Radio Communications (PIMRC), 2016

Nils Heitmann, Philipp H. Kindt, Samarjit Chakraborty.
EGON: Portable In-Situ Energy Measurementfor Low-Power Sensor Devices.
IEEE International Symposium on VLSI Design and Test (VDAT), 2016

```
Daniel Yunge, Philipp H. Kindt, Michael Balszun, Samarjit Chakraborty.
Hybrid Apps: Apps for the Internet of Things.
IEEE International Conference on Embedded Software and Systems (ICESS), 2015

Alma Pröbstl, Philipp H. Kindt, Emanuel Regnath, Samarjit Chakraborty.
Smart^2: Smart Charging for Smart Phones.
IEEE International Conference on Embedded and Real-Time
Computing Systems and Applications (RTCSA), 2015

Philipp H. Kindt, Jing Han, Nadja Peters, Samarjit Chakraborty.
ExPerio - Exploiting Periodicity  for Opportunistic
Energy-Efficient Data Transmission.
IEEE International Conference on Computer Communications (INFOCOM), 2015

Philipp H. Kindt, Daniel Yunge, Mathias Gopp, Samarjit Chakraborty.
Adaptive Online Power-Management for Bluetooth Low Energy.
IEEE International Conference on Computer Communications (INFOCOM), 2015
```

2 Fundamental Limits of Neighbor Discovery

2.1 Introduction

As already described in Chapter 1, reducing the energy consumption is one of the main concerns in MANETs. Since all devices in such MANETs run on batteries or rely on intermittently available energy-harvesting sources, the energy spent for communication needs to be as low as possible. Typically, MANET radios are duty-cycled and wake up only for short durations of time for carrying out the necessary communication and then go back to a sleep mode. While such duty-cycled communication schemes are easy to realize when the clocks of all devices are synchronized and their wakeup schedules are known by all participants of the network, asynchronous communication (i.e., communication without synchronized clocks) remains a challenging problem. One of the most important asynchronous procedures is establishing the first contact between different wireless devices, which is referred to as *ND*.

Neighbor Discovery: As already described, ND is used by a device for detecting other devices in range. This could be for clock synchronization and establishing a connection, after which more data can be exchanged in a synchronous fashion. Efficient ND is characterized by achieving the shortest possible discovery latency for a given energy budget. Towards this, a large number of ND protocols have been proposed till date, see [54, 39, 80, 71, 63, 68, 15, 23, 2, 55, 56, 10, 82, 67, 61, 22, 76, 87, 88, 11, 24, 32, 26, 69, 60, 70, 20, 86, 25, 81, 4, 73, 85, 18, 83, 6, 7, 9, 31, 78, 72, 19, 84, 33, 14, 77]. Among these, [63, 68, 15, 23, 2, 55, 56, 10, 82, 67, 61, 22] for example, concern *deterministic* discovery. Here, given the protocol parameters, an upper bound on the discovery latency can be determined. The problem of *pairwise discovery* between two devices is of fundamental importance, since in many scenarios, devices join the network gradually and only a master device and the newly joining one carry out the discovery procedure simultaneously.

Moreover, the process of discovering multiple devices always relies on pairwise ND.

Over the years, successive ND protocols have improved their discovery latencies for given energy budgets. However, despite the significant attention the ND problem has received over the past 15+ years, the fundamental question of what is the theoretically lowest possible discovery latency that any ND protocol could guarantee for a given energy budget still remains unanswered.

Performance of ND Protocols: In the absence of such a bound, the performance evaluations of different ND protocols have often been very subjective. The results of such evaluations relied on the choice of protocols, their parametrizations and the assumed setups. Hence, while a certain protocol might outperform others in such a comparison, it might perform differently if the parametrization or setup is changed. In addition, recall from Chapter 1 that most known protocols, e.g., [15, 2, 67], subdivide time into multiple slots and are hence referred to as *slotted*. Here, performance is quantified in terms of the worst-case number of slots until discovery is guaranteed. Though a certain protocol could perform better than another in terms of the number of slots, such comparisons are heavily dependent on the supported range of slot lengths. As a result, such comparisons in terms of slots and not directly in terms of time are often not meaningful. Moreover, despite slotted protocols having been studied thoroughly in the literature, many protocols that are frequently used in practice, e.g., BLE, do not rely on a slotted paradigm. They schedule reception windows and beacon transmissions with periodic intervals and offer three degrees of freedom that can be configured freely (viz., the periods for reception and transmission, and the length of the reception window). The high practical relevance of such *PI-based* protocols is underpinned by the 4.7 billion BLE-devices that were expected to be sold in 2018 [59]. It is known that the parametrizations for ND in BLE networks proposed by official specifications [42] lead to performances far from the optimum (see Chapter 3 for details). This has raised the interest to fully understand such *slotless* ND procedures. In particular, finding beneficial parametrizations for periodic interval-based protocols has been studied in the literature recently, e.g., in [22, 24]. However, until today, it is neither clear whether the proposed parametrizations are actually optimal, nor how such protocols compare to the slotted ones in terms of performance. In summary, despite the large volume of available literature, it is not possible to meaningfully assess and classify the performance of ND protocols in a purely objective fashion.

This Chapter: In this chapter, we study the fundamental limits of pairwise, deterministic ND. In particular, we establish a relationship between the optimal discovery latency, channel utilization (and hence beacon collision rate) and duty-cycle. No pairwise ND protocol can achieve lower discovery latencies than the ones we establish. The resulting bounds not only give important insights into the design of ND protocols, but will serve as a baseline for more objective performance comparisons. Surprisingly, our analysis shows that some recently proposed protocols actually perform optimally and cover the entire discovery latency/channel utilization/duty-cycle Pareto front. The optimality results of such protocols were not known prior to this work. In particular, the coverage of the Pareto front implies that there is no further potential for improvement.

However, there is still potential to improve the robustness against beacon collisions, which might occur frequently when many devices carry out ND simultaneously.

Principle of ND: In general, a radio can either be in a sleep state, listen to the channel or transmit a beacon. Hence, the basic building blocks of a ND protocol are given by these three operations and any ND protocol can be represented as a unique pattern of them. For a higher power-budget, the number of beacons and/or the number or lengths of reception windows can be increased and a discovery procedure is successful once a beacon overlaps with a reception window on another device. Since the design space of all possible reception and transmission patterns allows for an infinite number of possible configurations, determining the optimal pattern and its performance through any form of exhaustive search or numerical method is not possible. Further, as outlined above, most work on ND has focused on slotted protocols and therefore studied only a small part of the design space. As a result, the problem of assessing the optimal performance of ND has so far remained unsolved.

ND Scenarios: For different scenarios, the ND problem appears in different forms, and we provide bounds on the discovery latency for many of them. First, it is obvious that if two devices E and F both have the same beacon and reception patterns, their discovery properties are *symmetric*. This implies that device E discovers device F with the same worst-case latency for a given duty-cycle as F discovering E. Several publications, e.g., [88, 15], have studied this special case of *symmetric* duty-cycles, for which we present a bound on the discovery latency. If both devices run different patterns (for example, due to different duty-cycles), the discovery properties are *asymmetric*. For the asymmetric case, we provide a bound on the discovery latency when each device is aware of the other device's configuration.

Another important question we answer in this chapter is the partitioning of the duty-cycle, which corresponds to the energy-budget of a device. The duty-cycle of a device is the fraction of time it is active. On the other hand, channel utilization is the fraction of time a device occupies the channel, which is between zero and its duty-cycle. Beacon collision rates are solely determined by the channel utilizations of the devices in range. For the case when the channel-utilization (and hence collision rate) is unconstrained, we derive the ratio between transmission and reception times that minimizes the discovery latency.

In the case of many devices discovering each other, the channel utilization of each device has to be constrained for limiting the collision rate. In this chapter, we therefore not only derive bounds for the discovery latency that any protocol can guarantee for a given duty-cycle, but also for the case where both duty-cycle and the maximum channel-utilization are provided. To the best of our knowledge, no such protocol-agnostic bounds on discovery latency for the ND problem has been derived until now. In particular, we in this chapter make the following contributions.

Technical Contributions: We present the following bounds on the discovery latency of deterministic ND protocols.

1. The lowest discovery latency *any* symmetric and asymmetric pairwise ND protocol can guarantee for a given duty-cycle and hence energy consumption. Recall that in symmetric ND, all devices operate using the same duty-cycle, whereas in asymmetric ND devices use different duty-cycles.
2. A discovery latency bound for the case where the channel utilization is additionally constrained.
3. Bounds for the following three cases where two devices E and F discover each other. (a) Only E needs to discover F, whereas F does not need to discover E. (b) Either E discovers F or F discovers E, but both discovering each other is not possible. (c) Both E and F mutually discover each other.

We further study the relation between our bounds and previously known ones [87, 88, 55, 56], which are all limited to slotted protocols. These bounds are given in terms of a worst-case number of slots until discovery is guaranteed, where the discovery latency also depends on the slot length. However, how small a slot length can be is difficult to answer, while it is known that slot lengths cannot be made arbitrarily small. Therefore, discovery latencies in terms of time have not been derived, which we address in this chapter. Finally, while most previous work has focused on slotted protocols, we show that when channel utilization is unconstrained, only slotless protocols can perform optimally, whereas slotted ones cannot. This result is important because in many IoT scenarios devices join gradually and only a pair of devices participate in ND at any point in time. Here, channel utilization is therefore not of concern. We present an optimal ND solution that exploits this in Chapter 4.

Organization of this Chapter: The rest of this chapter is organized as follows. In Section 2.2, we present related work on discovery latency bounds of ND protocols. Next, in Section 2.3, we provide a formal description of a generic ND procedure. Based on this, in Section 2.4, we derive a list of properties that deterministic ND protocols need to guarantee. Recall that deterministic ND protocols are ones for which bounded discovery latencies can be guaranteed. We derive such latency bounds in Section 2.5. Finally, in Section 2.7, we relate the latency bounds of multiple existing ND protocols to the bounds obtained in this chapter. Throughout this chapter, we make a couple of simplifying assumptions. These assumptions are only for the ease of exposition and are relaxed in Section 2.6.

2.2 Related Work

In this section, we describe existing efforts to determine bounds on the discovery latency that any ND protocol can achieve, and relate them to the work presented in this chapter.

Bounds for Slotted Protocols: As discussed above, the vast majority of ND protocols proposed in the literature follow a slotted paradigm, in which reception and transmission are temporally coupled into slots. A bound on the discovery latency of slotted protocols has been studied in [87, 88]. Here, it has been shown that for guaranteeing discovery within T slots, every device needs to have at least $k = \sqrt{T}$ active slots. Therefore, if e.g., $k = 2$ out of $T = 4$ slots are active, then discovery can be guaranteed within four slots with a duty-cycle of 50%, whereas if $k = 4$ and $T = 16$, discovery can be guaranteed within 16 slots with a duty-cycle of 25%. Determining the schedule of active slots that realizes this bound relies on *cyclic difference sets* [87] (refer to Chapter 1 for details on this). Since only a very limited number of such difference sets are known, slotted protocols utilizing this bound can only be realized for a few duty-cycles that correspond to these known difference sets. Subsequently proposed protocols, such as Disco [15], Searchlight [2] and U-Connect [23] for the same discovery latency require more active slots than defined by this bound. But they are more flexible in terms of duty-cycles they can realize. Other recent work [55, 56] claims to have superseded this bound. By sending an additional beacon outside the slot boundaries in a schedule defined by difference sets, a tighter bound than described in [87, 88] can be reached.

Being on slotted protocols, the bounds in [87, 88, 55, 56] are all given in terms of a worst-case number of slots within which discovery is guaranteed. The corresponding bounds in terms of time depend on the slot length I. The minimum size of such a slot, among other factors, also depends on the hardware, and cannot be made arbitrarily small. Consequently, no bounds on the discovery latency in terms of time for slotted protocols have been known until now. This issue is addressed later in this chapter.

Bounds for PI-based Protocols: The only known bounds on PI-based discovery are described in this thesis. In particular, in Chapter 3, we will present a generic latency model of PI-based protocols. Given a tuple of parameter values (T_a, T_s, d_s), we will present a method to compute the worst-case discovery latency of PI-based protocols configured accordingly. However, since there are infinite numbers of possible parametrizations (T_a, T_s, d_s), and because of the structure of the required computation scheme, it does not answer which parametrizations lead to the lowest latency. Hence the performance limits of PI-based protocols cannot be identified using this model. We will address optimal parametrizations in Chapter 4.

Generic Approaches: Unlike the work described above that was specific to slotted or PI-based protocols, protocol-agnostic bounds were presented in [5, 3]. In particular, they give an asymptotic latency bound in the form of $\Theta(d)$, where d is "the discretized uncertainty period of the clock shift between the two processors" [5]. Hence, this bound depends on the degree of asynchrony between the clocks of a sender and a receiver. First, the asymptotic nature of such a bound is very different from the concrete time bounds that have been pursued within the computer communications community, e.g., [87, 88, 55, 56], and the ones presented by us in this chapter. Second, this community has also focused on bounds that depend on duty-cycle and hence energy budget, which are of direct practical relevance. For these reasons, the bounds from [5, 3] are not comparable to those that have been more commonly pursued, and also to those presented in this chapter.

Bounds for Randomized Protocols: A bound for randomized, non-deterministic ND protocols has been proposed in [69]. However, [69] does not consider energy efficiency and hence, each device operates on a duty-cycle of 100 %. The only factor that limits the discovery latencies in this model are packet collisions. Therefore, this bound is not comparable to the bounds derived in this work.

Related Problems: Related problems have been addressed in the context of other aspects of networking. For example, [11] studies the optimal throughput of packets transmitted by a number of unsynchronized nodes, which can be regarded as a form of ND. However, due to the different nature of these problems and the underlying assumptions, such solutions do not apply to the problems concerned in this thesis.

2.3 Neighbor Discovery Protocols

2.3.1 Definition

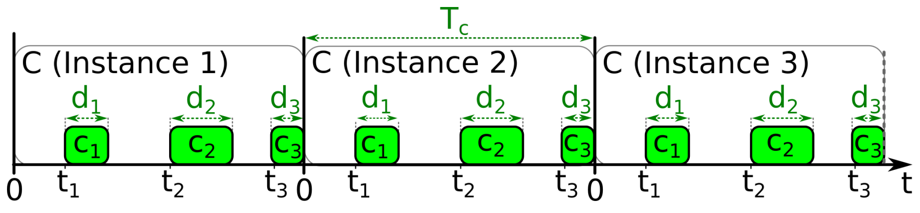
In this section, we formally define the ND procedure and its associated properties.

Definition 2.1 (Reception Window Sequence)**:** Let the time windows during which a device listens to the channel be given by the tuples $c_1 = (t_1, d_1), c_2 = (t_2, d_2), c_3 = (t_3, d_3), ...$, where each *reception window* c_i starts at time t_i and ends d_i time-units later (see Figure 2.1 a)). A *reception window sequence* $C = c_1, c_2, ..., c_n$ could be of finite or infinite length. In this chapter, for simplicity of notation, we refer to such finite length sequences by C and infinite length sequences by C_∞.

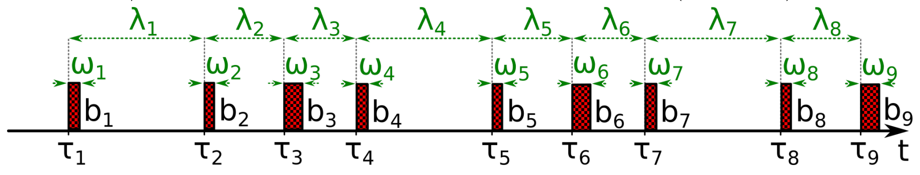
For the simplicity of exposition, throughout this chapter, we always assume that any C_∞ is an infinite concatenation of some finite length sequence C. For such C_∞, we define $n_C = |C|$ (i.e., the number of windows contained in C). Further, we denote the time between the ends of two consecutive instances of C as the *reception period* T_C. It is worth mentioning that all our bounds remain valid also for sequences C_∞ that are not given by concatenating the same C, as we show in Section 2.6.6. We assign a time-axis to every instance of C. For convenience, which will become clear later, the origin of time in a certain instance of C will start at the end of the last reception window of the previous instance, as depicted in Figure 2.1 a). In this figure, C consists of three reception windows (i.e., c_1, c_2, c_3), and three concatenated instances of C are shown. For example, the origin of the time-axis for Instance 2 lies at the end of c_3 in Instance 1.

Definition 2.2 (Beacon Sequence)**:** A sequence of beacons $B = b_1, b_2, ..., b_m$ sent at the time-instances $\tau_1, \tau_2, ..., \tau_m$, as depicted in Figure 2.1 b), is called a *beacon sequence* of length m. The transmission durations of these beacons are given by $\omega_1, \omega_2, ..., \omega_m$. A sequence of infinite length (i.e., $m \to \infty$) is denoted by B_∞.

We denote infinite length beacon sequences B_∞ that are given by concatenations of a finite beacon sequence B as *repetitive beacon sequences.* In such repetitive sequences, $m_B = |B|$ and the time between the endings of two consecutive instances of B is given by T_B. Unlike for reception window sequences, we do not restrict ourselves to repetitive



a): Reception window sequence. Here, $C = (c_1, c_2, c_3)$.


λ1 λ2 λ3 λ4 λ5 λ6 λ7 λ8
ω1 ω2 ω3 ω4 ω5 ω6 ω7 ω8 ω9
b1 b2 b3 b4 b5 b6 b7 b8 b9
τ1 τ2 τ3 τ4 τ5 τ6 τ7 τ8 τ9 t

b): Beacon sequence $B = b_1, b_2, ..., b_9$.

Figure 2.1: Reception window and beacon sequences.

infinite beacon sequences. However, we will prove that all beacon sequences that optimize the relevant metrics of a ND procedure are repetitive when the corresponding reception window sequence is also repetitive.

We indicate an arbitrary shorter sequence B' to be a part of a longer sequence B by using the notation $B' \in B$. For example, in Figure 2.1 b), $B' = b_2, b_3, b_4, b_5, b_6 \in B$. Further, the time between the beginnings of beacon b_i and beacon b_{i+1} is called the *beacon gap* λ_i. It is $\lambda_i = \tau_{i+1} - \tau_i$.

Definition 2.3 (ND Protocol)**:** A tuple of an infinite beacon and reception window sequence (B_∞, C_∞) is called a *ND* protocol.

In this chapter, unless explicitly stated, we assume that C_∞ and B_∞ stem from two different devices E and F. When it is necessary to explicitly specify the device that a sequence is scheduled on, we use the notation $C_{E,\infty}$ or $B_{F,\infty}$, where E and F refer to device E or F respectively. We also apply this notation to reception windows and beacons, e.g., $b_{E,1}$ refers to beacon 1 on device E and $c_{F,1}$ refers to reception window 1 on device F.

The most important properties of a ND protocol are its worst-case latency L, its duty-cycle η, and its channel utilization η_B, as defined next.

Definition 2.4 (Worst-Case Latency)**:** Given two devices E and F, where E runs an infinite beacon sequence and F an infinite reception window sequence, the *worst-case latency* L is the earliest possible time after which an overlap of a beacon from E with a reception window of F is guaranteed, measured from the point in time both devices come into the range of reception.

Definition 2.5 (Duty-Cycle)**:** The *transmission duty-cycle* η_B of a device is the fraction of time it spends for transmission, whereas the *reception duty-cycle* η_C is the fraction of time spent for reception. In general, depending on the configuration of the radio (e.g., transmit power and receiver gain), transmission incurs a different power consumption than

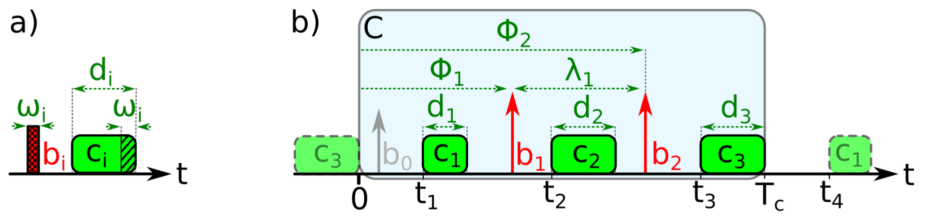


Figure 2.2: a) Beacons starting within the hatched area of window c_i can only fractionally coincide. b) Offset Φ_1 of the first beacon b_1 and Φ_2 of the second beacon b_2 in range.

reception. Therefore, the total *duty-cycle* η is given as a weighted sum $\eta = \eta_C + \alpha\eta_B$, where the weight α is the ratio of transmission and reception powers, i.e., $\alpha = {}^{P_{Tx}}/_{P_{Rx}}$. For a radio running a tuple of sequences (B_∞, C_∞), it is:

$$\eta_B = \lim_{m\to\infty} \frac{\sum_{i=1}^{m-1} \omega_i}{\tau_m - \tau_1}, \quad \eta_C = \lim_{n\to\infty} \frac{\sum_{i=1}^{n-1} d_i}{t_n - t_1}, \quad \eta = \alpha\eta_B + \eta_C \tag{2.1}$$

The transmission duty-cycle η_B is the same as the *channel utilization*. The duty-cycle η directly corresponds to the power consumption of an ideal radio. Non-ideal radios are discussed in Section 2.6.3.

It follows from the above that the duty-cycle of a tuple of sequences B_∞, C_∞, that are concatenations of finite length sequences B and C respectively, can be computed as follows.

$$\eta_B = \frac{\sum_{i=1}^{m_B} \omega_i}{T_B} = \frac{\sum_{i=1}^{m_B} \omega_i}{\sum_{i=1}^{m_B} \lambda_i}, \quad \eta_C = \frac{\sum_{i=1}^{n_C} d_i}{T_C}, \quad \eta = \alpha\eta_B + \eta_C \tag{2.2}$$

2.3.2 Beacon Length

A beacon needs to be transmitted entirely within a reception window of a receiving device for being received successfully. Each beacon has a certain transmission duration ω_i, and if the beacon transmission starts after the last ω_i time-units of a reception window (cf. after the start of the hatched area in Figure 2.2a)), it cannot be received successfully. Nevertheless, for simplicity of exposition, for now we assume that any overlap between a beacon and a reception window leads to a successful discovery. We further assume that all beacons have the same length ω and neglect the contribution of the transmission duration of the first successfully received beacon to the worst-case latency. We study the relaxation of these assumptions in Section 2.6.1 and 2.6.2.

2.3.3 Low Power Listening/Transmit

Some protocols, e.g., U-Connect [23], make use of a technique known as *low power listening/transmit*. Instead of sending short beacons that need to overlap with relatively longer reception phases, they use contiguous transmission phases that are significantly larger than one beacon transmission duration. The reception phases for detecting these beacons can in turn become as short as the hardware supports. Such protocols can be described by interchanging beacons and reception windows in the model presented above.

Thereby, reception window sequences are used to model transmissions, whereas beacon sequences describe the radio's reception pattern.

2.4 Deterministic Beacon Sequences

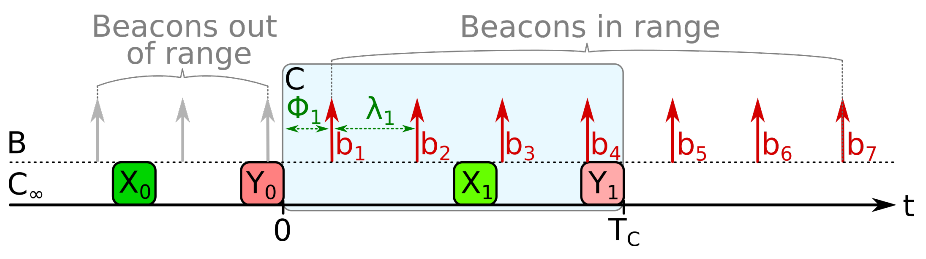
A device F can successfully discover another device E only if E sends a beacon during one of the reception windows of F. We refer to the other direction as E discovering F. In what follows, we first consider F discovering E only, and later generalize it towards mutual discovery.

On device E, let $B' = b_1, b_2, ...$ be a subsequence of B_∞. From here on, we will always assume that b_1 is the first beacon that is in range of a remote device F. This is because any prior beacons of B_∞, when E is not within the range of F, are not relevant for ND. Further, let F run an infinite reception window sequence C_∞. Though B_∞ and hence $B' \in B_\infty$ could be of infinite length, let us think of B' as a fixed-length sequence. This assumption is valid because in the case of a successful discovery, beacons that are sent thereafter are no longer relevant for the discovery procedure. Now recall that the reception windows of C_∞ are formed by concatenations of a finite sequence C and every instance of C has its own time origin, as defined by Definition 2.1 (cf. Figure 2.1 a)). The first beacon b_1 in B' lies within a certain instance of C and has a certain (random) offset Φ_1 from the time origin of this instance of C. This is depicted in Figure 2.2b), which shows an infinite beacon sequence consisting of concatenations of $C = c_1, c_2, c_3$, of which one full instance is depicted. In addition, the figure contains the last reception window c_3 of the preceding instance and the first reception window c_1 of the succeeding one. Further, three beacons b_0, b_1 and b_2 are shown, of which only b_1 and b_2 are in range. Here, B' consists of b_1, b_2 and some later beacons that are not shown in the figure. Beacon b_1 falls into the depicted instance of C and has an offset of Φ_1 time-units from its origin.

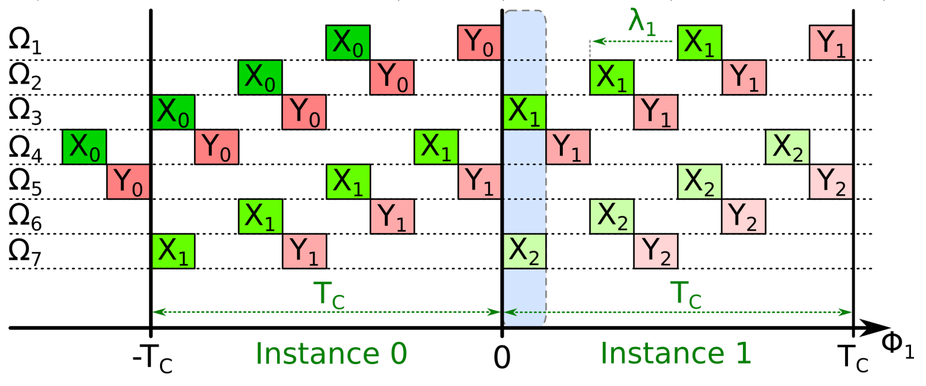
For some valuations of Φ_1, at least one beacon of B' will coincide with a reception window of C_∞. For other valuations of Φ_1, there might be no beacon in B' that coincides with any reception window of C_∞, irrespective of the length of B'. If an overlapping pair of a beacon and a reception window exists for ***all*** possible offsets Φ_1, the tuple (B', C_∞) guarantees discovery within a bounded amount of time and hence realizes deterministic ND. We, in the following, formalize the properties that such a tuple (B', C_∞) needs to fulfill for guaranteeing discovery.

2.4.1 Coverage and Determinism

A tuple (C_∞, B'), along with Φ_1, is depicted in Figure 2.2b). For a given (C_∞, B'), it is obvious that the offset Φ_1, which is a measure of the shift between B' and C_∞, solely determines whether a beacon in B' overlaps with a reception window in C_∞ or not. The time-duration after which such an overlap takes place, and hence the discovery latency, is also determined by Φ_1. For which values of Φ_1 will beacon b_1 fall into one of the reception windows? Clearly, these are given by the set $\Omega_1 = \{[t_1, t_1 + d_1], [t_2, t_2 + d_2], ...\}$ (cf. Figure 2.2b)). In other words, if Φ_1 lies within any interval belonging to Ω_1, then b_1



a): Example sequences $B' = (b_1, ..., b_7)$ and $C_\infty = (X_0, Y_0, X_1, Y_1, ...)$.



b): Coverage map for these sequences drawn over two periods.

Figure 2.3: Coverage maps.

is successfully received. Similarly, if Φ_2 is the offset of b_2, then for Φ_2 belonging to any interval in Ω_1, b_2 will be successfully received (see Figure 2.2b)).

Now, what are the offsets Φ_1 of b_1, such that beacon b_2 is successfully received? These are given by the set $\Omega_2 = \{[t_1 - \lambda_1, t_1 + d_1 - \lambda_1], [t_2 - \lambda_1, t_2 + d_2 - \lambda_1], ..\}$, where λ_1 is the time-distance between the beacons b_1 and b_2, as already defined in Section 2.3 (see Figure 2.2b)). Therefore, Ω_2 is obtained by shifting all elements of Ω_1 by λ_1 time-units to the left. Similarly, $\Omega_3 = \{[t_1 - (\lambda_1 + \lambda_2), t_1 + d_1 - (\lambda_1 + \lambda_2)], [t_2 - (\lambda_1 + \lambda_2), t_2 + d_2 - (\lambda_1 + \lambda_2)], ..\}$. Then Ω_k for $k = 3, 4, 5, ...$ is similarly defined as

$$\Omega_k = \left\{ \left[t_1 - \sum_{i=1}^{k-1} \lambda_i, t_1 + d_1 - \sum_{i=1}^{k-1} \lambda_i \right], \left[t_2 - \sum_{i=1}^{k-1} \lambda_i, t_2 + d_2 - \sum_{i=1}^{k-1} \lambda_i \right], ... \right\}. \qquad (2.3)$$

Now consider a beacon sequence $B' = b_1, ..., b_m$ of length m. If Φ_1 belongs to any interval in $\Omega_1 \cup \Omega_2 \cup ... \cup \Omega_m$, then one beacon from B' will be successfully received. We now extend this result and define a *coverage map*, which can be used to reason about valuations of the initial beacon offset Φ_1 that lead to successful discovery.

Coverage Maps

A coverage map is a formal representation of all offsets Φ_1 for which any beacon in B' overlaps with a reception window in C_∞. It also allows for a graphical representation, from which several properties of the tuple (B', C_∞) can be easily understood.

Recall that C_∞ is a repeated concatenation of a sequence of reception windows C (i.e., $C_\infty = C\ C\ C...$). Now, we need to be able to specify specific instances of C within

C_∞. For this purpose, let us consider a simple example where C has two reception windows X and Y, and C_∞ is therefore given by $C_\infty = X\ Y\ X\ Y...$, and in order to distinguish between different instances of these reception windows, we will denote $C_\infty = X_0\ Y_0\ X_1\ Y_1\ X_2\ Y_2...$. The reception windows X_i and X_{i+1}, as well as Y_i and Y_{i+1}, are T_c time-units apart (see Figure 2.3 a) and also Figure 2.1 a)).

Figure 2.3 a) shows a sequence of beacons $B' = b_1, ...b_7$ from a transmitting device. Below, two reception windows X_0, Y_0 from a receiving device are depicted, together with their periodic repetitions X_1, Y_1, which are T_C time-units later. Again, $b_1 \in B'$ has a certain random offset Φ_1 from the origin of C. Figure 2.3 b) shows the coverage map for the sequences in Figure 2.3 a).

Definition 2.6 (Covered)**:** An offset Φ_1 is *covered*, if at least one beacon in B' overlaps with any reception window in C_∞ for this offset.

Given the parameters of (B', C_∞), the *construction* of a coverage map as in Figure 2.3 b), is straightforward. We believe that the notion of such a coverage map and its use go beyond deriving latency bounds as done in this chapter. It would also be useful for analyzing and optimizing various kinds of different ND protocols, including already known ones.

From coverage maps, we can derive the following properties.

• **Beacon-to-beacon discovery latency l***: For a given offset Φ_1, let $l^*(\Phi_1)$ be the latency measured from the transmission time of the first beacon that is in range, to the first time a beacon is successfully received. In Figure 2.3, $l^*(\Phi_1) = \tau_i - \tau_1 = \sum_{k=1}^{i-1} \lambda_k$, where i is the smallest row number in which Φ_1 is *covered*. For example, for an offset Φ_1 slightly above 0 (i.e., an offset within the highlighted frame in Figure 2.3 b)), the beacon-to-beacon discovery latency will be $l^* = \tau_3 - \tau_1$, since b_3 is the earliest successful beacon for this offset.

• **Determinism**: By ensuring that all possible initial offsets are covered by at least one beacon, we can guarantee that B' is *deterministic* with respect to C_∞ (see next section for a formal definition of determinism).

• **Redundancy**: For certain valuations of Φ_1, one can see in Figure 2.3 b) that a beacon will be received by multiple reception windows. For example, for values of Φ_1 within the shaded frame, beacons b_3 and b_7 will be received by the windows X_1 and X_2, respectively. By integrating over the length of all reception windows, for which such duplicate receptions happen, we can quantify the degree of redundancy of a tuple (B', C_∞).

Determinism

Recall that protocols that can guarantee discovery for every possible initial offset are called *deterministic*. This is formalized below. In particular, we distinguish between a *beacon sequence* B' and a *protocol* (B_∞, C_∞) that can result in such a sequence.

Definition 2.7 (Deterministic ND Protocol)**:** A beacon sequence B' is *deterministic* in conjunction with an infinite reception window sequence C_∞, if all possible initial offsets

Φ_1 are covered by the tuple (B', C_∞). A ND protocol (B_∞, C_∞) is *deterministic*, if for all i, $B'_i = b_i, b_{i+1}, b_{i+2}, ...$ is a deterministic beacon sequence.

Hence, deterministic ND protocols (B_∞, C_∞) always guarantee a bounded discovery latency, no matter when a beacon of B_∞ comes within the range of a receiving device.

Lemma 2.1. *If a beacon sequence B' covers all offsets Φ_1 within $[0, T_C]$, then all possible valuations of Φ_1 are covered.*

Proof. Let us assume that a certain range of offsets $[\Phi_x, \Phi_y]$, where $\Phi_x, \Phi_y \leq T_C$, is covered by a beacon b_i in conjunction with a certain reception window c_j. Since the pattern of reception windows repeats every T_C time-units, any $\Phi_1 \in [\Phi_x + T_C, \Phi_y + T_C]$ will result in b_i being received by the reception window c_{j+n_C}, which is T_C time-units after c_j. □

Definition 2.8 (Redundant Sequences)**:** If any offset Φ_1 within $[0, T_C]$ is covered by more than one beacon, then the tuple (B', C_∞) is *redundant*. Otherwise, (B', C_∞) is *disjoint*, since no intervals in the corresponding coverage map overlap.

For example, in Figure 2.3 b), all offsets Φ_1 are covered and hence the corresponding tuple (B', C_∞) is deterministic. Further, since some offsets, e.g., the ones slightly above offset 0 (marked by the highlighted frame in Figure 2.3 b)) are covered twice, it is also redundant.

Coverage

For a tuple (B', C_∞), certain values of Φ_1 might be covered by multiple beacons, other values by exactly one beacon and yet others by no beacons. The notion of *coverage* quantifies how different values of $\Phi_1 \in [0, T_C]$ are covered. To understand this, recall that Ω_i is a set of intervals. Let us now consider those (full or partial) intervals of Ω_i that lie within $[0, T_C]$. The sum of the lengths of all such intervals for all Ω_i captures a notion of *coverage* that we formalize below.

Definition 2.9 (Coverage)**:** Given a tuple (B', C_∞), let a certain offset $\Phi_1 \in [0, T_C]$ be covered by k beacons, where $k \in \{0, 1, 2, ...\}$. Let us define an auxiliary function $\Lambda^*(\Phi_1) = k$. Then, the *coverage* Λ is defined as

$$\Lambda = \int_0^{T_C} \Lambda^*(\Phi_1) d\Phi_1. \tag{2.4}$$

For example, in Figure 2.3 b), if the lengths of X_i and Y_i are equal to unity and therefore $T_C = 8$, then $\Lambda = 14$. If $\Lambda < T_C$, a tuple (B', C_∞) cannot be deterministic, which implies that for certain values of Φ_1, no bounded discovery latency can be guaranteed. If $\Lambda = T_C$, then (B', C_∞) can either be deterministic and disjoint, or else, it will be redundant and not deterministic. If $\Lambda > T_C$, than (B', C_∞) cannot be disjoint, and may or may not be deterministic.

2.4.2 Minimum Coverage

While Λ quantifies the coverage due to all beacons in B', we now quantify the coverage induced by individual beacons.

Theorem 2.2 (Coverage per Beacon)**.** *Given a tuple (B', C_∞), every beacon $b_i \in B'$ induces a coverage of exactly $\sum_{k=1}^{n_C} d_k$ time-units.*

Proof. The first beacon b_1 in B' will cover exactly those time-units for which b_1 directly coincides with a reception window. The sum of such matching offsets is therefore $\sum_{k=1}^{n_C} d_k$ time-units. Every later beacon b_i will cover the same offsets shifted by the sum of beacon gaps $\sum_{k=1}^{i} \lambda_k$ to the left, which does not impact the amount of offsets covered. Since C_∞ is an infinite concatenation of a finite sequence C, for every covered offset that is shifted out of the considered range $[0, T_C]$, the same amount from a later period is shifted into that range, such that each beacon b_i covers exactly $\sum_{k=1}^{n_C} d_k$ time-units within $[0, T_C]$. □

From the above, we are able to derive a minimum length of B'.

Theorem 2.3 (Beaconing Theorem)**.** *Given a tuple (B', C_∞), the minimum number of beacons M a beacon sequence B' needs to consist of to guarantee deterministic discovery is:*

$$M = \left\lceil \frac{T_C}{\sum_{k=1}^{n_C} d_k} \right\rceil \tag{2.5}$$

Proof. From Theorem 2.2 it follows that every beacon induces a coverage of $\Lambda = \sum_{k=1}^{n_C} d_k$. For deterministic discovery, the coverage Λ has to be at least T_C. Therefore, the number of beacons needed for deterministic ND must be at least $\lceil T_C/\Lambda \rceil$. □

It is worth mentioning that Theorem 2.3 is a necessary, but not sufficient condition for deterministic ND. The positioning of the beacons, along with their number, together determine whether or not a tuple (B', C_∞) is deterministic.

2.5 Fundamental Bounds

In this section, we derive the lower bounds on the worst-case latency that a ND protocol could guarantee in different scenarios (e.g., symmetric or asymmetric discovery). In other words, given constraints like the duty-cycle, such a bound defines the best worst-case latency that any protocol could possibly realize. First, we consider the most simple case in which one device F runs an infinite reception window sequence $C_{F,\infty}$ without beaconing, whereas another device E only runs an infinite beacon sequence $B_{E,\infty}$ without ever listening to the channel. We refer to this as *unidirectional* beaconing.

2.5.1 Bound on Unidirectional Beaconing

The Coverage Bound

Consider a tuple (B', C_∞), where B' consists of M beacons and M is given by Theorem 2.3. Recall Theorem 2.3 and the subsequent discussion. If B' is disjoint and deterministic,

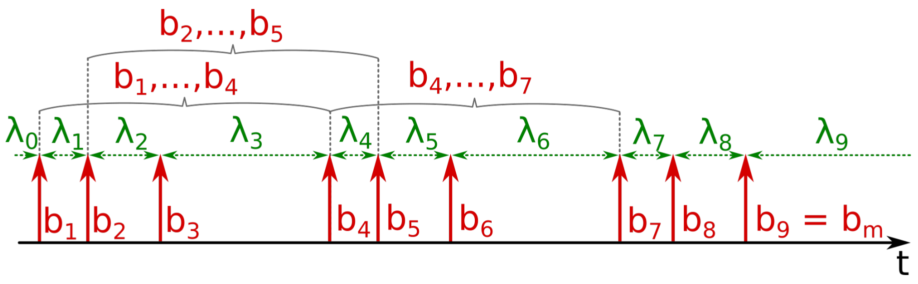


Figure 2.4: Partial sequences of an infinite beacon sequence.

then for every value of Φ_1, there is exactly one beacon in B' that overlaps with a reception window in C_∞. What are the beacon gaps λ_i using which such M beacons need to be spaced for minimizing the discovery latency?

The worst-case beacon-to-beacon discovery latency l^*, measured from the first beacon in range to the earliest successfully received one, is given by the sum of the $M-1$ beacon gaps between these beacons. The first beacon in B' is the first beacon that was sent when the transmitter came within the range of the receiver. To measure the worst-case discovery latency L, time begins when the two devices come in range, which might be earlier than the time the first beacon in B' was sent. How much earlier? At most by the beacon gap that precedes B'. Recall that B' belongs to an infinite sequence B_∞. Hence, the lowest worst-case latency is achieved if the sum of these M beacon gaps is minimized. At the same time, all offsets in $[0, T_C]$ need to be covered exactly once for ensuring determinism.

However, the following arguments rule out such M consecutive beacon gaps to be arbitrarily short. B_∞ has a transmission duty-cycle η_B, defined by the energy budget of the transmitter. Obviously, η_B determines the average beacon gap $\overline{\lambda}$. If the sum of a certain M consecutive beacon gaps becomes smaller than $M \cdot \overline{\lambda}$, then the sum of a different M consecutive beacon gaps within B_∞ needs to exceed $M \cdot \overline{\lambda}$ in order to respect average beacon gap of $\overline{\lambda}$ defined by η_B. Since any beacon in B_∞ could be the first beacon in range, the M beacons with the largest sum of beacon gaps determine the worst-case latency L. Hence, in an optimal B_∞, every sum of M consecutive beacon gaps must be equal to $M \cdot \overline{\lambda}$. It is worth noting that this requirement does not necessarily require equal beacon gaps, because the above property has to hold for a specific value of M given by Theorem 2.3. This is formalized in Lemma 2.5.

Quick Reference of Symbols

L	Worst-case latency
l^*	Worst-case beacon-to-beacon latency
T_C	Period of a reception schedule
d_k	Duration of reception window k
ω	Beacon transmission duration
η_B	Duty-cycle for beaconing
$\overline{\lambda}$	Mean beacon gap length
τ_i	Time at which beacon i is sent
M	Minimum number of beacons in a deterministic beacon sequence

A full table of symbols is given in Appendix A.

To illustrate the above, consider the following example. Figure 2.4 shows a sequence $B' = b_1, ..., b_7$. Here, let the minimum number M of beacons for deterministic ND be equal to 4 and let the partial sequences $(b_1, ..., b_4)$, $(b_2, ..., b_5)$, $(b_4, ..., b_7)$ be deterministic. Consider the sequence $b_1, ..., b_4$. Let us assume that b_4 would be sent somewhat earlier than depicted. Then, by decreasing λ_3, the beacon gap λ_4 would increase accordingly, and though the sequence $b_1, ..., b_4$ would result in a shorter discovery latency for all possible offsets, the sequence $b_4, ..., b_7$ would lead to a larger worst-case latency. The above observations are formalized below.

Theorem 2.4 (Coverage Bound)**.** *The lowest worst-case latency that can be guaranteed by a tuple (B_∞, C_∞) is:*

$$L = \left\lceil \frac{T_C}{\sum_{i=1}^{n_C} d_i} \right\rceil \frac{\omega}{\eta_B} \tag{2.6}$$

Proof. Consider a sequence $B' = b_1, ..., b_m$ with $m >> M$. In B', if any sum of M consecutive beacon gaps is less than $M \cdot \overline{\lambda}$, then the sum of a different M consecutive beacon gaps will exceed $M \cdot \overline{\lambda}$ and will define L. Since this is true for every m, it also holds for B_∞. The mean beacon gap is given by $\overline{\lambda} = {}^{(\tau_m - \tau_1)}/_{(m-1)}$ and the worst-case latency by $L = M \cdot \overline{\lambda}$. Expressing the mean beacon gap by the duty-cycle for transmission (cf. Equation 2.1) and expanding M using Theorem 2.3 leads to Equation 2.6. □

Lemma 2.5 (Repetitive Beacon Sequences)**.** *Given a repetitive C_∞, every B_∞ that guarantees the lowest worst-case latency is repetitive, with a period of $m_B = M$ beacons or $T_B = M \cdot \frac{\omega}{\eta_B}$ time-units.*

Optimal Reception Window Sequences

We know that in an optimal beacon sequence, the sum of every M consecutive beacon gaps is T_B. The corresponding reception window sequence must be such that all offsets in $[0, T_C]$ are covered by such a beacon sequence. While there can be multiple such C_∞ for a given B_∞, the ones that are optimal must fulfill the following property.

Theorem 2.6 (Overlap Theorem)**.** *For a given duty-cycle, every C_∞ that minimizes the worst-case latency is given by the following equation.*

$$T_C = k \cdot \sum_{i=1}^{n_C} d_i, \quad k \in \mathbb{N} \tag{2.7}$$

Proof. Let us assume that the length of T_C is equal to $k \cdot \sum_{i=1}^{n_C} d_i - \Delta$, where k is an integer and $\Delta \in [0, \sum_{i=1}^{n_C} d_i)$. Theorem 2.4 implies the same worst-case latency for all values of Δ, since the ceiling function in Equation 2.6 does not change L. With $T_C = k \cdot \sum_{i=1}^{n_C} d_i - \Delta$, the reception duty-cycle is given by (cf. Equation 2.2):

$$\eta_C = \frac{\sum_{i=1}^{n_C} d_i}{k \cdot \sum_{i=1}^{n_C} d_i - \Delta} \tag{2.8}$$

From Equation 2.8 follows that the duty-cycle is minimized when $\Delta = 0$, and hence $T_C = k \cdot \sum_{i=1}^{n_C} d_i$. □

The intuition behind Theorem 2.6 is that if Equation 2.8 is not satisfied, then T_C can be increased and therefore, the reception duty-cycle η_C can be reduced without requiring any additional beacons to guarantee discovery with the same L. In other words, the coverage intrinsically induced if Equation 2.8 is not satisfied exceeds what is needed for determinism. By combining Theorem 2.4 and 2.6, we can derive a bound for unidirectional beaconing.

Theorem 2.7 (Fundamental Bound for Unidirectional Beaconing)**.** *Given a device E that runs an infinite beacon sequence $B_{E,\infty}$ with a duty-cycle of $\eta_{B,E}$ and a device F that runs an infinite reception window sequence $C_{F,\infty}$ with a duty-cycle of $\eta_{C,F}$, the minimum worst-case latency that can be guaranteed for F discovering E is as follows.*

$$L = \left\lceil \frac{1}{\eta_{C,F}} \right\rceil \frac{\omega}{\eta_B}, \tag{2.9}$$

Clearly, optimal values of $\eta_{C,F}$ are of the form $1/k$, $k \in \mathbb{N}$ and other values of $\eta_{C,F}$ do not lead to an improved L compared to them.

Proof. By combining $T_C = k \cdot \sum_{k=1}^{n} d_k$ from Theorem 2.6 and Equation 2.1, we can write Equation 2.6 as follows.

$$L = \frac{T_C}{\sum_{i=1}^{n_C} d_i} \cdot \frac{\omega}{\eta_B} = \frac{\omega}{\eta_B \cdot \eta_C} \tag{2.10}$$

This holds true for $\eta_{C,F}$ in the form of $1/k$, $k \in \mathbb{N}$. The proof for other duty-cycles follows from the above discussion. □

2.5.2 Symmetric ND Protocols

In this section, we extend Theorem 2.7 towards bidirectional (i.e., device E discovers device F and vice-versa), symmetric (i.e., both devices E and F use the same duty-cycle η) ND. For achieving bidirectional discovery, every device runs both a beacon and a reception window sequence, and we assume that B_∞ and C_∞ can be designed such that both sequences on the same device never overlap with each other. We relax this assumption in Section 2.6.5.

Bi-Directional Discovery

We can achieve bidirectional discovery by running the optimal sequences B_∞ and C_∞ we have identified for unidirectional beaconing on both devices simultaneously. The latency of each partial discovery procedure (viz., the discovery of E by F and of F by E) is bounded by Theorem 2.7. As a result, the worst-case latency for both partial discoveries being successful will also be bounded by Theorem 2.7. Since both devices transmit and receive, we can optimize the share between η_B and η_C, which leads to the following bound.

Theorem 2.8 (Symmetric Bound for Bi-Directional ND Protocols)**.** *For a given duty-cycle η, no bi-directional symmetric ND protocol (i.e. every device runs the same tuple*

(B_∞, C_∞)) can guarantee a lower worst-case latency than the following.

$$L = \min\left(\underbrace{\left\lceil \frac{2}{\eta} \right\rceil^2 \cdot \frac{\omega\alpha}{\eta \left\lceil \frac{2}{\eta} \right\rceil - 1}}_{\mathbb{A}}, \underbrace{\left\lfloor \frac{2}{\eta} \right\rfloor^2 \cdot \frac{\omega\alpha}{\eta \left\lfloor \frac{2}{\eta} \right\rfloor - 1}}_{\mathbb{B}} \right) \tag{2.11}$$

Proof. Because of Theorem 2.6, optimal reception duty-cycles are given by $1/\eta_C = k, k = 1, 2, 3, ...$ By inserting $\eta = \alpha\eta_B + \eta_C$ (cf. Definition 2.5) into Equation 2.11 and setting $1/\eta_C = k$, we obtain

$$L = \frac{k^2\omega\alpha}{k\eta - 1}, k \in \mathbb{N} \tag{2.12}$$

We now have to find the value of k that minimizes L. Let us for now allow non-integer values of k in Equation 2.12. By forming the first and second derivative of Equation 2.12 by k, one can show that a local minimum of L exists for $k = 2/\eta$, which is a non-integer number for most values of η. By analyzing dL/dk, we can further show that Equation 2.12 is monotonically decreasing for values of $k < 2/\eta$ and monotonically increasing for values of $k > 2/\eta$. Hence, the only integer values of k that potentially minimize L are $\lceil 2/\eta \rceil$ and $\lfloor 2/\eta \rfloor$. Inserting $k = \lceil 2/\eta \rceil$ or $k = \lfloor 2/\eta \rfloor$ into Equation 2.10 and taking the minimum latency among both possibilities leads to Equation 2.11. □

In fact, Theorem 2.8 also holds true for unidirectional beaconing, if the joint duty-cycle $\eta = \alpha \cdot \eta_{B,E} + \eta_{C,F}$ of two devices E and F is to be optimized. Further, one can easily see that for small values of η, the ceiling function in Equation 2.11 only marginally affects the value of L, which can therefore be approximated by

$$L = \frac{4\alpha\omega}{\eta^2}. \tag{2.13}$$

Collision-Constrained Discovery

For achieving the bound given by Theorem 2.8, we have assumed that the beacons of multiple devices never collide. This assumption is reasonable for a pair of radios, in which collisions only rarely occur. However, as soon as more than two radios are carrying out the ND procedure simultaneously, collisions become inevitable and some of the discovery attempts fail. As a result, some devices might discover each other after the theoretical worst-case latency has passed, or, depending on the protocol design, might not discover each other at all. Therefore, it is often required to limit the channel utilization and hence collision rate, which leads to an increased worst-case latency bound.

In protocols with disjoint sequences (i.e., every Φ_1 is covered exactly once), every collision will lead to a failure of discovering within L. The collision probability is solely determined by the channel utilization η_B. We in this section study the worst-case latency that can be achieved if both η and η_B (and hence the collision probability) are given. We in addition discuss possibilities to reduce the number of failed discoveries for a given collision probability in Section 2.8.1.

Consider a number of S senders, of which each occupies the channel by a time-fraction of η_B. The first beacon of an additional sender that starts transmitting (or comes into range) at any random point in time will face a collision probability of (cf. [1]):

$$P_c = 1 - e^{-2(S-1)\cdot\eta_B} \tag{2.14}$$

Once a beacon has collided, the repetitiveness of infinite beacon sequences (cf. Lemma 2.5) implies that the fraction of later beacons colliding with this device is predefined. Nevertheless, since all offsets between the two sequences occur with the same probability, the collision probability of every individual beacon is given by Equation 2.14. When constraining the channel utilization to a maximum value $\eta_{B,m}$ that must never be exceeded, the following latency bound applies.

Theorem 2.9 (Bound for Symmetric ND with Constrained Channel Utilization)**.** *For a given upper bound on the channel utilization $\eta_{B,m}$, no symmetric ND protocol can guarantee a lower worst-case latency than the following.*

$$L = \begin{cases} \min(\mathbb{A}, \mathbb{B}), & \text{if } \eta \leq \eta_{C,o} + \alpha\eta_{B,m} \\ \left\lceil \frac{1}{\eta - \alpha\eta_{B,m}} \right\rceil \cdot \frac{\omega}{\eta_{B,m}}, & \text{if } \eta > \eta_{C,o} + \alpha\eta_{B,m} \end{cases} \tag{2.15}$$

Here, $\mathbb{A}$ and $\mathbb{B}$ are given by Equation 2.11 and $\eta_{C,o} = 1/\lceil 2/\eta \rceil$, if $\mathbb{A} \leq \mathbb{B}$, and $1/\lfloor 2/\eta \rfloor$, otherwise.

Proof. Given η, if the channel utilization that results from choosing the optimal value of η_C (see proof of Theorem 2.8) does not exceed $\eta_{B,m}$, the bound given by Equation 2.11 remains unchanged. Otherwise, the bound is obtained from Equation 2.9 by eliminating η_C using $\eta = \alpha\eta_{B,m} + \eta_C$ (cf. Definition 2.5). □

Even when both devices E and F transmit as well as receive, it is possible to design *unidirectional* protocols in which only one of the two devices, E or F, can discover the other. Here, the beacons on both devices contribute to a joint notion of coverage, leading to a reduced latency bound compared to the case where both devices can discover each other mutually. A bound for this possibility is described next.

2.5.3 Mutual Exclusive Unidirectional Discovery

In Section 2.5.1, we have studied unidirectional discovery in the sense that one device F could discover E without E discovering F. However, it is also possible to design the tuple (B_∞, C_∞) on each device such that either device E or F can directly discover its opposite, which we study in this section.

This form of unidirectional discovery is realized using beacon sequences $B \in B_\infty$, in which the beacons on one device are sent with a fixed temporal relation to the reception windows on the same device. For example, let beacon $b_{F,1}$ on device F be sent by ζ time-units after reception window $X_{F,1}$, as depicted in Figure 2.5. Further, let such a relation exist on both devices and in every period T_C of the reception window sequence. As

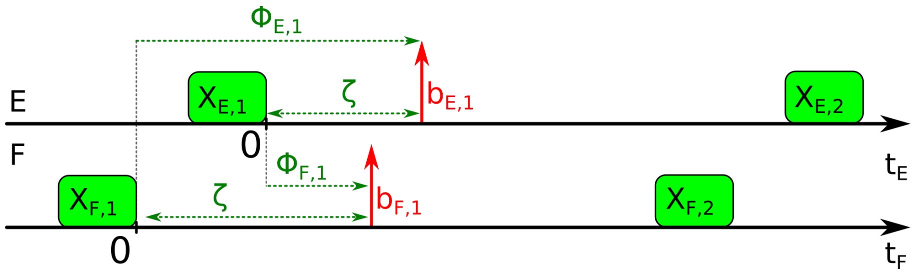


Figure 2.5: Correlated offsets $\Phi_{F,1}$ and $\Phi_{E,1}$ in the sequences of two devices E and F.

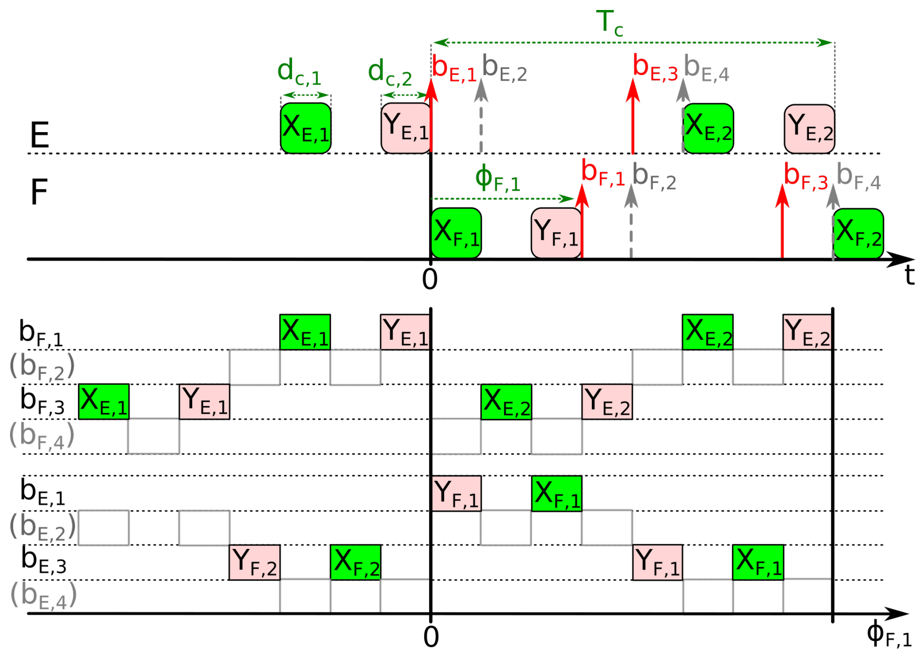


Figure 2.6: Quadruple of sequences $(\mathbf{C_{E,\infty}}, \mathbf{B_{E,\infty}}, \mathbf{C_{F,\infty}}, \mathbf{B_{F,\infty}})$ that exploits temporal correlations.

previously explained $b_{F,1}$ has a random offset of $\Phi_{F,1}$ time-units from the coordinate origin of device E. The temporal correlation between B_∞ and C_∞ on every device implies that the offset $\Phi_{E,1}$ beacon $b_{E,1}$ has from the coordinate origin of device F is fully determined by $\Phi_{F,1}$ (cf. Figure 2.5). It is:

$$\Phi_{E,1} = \zeta + (\zeta - \Phi_{F,1}) = 2 \cdot \zeta - \Phi_{F,1} \tag{2.16}$$

By exploiting this temporal relation, a quadruple of sequences $(C_{F,\infty}, B_{E,\infty}, C_{E,\infty}, B_{F,\infty})$ can guarantee deterministic one-way discovery, even if the pair $(C_{E,\infty}, B_{F,\infty})$ only covers half of the offsets $\Phi_{F,1} \in [0, T_C]$, by having the pair $(C_{F,\infty}, B_{E,\infty})$ covering the remaining ones. Thereby, the number of beacons that need to be sent per device for guaranteeing one-way discovery can be halved. The upper part of Figure 2.6 depicts the beacons (arrows) and reception windows (rectangles) of two devices E and F. On each device, the reception windows and beacons have a fixed temporal relation, whereas beacon $b_{F,1}$ has a random offset $\Phi_{F,1}$ from the coordinate origin of device E. Dashed arrows depict beacons that would need to be sent if every device would have to cover all offsets in the entire period T_C on its own. When exploiting temporal correlations between B_∞ and C_∞ on the same device, these beacons can be omitted without increasing the one-way worst-case latency. The lower part of Figure 2.6 depicts the coverage map of the beacons $b_{F,1}, ..., b_{F,4}$ of device F and $b_{E,1}, ..., b_{E,4}$ of device E. This coverage map represents all offsets $\Phi_{F,1}$, for which either a beacon from device F overlaps with a reception window of device E or a beacon from device E overlaps with a reception window of device F. Covered offsets of omitted beacons have been left white. As can be seen, every possible initial offset $\Phi_{F,1}$ is covered by either a beacon of $B_{F,\infty}$ falling into a reception window of $C_{E,\infty}$, or a beacon of $B_{E,\infty}$ falling into a reception window of $C_{F,\infty}$, and hence the number of beacons per device is halved compared to direct bi-directional discovery. This leads to the following latency bound, which is lower than the one given by Theorem 2.8. Since there are no further possibilities to improve pairwise discovery, this is also the tightest fundamental bound for all pairwise deterministic ND protocols.

Theorem 2.10. *The lowest worst-case latency a pair of devices can guarantee for mutual exclusive one-way discovery (i.e., either of both devices can discover its opposite one) is given by:*

$$L = \min\left(\left\lceil \frac{1}{\eta} \right\rceil^2 \cdot \frac{\omega\alpha}{\eta \cdot \lceil 1/\eta \rceil - 1/2}, \left\lfloor \frac{1}{\eta} \right\rfloor^2 \cdot \frac{\omega\alpha}{\eta \cdot \lfloor 1/\eta \rfloor - 1/2} \right) \tag{2.17}$$

Proof. For a given set of offsets Ω_F covered by $B_F \in B_{F,\infty}$ on device E, Equation 2.16 defines a set of offsets Ω_E that are automatically covered by $B_E \in B_{E,\infty}$ on device F, and vice-versa. If Ω_F and Ω_E are disjoint, the amount of offsets contained in $\Omega_F \cup \Omega_E$ must sum up to T_C time-units for guaranteeing one-way discovery. Hence, the beacon sequence on every device needs to cover only $1/2 \cdot T_C$ time-units to guarantee one-way determinism, and Equation 2.6 becomes:

$$L = \left\lceil \frac{T_C}{2 \cdot \sum_{k=1}^{n_C} d_k} \right\rceil \frac{\omega}{\eta_B} \tag{2.18}$$

The rest of this proof is identical to the one for direct symmetric discovery (cf. Theorem 2.8). □

Theorem 2.10 is valid for one-way discovery (i.e., device E discovers device F **or** vice-versa). An indirect reverse discovery can be realized as follows. Each device transmits its next point in time at which it listens to the channel along with its beacons. The receiving device then schedules an additional beacon at the received point in time. This technique is called *mutual assistance*, and is actually a form of synchronous connectivity. Here, the latency for two-way discovery will be increased by the maximum temporal distance between any beacon and its succeeding reception window on the same device. An upper bound for this penalty for two-way discovery is T_C time units, which can be reduced significantly in sequences with more than one reception window per period T_C.

2.5.4 Asymmetric Discovery

So far, we have assumed that two devices E and F have the same duty-cycle, i.e., $\eta_E = \eta_F = \eta$. Next, we study the latencies of *asymmetric* protocols with $\eta_E \neq \eta_F$. We thereby assume that each device knows the duty-cycle of and hence the sequences on its opposite device. This is relevant e.g., when connecting a gadget with limited power supply to a smartphone using BLE. Here, different sequences on both devices that account for the different power budgets can be determined in the specification documents of the service offered by the gadget. The case of every device being allowed to choose its duty-cycle autonomously during runtime is also relevant. The possible degradation of the optimal performance for this case needs to be studied in further work. Whereas all our previously presented bounds are actually reachable by practical protocols, the bound we present for asymmetric ND can only be reached for tuples of duty-cycles (η_E, η_F), for which $2/\eta_F$ and $2/\eta_E$ are integers. For other duty-cycles, the achievable performance will lie slightly below.

Theorem 2.11 (Bound for Asymmetric ND)**.** *Consider two devices E and F with duty-cycles η_E and η_F, where $\frac{2}{\eta_F}$ and $\frac{2}{\eta_E}$ are integers. The lowest worst-case latency for two-way discovery is as follows.*

$$L = \frac{4\alpha\omega}{\eta_E \eta_F} \tag{2.19}$$

Proof. According to Theorem 2.7, if $1/\eta_{C,E}$ and $1/\eta_{C,F}$ are integers, the lowest worst-case one-way discovery latency L_F for device F discovering device E and the latency L_E for the reverse direction are as follows.

$$L_F = \frac{\omega}{\eta_{C,F} \cdot \eta_{B,E}}, \quad L_E = \frac{\omega}{\eta_{C,E} \cdot \eta_{B,F}} \tag{2.20}$$

The global worst-case latency for two-way discovery is given by $L = max(L_E, L_F)$. Because of this, every optimal asymmetric ND protocol must fulfill $L_F = L_E$, since in cases of e.g., $L_F > L_E$, one could decrease the reception duty-cycle $\eta_{C,F}$ of device F and still achieve the same two-way discovery latency L. From $L_E = L_F$ and Equation 2.20 follows that $\eta_{C,F}/\eta_{C,E} = \eta_{B,F}/\eta_{B,E} = const = \mu$. By substituting $\eta_{B,E}$ by $\eta_{B,F}/\mu$ in L_F (cf. Equation 2.20) and by substituting $\eta_{C,F} = \eta_F - \alpha\eta_{B,F}$, we obtain:

$$L_F = \frac{\omega\mu}{(\eta_F - \alpha\eta_{B,F})\eta_{B,F}} \tag{2.21}$$

By differentiating L_F by $\eta_{B,F}$, we can show that L_F is minimal for $\eta_{B,F} = \eta_F/2\alpha$ and hence $\eta_{C,F} = \eta_F/2$. Similarly, L_E has a local minimum at $\eta_{B,E} = \eta_E/2\alpha$. We note that if $2/\eta_E$ and $2/\eta_F$ are integers, also $1/\eta_{C,E}$ and $1/\eta_{C,F}$ are integers. When re-substituting μ by $\eta_{B,F}/\eta_{B,E}$ and replacing $\eta_{B,F}$ and $\eta_{B,E}$ by their optimal values, we obtain Equation 2.20. □

2.6 Relaxation of Assumptions

In Section 2.4, for the sake of ease of presentation, we have made multiple simplifying assumptions. In this Section, we relax all assumptions that have an impact on the discovery latency, study how the fundamental bounds are impacted by this and numerically evaluate the difference between the ideal and real bounds. In particular, we consider the bound for unidirectional beaconing from Theorem 2.7. We thereby consider only optimal reception duty-cycles η_C, since other values of η_C do not lead to any improvement of L (see Theorem 2.7).

2.6.1 Successful Reception of All Beacons

Throughout this chapter, we have assumed that also beacons that only partially overlap with a reception window are received successfully. To account for the fact that beacons cannot be received if their transmissions start within the last ω time-units of each reception window (since they must entirely overlap with the window), we have to artificially shorten the actual length of each reception window d_k by one beacon transmission duration ω when computing discovery latencies, while still accounting for the full length of each reception window in computations of the duty-cycle. As a result, the coverage per beacon Λ in Equation 2.6 from Theorem 2.4 needs to be reduced by one beacon transmission duration ω for each reception window, such that a modified bound can be given as follows.

$$L = \left\lceil \frac{T_C}{\sum_{k=1}^{n_C}(d_k - \omega)} \right\rceil \frac{\omega}{\eta_B} \tag{2.22}$$

Clearly, this increases the worst-case latency that can be achieved for a given reception duty-cycle $\eta_C = \sum_{k=1}^{n_C} d_k / T_C$. From this and Equation 2.22 follows that for a given reception duty-cycle η_C, the increase of L becomes larger for higher numbers of reception windows n_C per period T_C. Hence, the tightest bound can be achieved for $n_C = 1$, for which the term $\sum_{k=1}^{n_C} d_k$ becomes d_1. Using this and by restricting η_C to optimal values and hence setting $T_C = k \cdot (d_1 - \omega)$ (cf. Theorem 2.6), we can write Equation 2.22 as:

$$L = \frac{T_C \omega}{T_C \eta_B \eta_C - \eta_B \omega} \tag{2.23}$$

By examining the first derivative of Equation 2.23, one can show that L becomes smaller for growing values of T_C. However, L cannot become arbitrarily large. If $n_C = 1$ (i.e., one window per reception period), T_C cannot exceed L time-units because of the following reason. Consider a beacon that is sent at the very beginning of a reception window of another device. Let us assume that two devices come into range infinitesimally after this beacon has been sent. Since optimal beacon sequences do not contain more than one beacon within L time-units that overlap the same reception window, the next successful beacon will overlap with the subsequent reception window, which begins T_C time-units later. In other words, T_C must not exceed L time-units, since L would otherwise scale with T_C. For $n_C > 1$, it is $T_C \leq n_C L$. Hence, we can to substitute T_C by L in Equation 2.23. Solving this equation by L leads to the following bound:

$$L = \frac{\omega + \eta_B \omega}{\eta_B \eta_C} \tag{2.24}$$

One can show that this bound is independent of the number of reception windows n_C per reception period.

2.6.2 Neglecting the First Successful Beacon

Throughout this chapter, we have neglected the transmission duration of the first successfully received beacon. We can account for this by adding ω time-units to Equation 2.22. By forming the first and second derivative, we can show that the optimal share between transmission and reception is not influenced by this. When accounting for this beacon, all our presented bounds become by ω time-units longer (e.g., Theorem 2.8 becomes $L = {}^{4\alpha\omega}/_{\eta^2} + \omega$). Besides from this, there are no changes, since finding the optimal beaconing duty-cycle η_B is the only step that is potentially sensitive on adding ω to L.

2.6.3 Radio Overheads

Throughout this chapter, we have assumed that the radios do not require any energy to switch from sleep mode to transmission or reception, and vice-versa. We now assume an overhead d_{oTx} to switch the radio from the sleep mode to transmission and back, and an overhead d_{oRx} to switch from the sleep mode to reception and back. These overheads can be regarded as effective durations of additional active time, i.e., as the actual durations that are needed to switch the radio's mode of operation, weighted by the quotient of the average power consumption during the switching phase over the power consumption for reception. For the sake of simplicity of exposition, we also assume the same overheads for switching directly between reception and transmission, without going to a sleep mode in between.

When accounting for these offsets, one can derive from Equation 2.2 that the duty-cycle for transmission η_B and for reception η_C become:

$$\eta_B = \sum_{i=1}^{m_B} \frac{\omega_i + d_{oTx}}{\lambda_i}, \quad \eta_C = \frac{\sum_{k=1}^{n_C}(d_k + d_{oRx})}{T_C} \tag{2.25}$$

With this, Equation 2.9, from which all other bounds are derived, becomes:

$$L = \left\lceil \frac{1}{\eta_C - \frac{n_C d_{oRx}}{T_C}} \right\rceil \cdot \frac{\omega + d_{oTx}}{\eta_B} \tag{2.26}$$

We only consider optimal values of η_C, which are given by $\eta_C = {}^1/_k + {}^{n_C d_{oRx}}/_{T_C}$. Large values of T_C minimize L, and hence, as explained in Section 2.6.1, we set $T_C = n_C \cdot L$. This leads to the following bound.

$$L = \frac{d_{oTx} + \omega + \eta_B \cdot d_{oRx}}{\eta_B \eta_C} \tag{2.27}$$

2.6.4 Evaluation

In this section, we numerically evaluate the impact of the simplifying assumptions described so far on the latency bound. After that, we will discuss the implications of additional assumptions, which do not have an impact on the latency bounds.

We assume a transmission duration of 32 µs, which corresponds to a 4-byte beacon on a 1 MBit/s - radio used for e.g, BLE. We consider a range of duty-cycles η_B of the sender and η_C of the receiver between 0.055 % and 5.55 %. This range of duty-cycles leads to a practically relevant range of discovery latencies from 0.1 s to 100 s for optimal protocols on ideal hardware platforms (i.e., for Equation 2.9). We only consider optimal values of η_C. We further assume $\alpha = 1$.

Let L_i denote the ideal latency bound (viz., Equation 2.9) and L_r the latency bound with relaxed assumptions. As can be seen from Figure 2.7, in the considered range of duty-cycles, the relative deviation $^{(L_r - L_i)}/_{L_i}$ ranges between nearly 0 % to 6 %.

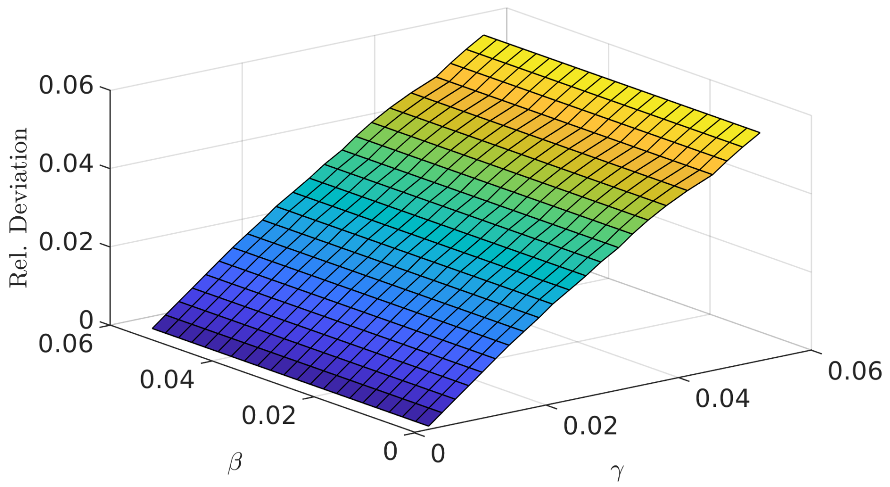


Figure 2.7: Relative difference between real and ideal bound on radios without switching overheads.

While Figure 2.7 provides a platform-independent comparison for any ideal 1 MBit/s radio, what performance can be achieved on existing hardware platforms? For a Nordic nRF51822 SOC [53], the switching overheads are approximately given by $d_{oRx} = d_{oTx} =$ 140 µs. Within the considered range of duty-cycles, the relative deviation to the ideal bound ranges between 438 % and 467 %.

2.6.5 Implications of Same Sequences on Both Devices

Throughout this chapter, we have assumed that C_∞ does not impose any constraints on scheduling the beacons in B_∞ on the same device. In this section, we study the relaxation of this assumption.

Symmetric Sequences

We first study the case in which both devices E and F run the same tuple of sequences (B_∞, C_∞). Here, B_∞ is designed such that a beacon overlap with C_∞ is guaranteed for all initial offsets. Hence, not only an overlap of a beacon of B_F with $C_{E,\infty}$ is guaranteed, but also an overlap of a beacon of B_E with $C_{E,\infty}$. Such an overlap implies that the affected reception window needs to be interrupted for a certain amount of time.

For an ideal radio (i.e., a radio that does not require any time to switch from reception to transmission and vice-versa, see Section 2.6.3), this amount of time is identical to one beacon transmission duration ω. A beacon sent by another device within this period of time would collide and therefore would not be received successfully, even if the radio was able to receive and transmit simultaneously.

However, a real-world radio needs a certain amount of time d_{oTxRx} to switch from transmission to reception and an overhead d_{oRxTx} to switch from reception to transmission, during which no communication can be carried out. We in the following analyze the impact of this. Towards this, we next compute the time-fraction of all reception windows in C_∞, during which the radio is unable to receive.

Since an optimal tuple of sequences (C_∞, B') is designed such that every initial offset is covered exactly once, exactly one beacon of B' will overlap with a reception window for every possible initial offset. For every such overlap, the radio is unable to receive incoming beacons for $d_{oTxRx} + d_{oRxTx} + \omega$ time-units within the affected reception windows.

In a tuple B_∞, C_∞, how frequent do such overlaps occur and which fraction of the total reception time is "blocked" by them? In optimal protocols, exactly one beacon overlaps with a reception window per worst-case latency L (cf. Section 2.5.1). From Theorem 2.10 follows that for optimal values of η_C, $L = T_C \cdot {}^{1}/_{(\sum_{i=1}^{n_C} c_i)} \cdot {}^{\omega}/_{\eta_B}$, and hence L is always divisible by T_C. In every instance of T_C, there are $\sum_{i=1}^{n_C} d_i$ time-units during which the radio is scanning, and therefore, the radio spends ${}^{L}/_{T_c} \cdot \sum_{i=1}^{n_C} d_i = {}^{\omega}/_{\eta_B}$ time-units per worst-case latency L for scanning. The probability of failed discoveries is identical to the fraction of "blocked" time per L, which leads to the following equation.

$$P_{fail} = \frac{\eta_B}{\omega} \cdot (d_{oTxRx} + d_{oRxTx} + \omega) \tag{2.28}$$

In this equation, we assume that the amount of time during which the radio is "blocked" per beacon that overlaps with a reception window of the same device is always identical to $d_{oTxRx} + d_{oRxTx} + \omega$ time-units. We in the following prove this assumption.

Recall from Section 2.4.1 that every beacon of a deterministic sequence B', in conjunction with a reception window from C_∞ of a remote device, leads to a certain contiguous range of covered offsets, which we in the following call a *coverage image*. If the initial offset Φ_1 lies within one of these coverage images, B' is received successfully. Figure 2.8 exemplifies a coverage map of a non-redundant and deterministic (and hence potentially optimal) ND protocol. Here, $C \in C_\infty$ consists of only one reception window and hence, there is one coverage image per beacon. Recall that if a remote device sends a beacon during the last ω time-units of every scan window, it is not received successfully (cf. Section 2.6.1). We can therefore subdivide every coverage image of an optimal protocol into the following three parts $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ (cf. Figure 2.8).

- Part $\mathcal{C}$ has a length of ω time-units, and a beacon of the remote device that falls into this part will not be received successfully. Therefore, such Parts $\mathcal{C}$ do not contribute to the overall coverage.

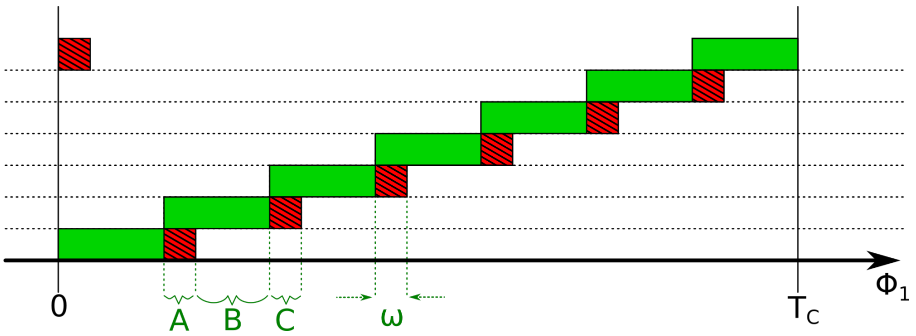


Figure 2.8: Coverage map of a deterministic beacon sequence B'_F in conjunction with a certain $C_{E,\infty}$. The offsets covered by any reception window are composed by a Part $\mathcal{A}$ that overlaps with the last ω time-units of another reception window, a Part $\mathcal{B}$ that is disjoint and a Part $\mathcal{C}$, during which an incoming beacon is not successfully received.

- To nevertheless ensure discovery if a beacon falls into such a Part $\mathcal{C}$ of a coverage image, each Part $\mathcal{C}$ is also covered by the Part $\mathcal{A}$ of another coverage image, which also has a length of ω time-units.
- The remaining part B is disjoint, i.e., no part of any other coverage image overlaps with it.

On a device E, we know that exactly one beacon of $B_{E,\infty}$ will overlap with at least one reception window of $C_{E,\infty}$ per L, which effectively interrupts or shortens the affected scan window. Such an overlap could happen in one of the following three ways.

1. The overlapping beacon falls into Part $\mathcal{B}$, such that a contiguous duration of $d_{oTxRx} + d_{oRxTx} + \omega$ time-units is blocked (e.g., it falls into the center of Part $\mathcal{B}$).
2. The overlapping beacon falls into the beginning (e.g., Part $\mathcal{A}$) of the scan window. Therefore, the "blocked" amount of time would also overlap with the neighboring Part $\mathcal{B}$ (cf. Figure 2.8). Hence, the amount of occupied scanning time is equal to is $d_{oTxRx} + d_{oRxTx} + \omega$ also for this situation.
3. The same holds true for a beacon falling into the end of the scan window (e.g., into Part $\mathcal{C}$), where parts of the "blocked" amount of time overlap with a Part $\mathcal{A}$ and possibly also $\mathcal{B}$ of another scan window.

Hence, in all three cases, the amount of "blocked" time is $d_{oTxRx} + d_{oRxTx} + \omega$.

Asymmetric Sequences

For asymmetric discovery (i.e., both devices have different duty-cycles), a quadruple of beacon- and reception window sequences can be designed such that B_∞ and C_∞ on the same device never overlap, while allowing for optimal (i.e., disjoint coverage) and deterministic two-way discovery between the two devices. Figure 2.9 depicts a pair of tuples $(B_{F,\infty}, C_{F,\infty})$ and $(B_{E,\infty}, C_{E,\infty})$ along with the corresponding coverage maps. As

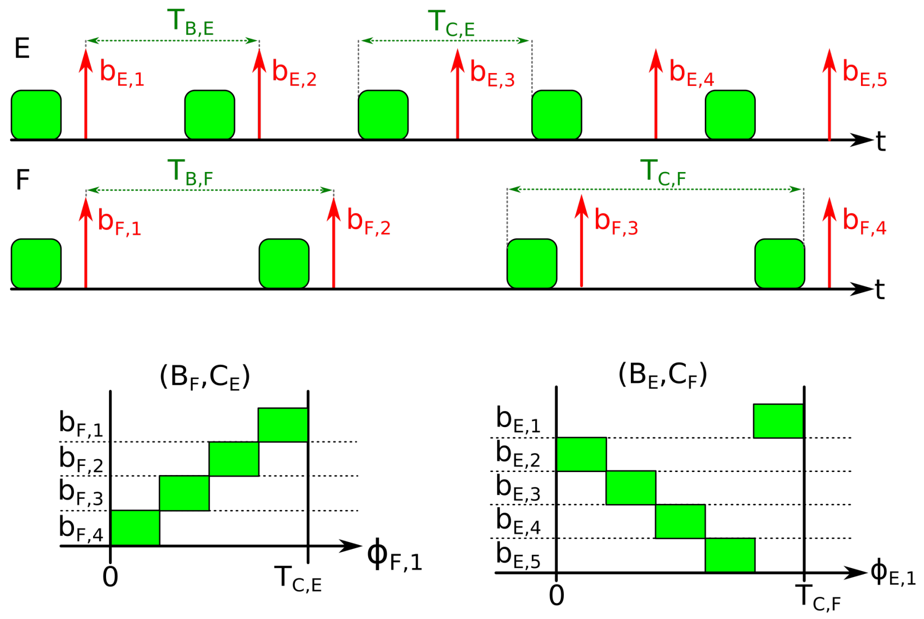


Figure 2.9: Asymmetric sequences and their coverage maps.

can be seen, $(B_{F,\infty}, C_{E,\infty})$ and $(B_{E,\infty}, C_{F,\infty})$ realize disjoint and deterministic discovery, while the sequences on the same device never overlap.

2.6.6 Non-Repetitive Reception Window Sequences

Throughout this chapter, we have restricted our considerations to infinite length reception window sequences C_∞ that are given by concatenations of some finite sequence C. Though all currently known deterministic ND protocols are constructed accordingly, reception window sequences that continuously alter over time are also feasible. In what follows, we study such sequences and establish that all our presented bounds remain valid for them.

Let us consider an arbitrary pattern of reception windows of infinite length C_∞. Such a C_∞ is characterized by its reception duty-cycle η_C. As in Section 2.4, we consider a sequence B' that consists of those beacons that are sent after both devices have come into range. Obviously, the first beacon $b_1 \in B'$ is received successfully if it directly overlaps with one of the reception windows. The fraction of time-units at which a transmission of b_1 leads to a reception is therefore identical to η_C. Another beacon that is sent by λ_1 time units later leads to additional points in time at which b_1 can be sent, such that one beacon out of b_1, b_2 is received successfully. These additional points in time lie λ_1 time-units earlier. Hence, like in Section 2.4, such points in time for later beacons are given by translating those of earlier ones to the left. If every point in time is covered by exactly one such translation, the tuple (B', C_∞) is disjoint and deterministic, and hence potentially optimal. This holds also true for cases in which C_∞ is not an infinite concatenation of the same C. The number of beacons M that need to be sent for guaranteeing deterministic discovery is therefore identical to the number of translations of the reception pattern C_∞, such that every point in time overlaps with exactly one such translation. It is:

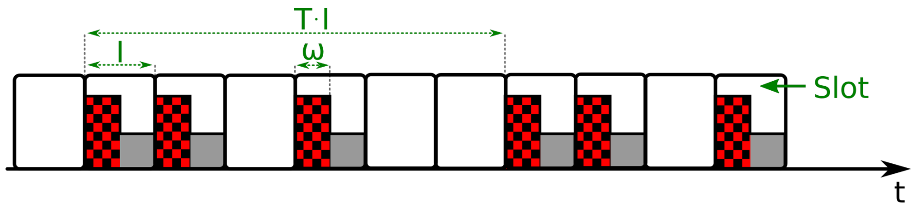


Figure 2.10: Slotted schedule proposed in [88]. Hatched bars depict beacons, smaller rectangles reception windows.

$$M = \left\lceil \frac{1}{\eta_C} \right\rceil \tag{2.29}$$

This is identical to Theorem 2.3, and hence all bounds remain unchanged.

2.7 Previously Known Protocols

In this section, we relate the worst-case performance of popular protocols and previously known bounds to the fundamental limits described in the previous section. Due to their relevance in practice, we consider only small duty-cycles η, for which the bound for symmetric protocols is given by Equation 2.13.

2.7.1 Worst-Case Bound of Slotted Protocols

As already described in Section 2.2, a worst-case number of slots within which discovery can be guaranteed is known for slotted protocols [87, 88]. The corresponding worst-case latency in terms of time is proportional to the slot length I, for which there is no known lower limit. In this section, we for the first time transform this worst-case number of slots into a latency bound and establish the relations to the fundamental bounds on ND presented in this chapter. We will also address the bound presented in [55, 56], which has been claimed to be tighter than the bound in [87, 88].

Latency/Duty-Cycle Bound

According to [87, 88], no symmetric slotted protocol can guarantee discovery within T slots by using less than $k \geq \sqrt{T}$ active slots per T. The associated worst-case latency L is $T \cdot I$ time-units, which is directly proportional to the slot length I. We in the following derive a theoretical lower limit for I and hence for L.

Slotted protocols can only function properly if the beacon length ω is "at least one order of magnitude smaller than I" [88]. If this requirement is not fulfilled, often a beacon might not overlap with a reception window even though the active slots of two devices overlap, as illustrated in Figure 2.10. Here, the slot length I in a slot design as proposed in [88] has been set to $2{\cdot}\omega$. As can be seen, practically none of the offsets for which two active slots overlap would lead to a successful reception, since every beacon would only partially overlap with a reception window. If I would be increased, the fraction of successful offsets would gradually become larger. For achieving zero collisions independently of the slot length, let us assume a full duplex radio, which can both transmit and receive during the same points in time. Then, the theoretical limit on the slot length I becomes as low as one beacon transmission duration ω, which leads to the following duty-cycle:

$$\eta = \frac{k \cdot (I + \alpha\omega)}{T \cdot I} = \frac{k \cdot (I + \alpha\omega)}{L} \tag{2.30}$$

Since the limit from [87, 88] requires that $k \geq \sqrt{T} = \sqrt{L/I}$, with a slot length of $I = \omega$, Equation 2.30 leads to the following latency limit:

$$L \geq \frac{\omega(1 + 2\alpha + \alpha^2)}{\eta^2} \tag{2.31}$$

For $\alpha = 1$, this bound becomes $\frac{4\omega}{\eta^2}$ and hence identical to the fundamental bound for symmetric protocols given by Theorem 2.8. For all other values of α, this bound exceeds the one given by Theorem 2.8.

However, the assumption of full-duplex radios is not fulfilled by most wireless devices. Further, every wireless radio requires a turnaround time to switch from transmission to reception, during which the radio is unable to receive any beacons. Even for recent radios, this time is large against the beacon transmission duration ω (e.g, for the nRF51822 radio [53], it lies around 140 µs, whereas beacons can be as short as 32 µs). Therefore, I will be orders of magnitude larger than ω, which linearly increases the worst-case latency slotted protocols can guarantee in practice. It is worth mentioning that this increase occurs in addition to the duty-cycle overhead induced by the turnaround times of the radio.

We now study the bound presented in [55, 56], which has been claimed to be lower in terms of slots than the one presented in [87, 88]. It is achieved by assuming two beacon transmissions per active slot ([87, 88] assumes only one), of which one beacon is sent slightly outside of the slot boundaries. By accounting for the two beacons per active slot, Equation 2.30 becomes $\eta = \frac{k \cdot (I + 2\alpha\omega)}{L}$, which leads to the following bound for the protocols proposed in [55, 56]:

$$L \geq \frac{\omega(\frac{1}{2} + 2\alpha + 2\alpha^2)}{\eta^2} \tag{2.32}$$

This bound becomes minimal for $\alpha = 1/2$, for which it is identical to the bound in Theorem 2.8. Hence, the bound in [55, 56] is lower in terms of slots than the bound in [87, 88], but indentical or larger in terms of time.

Latency/Duty-Cycle/Channel Utilization Bound

All previously known bounds for slotted protocols are in the form of relations between the worst-case number of slots and the duty-cycle. The channel utilization, which is directly related to the beacon collision rate, has not been considered before. However, in slotted protocols, the channel utilization depends both on the number of active slots per period and on the slot length. For sufficiently large slot lengths, the turnaround times of the radio only play a negligible role. Further, the time for reception in each slot approaches nearly the whole slot length I. Hence, for $I >> \omega$, we can compute the duty-cycle of slotted protocols as follows.

$$\eta_B = \frac{k\omega}{IT}, \quad \eta_C = \frac{kI}{IT} = \frac{k}{T}, \quad \eta = \eta_C + \alpha\eta_B \tag{2.33}$$

With the requirement of $k \geq \sqrt{T}$ from [87, 88], one can express the slot length I by the desired channel utilization η_B in Equation 2.33, which results in the following bound.

$$L \geq \frac{\omega}{\eta\eta_B - \alpha\eta_B^2} \tag{2.34}$$

From comparing Theorem 2.9 (cf. Equation 2.15) to Equation 2.34, it follows that if $\eta_{B,m}$ lies below $\eta/2\alpha$, the worst-case latency a slotted protocol can guarantee with a channel-utilization of $\eta_B = \eta_{B,m}$ is identical to the corresponding fundamental bound (recall that we only consider optimal duty-cycles). For $\eta_{B,m} > \eta/2\alpha$, slotted protocols cannot reach the fundamental bound from Theorem 2.9. In practice, this means that slotted protocols can potentially perform optimally in busy networks with many devices discovering each other simultaneously, but cannot offer optimal performance in networks in which new devices join gradually and hence only a master node and the joining device need to carry out ND at the same time.

We in the following evaluate the popular protocols Disco [15], Searchlight-Striped [2], U-Connect [23] and diffcode-based protocols [87] and compare them to the fundamental bound given by Theorem 2.9. In Disco, active slots are repeated after every p_1 and p_2 slots, where p_1 and p_2 are coprimal numbers. The Chinese Remainder Theorem implies that there is a pair of overlapping slots among two devices every $p_1 \cdot p_2$ time-units. U-Connect also relies on coprimal numbers for achieving determinism. In contrast, Seachlight defines a period of T and a hyper-period of T^2 slots. The first slot of each period is active, whereas a second active slot per period systematically changes its position, until all possible positions have been probed. Diffcode-based solutions are built on the theory of block designs and hence guarantee a pair of overlapping slots among two devices with the minimum possible number of active slots per worst-case latency. More details on these protocols can be found in [8].

Slot length-dependent equations on the worst-case latency and duty-cycle of these protocols are available from the literature. When assuming sufficiently large slots and by expressing the slot length I by the channel utilization η_B similarly to Equation 2.33, one can derive the relations between the worst-case latency, duty-cycle and channel utilization given in Table 2.2. Clearly, only Diffcode-based schedules reach the optimal performance in this metric, whereas all other ones perform below the optimum.

In summary, slotted protocols can perform optimal in the laten-cy/duty-cycle/channel utilization metric, if the channel utilization remains low. In the latency/duty-cycle metric, however, higher required channel utilizations prevent slotted protocols from performing optimally.

Protocol	$L(\eta_B, \eta)$
Diffcodes [87]	$\frac{\omega}{\eta\eta_B - \alpha\eta_B^2}$
Disco [15]	$\frac{8\omega}{\eta\eta_B - \alpha\eta_B^2}$
Searchlight-S [2]	$\frac{2\omega}{\eta\eta_B - \alpha\eta_B^2}$
U-Connect [23]	$\frac{\left(3\omega + \sqrt{\omega^2(8\eta - 8\alpha\eta_B + 9)}\right)^2}{8\omega\eta_B\eta - 8\omega\alpha\eta_B^2}$

Table 2.2: Worst-case latencies of slotted protocols.

2.7.2 Worst-Case Bound of PI-based protocols

In the previous section, we have established that slotted protocols can realize the bound in the latency/duty-cycle metric given by Theorem 2.8 in theory, but due to failed receptions and radio turnaround times, all practical protocols can only achieve worst-case latencies that lie orders of magnitude higher. If slotted protocols do not perform optimal, can slotless protocols do? Since they allow for temporally decoupling transmission and reception, the constraints of slotted approaches do not apply to them.

As already mentioned, the worst-case performance of slotless protocols has not been clear previously to this work. Further, the performance of PI-based protocols depends on their configuration. We will describe two slotless, PI-based parametrization schemes in Chapter 4, which are referred to as $PI-0M$ and $PI-kM^+$. As we will describe later, the parametrization schemes $PI-0M$ and $PI-kM^+$, which are presented in Chapter 4, and the protocol *Griassdi*, which is described in Chapter 5, achieve almost identical worst-case latencies. We formally proof the optimality of accordingly parametrized PI-based protocols in the latency/duty-cycle metric in Section 4.7. This will lead to the insight that PI-based protocols can realize the bound given by Theorem 2.8.

2.8 Conclusion

In this section, we first describe open problems left for future research and then summarize the main results of this chapter.

2.8.1 Open Problems

Problems On Fundamental Limits

Regarding the future work on fundamental limits, there are two important problems left open. First, what is the lowest latency an asymmetric protocol can guarantee, if the duty-cycles of all devices are unknown? An what is the bound for asymmetric ND for duty-cycles for which $2/\eta$ is not an integer?

Second, the bounds derived so far are valid for a pair of devices discovering each other. For unidirectional beaconing, protocols in which 100 % of all discovery attempts are successful within L time-units can be realized in practice. For increasing numbers of devices discovering each other simultaneously, it is inevitable that their beacons will collide and hence, an increasing number of discovery attempts will fail. Therefore, generalized performance bounds for multi-device scenarios need to be derived. Such bounds are in the form of a function $L(\eta_B, \eta_C, S, P_f)$, which needs to be interpreted as follows. For a given number of devices S with duty-cycles η_B and η_C, in no ND protocol, a fraction of at least $1 - P_f$ of all discovery attempts will terminate successfully within less than L time-units. Clearly, for $S \to 1$ and $P_f \to 0$, this bound converges to L from Equation 2.9. The following two mechanisms determine the performance in multi-device scenarios.

1) Lowering the Channel Utilization: The rate of collisions directly correlates to the channel utilization η_B, as described by Equation 2.14. Hence, devices can reduce the failure probability P_f by reducing η_B, which will, however, negatively affect the discovery latencies achieved in the two-device case (cf. Equation 2.9).

2) Redundant Coverage: Optimality in the $L(\eta_B, \eta_C)$ - metric for two devices implies that every initial offset is covered exactly once (cf. Theorems 2.3 and 2.6) and hence, every collision leads to a failed discovery. However, an ND protocol might cover multiple or all initial offsets more than once. Hence, for such offsets, more than one beacon would overlap with a reception window, and as long as one of them is not subjected to collisions, the discovery procedure will succeed. Moreover, it seems feasible to construct protocols that first cover every offset exactly once by a beacon sequence B' of length M. In addition, the same offsets are then covered again by concatenations of multiple instances of B'. In other words, such protocols would guarantee short latencies in the two-device case while performing potentially optimally also in multi-device scenarios.

The collision of a pair of beacons from two devices often induces an increased collision probability of subsequent pairs of beacons. For example, consider protocols in which beacons are sent with periodic intervals. Since all devices transmit with the same interval, a collision implies that all later beacons will also collide. To make protocols robust against failures due to collisions, a beacon schedule needs to fulfill the following property. Given any two beacons that both overlap with a reception window for the same offset Φ_1, their individual collision probabilities should exhibit the lowest possible correlation. It is currently not clear which degree of such a decorrelation can be actually achieved. Further, measures for decorrelating collision probabilities might reduce the latency performance, because they could prevent beacons from being sent at their optimal points in time. Hence, not all initial offsets can be covered with the fewest possible number of beacons, making additional beacon transmissions necessary. Besides open questions on decorrelating collisions, for protocols being optimal in the multiple-device case, how many times should every initial offset be covered? These questions need to be studied further in order to derive agnostic bounds in the form of $L(\eta_B, \eta_C, S, P_f)$.

Problems in Protocol Design

Our results also outline two important directions for the development of future ND protocols. First, there is no existing protocol which, for *every* duty-cycle and every required collision rate, could realize the optimal performance predicted by Theorem 2.9. Second, protocols that contain decorrelation mechanisms to make the collision of each beacon independent from the occurrence of previous collisions have not been studied thoroughly. Though BLE applies some random delay for scheduling its beacons [43], the optimal randomization technique to obtain the best trade-off between robustness and worst-case latency remains an open question.

2.8.2 Concluding Remarks

We have presented and proven the correctness of multiple fundamental bounds on the performance of deterministic ND protocols. In particular, we have presented bounds for unidirectional beaconing, for symmetric and for asymmetric bi-directional ND. Further, we have shown that in the latency/duty-cycle metric, only slotless protocols can reach optimal performance. These results underpin the need for PI-based, slotless protocols. The rest of this thesis is concerned with the analysis and design of such PI-based protocols. The prerequisite for identifying optimal designs is to establish a profound understanding of the mechanisms that determine the discovery latencies. In the next chapter, we present a comprehensive theory to compute mean and worst-case latencies of PI-based protocols for all possible parameter values.

3 Periodic Interval-Based Neighbor Discovery

3.1 Introduction

In the previous chapter, we have identified multiple fundamental bounds on ND. We have concluded that no slotted protocol can perform optimal in the latency/duty-cycle metric. Whereas slotted protocols require that beacon transmissions are temporally coupled to reception phases, slotless, PI-based solutions break away from this constraint. Packet transmissions take place with a period T_a, while receptions take place for a certain duration d_s once per period T_s. The parameters T_a, T_s and d_s are drawn over continuous time and can be chosen freely by the protocol designer.

Note that T_s corresponds to the reception period T_c defined in Chapter 2, T_a to the beacon period T_b and d_s to the duration of the first reception window d_1. To be compatible with the naming conventions of the Bluetooth specification [43], which denotes the interval for reception as the *scan interval*, the duration of one reception phase as the *scan window* and the interval for packet transmissions as the *advertising interval*, we use a different set of symbols than in Chapter 2 for the rest of this thesis. To make unambiguously clear that a PI-based solution is considered, we will use the notation of T_a (instead of T_b), T_s (instead of T_c) and d_s (instead of d_i with $i = 1$) in the remaining chapters of this thesis. Further, in Chapter 2, L had denoted the worst-case latency measured from the point in time at which both devices have come into range. We in this section consider the packet-to-packet worst-case latency l^*. In the context of BLE networks, the packet-to-packet latency is practically most relevant. Here, a device is often switched on within the range of a receiver and starts transmitting its first packet immediately thereafter. In such scenarios, the packet-to-packet latency is the one observed in practice. In contrast, the worst-case latency L, as defined in Chapter 2 is the largest latency observed when bringing an already running sender within the range of a receiver. In PI-based protocols, both latencies differ by exactly T_a time-units, and hence both terms are interchangeable.

In this chapter, unless explicitly stated, discovery latencies are measured from the first packet in range to the fist successfully received one. Though PI-based schemes can also be used for symmetric two-way discovery, most existing PI-based protocols (e.g., BLE and ANT/ANT+) assign different roles to different devices. In particular, they assume that one device only broadcasts advertising packets, whereas the other one only scans.

Motivation for Modeling PI-Based Protocols: Since PI-based protocols decouple reception from transmission, they are not subjected to the limitations of slotted protocols described in Chapter 2. Therefore, they can potentially perform optimally in the latency/duty-cycle design space. This underlines the importance of understanding their behavior. Besides designing protocols with potentially better latency/duty-cycle relations, understanding PI-based protocols is also of tremendous importance for modeling and optimizing many existing networks. Though most recent research has considered slotted discovery, adaptations of the slotless scheme described above are widely used in practice. The ANT/ANT+[46] protocol, which is used in over 100 million devices [47], implements this scheme [45]. Further, it has been adopted and slightly modified for BLE, in which an advertiser periodically sends up to three packets in a row on different channels [41]. The corresponding period consists of the sum of an interval T_a and a random delay of up to 10 ms. On the remote side, the scanner listens periodically on one of these three channels for a duration of d_s, thereby toggling the channel once per interval T_s.

As we will describe in detail in Section 3.2, even though such protocols are widely used e.g., in billions of BLE devices, their behavior could not be fully analyzed prior to this work. Except for the trivial case of $T_a \leq d_s - \omega$, no theory for computing their latencies has been known. In particular, it is has not been clear whether they can guarantee any upper latency bounds. Moreover, the impact of different parameter valuations for T_a, T_s and d_s could not be studied in a systematical manner. As a result, there has been no feasible way to choose optimal protocol parameters, e.g., for BLE. However, parameter optimization is extremely important since, as we will show, unfavorable parametrizations risk lying in a hyperbolic peak and therefore cause long mean latencies and high energy consumptions for the connection setup.

Difference from Modeling Slotted Protocols: The main reason for this lack of understanding is the complex analysis of PI-based, slotless protocols. Whenever two slots in a slotted protocol overlap in time, discovery is guaranteed, regardless of the actual temporal length of the overlap. Therefore, only a finite number of different temporal offsets between the active slots of two devices need to be considered, which makes the analysis simple. In contrast, in slotless, PI-based protocols, there could be an infinite number of initial offsets with different associated latencies. This makes existing models for slotted protocols unsuitable for slotless solutions. The theory presented in this chapter shows that the range of possible initial offsets can be subdivided into multiple, variable-length partitions with constant discovery latencies and hence the number of possibilities becomes finite also for PI-based protocols. Another challenge is finding the borders of these partitions, which is not straightforward.

In this chapter, we for the first time present a mathematical model that can compute the mean and worst-case ND latencies of periodic interval-based, slotless protocols for all possible parameter values.

Modeling PI-Based Protocols: The key idea of our proposed model is to track the change in the temporal distance between neighboring advertising packets and scan windows over time, rather than considering the corresponding absolute points in time. This temporal distance $\Phi[k]$ is depicted in Figure 3.1 for $k = 0$ to $k = 3$. Note that for

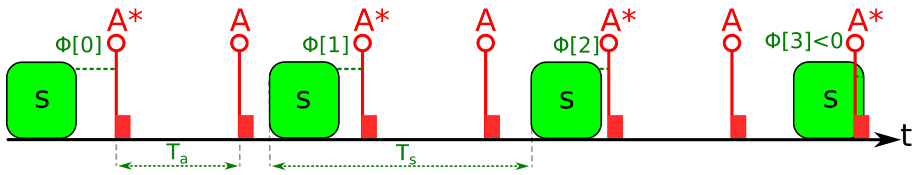


Figure 3.1: Shrinking distances $\Phi[k]$ between neighboring advertising packets (A) and scan windows (S). In this example, the index k is a counting number for the scan windows.

the rest of this thesis, we apply a slightly different notation than in Chapter 2 to the offsets between packets and scan windows: Wheres Φ_k, $k = 1, 2, ...$ in Chapter 2 denoted the offset of the k'th packet considered from the end of the last reception window in a sequence, $\Phi[k]$ is the offset between suitable pairs of neighboring advertising packets and scan windows. The index k in $\Phi[k]$ defines the individual instances of such pairs. We will describe which pairs are suitable, later. The initial offset of the first scan window and advertising packet in range is always denoted with $\Phi[0]$. In the example of Figure 3.1, such suitable pairs are formed by each scan window and its right neighboring advertising packet. An advertising packet is the *right neighbor* of a scan window k, if it is the closest packet temporally on the right of it. In Figure 3.1, such neighboring advertising packets are tagged with $A*$, whereas the remaining ones are labeled with A. In this example, after each instance k of the scan interval T_s, the temporal distance $\Phi[k]$ shrinks by a constant value. This shrinkage occurs repeatedly for multiple subsequent intervals, until the temporal distance falls below the length of one scan window d_s. Therefore, a match between the advertiser and the scanner occurs, as for the last scan window in Figure 3.1. In other examples, $\Phi[k]$ grows instead of shrinking, or the distance $\Phi[k]$ shrinks or grows when considering suitable multiples of T_a and T_s. Nevertheless, the exemplified concept can be generalized to all possible constellations. These generalizations require a more elaborate procedure, which is described in Section 3.4. By taking into account all possible initial offsets between the first advertising packet and the first scan window, the exact expected value of the discovery latency and its maximum value can be calculated using analytical methods with low computational complexities.
Comparison with Related Models: This technique differs significantly from all known models for PI-based protocols. All existing models are restricted to the trivial case of $T_a \leq d_s - \omega$, and rely on counting the number of T_a-intervals until reaching the time-instance at which the scanner switches on its receiver for the first time. Parametrizations with $T_a > d_s - \omega$ are very relevant in practice, since they enable low duty-cycles of the advertiser and are widely used in practice, e.g., as a recommended parametrization in the BLE find me profile [42]. To the best of our knowledge, the technique of tracking the change in the temporal distance among neighboring periodic events, as proposed in this chapter, has not been known before, neither for ND nor in other domains. This generic technique is suitable for modeling possible overlaps of all kinds of periodically repeated

actions, and can therefore also be applied in other applications, e.g., for accelerating discrete-event simulations [27].

Implications of our Results: With our proposed theory, we can for the first time show that a maximum discovery latency can be guaranteed for all parametrizations except for a finite number of singularities, at which the latency converges towards infinity. This insight allows interval-based protocols even with $T_a > d_s - \omega$ to be applied in scenarios where deterministic maximum latencies are required, in which currently only slotted protocols are used. In addition, our proposed theory can be used to find optimal parametrizations of protocols like ANT/ANT+ or BLE. In the Chapters 4 and 5, we will show that optimized PI-based protocols can significantly outperform all previously known discovery protocols, e.g., by achieving approximately 50× shorter discovery-latencies than the Searchlight [2] protocol for a given duty-cycle.

Further, slotted protocols, which we will describe in detail in Section 3.2, typically restrict the set of possible duty-cycles to a finite number. In contrast, interval-based protocols allow the highest flexibility in choosing interval lengths and duty-cycles, thereby offering the most fine-grained trade-off between latencies and energy-consumptions. As we will show, they nevertheless offer deterministic discovery latencies for almost all parametrizations.

Our Contributions: In this chapter, we present the first generic mathematical theory on the discovery latencies of PI-based protocols. We thereby make the following contributions:

- We propose a model that can compute the exact mean and maximum discovery latencies of PI-based protocols for the complete range of possible parametrizations, including $T_a > d_s - \omega$. This theory leads to important new insights into such protocols and provides a full understanding of the probabilistic processes involved.
- By using our model, we demonstrate that the discovery-latency is bounded for almost all parametrizations. The popular belief until now was that parametrizations with $T_a > d_s - \omega$ lead to unbounded discovery latencies. This new finding has important implications on the design of wireless protocols and the parametrization of protocols like BLE.
- We evaluate our proposed solution both with comprehensive discrete-event simulations and real-world measurements, which are based on an implementation on two wireless radios. Thereby, we demonstrate that our predicted latency bound is safe (i.e., the latency never exceeds this bound) and tight (i.e., the bound can be reached in practice), and that our estimated mean latencies are precise.

Organization of this Chapter: The rest of this chapter is organized as follows. In Section 3.2, we present related work on modeling PI-based protocols. We formally define the problem of PI-based ND in Section 3.3. In Section 3.4, we present our proposed theory for computing the discovery latencies. This theory involves multiple claims, which are formally proven in Section 3.5. In Section 3.6, we present an efficient implementation of our proposed technique. Readers who are only interested in understanding our proposed theory may only focus on Section 3.4. Only those wo are interested in an implementation may continue to Section 3.6. We describe how our publicly available implementation of the model can be used in Section 3.7. Next, we evaluate our theory with comprehensive discrete event simulations and real-world measurements in Section 3.8. In Section 3.9, we describe how the parameter values of interval-based protocols can be chosen in an optimal fashion. Finally, we summarize our results and discuss their implications in Section 3.10. To ease the readability of this chapter, a complete table of symbols used is given in Appendix A. The notations we introduce in this chapter remain valid throughout the remaining chapters of this thesis.

3.2 Related Work

In this section, we provide an overview on existing models for periodic interval-based, slotless protocols, such as BLE. As already mentioned, exact models only exist for trivial cases, e.g., $T_a \leq d_s - \omega$. The case of $T_a \leq d_s - \omega$ implies that the duration of one scan window must be larger than the time between two advertising packets. Therefore, every scan attempt is successful and the discovery latency is limited to roughly $1 \cdot T_s$. The first known model (for $T_a \leq d_s - \omega$) has been presented along with the introduction of the STEM-B protocol [63], which is one of the first known PI-based protocols. In [36, 35] and [34], this solution has been adopted to the BLE protocol to account for multiple channels, but the limitation to $T_a \leq d_s - \omega$ remains. Other models, e.g., [58], study the special case of *continuous scanning*, in which the scan window d_s is equal to the scan interval T_s and hence, the scanner is (almost) always in reception mode.

For the general case, as [45] states, there is no known model to compute the discovery latencies of PI-based protocols. Previous work has studied the discovery latencies of PI-based protocols using simulations [30]. Simulations are computationally very complex, since the discovery process needs to be simulated repeatedly for a large number of initial time offsets in order to assess the mean discovery latencies. An attempt to reduce this complexity has been proposed in [28]. However, the resulting complexity is still impractically high, and only an estimation of the mean latency can be given. No estimate on the maximum discovery latency can be provided.

An attempt to derive a probabilistic model for $T_a > d_s - \omega$ has been made in [13], [12]. Here, the probability that an advertising packet meets a scan window has been assumed to be constant for all packets on a particular channel. In particular, if p_m is the probability of a miss of the first advertising packet on a particular channel, then it is assumed that the miss-probability of the k'th advertising packet is p_m^k. However, the periodic nature of BLE implies that a strong correlation between the probability and the interval instance exists and hence $p_m(k)$ is a function of the interval count k. Hence, as we will also show in Section 3.8.4, the results obtained from such models do not correlate well with the actual discovery latencies. Also, under the assumption of constant matching probabilities for all packets, a deterministic discovery is not guaranteed and therefore no upper bound can be inferred.

In [40], the discovery latency of BLE has been computed by applying the probabilistic model checker PRISM [29] for some discrete values of $T_a > d_s - \omega$, T_s and d_s. However, in

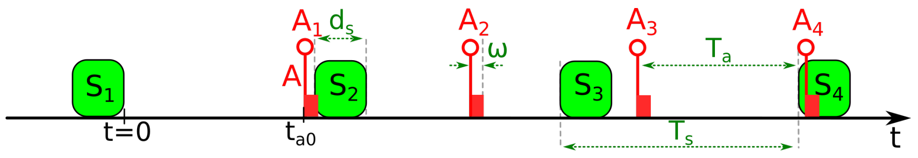


Figure 3.2: Sequence of advertising packets ($A_1 ... A_4$) and scan windows ($S_1 ... S_4$) for neighbor discovery.

our opinion, the necessary details to understand this work have not been provided in [40]. In addition, the initial offset between the first advertising packet and scan window has been assumed to be fixed. As a result, this approach is unable to determine the mean- and worst-case discovery latencies among all possible initial offsets.

Another attempt to model the neighbor discovery latency of BLE has been presented in [21]. In this model, the range of possible initial offsets is discretized into equal-length partitions. The discretized advertising interval T_a is rounded to the next integer that is coprime to the discretized scan interval T_s and hence, the Chinese remainder theorem applies. Due to the error introduced by this, and since it is not clear which partition length is sufficiently small, this model can only approximate the discovery latencies and the actual error cannot be quantified. Moreover, it relies on computing the modular multiplicative inverse repeatedly for every possible initial offset, which is computationally expensive. Especially, if the discretization is carried out with fine granularities and hence, high accuracies, the computational complexity grows. Therefore, this model shows a similar behavior to that of previous accelerated simulation models, such as, e.g., [28].

Usually, PI-based protocols are classified as non-deterministic, since no upper bounds for their latencies could be determined until now. To overcome the lack of determinism, it has been proposed to configure the BLE protocol such that its parameter values fulfill the Chinese remainder theorem [24]. However, such configurations restrict the range of possible valuations unnecessarily. As we will show, the vast majority of parametrizations is deterministic.

In summary, to the best of our knowledge, no valid model for PI-based protocols for the general case ($T_a > d_s - \omega$, $T_s > d_s$) has been known prior to this work.

3.3 Problem Formulation

In this section, we formally define the problem that is addressed in this chapter. We consider two devices that need to discover each other using a wireless protocol. We recall that one device, referred to as the *advertiser* A, periodically sends out *advertising packets*, starting from an uniformly distributed[1] random point in time t_{a0}. Transmitting a packet takes ω time-units. The packets are sent periodically with the *advertising interval* T_a. The remote device is called the *scanner* S. It periodically switches on its receiver once per *scan interval* T_s and listens for advertising packets for a duration called the *scan*

[1]The assumption of uniformly distributed initial time-offsets influences the computed mean discovery latencies, but not the worst-case latencies.

window d_s. The ND procedure is successful once an advertising packet has been received successfully by the scanner. Therefore, both devices need to be awake simultaneously for at least one packet transmission duration ω. In Figure 3.2, the first advertising packet (labeled with A_1) is not received successfully, as it only partially overlaps with the scan window S_2. The second and third advertising packets A_2 and A_3 are not received either, because they do not overlap with any scan window at all. The fourth packet A_4 lies entirely within the scan window S_4 and is therefore received successfully. We assume no packet loss and only one channel. Then, the necessary and sufficient condition for a successful reception is given by the existence of some $i, j \in \mathbb{N}_0$, such that

$$j \cdot T_s - d_s \leq t_{a0} + i \cdot T_a \leq j \cdot T_s - \omega, \tag{3.1}$$

where t_{a0} is the random point in time at which the first advertising packet is sent and i and j are the indices or the counting numbers of all advertising packets/scan windows. The questions answered in this chapter are the following. What are the maximum and average times for an advertising packet to lie within a scan window for the first time (as given by Inequality 3.1), i.e., what are the max and mean discovery latencies?

3.4 Protocol Analysis

In this section, we describe the key concepts of our proposed solution to compute the mean- and maximum ND latencies. To develop an intuitive understanding of this, we first present four different examples. Next, we formalize and generalize these examples. Towards this, without any loss of generality, we first set the duration of the advertising packets ω to 0. Because a successful reception occurs only if the advertising packet is received entirely by the scan window, this measure can be compensated for by shortening the scan window d_s by ω time units. Further, we have to add ω time units to all computed latencies. Recall that in this Chapter, unless explicitly stated, all discovery latencies are defined as the latency from the first packet sent in range to the first successfully received packet.

3.4.1 Examples

a) $\mathbf{T_a \leq d_s}$: Situations with $T_a \leq d_s$ form the most simple cases. Consider the situation depicted in Figure 3.3 a). Again, the straight lines with circles on their heads depict the starting times of the advertising packets, whereas the boxes depict the scan windows. Here, $T_a \leq d_s$ and hence a successful match is guaranteed within one scan interval. The temporal distance between the second scan window and each advertising packet shrinks by T_a time units with every packet sent. Given an initial offset $\Phi[0]$ between the end of the first scan window and the first advertising packet, the discovery-latency is defined by the number of advertising intervals that need to pass until the scan window is reached. It follows that the ND latency l is:

$$l(\Phi[0]) = \begin{cases} \left\lceil \frac{T_s - \Phi[0] - d_s}{T_a} \right\rceil \cdot T_a, & \text{if} \quad \Phi[0] \leq T_s - d_s, \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

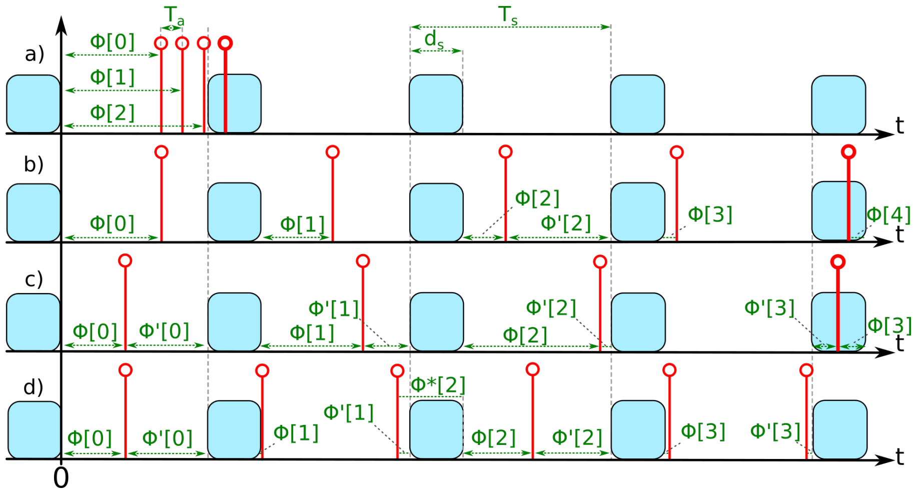


Figure 3.3: Examples for different situations of ND processes. The rounded boxes depict the scan windows, whereas the lines with circles at their heads depict the advertising packets.

As can be seen easily from the figure, the maximum discovery latency for this example is $l^* = \lceil \frac{T_s - d_s}{T_a} \rceil \cdot T_a$.

b) $\mathbf{T_a > d_s}$: Whenever $T_a > d_s$, as depicted in Figure 3.3 b), a successful discovery within one scan interval is not guaranteed anymore. Here, it is beneficial to describe the problem using a different representation. Instead of considering the absolute points in time the advertising packets and scan windows begin at, we examine the relative time differences $\Phi[k]$ between each scan window $k = 0, 1, 2, ...$ and its neighboring advertising packets. As we will introduce later, there are "shrinking" and "growing" constellations of scan windows and advertising packets. The example in Figure 3.3 b) is shrinking, and here only each closest neighboring advertising packet that is temporally on the right of a scan window (i.e., the first advertising packet on the right of each scan window) is relevant.

From Figure 3.3 b), one can observe that in this example, the temporal distance $\Phi[k]$ is monotonically shrinking for increasing indices k, and the amount of shrinkage is constant per scan interval k. This shrinkage is denoted as γ_0, and it is $\gamma_0 = \Phi[k] - \Phi[k+1]$. Here, one can observe that a match occurs after $\Phi[k]$ becomes smaller than 0 (cf. $\Phi[4] < 0$ in Figure 3.3 b). This holds true for any initial offset, as long as the shrinkage γ_0 is smaller or equal than d_s .

In the previous example with $T_a \leq d_s$, (cf. Figure 3.3 a)), we had counted the number of advertising intervals that fit into the temporal distance to the next scan-window (i.e., $T_s - d_s - \Phi[0]$ time units) for computing the discovery latency. The example from Figure 3.3 b) can be handled in the same way - with the only difference being that we have to count the number of γ_0-intervals that fit into the initial offset, instead of the number of advertising intervals. Hence, the discovery latency is defined by Equation 3.3.

$$l(\Phi[0]) = \begin{cases} \left\lceil \frac{\Phi[0]}{\gamma_0} \right\rceil \cdot T_a & \text{, if} \quad 0 \leq \Phi[0] \leq T_s - d_s \\ 0 & \text{, else.} \end{cases} \tag{3.3}$$

c) $\mathbf{T_a > d_s}$, growing: In the previously examined situation (depicted in Figure 3.3 b)), the offset $\Phi[k]$ shrank for increasing values of k. However, there are cases in which $\Phi[k]$ becomes larger with increasing values of k, as depicted in Figure 3.3 c). Whereas the distance from a scan window k to its right neighboring advertising packet $\Phi[k]$ grows with increasing values of k, the distance to its left neighboring advertising packet $\Phi'[k]$ shrinks. Growing constellations can therefore be handled similarly to shrinking ones.
d) $\boldsymbol{\gamma_0} > \mathbf{d_s}$: The examples b) and c) both had $\gamma_0 \leq d_s$ in common. However, for $\gamma_0 > d_s$, the computation becomes more involved. In the example from Figure 3.3 d), the distance $\Phi[0]$ shrinks from $\Phi[0]$ to $\Phi[1]$ by γ_0 time units. After $\Phi[1]$, this distance Φ_2^* (we label this distance with *, because it is no longer the distance to the closest advertising packet on the right of the scan window) becomes smaller than $-d_s$, since the advertising packet has moved to the left side of the scan window. Because advertising packets on the left have no chance of hitting the scan window considered (given the situation is shrinking), the advertising packet that needs to be considered changes (e.g., from $\Phi[2]^*$ to $\Phi[2]$ in the figure). One advertising packet is skipped and the offset $\Phi[k]$ becomes larger again for this packet (i.e. $\Phi[2] > \Phi[1]$).

We say that after $\Phi[1]$, the advertising packet "overtakes" the scan window. After that, from $\Phi[2]$ to $\Phi[3]$, the distance shrinks again by γ_0, until the relevant packet changes the next time. Shrinking the initial offset successively by γ_0 time units cannot guarantee discovery, since the "overtaking" might repeat. However, let us consider two consecutive skipped advertising packets, e.g., the packets with the offsets $\Phi'[1]$ and $\Phi'[3]$ in Figure 3.3 d). Both packets fulfill the condition that they are the first ones after an advertising packet has "overtaken" the corresponding scan window, and it is $\Phi'[1] - \Phi'[3] = \mathit{const.} = \gamma_1 < \gamma_0$. In other words, when considering appropriate multiples of the advertising interval (here: $i_2 = 3 \cdot T_a$) and scan interval (here: $j_2 = 2 \cdot T_s$), given an offset $\Phi[k]$ between a scan window k and its neighboring advertising packet, the offset between a scan window that is i_n scan intervals and j_n advertising intervals later will be $\Phi[k] - \gamma_1$. In case 3.3 d), $\gamma_1 < d_s$ and therefore the advertising packets and scan windows will match guaranteed after the initial offset $\Phi[0]$ has shrunk by a certain number of steps of length γ_1.

3.4.2 γ-Sequences

We now generalize the above examples. Towards this, let us first define the concept of γ-sequences and state multiple properties, which we claim to be true. We will then use these properties to compute the ND latencies for arbitrary parametrizations. We will focus our attention on proving these claims in Section 3.5.

Definition 3.1: γ-sequences: Let there be a set of parameters (T_a, T_s, d_s) and pairs of integers $(i_n,\ j_n)$, $n = 0, 1, 2, ...n_m$, with $i_n \geq i_{n-1}$ and $j_n \geq j_{n-1}$. We call a sequence of scan windows and advertising packets defined by $(1{\cdot}i_n{\cdot}T_a,\ 1{\cdot}j_n{\cdot}T_s)$, $(2{\cdot}i_n{\cdot}T_a,\ 2{\cdot}j_n{\cdot}T_s), ...$ a *γ-sequence* of order n, if the following properties are fulfilled.

Property 3.1: If the time offset between any scan window and advertising packet is Φ, the time offset between the scan window that is j_n scan-intervals later and the advertising packet that is i_n advertising intervals later will be $\Phi \pm \gamma_n$. Sequences of tuples of scan windows and advertising packets, in which such offsets always decrease, are called *shrinking* γ-sequences, whereas constellations, in which these offsets increase, are called *growing* γ-sequences.

Property 3.2: $0 \leq \gamma_n \leq \min(T_a, T_s),\ \forall n$

Definition 3.1.1: Mode m: We define $m_n \in (g, s, c)$ as the *mode* of a γ-sequence of order n. If the offset Φ between any scan window and any advertising packet becomes larger every i_n advertising packets and j_n scan-windows, then $m_n = g$ (growing). If it becomes smaller, then $m_n = s$ (shrinking). If $\gamma_n = 0$, then there is no shrinkage or growth, and $m_n = c$ (coupling).

Property 3.3: $\gamma_n = |T_s - |\sum_{k=0}^{n-1} a_k \cdot \gamma_k||$, $a_k \in \mathbb{Z} \setminus 0$. Here, the only valid choices of a_k are those that satisfy the following: i) $\gamma_n \leq \frac{1}{2}\gamma_{n-1}$, ii) the sum $\sum_{k=0}^{n-1} |a_k| \cdot (i_k T_a)$ is minimized, and iii) for a shrinking sequence, $a_k < 0$, otherwise, $a_k > 0$.

Property 3.4: For any γ_n, $i_n \cdot T_a$ has always at least the value of the smallest possible sum $\sum_{k=0}^{n-1} |a_k| \cdot (i_k T_a)$, for which $|\sum_{k=0}^{n-1} a_k \gamma_k| \geq T_s - \gamma_n \geq T_s - \frac{1}{2}\gamma_{n-1}$.

Property 3.5: The smallest n for which $\gamma_n \leq d_s$ is called the *maximum order* n_m for the given parametrization. It is

$$n_m = \begin{cases} \left\lceil \frac{\ln(\min(T_a, T_s)) - \ln(d_s)}{\ln(2)} \right\rceil, & \text{if } \min(T_a, T_s) > d_s, \\ 0, & \text{otherwise.} \end{cases} \tag{3.4}$$

Definition 3.1.2: Penalty σ: Let there be a γ-sequence with a shrinkage/growth γ_n. Then $\sigma_n = i_n \cdot T_a$ is called the *penalty* for increasing or decreasing any offset between a scan window and an advertising packet by γ_n time units.

Lemma 3.1. *Property 3.3 implies that* $\sigma_n = \sum_{k=0}^{n-1} |a_k| \cdot \sigma_k = i_n \cdot T_a$*, where* a_k *are equal to the coefficients in Property 3.3. Further,* $\sigma_n > \sigma_{n-1}$*, since* $i_n > i_{n-1}$.

Let us assume that the properties above hold true. We can then make use of them to solve the ND problem, as defined by Inequality 3.1, as follows.

1. First, $\gamma_0, \gamma_1, ..., \gamma_{n_m}$ need to be computed. This computation is described in detail in Section 3.4.3.
2. Next, the entire range of possible initial offsets is subdivided into partitions with equal latencies. For each partition, the latency can be derived by identifying a set of γ-sequences that leads to a coincidence of a packet and a corresponding scan window. This procedure is described in Section 3.4.5.
3. From the set of partitions with corresponding latencies, one can extract the global worst-case and mean latencies.

3.4.3 Computation of the γ - Parameter

This section is dedicated to the computation of γ_n. Property 3.3 implies that γ_n is defined by the absolute difference between T_s and the absolute value of a linear combination of lower-order γ-values, such that i_n (and hence σ_n) is minimized, under the constraint that $\gamma_n < \frac{1}{2}\gamma_{n-1}$. We in the following present a recursive scheme to compute γ_n given $\gamma_k,\ k \in 0..n-1$. First, we present the computation of the initial value γ_0 of this recursion.

Initial Values

For realizing a shrinkage or growth of γ_n time units every i_n advertising packets and j_n scan windows, the amount of time defined by $i_n \cdot T_a$ must be by γ_n time units shorter (for a shrinkage) or longer (for a growth) than $j_n \cdot T_s$. Hence, we have to identify the smallest i_n and a suitable j_n, such that $|i_n T_a - j_n T_s| = \gamma_n < \min(T_a, T_s)$ (cf. Property 3.2). If $T_a < T_s$, every single advertising packet is part of a growing γ-sequence with $i_0 = 1, j_0 = 0$, and it is $\gamma_0 = T_a$. If $T_a > T_s$, the γ-sequence of order 0 implies $i_n = 1$, since this is the lowest possible multiple of T_a. There are two possible values for j_n, such that Property 3.2 (i.e., $\gamma_0 < T_s$) is fulfilled. The first one is defined by the smallest multiple of T_s that exceeds T_a, which forms a shrinking sequence with $\gamma_{s,0}$. The other one is defined by the largest multiple of T_s that does not exceed T_a, which forms a growing Sequence with $\gamma_{g,0}$. It is

$$\gamma_{s,0} = \left\lceil \frac{T_a}{T_s} \right\rceil \cdot T_s - T_a,\ \gamma_{g,0} = T_a - \left\lfloor \frac{T_a}{T_s} \right\rfloor \cdot T_s. \tag{3.5}$$

Though both $\gamma_{s,0}$ and $\gamma_{g,0}$ are valid values for γ_0, we choose $\gamma_0 = \min(\gamma_{g,0}, \gamma_{s,0})$, and set $m_0 = s$ or g, accordingly, because this will result in a lower highest order n_m. Next, we describe how to compute γ_n with $n > 0$.

Recursion Scheme

We in the following assume two sequences with $m_{n-1} = m_n = g$, as depicted in Figure 3.4, and restrict our description to this. We generalize it afterwards.

Let γ_{n-1} describe a certain offset-shrinkage or -growth (cf. Property 3.1). Since this shrinkage or growth applies for the offset between any pair of scan windows and advertising packets, also offsets that result from a previous shrinkage/growth are again shrunk or grown by γ_{n-1} time units, whenever the additional advertising and scan intervals pass. Hence, linear combinations $\upsilon_{\gamma_n} = \sum_{k=0}^{n} a_k \gamma_k$, with $a_k \geq 0$, if $m_k = g$ and $a_k < 0$, if $m_k = s$, will result in an effective growth or shrinkage of υ_{γ_n} time units. Further, if $\upsilon_{\gamma_n} > 0$, we can form differences $\upsilon_{\gamma_n} - T_s$ and obtain an effective shrinkage or growth $\gamma'_n = |\upsilon_{\gamma_n} - T_s|$. If υ_{γ_n} involves i_n advertising intervals and j_n scan intervals, γ'_n can be regarded as the shrinkage (if $\upsilon_{\gamma_n} - T_s < 0$) or growth (if $\upsilon_{\gamma_n} - T_s > 0$) after every i_n advertising intervals and $j_n + 1$ scan intervals. Similarly, if $\upsilon_{\gamma_n} < 0$, $\gamma'_n = \upsilon_{\gamma_n} + T_s$ is the shrinkage (if $\upsilon_{\gamma_n} < -T_s$) or growth (if $\upsilon_{\gamma_n} > -T_s$) after i_n advertising and $j_n + 1$ scan intervals.

We can use this to construct $\gamma_n = |T_s - |\sum_{k=0}^{n-1} a_k \cdot \gamma_k||$, $a_k \in \mathbb{Z} \setminus 0$, by choosing a_k such that $|\upsilon_{\gamma_n}| = |\sum_{k=0}^{n-1} a_k \cdot \gamma_k|$ differs by less than $\frac{1}{2} \cdot \gamma_{n-1}$ time-units from T_s and hence, $\gamma_n < \frac{1}{2}\gamma_{n-1}$. There are different choices of a_k that fulfill $\gamma_n < \frac{1}{2} \cdot \gamma_{n-1}$, and we have to identify the one that leads to the lowest penalty $\sigma_n = \sum_{k=0}^{n-1} |a_k| \cdot \sigma_k$. Property 3.4 implies that shrinking or growing any offset by γ_n time-units always incurs at least the penalty

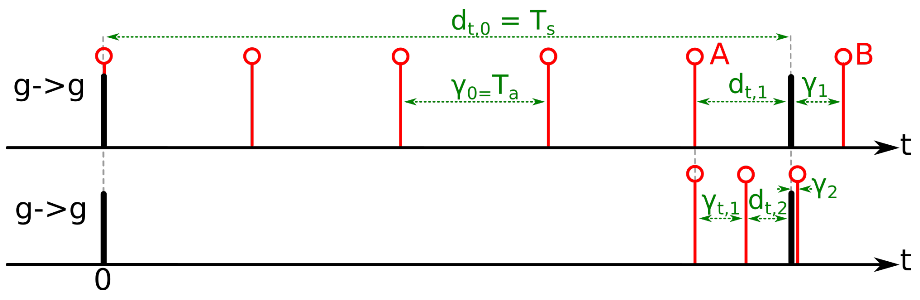


Figure 3.4: Computation of γ_2 given γ_1 for growing sequences.

σ_n of the linear combination $|\sum_{k=0}^{n-1} a_k\gamma_k|$ that i) reaches or exceeds $T_s - \frac{1}{2}\gamma_n$, and ii) has the lowest possible corresponding sum $\sum_{k=0}^{n-1} |a_k|\sigma_k$. Since we attempt to find the "best" (in terms of penalty) linear combination $|\sum_{k=0}^{n-1} a_k\gamma_k| > T_s - \frac{1}{2}\gamma_{n-1}$, Property 3.4 implies that one instance of γ_{n-1} in such a linear combination incurs already the penalty of the "best" linear combination that contains no instance of γ_{n-1}, if such a combination exists. As a result, coefficients a_k for higher sequence orders k always need to be minimized at the cost of larger coefficients a_k for lower orders k.

We now study all different constellations, which are given by the mode m_n of the current and m_{n+1} of the next sequence order.

Growing → Growing: In the example depicted in Figure 3.4, γ_1 and γ_2 are computed given γ_0. Here, $m_1 = m_2 = g$, and therefore $\upsilon_{\gamma_{n-1}}$ must always exceed T_s. Hence, it is $\gamma_n = \upsilon_{\gamma_{n-1}} - T_s$. For computing γ_1, let us find the lowest multiple Q_0 of γ_0 time units, such that $(Q_0 + 1) \cdot \gamma_0$ exceeds T_s. We can regard this as finding the number of γ_0 time-intervals that "fit" into a certain *distance to travel* $d_{t,0} = T_s$, i.e., $Q_0 = \lfloor \frac{d_{t,0}}{\gamma_0} \rfloor$. As can be seen in Figure 3.4, $\gamma_1 = (Q_0 + 1)\gamma_0 - T_s$, since this is the linear combination with the lowest sum of penalties (here: $(Q_0 + 1) \cdot \sigma_0$) that fulfills $\gamma_1 \leq \frac{1}{2}\gamma_0$.

For computing γ_2, we consider the largest multiple Q_0 of γ_0-intervals that does not exceed T_s. The remaining difference from T_s is $d_{t,1} = d_{t,0} - Q_0\gamma_0$. We now identify the largest multiple of γ_1 time units that "fit" into $d_{t,1}$, i.e., $Q_1 = \lfloor \frac{d_{t,1}}{\gamma_1} \rfloor$. Using this, $\gamma_2 = Q_0\gamma_0 + (Q_1 + 1)\gamma_1 - T_s$.
Any linear combination that involves a higher number of γ_1 time units than Q_1 would violate Property 3.4. This scheme can be

Quick Reference of Symbols

γ	Growth/Shrinkage per interval
m	Mode of a sequence (s ← shrinking, g ← growing, c ← coupling).
d_t	Distance left to travel
σ, σ_s	Penalty related to γ / Penalty related to d_t
Q	Number of γ intervals that fit into d_t.

A full table of symbols is given in Appendix A.

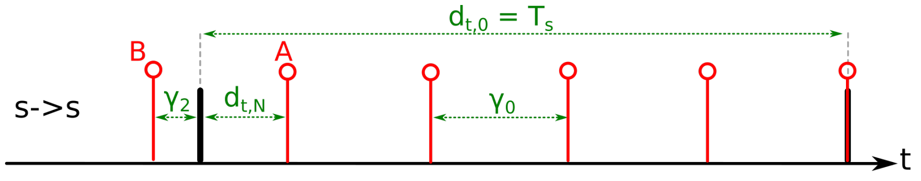


Figure 3.5: Computation of γ_1 given γ_0 for $m_0 = m_1 = s$.

generalized for arbitrary orders n. With $Q_n = \lfloor \frac{d_{t,n}}{\gamma_n} \rfloor$, it is

$$\begin{aligned} \gamma_n &= (Q_{n-1} + 1) \cdot \gamma_{n-1} - d_{t,n-1}, \\ d_{t,n} &= d_{t,n-1} - Q_{n-1} \cdot \gamma_{n-1}, \\ \sigma_n &= \sigma_{s,n-1} + (Q_{n-1} + 1) \cdot \sigma_{n-1}, \\ \sigma_{s,n} &= \sigma_{s,n-1} + Q_{n-1} \cdot \sigma_{n-1}. \end{aligned} \tag{3.6}$$

The first distance to travel $d_{t,0}$ is initialized by T_s, $\sigma_{s,0}$ is initialized by 0 and σ_0 by T_a. Here, $\sigma_{s,n}$ is the penalty (i.e., sum of advertising intervals) for any linear combination $\sum_{k=0}^{n-1} a_k \gamma_k$ that is by $d_{t,n}$ smaller than T_s.

In fact, one can show that Equation 3.6 holds true whenever $m_n = m_{n-1}$ (i.e., also for $m_n = s = m_{n-1}$). Similar considerations also exist for $m_n \neq m_{n-1}$, which lead to slightly different equations, as described below. Therefore, depending on m_n and m_{n-1}, one out of two sets of equations has to be applied. This requires the mode m_n, which can be computed given γ_{n-1}, m_{n-1} and d_{t,n_1}, as defined in Table 3.2.

Shrinking → Shrinking: A situation with $m_0 = m_1 = s$ is shown in Figure 3.6. For $m_0 = m_1 = g$, to compute γ_1, we have formed the linear combination of γ_0-values that exceeds T_s by no more than $\frac{1}{2}\gamma_0$ with the smallest possible corresponding sum $\sum_{k=0}^{n-1} a_k \cdot \sigma_k$. Here, since $m_0 = s$, the coefficient a_0 of the linear combination is negative and hence we form the negative combination of γ_0 time-units for which the absolute value exceeds T_s by no more than $\frac{1}{2}\gamma_0$ time-units. This leads exactly to the same scheme as given by Equation 3.6.

Growing → Shrinking and Shrinking → Growing: Figure 3.6 depicts a situation in which $m_0 = g$, $m_1 = s$ and $m_2 = g$. When comparing this to the situation in the Figures 3.4 and 3.5, it becomes clear that when computing γ_n, a mode-change $m_{n-1} \neq m_n$ incurs one instance of γ_{n-1} fewer than for $m_n = m_{n-1}$. This leads to the following Equations, which hold true for $m_n = s$, $m_{n+1} = g$ and $m_n = g$, $m_{n+1} = s$.

$d_{t,n-1} - \lfloor \frac{d_{t,n-1}}{\gamma_{n-1}} \rfloor$	$< 1/2 \cdot \gamma_{n-1}$	$> 1/2 \cdot \gamma_{n-1}$	$= 1/2 \cdot \gamma_{n-1}$
$m_{n-1} = s$	g	s	c
$m_{n-1} = g$	s	g	c
$m_{n-1} = c$	-	-	-

Table 3.2: Recursive definition of the sequence mode m_n given m_{n-1}

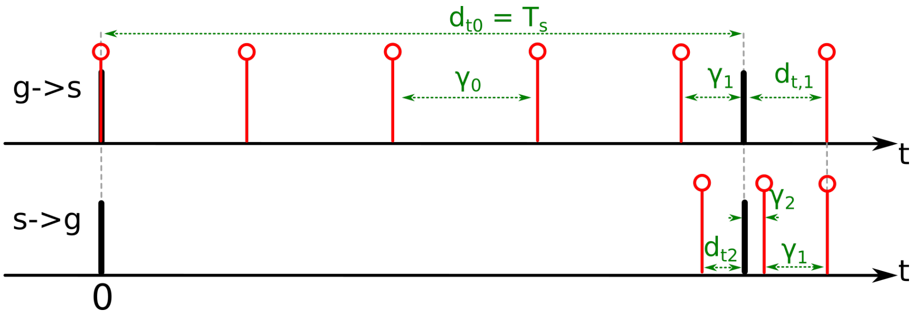


Figure 3.6: Computation of γ_1 given γ_0 for $m_0 \neq m_1 \neq m_2$.

$$\begin{aligned}
\gamma_n &= d_{t,n-1} - Q_{n-1} \cdot \gamma_{n-1}.\\
d_{t,n} &= (Q_{n-1}+1) \cdot \gamma_{n-1} - d_{t,n-1}.\\
\sigma_n &= \sigma_{s,n-1} + Q_{n-1} \cdot \sigma_{n-1}.\\
\sigma_{s,n} &= \sigma_{s,n-1} + (Q_{n-1}+1) \cdot \sigma_{n-1}.
\end{aligned} \tag{3.7}$$

3.4.4 Relation to the Euclidean Algorithm

Property 3.3 defines γ_n as $\gamma_n = |T_s - \upsilon_{\gamma_{n-1}}|$, where $\upsilon_{\gamma_{n-1}}$ is a linear combination $\sum_{k=0}^{n-1} a_k \cdot \gamma_k$. Let us consider two possibilities for $\upsilon_{\gamma_{n-1}}$, both assuming $m_0 = g$. The one, υ_s, is the linear combination $\sum_{k=0}^{n-1} a_k \gamma_k$ that approximates T_s as close as possible, without exceeding it. The other one, $\upsilon_g = \upsilon_1 + \gamma_{n-1}$ slightly exceeds T_s, and $\min(|T_s - \upsilon_1|, |T_s - \upsilon_2|) < \frac{1}{2}\gamma_{n-1}$. If $\upsilon_s < \upsilon_g$, $m_n = s$ and $\gamma_n = T_s - \upsilon_s$. Otherwise, $m_n = g$ and $\gamma_n = \upsilon_g - T_s$. This shares similarities with the method of least absolute remainders, which the mathematician Leopold Kronecker has shown to be the most efficient version of the Euclidean algorithm [57]. For determining the next remainder (here: $|T_s - \upsilon_{\gamma_{n-1}}|$), rather than dividing the smallest previous reminder by its predecessor (as in the Euclidean algorithm), we divide the larger previous reminder (i.e., $d_{t,n-1}$) by the smaller one (i.e. γ_{n-1}) in our recursive equations.

3.4.5 Latency Computation

The first packet sent by a device that comes into the range of a second, scanning device might fall into any part of its scan interval with an equal probability. Hence, the valid range of initial offsets between a scan window and an advertising packet is restricted to $\Phi[0] \in [0, T_s]$. For computing the discovery latency for a given $\Phi[0]$, we have to identify the linear combination $\upsilon = \sum_{k=0}^{n_m} a_k \cdot \gamma_k$ that either fulfills $-d_s \leq \Phi[0] + \upsilon \leq 0$ (i.e., offset + linear combination ends in a scan window temporally on the left), or $T_s - d_s \leq \Phi[0] + \upsilon \leq T_s$ (i.e., offset + linear combination ends in a scan window temporally on the right), and at the same time minimizes the corresponding sum of penalties $\sum_{k=0}^{n_m} |a_k| \cdot \sigma_n$. Minimizing the sum of penalties is required, because a linear combination with a higher sum of penalties would relate to a later coincidence of an advertising packet and a scan window, whereas the discovery latency is given by the first rendezvous. The intuition behind the latency computation procedure, along with an example, is given here. It is followed by a full algorithm in Section 3.6.

As already explained, a γ-sequence incurs a shrinkage or growth of the temporal offset ϕ between any scan window and any advertising packet after every i_n advertising and j_n scan intervals. Equivalently, one could describe this as *one* advertising packet being "shifted" by multiples of γ_n towards the left (if $m_n = s$) or the right (if $m_n = g$) within *one* scan-interval. By summing up the penalties σ_n of every "shift" of length γ_n, one can account for the fact that each shrinkage or growth actually occurs for later pairs of scan windows and advertising packets. With this representation, the problem of finding such linear combinations υ can be solved by finding the "best" (in terms of penalties) linear combination of "shifts" of length γ_n, until the advertising packet reaches either the scan window on the left or on the right of it. Again, coefficients a_k for high orders k always have to be minimized at the cost of lower-order ones, as proven in Section 3.5.

With these considerations, the latency computation can be carried out as follows.

1. First, we create a set of partitions, such that the corresponding scan window (i.e., the left one for $m_n = s$ or the right one for $m_n = g$) is either reached or "overtaken" after the same multiple of γ_0 time units from within each partition.

2. Each such partition is then subdivided into two other partitions. One of them contains all initial offsets from which the scan window can be reached (without "overtaking") after a multiple of γ_n time units. We call them *complete* partitions, since they do not need to be processed further, because discovery occurs after the corresponding multiple of γ_0 time units. The other ones contain the remaining offsets.

3. The partitions with the remaining offsets need to be sub-partitioned again, such that after the same number of γ_1 time units per partition, either the closest scan window or complete partition on the left (if $m_1 = s$) or right (if $m_1 = g$ is reached) or "overtaken".

4. Steps 2) and 3) are repeated for higher orders n, until $\gamma_n < d_s$.

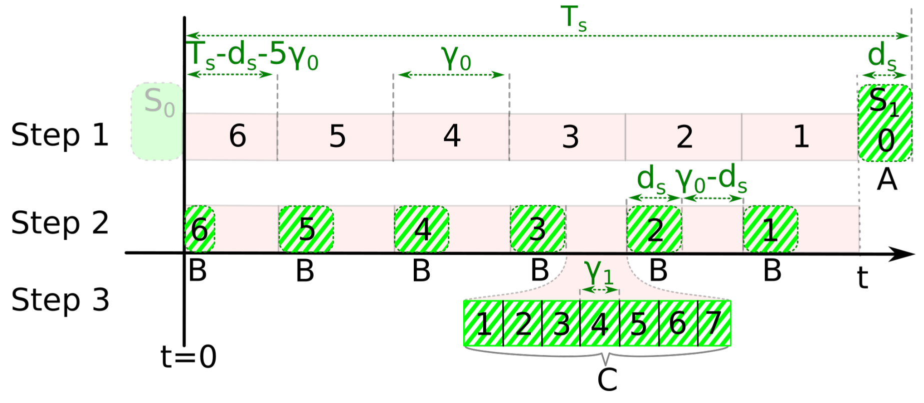


Figure 3.7: Computing the mean discovery latency for a higher order sequence.

In the following, we illustrate this scheme using an example. Figure 3.7 exemplifies a partitioning of the entire range of initial offsets for a growing sequence with parameter γ_0

and a shrinking sequence with γ_1. In Figure 3.7, the hatched rounded rectangles depict the complete partitions, whereas the remaining rectangles depict temporary partitioning steps that need to be further subdivided later. In **Step 1** (cf. Figure 3.7), we subdivide the entire range $[0, T_s]$ into partitions of possible initial offsets, from which the scan window S_1 can be reached or "overtaken" using same multiple of γ_0 time units (i.e., the number of γ_0-intervals represented by the digits in each rectangle in Figure 3.7). A special case of this partitioning scheme is the scan window S_1 itself (cf. **A** in Figure 3.7)), since an an advertising packet falling into this scan window has zero latency.

In **Step 2**, each of these partitions is further subdivided into parts from which a multiple of γ_1 time units can reach the scan window S_1 (i.e., the hatched boxes in Figure 3.7), and the remaining parts, from which S_1 would be "overtaken". For the hatched parts denoted with **B** in the figure, no further partitioning is needed, since the discovery latency for an initial offset falling into such a part is defined by the number of γ_0-intervals until reaching the scan window S_1. These partitions are called *complete.*

However, the non-hatched parts from Step 2, which would "overtake" S_1, need to be further subdivided in **Step 3**. Each of them is subdivided into multiple partitions of length γ_1 (except for the leftmost one, which is shortened to fit within the $\gamma_0 - d_s$ time units of the containing partition obtained from Step 2). These partitions are marked with **C** in the figure. In each of these partitions, the number of γ_1 - steps a packet has to be "shifted" by to reach its closest complete partition (or S_0 from the leftmost partition, respectively) is equal. Once a complete partition has been reached (or S_0 from the leftmost partition), S_1 is approached in steps of γ_0, as described before. Since $\gamma_1 < d_s$, no parts of these partitions can "overtake" the corresponding partition of type **B** (or the scan-window S_0, respectively). Therefore, the partitioning process is complete.

The discovery latency of any partition can be computed by summing up the appropriate penalties. For every partition, we can count the number of γ_n - intervals until reaching the scan-window (S_1 in Figure 3.7). For example, from a partition of type **C**, we count the number of γ_1-intervals n_1 until reaching a partition of type **B**. From there, we count the number of γ_0-intervals n_0 until reaching the scan window S_1. The digits in each partition in Figure 3.7 correspond to n_0 and n_1. The discovery latency l is then $n_1 \cdot \sigma_1 + n_0 \cdot \sigma_0$.

This scheme, together with Property 3.5, also implies that a scan window can always be reached within a finite number of steps. Therefore, the worst-case latency is always bounded, if $m_n \neq c \; \forall n < n_m$.

3.5 Proofs of the Properties

In Section 3.4.2, we have defined γ-sequences and have introduced multiple properties, for which we claimed that they hold true for some (i_n, j_n). In this section, we proof that these properties can be fulfilled.

Property 3.1: Constant Shrinkage/Growth

Given an offset Φ between a scan window and an advertising packet, Property 3.1 implies that the offset Φ' that is j_n scan intervals and i_n advertising intervals later will be by $\gamma_n \geq 0$ time-units larger or smaller than Φ. Let us consider $i_n \cdot T_a$ and $j_n \cdot T_s$ time-units. Their difference is $|j_n \cdot T_s - i_n \cdot T_a| = \gamma_n$. Let t_s be an arbitrary point in time a scan window starts at, and t_a an arbitrary point in time an advertising packet is sent at, and $\Phi = t_s - t_a$. Then, for the offset Φ' that is i_n advertising intervals and j_n scan intervals later, it is

$$\begin{aligned} \Phi' &= t_s + j_n \cdot T_s - (t_a + i_n \cdot T_a) = \\ &= t_s - t_a + (j_n \cdot T_s - i_n \cdot T_a) = \\ &= \Phi \pm \gamma \ \forall (t_a, t_s). \end{aligned} \tag{3.8}$$

□

Property 3.2: Range of γ

Property 3.2 implies that there exist pairs (i_n, j_n), such that $0 \leq \gamma_n \leq \min(T_s, T_a)$. Let us first assume $T_s < T_a$. It is $\gamma = |j_n \cdot T_s - i_n \cdot T_a|$. We choose $j_n = \lceil \frac{i_n T_a}{T_s} \rceil$, and it is

$$\begin{aligned} \gamma_n &= \left| \left\lceil \frac{i_n T_a}{T_s} \right\rceil T_s - i_n T_a \right| \\ &\leq \left| \left(\frac{i_n \cdot T_a}{T_s} + 1 \right) T_s - i_n \cdot T_a \right| \\ &= |i_n T_a + T_s - i_n T_a| \\ &= T_s \end{aligned} \tag{3.9}$$

Similarly, if $T_s > T_a$, we can choose $j_n = \lceil \frac{i_n T_s}{T_a} \rceil$, which will result in $\gamma_n \leq T_a$. □

Property 3.3: Construction of γ

Property 3.3 claims that there are linear combinations $\gamma_n = |T_s - |\sum_{k=0}^{n-1} a_k \cdot \gamma_k||$, $a_k \in \mathbb{Z} \backslash 0$, which fulfill the following three properties.

- $\gamma_n \leq \frac{1}{2}\gamma_{n-1}$
- the sum $\sum_{k=0}^{n-1} |a_k| \cdot (i_k T_a)$ is minimized, and
- for a shrinking sequence, $a_k < 0$, otherwise, $a_k > 0$

A valid constructive algorithm for linear combinations that fulfill these properties has been presented in Section 3.6, which proofs the existence of such linear combinations. □

Property 3.4: Priority of Lower Orders

This property claims that for any γ_n, $\sigma_n = i_n \cdot T_a$ is always at least as large as the smallest possible linear combination $\sum_{k=0}^{n-1} |a_k| \cdot \sigma_k$, for which $|\sum_{k=0}^{n-1} a_k \gamma_k| \geq T_s - \gamma_n \geq T_s - \frac{1}{2}\gamma_{n-1}$.

The construction of γ_n is defined by $\gamma_n = |T_s - |\sum_{k=0}^{n-1} a_k \cdot \gamma_k||$, $a_k \in \mathbb{Z} \setminus 0$, such that i) $\gamma_n < \frac{1}{2}\gamma_{n-1}$ and ii) the corresponding penalty σ_n is minimized. The necessary condition $\gamma_n = |T_s - |\sum_{k=0}^{n-1} a_k \cdot \gamma_k|| < \frac{1}{2}\gamma_{n-1}$ implies that $|\sum_{k=0}^{n-1} a_k \cdot \gamma_k| \geq T_s - \frac{1}{2} \cdot \gamma_{n-1}$. This sum has a corresponding sum of penalties $\sum_{k=0}^{n-1} |a_k| \cdot \sigma_k$. Since $\gamma_n = |T_s - |\sum_{k=0}^{n-1} a_k \cdot \gamma_k||$, it is $|\sum_{k=0}^{n-1} a_k \cdot \gamma_k| \geq T_s - \gamma_n$. □

Property 3.5: Highest Possible Order

The maximum number of iterations of Algorithm 3.1 is identical to the maximum order n_m. In the following, we proof that the maximum order is bounded, as described by Property 3.5.

Property 3.3 implies that $\gamma_n \leq \frac{1}{2}\gamma_{n-1}$. Hence, in the worst-case, $\gamma_n = \frac{1}{2}\gamma_{n-1}\ \forall n$. Using this, we can define a worst-case γ-parameter for the order n, $\gamma_{wc,n}$, by assuming a maximum initial value γ_0 of $\min(T_a, T_s)$ (cf. Property 3.2). It is:

$$\gamma_{n,wc} = \left(\frac{1}{2}\right)^n \cdot \min(T_a, T_s). \tag{3.10}$$

In the worst-case, the highest-order sequence is the first one for which $\gamma_{n,wc} \leq d_s$ holds true. Hence, it is

$$\left(\frac{1}{2}\right)^{n_m} \cdot \min(T_a, T_s) \leq d_s. \tag{3.11}$$

Solving this inequality leads to a maximum order n_m, as given by Equation 3.12.

$$n_m = \left\lceil \frac{\ln(\min(T_a, T_s)) - \ln(d_s)}{\ln(2)} \right\rceil \tag{3.12}$$

In addition, it is intuitively clear that if $\min(T_a, T_s) \leq d_s$, only order-0-sequences can occur (the advertiser can never “overtake” the scanner) and therefore, $n_m = 0$. Further, we have assumed d_s to be shortened by ω time-units, as described in Section 3.4, and the equation above is only valid for shortened values of d_s.

The limited number of sequence orders also implies that whenever $\gamma_n \neq 0\ \forall n \leq n_m$, the worst-case discovery latency is bounded. This is because the discovery latency is defined by the linear combination of γ-intervals that “fits” into the scan interval, which is always finite if $\gamma \neq 0$.

□

Proof of Latency Computation

In Section 3.4, we have formed linear combinations $\upsilon = \sum_{k=0}^{n} a_k \cdot \gamma_k$, such that a packet that is sent with a certain initial offset $\Phi[0]$ from the first scan window is shifted by υ towards the next (if $m_n = g$) or previous (if $m_n = s$) scan window. We have claimed that coefficients a_k for high values of k always need to be minimized, no matter how by how much coefficients a_k for lower values of k need to be increased to compensate for this minimization. Algorithm 3.1, which we will present in Section 3.6, will also rely on this. In what follows, we proof this claim.

Let us consider two linear combinations υ_1 and υ_2 that both fulfill either $-d_s \leq \Phi[0] + \upsilon \leq 0$ (i.e., a sequence that reaches a scan window temporally on the left), or $T_s - d_s \leq \Phi[0] + \upsilon \leq T_s$ (i.e., a sequence that reaches a scan window temporally on the right), respectively, and hence lead to a coincidence of the advertising packet and a scan window. Further, $\upsilon_1 = \sum_{k=0}^{n} a_{k,1} \cdot \gamma_k$, whereas $\upsilon_2 = \sum_{k=0}^{n-1} a_{k,2} \cdot \gamma_k$. The coefficients $a_{k,1}$ and $a_{k,2}$ are chosen such that the corresponding sum of penalties is minimized. We now study whether υ_1 or υ_2 has a lower sum of penalties.

If $m_{n-1} = m_n$, then $\sum_{k=0}^{n-1} a_{k,2} \cdot \gamma_k - \sum_{k=0}^{n-1} a_{k,1} \cdot \gamma_k \leq \gamma_{n-1}$, since $\gamma_n \leq \frac{1}{2}\gamma_{n-1}$. This means that by having $1 \cdot \gamma_n$ in υ_1, at most one instance of γ_{n-1} can be saved. Because $\sigma_n \geq \sigma_{n-1}$, υ_2 always has a lower sum of penalties than υ_1.

If $m_{n-1} \neq m_n$, the initial offset $\Phi[0]$ could potentially form a situation in which υ_1 is defined by $-1 \cdot \gamma_n$, and $\upsilon_2 = \sum_{k=0}^{n-1} a_k \gamma_k \geq T_s - d_s$ in the worst-case. In other words, there might be situations in which either $-1 \cdot \gamma_n$, or an appropriate number of γ_{n-1} time-units of length $T_s - d_s$ or longer, will lead to a match. Such a situation with $\gamma_{n-1} = g$ and $\gamma_n = s$ is depicted in Figure 3.8. We now study which of both possibilities υ_1, υ_2 has a lower sum of corresponding penalties.

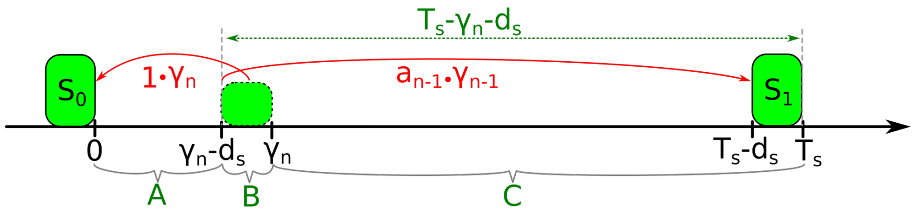


Figure 3.8: Proof for Latency Computation

Here, the range of initial offsets $\Phi[0]$ between $\gamma_n - d_s$ and γ_n (Area **B** in Figure 3.8) forms the only case in which the left scan window S_0 can be reached within $|-1| \cdot \gamma_n$ time-units. From Property 3.4 follows that the corresponding penalty $1 \cdot \sigma_n$ will be at least the lowest possible sum of penalties that corresponds to a linear combination $|\sum_{k=0}^{n-1} a_k \cdot \gamma_k| \geq T_s - \gamma_n$. However, as can be seen from the figure, the temporal distance from the right scan window S_1 to any point within area **B** is not larger than $T_s - d_s - \gamma_n < T_s - \gamma_n$ time-units. Hence, if there is a linear combination such that $\Phi[0] + \sum_{k=0}^{n-1} a_k \cdot \gamma_k$ lies within S_1, the penalty of σ_n will be at least as high as the sum of penalties $\sum_{k=0}^{n-1} |a_k| \cdot \sigma_k$. For the same reason, in area **C**, the latency of a suitable linear combination of $\sum_{k=0}^{n-1} a_k \cdot \gamma_k$ will be always lower than any linear combination that contains at least $1 \cdot \gamma_n$. Initial offsets $\Phi[0] < \gamma_n - d_s$ in area **A** can lead to a match either using a sequence $a_n \cdot \gamma_n + \sum_{k=0}^{n-1} a_k \gamma_k$, with $|a_n| \geq 1$ and $|a_k| > 0$ for at least some k, or by linear combinations without γ_n, i.e., $\sum_{k=0}^{n-1} a_k \cdot \gamma_k$. Clearly, $\sum_{k=0}^{n-1} a_k \cdot \gamma_k$ is the possibility with the lowest sum of penalties. $\square$

3.6 Implementation of the Model

An intuitive but computationally complex approach for computing the mean and worst-case ND latencies would be iterating over every single partition and computing the latencies for each of them, as outlined in the previous section. We will present an algorithm based on this scheme in Chapter 5. However, the computation can be done more efficiently, without carrying out the entire partitioning explicitly. Therefore, in what follows, we present an alternative, more efficient algorithm for computing the discovery latencies, which is based on two main concepts:

- Discrete summations: Instead of iterating over every single partition of a given sequence order, we account for multiple of them at once by appropriate discrete summations.
- Separation of sequence orders: Rather than carrying out the entire partitioning for all γ-sequences, we first only perform the partitioning for a single sequence order. After each order, the results are merged, since the resulting latency distributions are identical for many of the processed partitions. Therefore, the computational complexity of the next iteration is reduced.

We provide a ready-to-use MATLAB-implementation of this algorithm for download (cf. Section 3.7).

The algorithm we propose handles probability distributions that consist of multiple probability values over piecewise constant time intervals. For this purpose, we define a data structure called a *probability buffer* Γ. It is described next.

3.6.1 Probability Buffers

A *probability buffer* consists of a set of k probability densities $\Gamma[k]$, which are called *segments*. Each of them is defined over a unique, disjoint time-interval, which consists of a start time $t_s[k]$ and an end time $t_e[k]$. The elements are sorted by their starting times $t_s[k]$ in an ascending order. We define $\|\Gamma\|$ as the number of segments in the probability buffer Γ. For each probability buffer, we define an operation *add*(t_{ss}, t_{ee}, p), which creates a new segment from t_{ss} to t_{ee} with probability p. Existing probability segments that lie at least partially within $[t_{ss}, t_{ee}]$ are split and their probabilities are added for the overlapping time duration. To add a segment from t_{ss} to t_{ee} with probability p, we introduce the following notation: $[t_{ss}, t_{ee}] \leftarrow p$. Further, the term $\Gamma[k]$ represents the probability of the segment k.

3.6.2 Example

To provide the intuition behind our proposed algorithm, we first present an example for a ND problem that is solved by it, as shown in Figure 3.9. After that, we formally define this algorithm. As in the previous example from Figure 3.7, there is one growing sequence of order 0 and a shrinking sequence of order 1.

In **Iteration 1** of our proposed algorithm (upper part of Figure 3.9), the partitioning for the order-0 sequence, as already described, is carried out. Again, there are *hitting* partitions (1-6 in the figure), from which the scan window on the right (S_1) is reached after 1 to 6 steps of length γ_0. In addition, there are *missing* partitions (a-g), from which

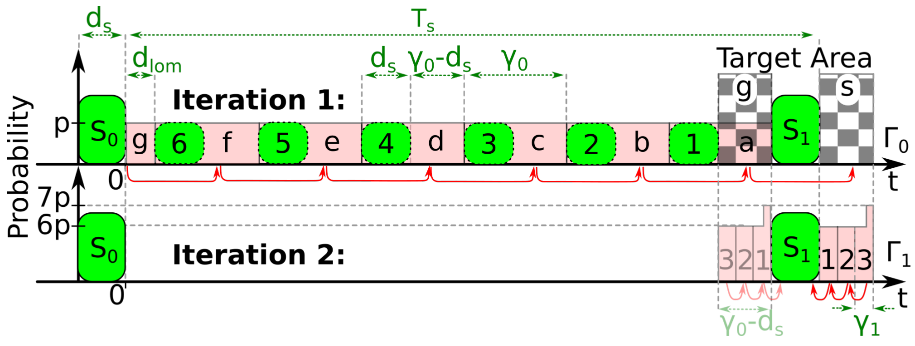


Figure 3.9: Solution scheme of our proposed algorithm.

the scan window S_1 is "overtaken" after 1 to 7 steps of length γ_0. In this example, each hitting partition has a length of d_s, whereas each missing partition has length $\gamma_0 - d_s$, except for partition g, which is shortened to fit within $T_s - d_s$.

Since the first advertising packet might fall into every part of $[0, T_s]$ with an equal probability, we assign a probability density of $p = {}^{1}/_{T_s}$ to every partition that is created in Iteration 1. We can compose the mean discovery latency $\bar{l}$ as a sum of multiple probability-weighted latencies $\sum_k l_p[k] \cdot p[k] \cdot (t_e[k] - t_s[k]) = \sum_k \overline{l_p[k]}$. Here, $l_p[k]$ is the discovery latency given the initial offset $\Phi[0]$ falls into a certain partition k and $p[k]$ is the corresponding probability. Similarly, $\overline{l_p[k]}$ is the latency $l_p[k]$ weighted by the corresponding probability density $p[k]$ (which has a value of $p = {}^{1}/_{T_s}$ in Iteration 1 of the algorithm for all values of k). We call such a probability-weighted latency a *partial latency* $\overline{l_p}$. Each partial latency corresponds to a certain partition type (e.g., to all hitting partitions from Iteration 1, etc.)

Quick Reference of Symbols

Γ	Probability buffer
$\|\Gamma\|$	Number of segments in Γ
$\Gamma[k]$	Probability density of segment $k \in \Gamma$
$t_s[k]$	Starting time of a segment $k \in \Gamma$
$t_e[k]$	Ending time of a segment $k \in \Gamma$
γ	Growth/Shrinkage per interval
$\vec{\gamma}$	Vector of γ-parameters for all orders
m	Mode of a sequence
σ, σ_s	Penalties
d_{lo}	Leftover duration
d_{loh}	Hitting duration of a leftover partition
d_{lom}	Missing duration of a leftover partition
$\overline{l_p}$	Weighted partial latency of a partition

A full table of symbols is given in Appendix A.

In the Example from Figure 3.9, all hitting partitions (1-6) in Iteration 1 will incur a partial latency of $\overline{l_p[0]} = d_s \cdot p \cdot \sum_{k=1}^{6} k \cdot \sigma_0$, with $p = \frac{1}{T_s}$, and the worst-case latency among all hitting parts is $6 \cdot \sigma_0$. An advertising packet that is sent within a missing partition (a-g) will "overtake" the scan window S_1 after the corresponding number of advertising intervals. After this overtaking has taken place, the subsequent advertising packet will be sent from one of the two *target areas* depicted in the figure, depending on the mode m_{n+1} of the next higher-order process. In our example, $m_{n+1} = s$, and the subsequent packet will be sent from target area s. This can be represented by "shifting" every missing partition by the appropriate multiple of γ_0 time-units, such that they lie within the target are s. Since multiple partitions are shifted into the same area, their probabilities have to be added. For the superposition of the partitions (a, b, c, d, e, f) in Figure 3.9, the resulting probability density will be $6 \cdot p$, since each of these partitions has a probability density of $p = \frac{1}{T_s}$. In addition, partition g will increase this merged probability density by $1 \cdot p$ within its corresponding part of the target area, as shown in the figure. The shifting procedure of all missing partitions into the target area s incurs a partial latency $\overline{l_p[1]} = p \cdot \sigma_0 \cdot ((\gamma_0 - d_s) \cdot (1 + 2 + 3 + 5 + 6) + d_{lom} \cdot 7)$. Here, d_{lom} is the length of partition g. Similarly, the worst-case latency for a packet being shifted to target area s is $7\sigma_0$.

The resulting probability distribution within target area s is shown in the lower part of Figure 3.9, which is represented by the probability buffer Γ_1. In **Iteration 2**, Γ_1 is subdivided into multiple partitions, such that S_1 is reached (from target area s) after the same multiple of γ_1 time-units. Again, a partial latency $\overline{l_p[2]}$ for reaching the scan window from from the sub-partitions (1,2,3) of Γ_1 (cf. lower part of Figure 3.9) can be computed. Here, $\gamma_1 \leq d_s$, and the computation is complete. In other cases, Γ_1 needs to be sub-partitioned into hitting and missing parts again, and the procedure repeats.

As can be seen from the Figure, Γ_1 can represent the entire probability distribution after Iteration 1 using only two segments (i.e., one within $[T_s, T_s + \gamma_0 - d_s - d_{lom}]$, and one within $[T_s + \gamma_0 - d_s - d_{lom}, T_s + \gamma_0 - d_s]$). In contrast, in the partitioning scheme described in Section 3.4.5, *every single* missing partition (e.g, all non-hatched partitions in Step 2 in Figure 3.7) need to be processed individually. Hence, this algorithm greatly reduces the computational complexity.

3.6.3 Algorithm Overview

A generic algorithm that corresponds to the described example is shown in Figure 3.10. In addition, a formal definition is given by Algorithm 3.1. This algorithm works as follows.

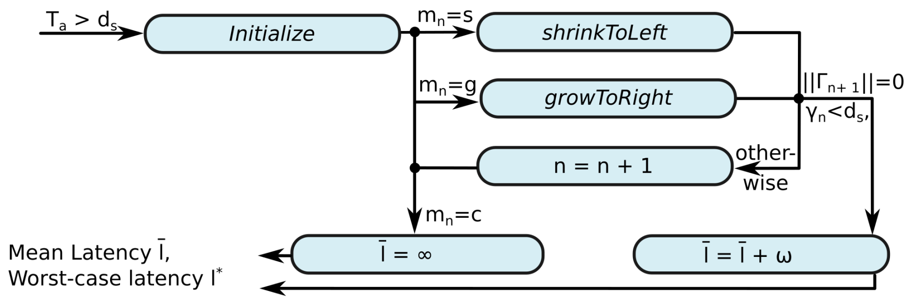


Figure 3.10: Algorithmic solution to assess the discovery latency.

First, the *Initialize()*-function is called, which initializes the mean ND latency $\bar{l}$ by 0. Further, an initial probability buffer Γ_0 is created. It contains one segment in $[0, T_s - d_s]$ with a probability density of $\frac{1}{T_s}$. The case of the first packet being sent within $[T_s - d_s, d_s]$ is accounted for implicitly by setting the initial probability density to $\frac{1}{T_s}$.

Algorithm 3.1 High-level algorithm for computing l

```
Require: $T_a$, $\omega$, $T_s$, $d_s$
Ensure: $\bar{l}$
 1: $(\bar{l}, \Gamma_0, \vec{\gamma}, \vec{\sigma}, \vec{m}) \leftarrow Initialize(T_a, T_s, d_s)$
 2: if $T_a \leq T_s$ then
 3:     $n \leftarrow 0$
 4: else
 5:     $n \leftarrow 1$
 6: end if
 7: while $\gamma_n \geq d_s$ do
 8:     if $m_n = g$ then
 9:         $(\bar{l}, \Gamma_{n+1}) \leftarrow growToRight(\bar{l}, \Gamma_n, \gamma_n, \sigma_n, m_{n+1})$
10:     else if $m_n = s$ then
11:         $(\bar{l}, \Gamma_{n+1}) \leftarrow shrinkToLeft(\bar{l}, \Gamma_n, \gamma_n, \sigma_n, m_{n+1})$
12:     else
13:         $\bar{l} \leftarrow \infty$; return
14:     end if
15:     if $(\|\Gamma_n\| = \emptyset) \vee (\gamma_n < d_s)$ then
16:         break loop
17:     else
18:         $n \leftarrow n + 1$
19:     end if
20: end while
21: $\bar{l} = \bar{l} + \omega$
```

Next, an iterative scheme begins. Depending on the mode m_n, either the *growToRight()*- or the *shrinkToLeft()*-function is called. If $m_n = c$, the algorithm is aborted with $\bar{l} \leftarrow \infty$, instead.

growToRight() performs the partitioning of its input probability distribution and the shifting to the appropriate target area, which results into the new probability buffer Γ_{n+1} and multiple probability-weighted partial latencies $\overline{l_p[k]}$. In addition to these latencies $\overline{l_p[k]}$, every partition has a largest sum of penalties $S = \sum \sigma$, which corresponds to the maximum amount of time until discovery takes place for any offset within this partition. The algorithm computes the largest sum of penalties S for every resulting partition as follows. If a partition to be processed has a largest sum of penalties of S_1, and the highest number of γ-intervals until reaching the next target area from this partition is n_i, then the resulting partition in Γ_{n+1} will have the largest sum of penalties $S_2 = S_1 + n_i \cdot \sigma_n$ assigned to it. When multiple partitions are merged within one target area, the maximum value of S among the merged partitions is always assigned to the resulting partition. If a partition is not part of a previously processed, lower-order partition, this recursion is initialized by 0. For the sake of simplicity of exposition, we only describe the computation of the weighted partial latencies $\overline{l_p}$ in detail. The computation of S works similarly. Whereas the *growToRight()*-function handles growing sequences, the *shrinkToLeft()*-function provides the equivalent functionality for shrinking ones.

After a call of *growToRight()* or *shrinkToLeft()*, one out of two different steps is performed:

1. If $\gamma_n < d_s$ or Γ_{m+1} is empty, then all existing partitions have been examined and $\bar{l}$ has been computed. The algorithm terminates. The number of iterations until this occurs is always bounded (cf. Property 3.5).

2. In all other cases, n is increased and the scheme is repeated by another iteration.

The mean packet-to-packet discovery latency is the sum of weighted partial latencies $\sum \bar{l} = E(d)$, and the worst-case latency l^* is the largest sum of penalties S in the last iteration of the algorithm. The values of γ_n, m_n and σ_n are computed at the beginning of every iteration. Finally, since we have assumed the packet length ω to be 0, we add ω to $\bar{l}$ and l_m to account for the last, successful advertising packet sent. Next, we describe the *growToRight()*-function detail.

3.6.4 *growToRight()*-Function

Based on a given probability buffer Γ_n, which either results from the *Initialize()* -function or from a previous iteration of the algorithm, the *growToRight()*-function iterates through all segments of Γ_n and processes each of them separately. This results into multiple partial latencies $\overline{l_p}$ and one resulting probability buffer Γ_{n+1}.

Quick Reference of Symbols

Γ	Probability buffer
$\Gamma[k]$	Probability density of segment $k \in \Gamma$
$t_s[k]$	Starting time of a segment $k \in \Gamma$
$t_e[k]$	Ending time of a segment $k \in \Gamma$
γ	Growth/Shrinkage per interval
m	Mode of a sequence
σ, σ_s	Penalties
N_l, N_u	Minimum/Maximum number of γ-intervals until reaching a target area for a segment of Γ
d_{Nl}, d_f, d_{Nu}	Length of the part of a segment that has $N_l - 1/N_l ... N_u/N_u + 1$ intervals to travel until reaching a target area

A full table of symbols is given in Appendix A.

We first split up each segment of the probability buffer Γ_n into multiple partitions with constant values of N_{sh}. N_{sh} denotes the number of steps of length γ_n until reaching the right scan window (S_1 in Figure 3.9) or the appropriate target area (target area s for $m_{n+1} = s$ or target area g for $m_{n+1} = g$), respectively. Next, every such partition is then subdivided into a hitting and a missing part. For every hitting and missing part, a partial latency $\overline{l_p}$ for reaching the scan window is computed. In addition, every missing part contributes to a new segment in Γ_{n+1}, which is obtained by shifting this missing part by $N_{sh} \cdot \gamma_n$ time-units. If $m_n + 1 = g$, for any given point in time $t \in \Gamma_n$, N_{sh} is given by $N_{sh} = \lfloor \frac{t_s - d_s - t}{\gamma_n} \rfloor$. For $m_{n+1} = s$, the next higher-order sequence shrinks and therefore the advertising packets will approach the temporally right scan window (S_1 in Figure 3.9) from the right target area (i.e., target area s in Figure 3.9). Further, for $m_{n+1} = s$, the temporally right scan window in Γ_n (i.e.,

S_1 in Figure 3.9) is regarded as the temporally left one in Γ_{n+1}. This implies that the coordinate system of Γ_{n+1} has its origin at the end of the temporally right scan window (S_1) of Γ_n. Therefore, in addition to shifting every missing partition by $N_{sh} = \lceil \frac{Ts-d_s-t}{\gamma_n} \rceil$ steps of γ_n time-units towards the right, the partition has to be transformed into the new coordinate system. The shifting and the coordinate transformation of every point in time t within Γ_n results into a point in time t' in Γ_n+1 of $t' = t - T_s + N_{sh}(t) \cdot \gamma_n$.

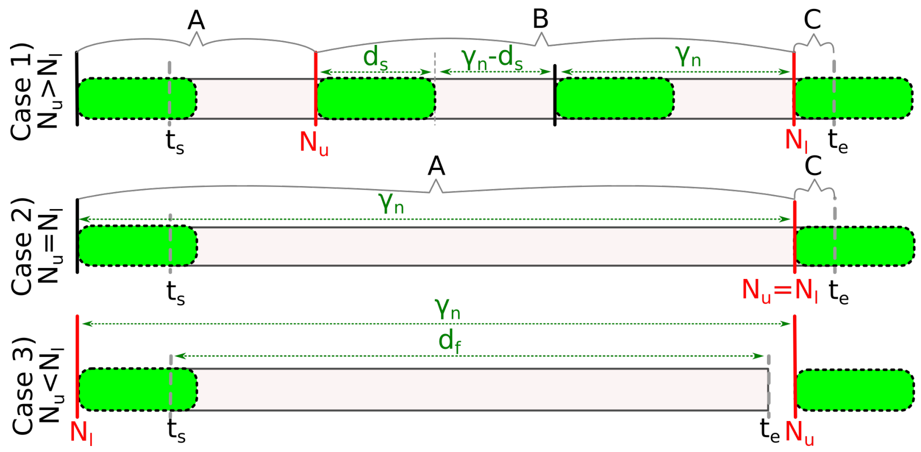


Figure 3.11: Processing of one probability buffer segment $\Gamma[k]$ for growing γ-sequences

In what follows, we formally define the procedure of processing a segment $k \in \Gamma_n$. In particular, based on multiple cases, we define the segments that need to be added to Γ_{n+1} and the partial latencies $\overline{l_p}$ need to be summed up to obtain the mean discovery latency. We first compute the minimum (N_l) and maximum number (N_u) of γ-intervals that fit into the distance from every point of this segment to the right scan window, as defined by Equation 3.13.

$$N_l = \left\lceil \frac{T_s - d_s - t_e[k]}{\gamma_n} \right\rceil, \; N_u = \left\lfloor \frac{T_s - d_s - t_s[k]}{\gamma_n} \right\rfloor. \tag{3.13}$$

Figure 3.11 depicts a segment of Γ_n and three possible cases for different values of N_l and N_u. The boxes with dotted rounded borders depict the hitting parts of this segment, whereas the remaining parts represent the missing ones. The figure also shows N_u and N_l and the borders t_s and t_e of the segment. We distinguish between 3 regions in each case: **A** (probability left of N_u), **B** (probability in between N_l and N_u) and **C** (probability right of N_l). The partial latency $\overline{l_p}$ of the considered segment $\Gamma[k]$ is the sum of all three partial latencies of these regions, viz., $\overline{l_A} + \overline{l_B} + \overline{l_C}$. Further, d_{Nu} is the temporal length of the part left of N_u, if there is any. Similarly, d_{Nl} is the part right of N_l and d_f is the part in between. These values can be computed as follows.

$$\begin{aligned} d_{Nu} &= (T_s - d_s - N_u\gamma_n) - t_s[k]. \\ d_f &= t_e[k] - t_s[k]. \\ d_{Nl} &= t_e[k] - (T_s - d_s - N_l\gamma_n). \end{aligned} \tag{3.14}$$

The segments added to Γ_{n+1} and the partial latencies depend on these values. There are 3 separate cases, as described below.

Case 1) and Case 2): $\mathbf{N_u \geq N_l}$
For $m_{n+1} = s$ one has the following partial latencies:

$$\begin{aligned}
\overline{l_A} &= \Gamma_n[k]\sigma_n d_{Nu}(N_u+1). \\
\overline{l_B} &= \begin{cases} 0, & \text{if} \quad N_u = N_l, \\ \frac{1}{2}\Gamma_n[k]\sigma_n\gamma_n \cdot (N_u - N_l)(N_u + N_l + 1), & \text{else.} \end{cases} \\
\overline{l_C} &= \Gamma_n[k]\sigma_n d_{Nl} N_l.
\end{aligned} \tag{3.15}$$

For $m_{n+1} = s$, the following segments are added to Γ_{n+1}:

$$\begin{aligned}
&[t_s[k] - T_s + N_u\gamma_n + \gamma_n, \gamma_n - d_s] \leftarrow \Gamma_n[k], && \text{if} \quad d_{Nu} < \gamma_n - d_s, \\
&[0, \gamma_n - d_s] \leftarrow \Gamma_n[k], && \text{else.} \\
&-, && \text{if} \quad N_u = N_l, \\
&[0, \gamma_n - d_s] \leftarrow \Gamma_n[k] \cdot (N_u - N_l), && \text{else.} \\
&-, && \text{if} \quad d_{Nl} < d_s, \\
&[0, t_e[k] - T_s + N_l\gamma_n] \leftarrow \Gamma_n[k], && \text{else.}
\end{aligned} \tag{3.16}$$

For $m_{n+1} = g$, the partial latencies are:

$$\begin{aligned}
\overline{l_A} &= \begin{cases} \Gamma_n[k]\sigma_n d_{Nu} N_u, & \text{if} \quad d_{Nu} < \gamma_n - d_s, \\ \Gamma_n[k]\sigma_n((N_u+1) \cdot (d_{Nu} - \gamma_n + d_s)) + (\gamma_n - d_s)N_u), & \text{else.} \end{cases} \\
\overline{l_B} &= \begin{cases} 0, & \text{if} \quad N_u = N_l, \\ \frac{1}{2}\Gamma_n[k]\sigma_n(N_u - N_l) \cdot (2d_s + \gamma_n \cdot (N_u + N_l - 1)), & \text{else.} \end{cases} \\
\overline{l_C} &= \begin{cases} \Gamma_n[k]\sigma_n d_{Nl} N_l, & \text{if} \quad d_{Nl} < d_s, \\ \Gamma_n[k]\sigma_n(d_s + d_{Nl} \cdot (N_l - 1)), & \text{else.} \end{cases}
\end{aligned} \tag{3.17}$$

For $m_{n+1} = g$, the following probabilities are added to Γ_{n+1}:

$$\begin{aligned}
&[t_s[k] + N_u\gamma_n, T_s - d_s] \leftarrow \Gamma_n[k], && \text{if} \quad d_{Nu} < \gamma_n - d_s, \\
&[T_s - \gamma_n, T_s - d_s] \leftarrow \Gamma_n[k], && \text{else.} \\
&-, && \text{if} \quad N_u = N_l, \\
&[T_s - \gamma_n, T_s - d_s] \leftarrow \Gamma_n[k] \cdot (N_u - N_l), && \text{else.} \\
&-, && \text{if} \quad d_{Nl} < d_s, \\
&[T_s - \gamma_n, t_e[k] + (N_l - 1)\gamma_n] \leftarrow \Gamma_n[k], && \text{else.}
\end{aligned} \tag{3.18}$$

If $\gamma_n < d_s$, no segments are added to Γ_{n+1} and the partial latency $\overline{l_p}$ can be simplified (both for $m_{n+1} = g$ and $m_{n+1} = s$) to $\overline{l_p} = \Gamma_n[k]\sigma_n \cdot (\frac{1}{2}d_{Nl}N_l\gamma_n(N_u - N_l) \cdot (N_u + N_l + 1) + d_{Nu}(N_u + 1))$.

Case 3): $\mathbf{N_u > N_l}$
From Figure 3.11 follows that this case has three subcases, depending on whether $t_s[k]$, $t_e[k]$ or both lie within an area for which the current sequence matches, or not. For $t_s[k] - (T_s - d_s - N_l\gamma_N) \leq d_s \wedge d_{Nl} \leq d_s$, both $t_s[k]$ and $t_e[k]$ lie within a matching area and therefore no probability is added to Γ_{n+1}. The latency $\overline{l_p}$ (both for $m_{n+1} = s$ and g) is $\Gamma_n[k]\sigma_n N_l(t_e[k] - t_s[k])$. For $t_s[k] - (T_s - d_s - N_l\gamma_N) \leq d_s \wedge d_{Nl} > d_s$, it is

$$\overline{l_p} = \begin{cases} \Gamma_n[k]\sigma_n N_l(t_e[k] - t_s[k]), & \text{if} \quad m_{n+1} = s, \\ \Gamma_n[k]\sigma_n(N_l(d_s - \gamma_n + d_{Nu}) + (N_l - 1)(d_{Nl} - d_s)), & \text{else,} \end{cases} \tag{3.19}$$

and the following segments are added to Γ_{n+1}:

$$\begin{aligned}
&[0, t_e[k] - T_s - N_l\gamma_n] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = s, \\
&[T_s - \gamma_n, t_e[k] + (N_l - 1)\gamma_n] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = g.
\end{aligned} \tag{3.20}$$

Finally, for $t_s[k] - (T_s - d_s - N_l\gamma_N) > d_s$, it is:

$$\begin{aligned}
&\overline{l_p} = \Gamma_n[k]\sigma_n N_l(t_e[k] - t_s[k]). && \\
&[t_s[k] - T_s + N_l\gamma_n, t_e[k] - T_s + N_l\gamma_n] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = s, \\
&[t_s[k] + N_u\gamma_n, t_e[k] + (N_l - 1)\gamma_n] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = g.
\end{aligned} \tag{3.21}$$

If $\gamma_n < d_s$, no segments are added to Γ_{n+1} and Case 3) can be simplified to

$$\overline{l_p} = \Gamma_n[k](t_e[k] - t_s[k])N_l\sigma_n \tag{3.22}$$

The worst-case latency is computed similarly to the mean latency by identifying the largest number of multiples of γ_n time-units until reaching either the appropriate target area (for missing partitions) or scan window (for hitting partitions) in each iteration.

Sequences with $m_n = s$ are processed by the *shrinkToLeft()*-function, as described in the next section.

3.6.5 *shrinkToLeft()*-Function

In Section 3.6.4, we have presented the *growToRight()*-function, which grows a given distance Φ towards the temporally right scan window. Whereas *growToRight()* only works for growing sequences (i.e., $m_n = g$), *shrinkToLeft()* does the equivalent computation for shrinking sequences with $m_n = s$ of any order n, thereby reducing Φ ins steps of γ_n towards the temporally left scan window.

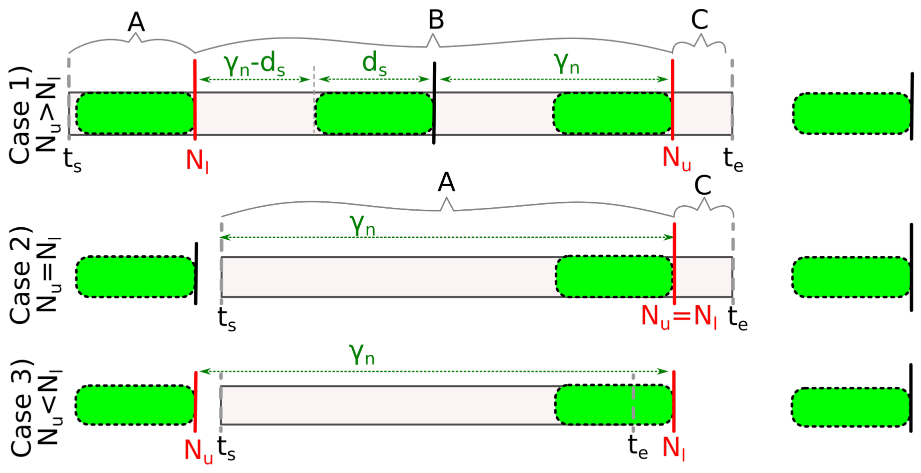


Figure 3.12: Processing of one probability buffer segment $\Gamma[k]$ for shrinking γ-sequences.

The partitioning of a segment for a shrinking sequence is depicted in Figure 3.12. The derivation of the equations is similar to the *growToRight()*-function and hence we only present the final results without further explanations. N_u and N_l are defined as

$$N_l = \left\lceil \frac{t_s[k]}{\gamma_n} \right\rceil, \quad N_u = \left\lfloor \frac{t_e[k]}{\gamma_n} \right\rfloor. \tag{3.23}$$

The variables d_{Nl}, d_{Nu} and d_f are, in analogy to *growToRight()*:

$$d_{Nl} = N_l\gamma_n - t_s[k], \quad d_{Nu} = t_e[k] - N_u\gamma_n, \quad d_f = t_e[k] - t_s[k]. \tag{3.24}$$

Like in *growToRight()*, there are three different cases.

Case 1) and Case 2): $\mathbf{N_u \geq N_l}$
For $m_{n+1} = s$, the partial latencies of the three parts **A**, **B** and **C**, as depicted in Figure 3.12, are defined as follows.

$$\begin{aligned}
\overline{l_A} &= \begin{cases} \Gamma_n[k]\sigma_n N_l d_{Nl}, & \text{if} \quad d_{Nl} < d_s, \\ \Gamma_n[k]\sigma_n(d_s N_l + (d_{Nl} - d_s)(N_l - 1)), & \text{else.} \end{cases} \\
\overline{l_B} &= \begin{cases} 0, & \text{if} \quad N_u = N_l, \\ \frac{1}{2}\Gamma_n[k]\sigma_n(N_u - N_l) \cdot (2d_s + \gamma_n(N_u + N_l - 1)), & \text{else.} \end{cases} \\
\overline{l_C} &= \begin{cases} \Gamma_n[k]\sigma_n N_u d_{Nu}, & \text{if} \quad N_l\gamma_n - t_e[k] \geq d_s, \\ \Gamma_n[k]\sigma_n((d_{Nl} - d_s)(N_l - 1) + (d_s - \gamma_n + d_{Nu})N_l), & \text{else.} \end{cases}
\end{aligned} \tag{3.25}$$

The following probability segments are added to Γ_{n+1} for $m_{n+1} = s$.

$$\begin{array}{ll}
-, & \text{if} \quad d_{Nl} < d_s, \\
[\gamma_n - d_{Nl}, \gamma_n - d_s] \leftarrow \Gamma_n[k], & \text{else.} \\
-, & \text{if} \quad N_u = N_l, \\
[0, \gamma_n - d_s] \leftarrow \Gamma_n[k] \cdot (N_u - N_l), & \text{else.} \\
[0, d_{Nu}] \leftarrow \Gamma_n[k], & \text{if} \quad d_{Nu} < \gamma_n - d_s, \\
[0, \gamma_n - d_s] \leftarrow \Gamma_n[k], & \text{else.}
\end{array} \tag{3.26}$$

For $m_{n+1} = g$, the following partial latencies $\overline{l_p}$ exist:

$$\begin{aligned}
\overline{l_A} &= \Gamma_n[k]\sigma_n N_l d_{Nl}. \\
\overline{l_B} &= \begin{cases} 0, & \text{if} \quad N_u = N_l, \\ \frac{1}{2}\Gamma_n[k]\sigma_n\gamma_n(N_u - N_l) \cdot (N_l + N_u + 1), & \text{else.} \end{cases} \\
\overline{l_C} &= \Gamma_n[k]\sigma_n(N_u + 1)d_{Nu}.
\end{aligned} \tag{3.27}$$

The following probability segments are added to Γ_{n+1} for $m_{n+1} = g$:

$$\begin{array}{ll}
-, & \text{if} \quad d_{Nl} < d_s, \\
[T_s - d_{Nl}, T_s - d_s] \leftarrow \Gamma_n[k], & \text{else.} \\
-, & \text{if} \quad N_u = N_l, \\
[T_s - \gamma_n, T_s - d_s] \leftarrow \Gamma_n[k] \cdot (N_u - N_l), & \text{else.} \\
[T_s - \gamma_n, T_s - \gamma_n + d_{Nu}] \leftarrow \Gamma_n[k], & \text{if} \quad d_{Nu} < \gamma_n - d_s, \\
[T_s - \gamma_n, T_s - d_s] \leftarrow \Gamma_n[k], & \text{else.}
\end{array} \tag{3.28}$$

If $\gamma_n < d_s$, then no probability is added to the resulting probability buffer and the partial latency contribution added to $\bar{l}$ can be computed by

$$\overline{l_p} = \Gamma_n[k]\sigma_n \cdot (\frac{1}{2} d_{Nl} N_l \gamma_n (N_u - N_l) \cdot (N_u + N_l + 1) + d_{Nu}(N_u + 1)). \tag{3.29}$$

Case 3): $\mathbf{N_u < N_l}$
Like in *growToRight()*, this case has three subcases. If $N_l\gamma_n - t_e[k] \geq d_s$ holds true, the partial latency is (for both $m_{n+1} = s$ and $m_{n+1} = g$)

$$\overline{l_p} = \Gamma_n[k] N_l \sigma_n d_f. \tag{3.30}$$

The following segments are added to Γ_{n+1}:

$$\begin{aligned} &[\gamma_n - d_{Nl}, d_{Nu}] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = s, \\ &[T_s - d_{Nl}, T_s + d_{Nu} - \gamma_n] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = g. \end{aligned} \tag{3.31}$$

For $N_l\gamma_n - t_e[k] < d_s \wedge d_{Nl} \geq d_s$, it is

$$\overline{l_p} = \begin{cases} \Gamma_n[k]\sigma_n((d_{Nl} - d_s)(N_l - 1) + N_l(d_s - \gamma_n + d_{Nu})), & \text{if} \quad m_{n+1} = s, \\ \Gamma_n[k]\sigma_n N_l (t_e[k] - t_s[k]), & \text{if} \quad m_{n+1} = g. \end{cases} \tag{3.32}$$

The following segments are added to Γ_{n+1}:

$$\begin{aligned} &[\gamma_n - d_{Nl}, \gamma_n - d_s] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = s, \\ &[T_s - d_{Nl}, T_s - d_s] \leftarrow \Gamma_n[k], && \text{if} \quad m_{n+1} = g. \end{aligned} \tag{3.33}$$

If $N_l\gamma_n - t_e[k] < d_s \wedge d_{Nl} < d_s$ holds true or if $\gamma_n \leq d_s$, no probability is added to the resulting buffer and the partial latency is $\overline{l_p} = \Gamma_n[k]\sigma_n N_l d_f$.

3.7 Software Library

An implementation of the model presented can be downloaded as a MATLAB library from `www.rcs.ei.tum.de/wireless_systems`. Further, it can also be downloaded or executed online using a web browser on `ieeexplore.ieee.org/document/8003429`. The function

```
function [dAvg, dMin, dMax, o] = getDiscoveryLatency(Ta,Ts,dA,ds,iL)
```

computes the mean- (*dAvg*), minimum- (*dMin*) and maximum- (*dMax*) discovery latency for a given parametrization. The parameters of this function are the advertising interval (*Ta*), the scan interval (*Ts*), the scan window (*ds*), the advertising packet-length (*dA*) and an iteration limit (*iL*) of the algorithm. The algorithm also returns the maximum sequence order (*o*) for the given parameters. An example for computing entire latency curves with sweeping parameters is also available for download.

3.8 Analysis and Evaluation

To evaluate the validity of our proposed theory, we compare its predicted latencies to measured latencies of $29,000$ discovery procedures, which have been carried out by two wireless radios. In addition, to cover a larger fraction of the whole parameter space, we have simulated $81,760,000$ discovery procedures and compare them to the values predicted by our theory. Towards this, we have carried out 7 different experiments. In Experiment a), we have measured latencies for sweeping advertising intervals and the fixed parameter values $T_s = 2.42\,\mathrm{s}, d_s = 0.59\,\mathrm{s}$. In addition, we have simulated the latencies for sweeping values of T_a, while T_s and d_s have been set to the following values: b) $T_s = 2.56\,\mathrm{s}, d_s = 0.32\,\mathrm{s}$; c) $T_s = 2.56\,\mathrm{s}, d_s = 0.64\,\mathrm{s}$; d) $T_s = 7.68\,\mathrm{s}, d_s = 0.32\,\mathrm{s}$; e) $T_s = 2.56\,\mathrm{s}, d_s = 0.2015\,\mathrm{s}$, f) $T_s = 5.12\,\mathrm{s}, d_s = 0.32\,\mathrm{s}$. Further, to study the impact of different scan intervals, we present results obtained from our model for sweeping values of T_s with fixed values $T_a = 5.12\,\mathrm{s}, d_s = 0.64\,\mathrm{s}$ in Experiment g) (without comparing them to simulation results). The parameter values for these experiments have been chosen such that they study the impact of different values for T_a, T_s and d_s and such that the main characteristics of the latency curves can be examined easily. In addition, to allow for a quantitative comparison between ground-truth and computed values, we have calculated the maximum and root mean square deviations for these experiments. In particular, the following values are presented, which are based on the simulated or measured mean/max latencies l_{sim}/l_{mes} and the computed latencies l_{comp}.

- The Root Mean Square Error K with

$$K = \frac{\sqrt{\sum_{T_{a,min}}^{T_{a,max}} (l_{comp}(T_a) - l_{sim}(T_a))^2}}{\|T_{a,max} - T_{a,min}\|}, \tag{3.34}$$

  where the denominator is the number of data points with unique advertising intervals.
- The Normalized Root Mean Square Error κ with

$$\kappa = \frac{K^2}{\max(l_{sim}) - \min(l_{sim})}. \tag{3.35}$$

- The Maximum deviation $MD = \max(|l_{comp} - l_{sim}|)$.

3.8.1 Testbed Measurements

To compare the results of our proposed theory against real-world measurements, we conducted the following experiment. Based on the open-source BLE stack *BLESSED* [44], we have implemented the protocol studied in this chapter. In particular, *BLESSED* implements the BLE protocol, but does not make use of additional random delays added to T_a. Advertising and scanning are carried out on one channel. This is exactly the problem considered in this chapter. We have created two different firmwares for two NRF51822 USB dongles. One device acts as the advertiser. First, it waits for a random amount of time between $[0, T_s]$ and then starts advertising with a certain interval T_a.

The other firmware scans the channel with an interval $T_s = 2.42\,\mathrm{s}$ and a scan window $d_s = 0.59\,\mathrm{s}$ during all times. For every discovery procedure, the advertiser reports the point in time the advertising is started to a host PC via an USB connection. In the same way, the scanner reports the reception of a packet. With this experiment, the discovery latency can be measured directly. We have conducted the experiment in an anechoic chamber to avoid any interference from external devices (e.g. smartphones). To evaluate the accuracy of this setup, we have carried out 10,000 additional measurements, in which we have measured a maximum loopback latency of 8 ms between the PC and the wireless node. This means that every measured discovery latency is increased by up to this amount of time. The advertising interval has been swept between 0.1 s and 3.0 s in steps of 0.01 s. Each discovery procedure has been aborted after 60 s, if no packet has been received within this period. For each advertising interval, the experiment has been repeated 100 times. The total number of discovery-procedures carried out has been 29,100, which took about 43 hours of wall-clock time. Figure 3.13 depicts the upper

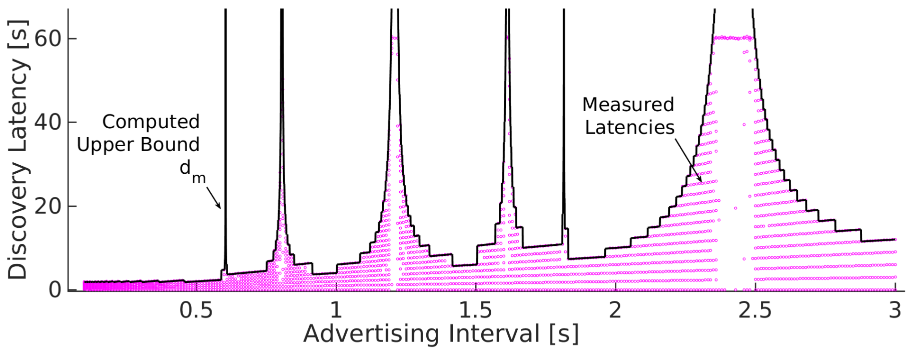


Figure 3.13: Measured latencies (circles) and computed upper bound (solid line) for Experiment a) ($T_s = 2.42s$ and $d_s = 0.59s$).

latency bound l^* predicted by our proposed theory (solid line). In addition, the 29,100 points in the figure show the measured discovery latencies of the wireless radios. As can be seen, our predicted bound is always reached, but never exceeded. Within latency peaks, most measurements have been aborted before successful discovery was achieved. The abortion took place after 60 s of wall-clock time, which explains the accumulation of latencies around 60 s within such areas.

To further quantify the accuracy of our theory, among the 100 repetitions for each advertising interval, the mean and maximum discovery latencies have been determined. Figure 3.14 a) shows the computed mean and worst-case latencies obtained from our theory, together with the measured values for a subset of the advertising intervals that have been realized. Each circle depicts the measured maximum latency out of 100 discovery procedures, and each cross the mean value. As can be seen, both for the mean and the worst-case latencies, the measurements lie in close proximity with our theory. Mean and maximum errors are given in Row a) of Table 3.5. Values with computed latencies higher than 60 s have been excluded from the computation of the errors, since they denote an aborted experiment, as already described. The computed upper latency bound has never been exceeded by more than 8.7 ms for any of the measured latencies.

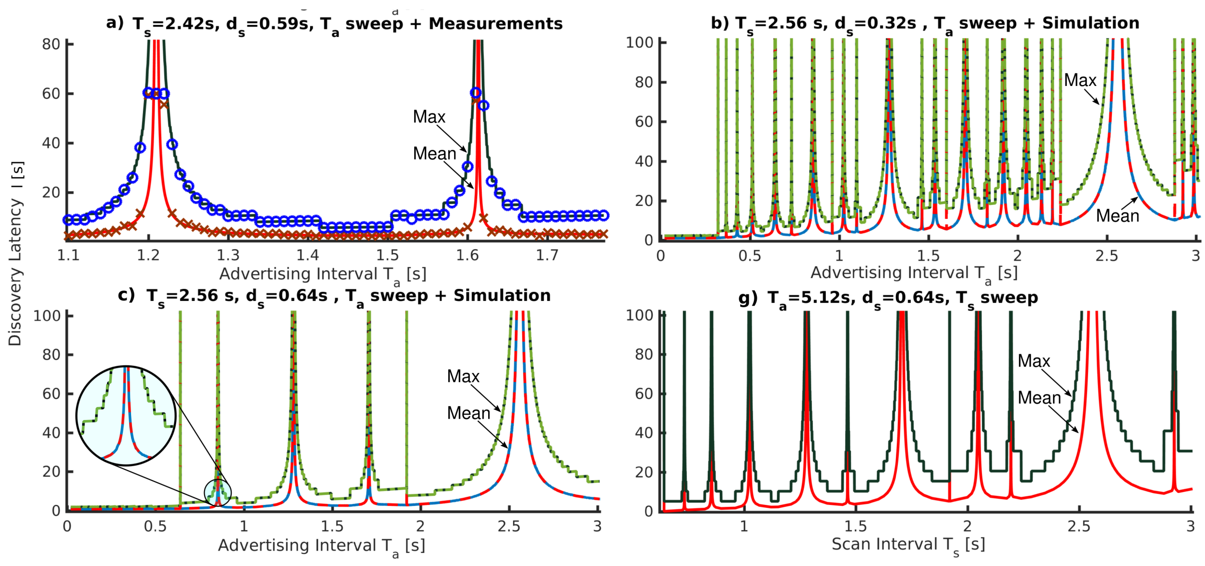


Figure 3.14: a): Measured (solid line) and computed (dashed line) mean and maximum discovery latencies. b), c): Simulated (solid lines) and computed (dashed lines) mean and maximum discovery latencies. Dashed and solid lines lie in such a close proximity that no difference is visible, indicating a low error between simulated and computed results. Figure g) shows the computed latencies for a fixed value of T_a and varying values of T_s.

This error is caused mainly by the latency of the USB connection. These measurement results fully confirm the validity of our proposed theory.

3.8.2 Discrete-Event Simulations

The high amount of wall-clock time required for the measurements limit both the number of repetitions for each advertising interval, as well as the number of examined parameter values. Therefore, we complemented the measurements by comprehensive simulation results. For this purpose, we have developed a custom discrete-event simulator in C++, which carries out the PI-based discovery-process considered in this chapter. The initial offset $\Phi[0]$ has been determined by an uniformly distributed random number generator in each simulation. The values of T_s and d_s have remained constant in each of our simulation experiments, while T_a has been varied in steps of 1.25 ms within [10 ms, 10.24 s − 1.25 ms]. The packet length ω has been set to 248 µs, which is a realistic value for protocols such as e.g., BLE. To determine the mean latency $\bar{l}$ and the maximum latency l^*, each simulation has been repeated 10,000 times for every value of T_a and the mean- and maximum latencies have been computed out of the 10,000 instances. The total number of simulations executed for each experiment has been 16,352,000, which took more than a month of CPU time per experiment. Each single simulation has been aborted after 1000 s of simulated time, even if the computed latencies have exceeded this limit.

The Figures 3.14 b) and c) show the simulated latencies for experiments b) and c) together with the latencies computed by our proposed algorithm for a subset of the

	$\overline{K}$	$\overline{\kappa}$	$\overline{MD}$	K_m	κ_m	MD_m
a)	0.55	0.933 %	3.62	1.12	1.908 %	7.32
b)	0.49	0.06 %	6.57	0.72	0.07 %	26.24
c)	0.35	0.05 %	6.35	0.41	0.04 %	25.62
d)	0.86	0.089 %	7.21	2.61	0.26 %	114.52
e)	0.61	0.066 %	9.02	0.66	0.067 %	38.60
f)	0.71	0.076 %	6.90	1.88	0.19 %	83.6

Table 3.5: Comparison of computation results against a) real-world measurements and b) - f) simulations. $\overline{K}$, $\overline{\kappa}$ and $\overline{MD}$ are RMSE, NRMSE and maximum deviations for the mean discovery latency, whereas K_m, κ_m and MD_m denote these values for the maximum discovery latency.

advertising intervals considered. In addition, the latencies predicted by our theory for Parametrization g), in which T_s instead of T_a is varied, are shown. The simulated curves are drawn using solid lines, whereas the curves obtained by our algorithm are drawn on top of them using dashed lines. As can be seen, both curves lie in such close proximity that no difference is visible in the figure. The only significant deviations from the simulation results exist at values of T_a at which the computed maximum latency has exceeded the maximum simulated time. Since each simulation has been aborted at 1000 s of simulated time, this behavior is expected.

To carry out a fair numerical comparison, we have excluded all values where the computed maximum latencies lie above $90\,\% \cdot 1000\,\mathrm{s}$ (1000 s is the maximum simulated time). The numerical results of this comparison are given in Rows b)-f) of Table 3.5. It can be observed that the root mean square errors K are negligibly low. This indicates a good match of our proposed theory. Since the maximum predicted latency might not have been reached by any of the 10,000 simulation repetitions carried out for each value of T_a, the maximum errors are higher than the mean ones, in particular for the worst-case latency MD_m. Our predicted upper bound has never been exceeded by any simulated value by more than $10 \times 10^{-10}\,\mathrm{s}$, which we considered as the floating point accuracy. Overall, these results fully confirm our proposed theory.

3.8.3 Discussion of the Results

The Figures 3.14 b) and c) show that the mean discovery latency has a nearly linear relation to the advertising interval for $T_a \leq d_s$. For $T_a > d_s$, both $\bar{l}(T_a)$ and $l^*(T_a)$ are composed of multiple peaks and minima. If T_s is increased or d_s is decreased, the number of peaks for different values of T_a is increased and these peaks tend to become narrower and steeper. Whereas the mean discovery latency is - except for some singularities - a smooth curve, the figures show (i.e., in the magnifying glass in Figure 3.14 c)) that the maximum discovery latency is composed of multiple straight lines with points of discontinuity separating them from each other. The situation shown in Figure 3.14 g), in which T_s is varied, has similar properties. Figure 3.14 reveals interesting results on the behavior of the ND latency for different values of T_a. First, we can observe that, except for the already mentioned singularities with $m_n = c$, the maximum latency is bounded for all values of T_a, T_s and d_s. To the best of our knowledge, this fact was previously unknown. It is now clear that interval-based protocols can guarantee deterministic latencies in general. Moreover, it shows that the advertising interval (and hence the duty-cycle) can be greatly increased without increasing the discovery latency by more than a nearly linear relation, if beneficial parametrizations are selected. The highly irregular curves also underpin the need for systematic parameter optimizations in PI-based networks.

Further, new protocols can be developed based on our theory. Such protocols can rely on one or more pairs of optimized advertising and scan intervals. These intervals do not necessarily have to be constant. As an example, for BLE, strategies that successively reduce the advertising interval (e.g., by half after each interval) have been proposed [35] and can now be optimized systematically using our theory.

Using our proposed solution, unsynchronized continuous data transmission with duty-cycled, PI-based protocols (also referred to as continuous broadcasting, e.g. in the context of BLE beacons) becomes feasible, since the broadcaster and the receivers can be configured such that the number of packet repetitions is optimized and a deterministic reception is guaranteed.

3.8.4 Comparison to Existing Models

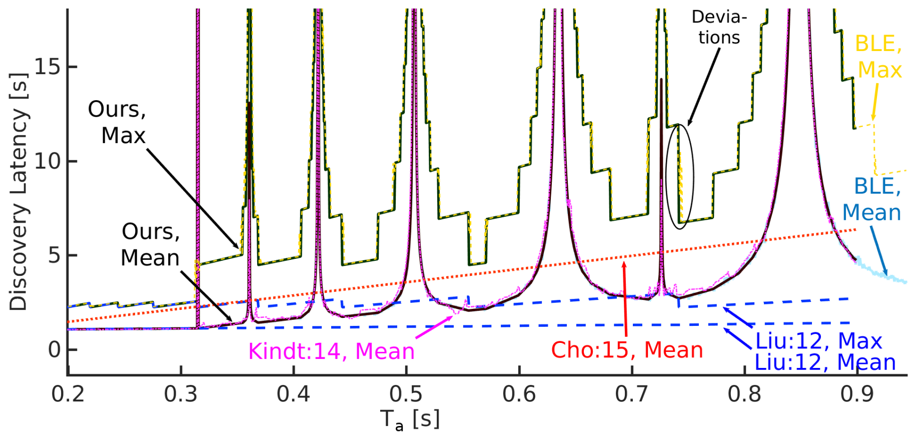


Figure 3.15: Comparison to the models Liu:12 [36], [35], [34], [63], Cho:15 ([13], [12]), Kindt:13 [28], our proposed model and a simulation of BLE.

In this section, we compare our proposed theory to existing models for BLE. We assume the same parametrization as for Figure 3.14 b). For a fair comparison, we have implemented modified versions of these models, such that they describe the ideal PI-based problem, rather than the 3-channel discovery procedure of BLE. We have considered the model by Liu et al. [36], [35], [34], which is identical to the model from [63] when considering single-channel discovery. Besides our proposed theory, this model is the only one that predicts upper latency bounds (for $T_a < d_s$, only). Further, we have considered the predicted mean latencies of the model by Cho et al. [13], [12] and the accelerated simulation model by Kindt et al. [28] (i.e., our own previous work), with $\Delta = 93.12\,\mathrm{s}$ and $\epsilon = 0.99$ into this comparison.

To compare to which extent existing models approximate the BLE protocol, we have conducted discrete-event simulations of the complete BLE discovery procedure, with 1000 repetitions per value of T_a. For this purpose, the simulator used for the evaluation of our model has been extended to account for 1) three channel discovery and 2) a random delay $\rho \in [0\,\mathrm{s}, 10\,\mathrm{ms}]$ being added to each advertising interval. As already described, the outputs of the simulator for the ideal PI discovery procedure are in line with the results obtained by our model, which are verified against real-world measurements (cf. Section 3.8.1). We have assumed an effective advertising interval of $T_a' = T_a + 0.005\,\mathrm{s}$ to account for the mean shift of BLE's random delay in all considered models, which increases their accuracies.

Comparison with other models: As shown in Figure 3.15, the model by Liu [36], [35], [34], [63] matches well for $T_a \leq d_s - \omega$, but is not valid for $T_a > d_s - \omega$. It can be seen from the figure that the model by Cho [13], [12] is unable to predict any latency peaks, due to the assumption of independent overlap probabilities. The simulation model by Kindt [28] models the overall trend, but is subjected to significant statistical variations.

Comparison with BLE: The results of this comparison are shown in Figure 3.15. As can be seen, our proposed model approximates the properties of BLE well. Small deviations of the predicted upper latency bound at border values can be observed e.g. around $T_a \approx 0.74s$, which are caused by the random delay and the multichannel discovery. While our model provides, to the best of or knowledge, the most accurate latency estimations for the BLE protocol that are available, it needs to be mentioned that for some parametrizations, the random delay could cause significant deviations from our predicted latencies. Studying the random delay and three-channel discovery needs to be studied in future research.

3.8.5 Computational Complexity

In this section, we evaluate the computational complexity of Algorithm 3.1. Our aim is to establish that it has reasonable computational overhead and can therefore be used to compute a large number of latency values in short amounts of time.

The algorithm processes each segment of a probability buffer Γ_n, evaluates the corresponding equations and adds new probability segments to the next probability buffer Γ_{n+1}. The computational complexity for processing any order n is therefore determined by the number of segments in Γ_n. Whereas the initial probability buffer contains only one segment, a conservative assumption is that each additional sequence-order $n \leftarrow n+1$ potentially triples the number of entries in Γ_{n+1}, since each segment of Γ_n might be subdivided twice in the worst-case. Therefore, the complexity grows exponentially with larger maximum orders n_m. However, the maximum order is bounded, and the

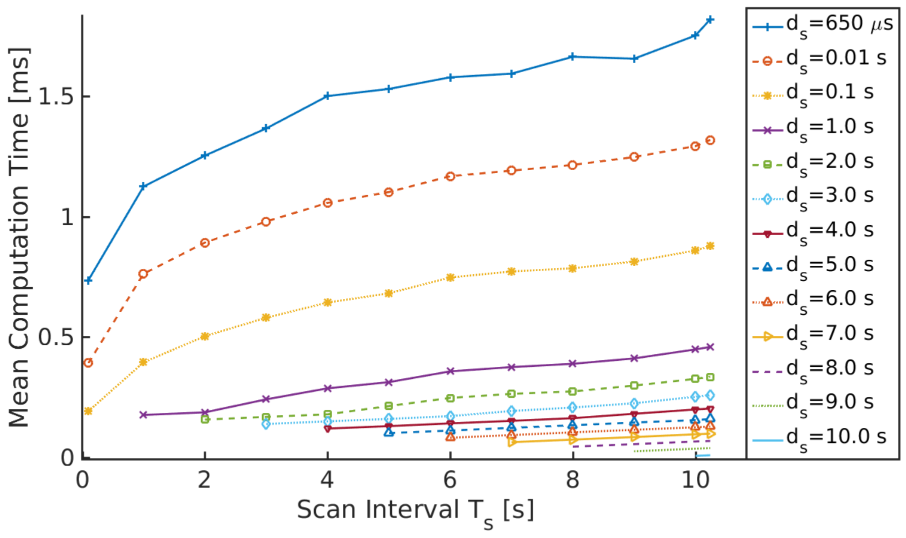


Figure 3.16: Mean computation time per advertising interval for computing the mean- and maximum latencies for all advertising intervals $T_a \in [0.2\,\mathrm{s}, 10.24\,\mathrm{s}]$ in steps of 650 µs.

bound grows only logarithmically with the smaller interval of T_a and T_s, as defined by Equation 3.4. Because practical problems have limited maximum orders (e.g., in BLE), the expected computational demands are low.

For a detailed evaluation, we have measured the computation time of a MATLAB-implementation using the tic/toc profiling mechanism of MATLAB [51]. The timing measurements were taken on a Lenovo Thinkpad X240 laptop with MATLAB R2015b under Linux Mint 17.2, using a single CPU core. For different values of d_s and T_s, we ran our proposed algorithm for all advertising intervals between [20 ms, 10.24 s] in steps of 650 µs, which is equivalent to the whole range of advertising intervals allowed in BLE. For each of these 16,353 executions of Algorithm 3.1, the execution time was measured. The mean time the algorithm took until termination among the 16,535 executions is shown in Figure 3.16 for multiple scan intervals and windows. The plot also includes the extreme case for BLE, with $d_s = 650\,\mu\mathrm{s}$ and $T_s = 10.24\,\mathrm{s}$. This parametrization leads to the maximum theoretically possible order of $n_m = 14$ for BLE[2]. The highest computation time reached has been 1.8 ms for $d_s = 650\,\mu\mathrm{s}$ and $T_s = 10.24\,\mathrm{s}$. The results show that our proposed algorithm is computationally cheap in real-world situations and multiple orders of magnitude faster than discrete-event simulations. Recall that for the curves in Figure 3.14, more than a month of computation time has been spent for each simulation.

3.9 Parameter Selection

In this section, we briefly describe how our proposed theory can be used to optimize the parametrization of PI-based discovery approaches. As shown in Section 3.8.5, Algorithm 3.1 is computationally cheap and therefore well suited to be executed

[2]During this experiment, the maximum problem order that actually appeared was 10.

repeatedly for a large set of parameters. For example, for a fixed tuple of values (T_s, d_s), all possible values of T_a allowed by the BLE specification [41] can be computed within a few seconds. With simulations, this is not possible due to the long simulation times and the limited accuracies. Therefore, for optimizing the parameters, we propose performing a design-space exploration based on our model. The optimization goal is specific to the application and is typically one out of the following possibilities:

1. Minimize the mean/maximum discovery latency as the first optimization goal. Since the same latencies can be achieved with multiple parametrizations, a second optimization criterion can be chosen.

2. Minimize the expected mean/maximum energy-consumption spent for the discovery. Given the energy E_a for an advertising packet and E_s for one scan window, it is reasonable to minimize the product of the latency and the energy consumption $1/T_a \cdot E_a \bar{l} \omega$ for the advertiser or $1/T_s \cdot E_s \bar{l} d_s$ for the scanner. When considering the maximum discovery latency L instead of $\bar{l}$ and by setting $E_a = E_s = 1$, this equation approximates the power/latency product defined in [23].

3. Minimize the joint energy consumption of both the advertiser and the scanner, such that their sum becomes minimal.

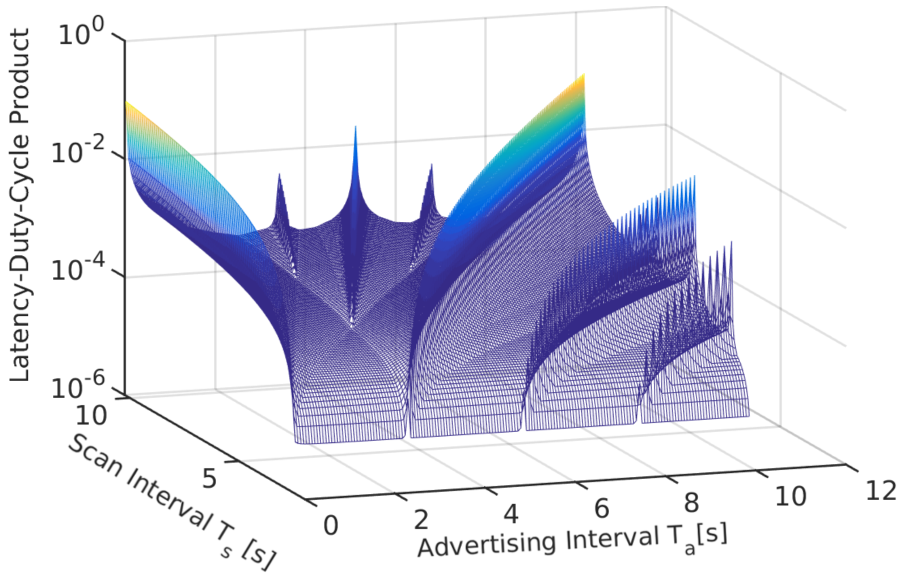


Figure 3.17: Latency/duty-cycle product for a fixed scan window $d_s = 2.5s$.

A design space exploration for minimizing the mean-power-latency product is shown in Figure 3.17. For a fixed scan window $d_s = 2.5s$, T_a and T_s have been varied in steps of $62.5ms$ to compute the latency/duty-cycle product $1/T_a \cdot \omega \bar{l}$ of the advertiser. In the figure, all latencies exceeding 1×10^{10} s have been truncated to this value. As can be seen, there are multiple optimal points for a given scan interval, such that the advertising interval can be chosen within a large set of values without affecting the energy efficiency. We derive such optimal parametrizations analytically in the next chapter.

3.10 Concluding Remarks

In this chapter, we have presented a complete theory for computing the ND latencies of PI-based, slotless protocols. Our results imply two major novel insights, which have important implications.

First, the discovery latencies of such protocols can now for the first time be computed accurately. Since PI-based concepts are widely used in practice (e.g., for the BLE protocol and for ANT/ANT+), understanding their latency behavior is of high importance. The specifications of some wireless services recommend certain parameter ranges in which latency-peaks can occur. For example, the parametrizations suggested by the BLE find-me profile [42] allow interval lengths that lie within such a peak, leading to long discovery latencies [28]. Therefore, such parametrizations should be checked using our theory in the future to avoid disadvantageous recommendations.

Second, our results have revealed that the latency is bounded for almost all parameter values. This fact has previously been unknown. Because of this, PI-based solutions become applicable to networks with deterministic latency demands, where currently slotted protocols are being used. As already described, periodic interval-based solutions provide the largest flexibility among all deterministic possibilities. They are not only capable of realizing practically every duty-cycle with an almost arbitrary granularity, but also decouple reception and transmission. Because of this, they can potentially achieve lower worst-case latencies than slotted protocols for a given duty-cycle. However, the performance of PI-based protocols clearly depends on their parametrization. For achieving low energy-consumptions and discovery latencies, we study optimal parametrizations in depth in the next chapter.

4 Design of Optimal Periodic Interval-Based Protocols

4.1 Introduction

In Chapter 3, we have presented a theory to compute the worst-case latencies of PI-based protocols for all possible values of their parameters T_a, T_s and d_s. This theory has revealed that deterministic discovery latencies can be achieved for all parameter values except for a finite number of singularities. Because of this, the question on how PI-based ND protocols compare against existing ones arises. However, as we have shown, their performance strongly depends on their parameter values.

In this chapter, we propose a computationally inexpensive optimization framework for PI-based ND protocols. Given a target duty-cycle, our framework will help in computing the protocol parameters that we show in this chapter outperform the latencies of a large number of well-known ND protocols, e.g., Disco [15], Searchlight [2], U-Connect [23] and G-Nihao [61]. Further, PI-based protocols configured using this framework outperform difference codes [55], which have been proven to provide the lowest-possible latencies among all slotted protocols. In fact, we prove that parametrizations resulting from our framework are optimal for all specified duty-cycles. In particular, no other parametrization will lead to lower worst-case latencies. Further, we show that no other ND protocol in which each device independently discovers its opposite can beat the latencies achieved.

We focus on networks with small numbers of devices carrying out the discovery simultaneously, which require different solutions for ND than larger networks. We next describe the differences of ND in small and large networks.

ND in Small vs. Large Networks: Many wireless networks, in particular in the personal area domain, consist of only a few devices being in discovery mode simultaneously. For example, in BLE piconets, two devices usually communicate in a synchronized fashion after the ND procedure has completed. Here, ND is only used for setting up the connection. Even in setups in which additional devices might join during runtime, only the connection master needs to continuously run a ND protocol. Hence, the number of devices performing ND simultaneously is limited. Though the total number of wireless devices is likely to increase in the future, networks with few devices being in discovery mode simultaneously will remain important. For example, IoT setups, in which new devices join the network gradually, are expected to grow in importance.

Whereas slotted ND protocols, e.g., [15, 23, 2, 55], can be efficient solutions in large networks with tens or hundreds of participating devices, they perform inefficiently in scenarios with only a few (e.g., up to ten) devices being in discovery mode simultaneously. This is because slotted schemes imply that transmission and reception take place jointly within each slot, which prevents a systematic optimization of the ratio between sending and receiving durations. In contrast, PI-based protocols can send the *optimal* rate of packets for achieving a low worst-case latency and energy consumption. Though this leads to a higher channel utilization and hence collision probability, the fraction of failed discoveries remains reasonably low if only a few devices carry out the discovery procedure at the same time. The collision probability increases with more devices being in discovery mode simultaneously, but the degradation of the rate of successful discoveries is graceful. Therefore, parameter optimizations for PI-based protocols are highly relevant in practice.

Optimizations in PI-Based Protocols: This chapter proposes an optimization framework for jointly minimizing the duty-cycle and worst-case latencies of PI-based protocols. While a significant amount of research has been carried out to analyze and improve slotted protocols, the optimization of PI-based ones has not been sufficiently studied. This is mainly due to the popular belief that they cannot guarantee deterministic latencies. As described in the previous Chapter, except for trivial cases like $T_a \leq d_s - \omega$ (i.e., the distance between two packets is below the scan window length) or $T_s = d_s$ (i.e., the scanner is always active), the discovery latencies of PI-based protocols have not been well understood prior to this work. Our generic model for PI-based protocols described in Chapter 3 is given in the form of a recursive computation scheme, and therefore cannot be "inverted" to identify beneficial parameter values. In other words, while Chapter 3 shows how worst case latencies can be computed from protocol parameters, it is not possible to use these results to determine parameters for achieving optimal discovery latencies for a given duty cycle (and hence energy budget). Performing an exhaustive search to find optimal parameter values is not practical either, since there are 3 degrees of freedom (viz., T_a, T_s and d_s), leading to a computationally infeasible procedure. Therefore, new methods to calculate optimal protocol parameters in an efficient manner are required. This chapter addresses this problem and proposes a framework to translate any given target duty-cycle into a set of optimal parameter values. In particular, T_a, T_s and d_s are derived from a target duty-cycle using closed-form equations, which makes exhaustive searches or other computationally expensive procedures unnecessary.

Moreover, while there is a large number publications that compare the performance of various slotted protocols against each other, what is the performance of PI-based protocols, such as BLE? Due to the lack of optimal parametrizations, their performance still remains unclear. This holds true particularly for networks with few devices carrying out ND in parallel, in which protocols such as BLE are frequently used. We in this chapter for the first time compare the performance of the BLE discovery procedure to existing ones.

Besides optimal parameter values, a methodology for comparing the performances of optimized slotless and slotted protocols is required. The worst-case latencies of slotted solutions are proportional to the slot length, for which no lower limit is known. This has prevented reasonable comparisons in the past. While we have identified the theoretical lower limits of the slot length in Chapter 2 for ideal radios, we in this chapter derive the limits of the slot length for real-world radios and hence make such comparisons possible.
Contributions and Organization: The major contributions of this chapter are the following.

1. We propose the first closed-form optimization framework for PI-based protocols in Section 4.3. It is given by a set of equations for translating any specified target duty-cycle η into optimal parameter values for T_a, T_s and d_s. Protocols configured accordingly achieve significantly lower worst-case latencies than a large number of known pairwise ND protocols (e.g., Searchlight, Disco, U-Connect, G-Nihao and difference codes) in small networks. We formally proof that the resulting latency-duty-cycle relations are optimal, and that no other direct, pairwise ND protocol can guarantee lower latencies for the same duty cycle.

2. As in any model, we made some idealistic assumptions for our framework. A real-world hardware has a number of non-idealities, like turnaround times and clock quantization errors. Therefore, we have taken steps to minimize the differences between the platform and the model. Building upon these steps, in Section 4.4, we present a real-world implementation of a PI-based protocol parameterized using our proposed framework on a wireless radio.

3. We show how the values obtained from our optimization framework can be tuned for configuring BLE in Section 4.5. To the best of our knowledge, we thereby propose the first closed-form parametrization framework to configure the BLE protocol towards low latencies and energy consumption.

4. A methodology for comparing the performances of slotted protocols to those of PI-based approaches is not available in the literature. The latencies of slotted protocols are proportional to the slot length, for which a lower limit has to be identified for reasonable comparisons. In any ND protocol, even in deterministic ones, every discovery attempt can fail with a certain probability, and two devices never discover each other in such cases. For slotted protocols, we identify the slot lengths that lead to the same failure rates as those of our proposed solution and compare the performance of optimized PI-based protocols with existing approaches

in Section 4.6. In the applications considered, our proposed solution significantly outperforms all slotted ND protocols. E.g., when allowing a rate of failed discoveries of 0.19 % for two devices, difference codes [55], which achieve the theoretically optimal latencies for slotted protocols, provide by 385.1× larger worst-case latencies than our solution. Further, using experimental measurements on 560, 000 discovery procedures, we demonstrate that the anticipated performance is reached in practice.

In the next section, we present related work. Finally, we discuss the implications of our theory in Section 4.8.

Recall that also in this chapter, as described in Chapter 3, l^* denotes the packet-to-packet worst-case latency. By optimizing l^*, our framework will also optimize L. To maintain comparability with the results from the previous chapters, we will evaluate our parametrizations in respect of L instead of l^*.

4.2 Related Work

In this section, we describe previous works that concern identifying beneficial parameter values for PI-based protocols. In addition, since we will compare the performances of PI-based protocols using our proposed parametrizations to multiple previously known protocols in this chapter, the slotted ND protocols described in Chapter 1 are also relevant. To avoid redundant descriptions, we do not describe these protocols in this section, but refer to Chapter 1 instead.

Due to the lack of appropriate analytical models, it has not been clear whether PI-based protocols can provide deterministic latencies beyond the special case of $T_a \leq d_s - \omega$. As a workaround, special parameterizations for BLE that fulfill the Chines Remainder Theorem have been proposed in [24]. Another attempt to derive beneficial parameters for a modified version of BLE has been presented in [22]. Since BLE applies a random delay that is added to each advertising interval, extending the scan window has been proposed as a countermeasure. The parametrizations proposed in [22] partially rely on exhaustive searches, which are feasible due to the restricted set of valid parameter choices in BLE. In particular, BLE only allows a finite set of values for T_a, T_s and d_s. In contrast, our approach provides optimizations for the full, continuous range of parameter values. In addition, our approach is fully closed-form and does not rely on exhaustive searches, which can only be done offline. Another attempt on parameter optimization has been proposed in [77]. However, it has been assumed that the length of each transmission is not given by ω, but there are certain fixed-length transmission windows that exceed ω. This is only applicable on radios supporting the low-power listening (LPL) paradigm (cf. [23]). Further, only one-way discovery and a restricted set of parameter values has been considered.

Though not being a PI-based scheme in the classical sense, some recently proposed pseudo-slotted protocols, such as e.g., Nihao [61], construct schedules that result into sequences of packets and reception windows that are similar to that of PI-based protocols. Therefore, they can also be regarded as parametrization schemes for PI-based protocols. Unlike most previously known slotted protocols, Nihao defines dedicated receive and transmit slots. In each receive slot, the radio listens to the channel during the whole slot length. In each transmit slot, one beacon is sent at the beginning of the slot, and the device goes back to a sleep mode afterwards. Such a scheme can be regarded as slotless, since transmission and reception are decoupled from each other. However, all points in time for transmission and reception are determined by a multiple of a certain duration, which is called the slot length I and which corresponds to the length of one reception phase. The assumption of slotted time thereby functions as a simplification of the design space.

In contrast, our proposed parametrization scheme considers continuous time. The resulting parametrizations also lead to a sequence of packets and reception windows whose points in time are given by multiples of the reception window length. In comparison to our proposed protocol, Nihao assumes a fixed slot length that is not adjusted online, whereas in our solution, the length of each reception window depends on the duty-cycle. Further, Nihao does not contain built-in optimizations of its parameters for low worst-case latencies. In particular, Nihao leaves open the values of multiple of its parameters (e.g., n and m, as described in [61]), which need to be chosen by the user.

Though there is a version of Nihao that has no adjustable parameters, which is called *S-Nihao*, we only consider its parameterizable version, which is called *G-Nihao* in this chapter. For S-Nihao, [61] states that it requires the quotient of the packet transmission duration over the slot length to be smaller than its duty-cycle. Therefore, it is restricted to relatively large duty-cycles or to very long slot lengths [61], which are infeasible in practice.

Another pseudo-slotted protocol is Lightning [76]. However, this protocol applies mutual assistance, as described in Chapter 2. Therefore, its performance is not directly comparable to the remaining protocols. The same holds true for Wifi-Lock [60].

In summary, no comparable framework for optimizing the parameters of PI-based protocols towards low discovery latencies has been known prior to this work. Further, no existing direct ND protocol (i.e., no protocol that does not rely on mutual assistance), including no slotted one, achieves latency/duty-cycle relations that are as low as those of PI-based protocols parameterized using our proposed scheme. It is worth mentioning that a mutually assisted version of our proposed protocol can be created easily. For such a protocol, Theorem 2.10 instead of Theorem 2.8 would apply for one-way discovery, with two-way discovery occurring only slightly later. Here, the number of beacons and/or reception windows and their length could be reduced, while obtaining the same two-way worst-case discovery latency. Hence, the latency/duty-cycle relation could be further improved. However, mutual assistance would lead to a higher rate of failed discoveries.

4.3 Optimization Framework

In this section, we describe our proposed optimization framework for parametrizing PI-based protocols. As already mentioned, in PI-based protocols, packets are transmitted with an interval T_a, whereas reception windows of length d_s are scheduled periodically every T_s time-units. Our objective is to derive a set of equations that translate any specified target duty-cycle η into parameter values for T_a, T_s and d_s, which are optimized for low latency-duty-cycle relations.

4.3.1 Overview

The duty-cycle η is the main energy metric of PI-based protocols. It is defined as

$$\eta = \frac{d_s}{T_s} + \alpha \cdot \frac{\omega}{T_a} = \frac{T_a d_s + \alpha T_s \omega}{T_a T_s}. \tag{4.1}$$

Here, α is a balancing factor to account for different energy consumptions for transmission and reception. Without loss of generality, we assume $\alpha = 1$ for the rest of this chapter. After a device comes into the range of reception of another device, the first packet of this device has a temporal offset of Φ_0 time-units to the scan window of the opposite device that directly precedes this packet (cf. Figure 4.1a)). Since there is no previous synchronization between any two devices, Φ_0 has a random value between 0 and T_s time-units. The worst-case latency l^* is given by the time until which a packet from one device overlaps with a scan window of the other device for *any* initial offset Φ_0, measured from the first packet that is sent in range.

For a given target duty-cycle η, there is in general a large set of values for T_s, T_s and d_s that correspond to this duty-cycle, of which multiple values are potentially optimal. In our proposed framework, we first pre-constrain the set of possible values by establishing multiple relations between these parameters. These relations mainly exploit the insight that for no initial offset, more than one packet that is sent within the worst-case latency should redundantly overlap with any scan window (refer to Chapter 2 for details on why this is a necessary condition for optimality). We then systematically optimize the parameter values that lie within the set of constrained values. Though the set of parameters that are excluded by these constraints might also contain optimal parameter values, we formally proof in Section 4.7 that for every η, the parameter values computed by our framework are always optimal.

Quick Reference of Symbols

η	Duty-cycle a protocol realizes
γ	Growth/Shrinkage in a γ-sequence
k, M	Integer numbers to parametrize T_a and T_s
M_{opt}, k_{opt}	Optimal values of k and M
$d_{s,m}$	Lower limit of the scan window d_s

A full table of symbols is given in Appendix A.

In particular, we consider three different cases that lead to different sets of pre-constrained values. As will become clear in Section 4.4, one of these cases is beneficial for real-world implementations of ideal PI-based schemes, whereas another one is beneficial for parametrizing BLE. These three cases are shown in Figure 4.1. The ruled boxes depict the scan windows, whereas the hatched vertical bars show the packets. First, we consider the situation with $T_a \leq d_s - \omega$, as depicted in Figure 4.1a).

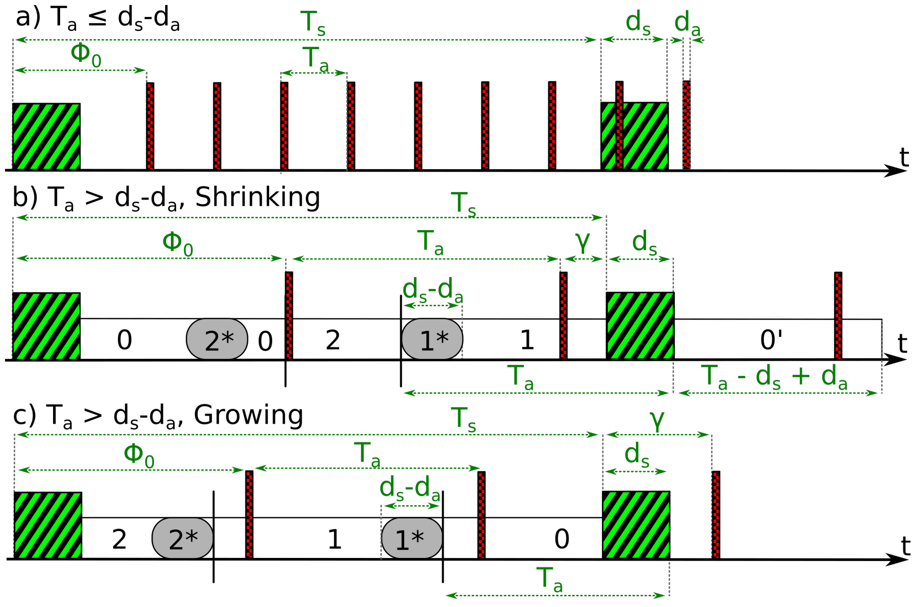


Figure 4.1: Model for a) $T_s \leq d_s - \omega$ and b), c) $T_s > d_s - \omega$.

4.3.2 Optimization Scheme a): PI − 0M

Figure 4.1a) shows the packet flow for a situation with $T_a \leq d_s - \omega$, which leads to a parametrization scheme we call $PI - 0M$ (as this scheme has one parameter M, as described below). Here, the temporal distance between any two packets is no more than $d_s - \omega$ time-units, and hence, there is no scan window into which no packet falls. One can derive from the figure that for every possible initial offset Φ_0, the time-distance measured from the first packet in range to the next packet that overlaps with a scan window never exceeds approximately T_s time-units. More precisely, the packet-to-packet worst-case latency l^* is given by:

$$l^* = \underbrace{\left\lceil \frac{T_s - (d_s + \omega)}{T_a} \right\rceil}_{\mathcal{A}} \cdot T_a + \underbrace{\omega}_{\mathcal{B}} \tag{4.2}$$

Equation 4.2 can be explained as follows. The distance between any two consecutive scan windows is given by $T_a - (d_s - \omega)$ time-units. The number of advertising intervals that "fit" into this distance is given by the term $\mathcal{A}$. In the worst case, after this number of advertising intervals plus one packet transmission duration ($\mathcal{B}$), discovery is guaranteed. We next attempt to minimize this latency.

Optimizing T_a

To minimize the ceiling-function $\mathcal{A}$ in Equation 4.2, it is beneficial to set T_a to its largest possible value. Since we for this parametrization scheme require $T_a \leq d_s - \omega$, this largest possible value is given by $T_a = d_s - \omega$. Figure 4.2 depicts the maximum discovery latencies l^* for sweeping values of T_a and T_s with a fixed value of d_s. As can be seen, the latency

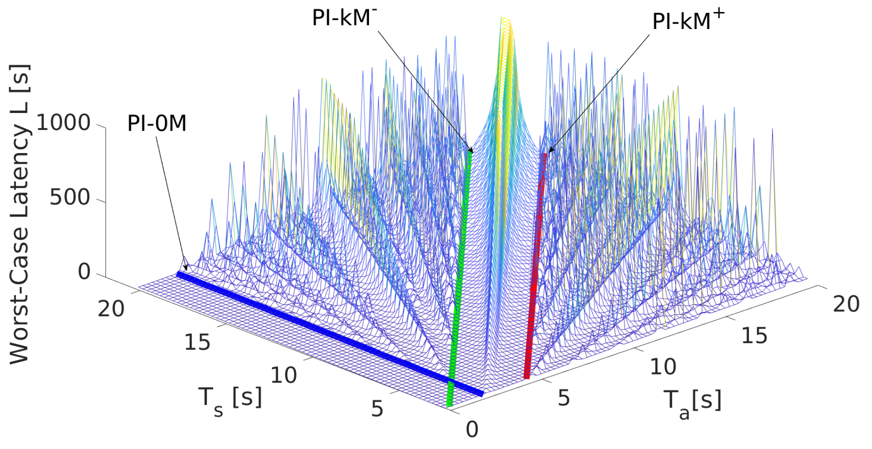


Figure 4.2: Discovery latencies for different values of T_a and T_s and valuations chosen for the $PI - 0M$, $PI - kM^+$ and $PI - kM^-$ parametrization schemes ($d_s = 2.1\,\mathrm{s}, k = 1$).

is subjected to strong variations for different parameter values. The line marked with $PI - 0M$ in Figure 4.2 visualizes the worst case latencies when choosing the advertising interval as $T_a = d_s - \omega$. As can be seen, for all values of T_s, these latencies lie within a local minimum.

Optimizing $\mathbf{T_s}$

Within certain bounds, the duty-cycle η defined by Equation 4.1 becomes smaller for increasing values of T_s, without affecting the worst-case latency l^*. However, if T_s exceeds certain thresholds, $\mathcal{A}$ in Equation 4.2 turns over to its next higher value. Therefore, we set T_s to the largest possible values that precede the steps of the ceiling-function $\mathcal{A}$ in Equation 4.2. Such values are given by the following relation:

$$\frac{T_s - (d_s - \omega)}{T_a} = M - \epsilon, M \in \mathbb{N} \tag{4.3}$$

We assume that ϵ is an arbitrary small value close to zero, such that $\mathcal{A}$ in Equation 4.2 is slightly below its next integer value. We will describe a practical valuation for ϵ in Section 4.4 and assume $\epsilon \leftarrow 0$ in the following. With this assumption, and by substituting $T_a = d_s - \omega$ in Equation 4.3, T_s becomes:

$$T_s = (M + 1)(d_s - \omega) \tag{4.4}$$

By inserting this into Equation 4.2, one can show that the worst-case latency l^* obtained using such parameterizations is $M(d_s - \omega)$ time-units. The specified target duty-cycle η determines the value of d_s. By rearranging Equation 4.1 and by substituting T_s with Equation 4.4 and T_a with $d_s - \omega$, we obtain:

$$d_s(\eta) = \omega + \frac{\omega(M + 2)}{\eta(M + 1) - 1} \tag{4.5}$$

The only parameter that is yet to be determined is M. By differentiating the product $\eta \cdot l^*$ (which we attempt to minimize), one can identify a local minimum at:

$$M_{opt} = \frac{\sqrt{1-\eta^2}+1}{\eta} - 1 \tag{4.6}$$

Using this, given a target duty-cycle η, T_a, T_s and d_s are given by closed-form equations. As we show in Section 4.7, the resulting values are optimal, and no other values lead to lower worst-case latencies. However, the hardware might be unable to realize reception phases shorter than a certain threshold $d_{s,m}$. Solving the inequality $d_s(\eta) < d_{s,m}$ using Equation 4.5 leads to the following constraints on M:

$$M > \frac{1}{\eta} - 1 = M_{min} \tag{4.7}$$

In addition, if $\eta > \frac{\omega}{d_{s,m}-\omega}$, $M \leq M_{max}$, where

$$M_{max} = \frac{d_{s,m}(\eta-1) - \omega(\eta+1)}{\omega(\eta+1) - \eta d_{s,m}}. \tag{4.8}$$

Considering these constraints, we always set M to round(M_{opt}), or to the value being closest to round(M_{opt}) that lies within $[\lceil M_{min} \rceil, \lfloor M_{max} \rfloor]$, respectively. Since $\lceil M_{min} \rceil \leq \lfloor M_{max} \rfloor$, a conservative bound on the maximum duty-cycle that can be realized using this scheme is obtained from solving $M_{min} + 1 \leq M_{max} - 1$ as:

$$\eta_{max} = \frac{3\omega + \sqrt{\omega(\omega + 8d_{s,m})}}{4(d_{s,m} - \omega)} \tag{4.9}$$

As will become clear in Section 4.4, the scheme above is beneficial for parametrizing BLE, in which the transmission times are given by periodic intervals plus a certain random delay. In contrast, the schemes described next will be more beneficial for implementing unmodified PI-based protocols.

4.3.3 Schemes b) and c): $\mathbf{PI - kM^+}$

We now consider values with $T_a > d_s - \omega$ and study two possible parametrization Schemes b) and c).

For Scheme b), let k be the largest multiple of T_a, such that $k \cdot T_a < T_s$ and $T_s - k \cdot T_a < 1/2 \cdot T_a$. For example, in Figure 4.1b), $k = 2$. We define $\gamma = T_s - k \cdot T_a$. As already mentioned, the first advertising packet has a temporal distance of Φ_0 time-units to its temporally left scan window. Clearly, after every scan interval, the temporal distance between a scan window and its temporally right advertising packet will be reduced by γ time-units, since $k \cdot T_a$ time-units are by exactly γ time-units shorter than T_s. For example, in Figure 4.1b), the temporal distance between the third packet and the second scan window is by γ time-units shorter than between the first packet and the first scan window. If $\gamma < d_s - \omega$, after a finite number of steps of length γ, the remaining offset will lie within $[0, d_s]$ and therefore the packet matches the corresponding scan window. Since the temporal distance between subsequent scan windows and their neighboring packets shrinks successively, Scheme b) is referred to as *shrinking*.

In Scheme c), let k be the smallest multiple of T_a, such that $k \cdot T_a > T_s$ and $k \cdot T_a - T_s < 1/2 \cdot T_a$. For example, in Figure 4.1c), $k = 3$. Here, the offset Φ_0 between the first packet and the closest scan window that is temporally left grows successively for later pairs of packets and windows. Hence, the offset to the next (i.e., temporally right) scan window shrinks by $\gamma = k \cdot T_a - T_s$ time-units per scan interval, and similar to Scheme b), there will be a packet overlapping with a scan window after a certain number of scan intervals.

After which time does such an overlap take place in the worst case? In both schemes, from some offsets within $[0, T_s]$ (viz., those indicated by the rounded boxes marked with * in Figure 4.1b)), the next scan window can be reached directly by multiples of T_a. From the remaining offsets, the right scan window depicted in the figure is approached as close as possible by multiples of T_a, and the resulting temporal distance is then reduced by multiple steps of γ, such that a later scan window is reached by the mechanism described above. The numbers in the rounded boxes in Figure 4.1b) (i.e., the boxes marked with *) are the number of T_a - intervals until reaching the scan window. The numbers in the remaining areas are the quantity of T_a - intervals for approaching the scan window as close as possible by multiples of T_a. From the resulting offsets, which lie in a proximity of less than T_a time-units from the scan window, a later scan window is reached after multiple additional steps of γ. For example, in Scheme b), the first packet might be sent within Area 2 in Figure 4.1b). Then, two advertising intervals pass until reaching the area behind the right scan window (indicated with $0'$ in Figure 4.1b). From there, the temporal distance to a later scan window is further reduced in multiples of γ until it is reached. With these considerations, one can derive the worst case latency for Scheme b) as follows:

$$l^* = \left\lceil \frac{T_s - T_a}{T_a} \right\rceil T_a + \left\lceil \frac{T_a - (d_s - \omega)}{\gamma} \right\rceil \cdot \left\lfloor \frac{T_s}{T_a} \right\rfloor T_a + \omega. \tag{4.10}$$

For Scheme c), it is similarly:

$$\begin{aligned} l^* = &\left\lceil \frac{T_s + d_s - \omega}{T_a} - 1 \right\rceil T_a + \\ &\left\lceil \frac{T_a - d_s + \omega}{\gamma} \right\rceil \cdot \left\lceil \frac{T_s}{T_a} \right\rceil T_a + \omega. \end{aligned} \tag{4.11}$$

With this, the product of the latency and the duty-cycle can be optimized by choosing T_a, T_s and d_s as follows.

Choosing $\mathbf{T_a}$

Equations 4.11 and 4.10 become minimal for large values of γ. The maximum possible value of γ is $d_s - \omega$, since larger values would lead to offset shrinkages or growths per scan interval of $\gamma > d_s - \omega$, and hence the scan window would be missed for some offsets.

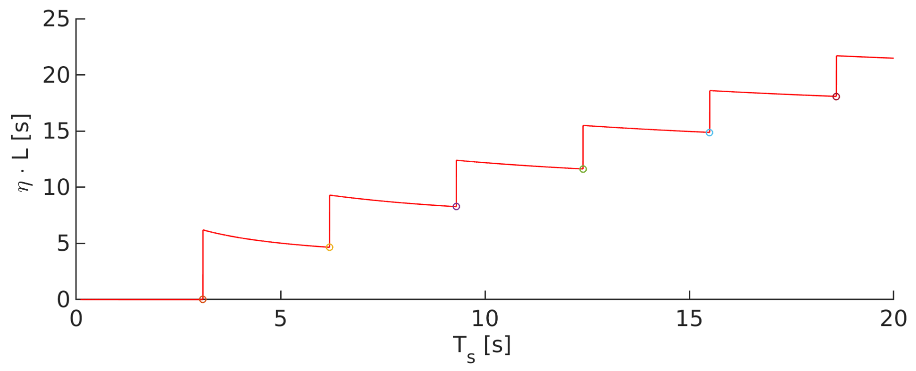


Figure 4.3: The product $\eta \cdot l^*$ for $d_s = 3.1\ s$, $k = 1$ when choosing T_a according to Equation 4.12.

From $\gamma = |k \cdot T_a - T_s|$, it follows that $\gamma = d_s - \omega$ can be realized by setting $k \cdot T_a$ by $d_s - \omega$ time-units shorter or longer than one scan interval:

$$k \cdot T_a = T_s \pm (d_s - \omega) \tag{4.12}$$

We refer to the parametrization scheme that defines $k \cdot T_a$ by $d_s - \omega$ longer than T_s as the $PI - kM^+$ scheme (because of the "+"-operation in Equation 4.12), and to the one that sets T_a by $d_s - \omega$ smaller than T_s as the $PI - kM^-$ scheme. We in the following analyze the $PI - kM^+$ scheme in detail. Figure 4.2 shows the intervals and corresponding latencies defined by Equation 4.12 for $k = 1$. As can be seen, these valuations lead to local latency minima.

Choosing $\mathbf{T_s}$

As already mentioned, our goal is to optimize the product of the latency and duty cycle. Figure 4.3 shows $\eta \cdot l^*$ for fixed values of d_s and k. Pareto-efficient values are the ones depicted by the circles, which we attempt to derive for arbitrary values of d_s in the following.

Due to the ceiling-functions in Equations 4.10 and 4.11, multiple values of T_s result into the same worst-case latencies. Since the duty-cycle η decreases for increasing values of T_s, the local optima are given by the values of T_s that maximize the terms within the ceiling functions in Equations 4.10 and 4.11, without causing them to turn over. Hence, we require:

$$\frac{T_s + (d_s - \omega)(1-k)}{k(d_s - \omega)} = M - \epsilon, M \in \mathbb{N} \tag{4.13}$$

ϵ is a number close to zero, as described for the $PI - 0M$ scheme. Solving Equation 4.13 by T_s leads to:

$$T_s = (k(M+1) - 1)(d_s - \omega) \tag{4.14}$$

When combining Equations 4.10, 4.11, 4.12 and 4.14, we can derive the following worst-case latencies for case b) and c):

$$l^*(k, M, d_s) = \begin{cases} (M-1)(M+1)(d_s - \omega) + \omega, & \text{if } k = 1, \\ k(M+1)(d_s - \omega) \cdot \frac{k(M+1)-2}{k} + \omega, & \text{else} \end{cases} \tag{4.15}$$

From Equation 4.1 follows that a protocol with such a parametrization has a duty-cycle of:

$$\eta(d_s, k, M) = \frac{d_s - \omega + k\omega + M(d_s + k\omega)}{(d_s - \omega)(M+1)(k + kM - 1)} \tag{4.16}$$

We can use this to express d_s by the specified target duty-cycle η. Compared to the initial situation, in which we had to choose 3 real-valued parameters T_s, T_a and d_s, we have simplified the problem to choosing two integer numbers k and M. All valuations of k and M lead to local latency minima, of which some have better latency-duty-cycle relations than others. In what follows, we derive the optimal values for k and M.

Optimizing k and M

By applying the theory of γ-sequences described in Chapter 3, one can show that a value of $M = 1$ leads to the situation depicted in Figure 4.1b), in which the offset Φ_i shrinks in steps of γ. For values of $M > 1$, the situation shown in Figure 4.1c) occurs, in which Φ_i grows in steps of γ. We can easily find the values of k that lead to a local latency minimum by solving $\frac{d}{dk}\left(\eta(k, M, \eta) \cdot l^*(k, M, \eta)\right) = 0$ using Equations 4.15 and 4.16 and by expressing d_s by the specified target duty-cycle η. The optimal values of k are given by:

$$k_{opt} = \frac{1}{M+1} + \frac{\sqrt{(1-\eta)(\eta(M+1)+1)} + 1}{\eta(M+1)} \tag{4.17}$$

For optimizing M, we set $k = k_{opt}$ and compute the derivative $\frac{d}{dM}(\eta(k_{opt}, M, \eta) \cdot l^*(k_{opt}, M, \eta))$, which represents the slope of the latency-duty-cycle product. The resulting terms are positive for all possible valuations, and hence low values of M lead to low products $\eta \cdot l^*$. As a result, $M = 1$ is the value with the highest performance, whereas all larger values of M will increase the product of the latency and duty-cycle. We will therefore first analyze the case of $M = 1$ in detail. As we will describe in Section 4.4, the case of $M = 2$ has beneficial properties to handle non-idealities of the radio in practical implementations. Moreover, our evaluations in Section 4.6 show that the difference in performance between $M = 1$ and $M = 2$ is negligible. We therefore also analyze the case of $M = 2$.

The $\mathbf{PI - k1^+}$ Scheme ($\mathbf{M = 1}$)

In the following, we assume $M = 1$ and analyze the resulting properties. We refer to the resulting parametrization scheme as $PI - k1^+$. For a given target duty-cycle η, we can rearrange Equation 4.16 to express d_s as follows:

$$d_s(\eta) = \frac{\omega(2k-1)(1+2\eta)}{2\eta(2k-1)-2}. \tag{4.18}$$

Still, with k, there is one degree of freedom left. We determine its value by analyzing the constraints on d_s. First, $d_s(\eta)$ needs to be larger than zero, and hence from Equation 4.18 follows that

$$k > \frac{1}{2\eta} + \frac{1}{2}. \tag{4.19}$$

In addition, since a practical radio cannot realize a scan window that is shorter than a certain threshold $d_{s,m}$, we require $d_s(\eta) \geq d_{s,m}$. By expressing d_s using Equation 4.18 and by solving this inequality, one can derive that k needs to fulfill the following constraints.

$$\begin{aligned} k \leq k_l, &\quad \text{if} \quad \eta > \tfrac{\omega}{2(d_{s,m}-\omega)}, \\ k \geq k_l, &\quad \text{if} \quad \eta < \tfrac{\omega}{2(d_{s,m}-\omega)}, \end{aligned} \tag{4.20}$$

with

$$k_l = \frac{2d_{s,m}(1+\eta) - \omega(1+2\eta)}{4\eta d_{s,m} - 2\omega(1+2\eta)}. \tag{4.21}$$

For $\eta = \frac{\omega}{2(d_{s,m}-\omega)}$, no constraints on k apply. Therefore, k is chosen as round(k_{opt}) (cf. Equation 4.17), or the value being as closest to k_{opt} as allowed by these constrains.

With this scheme, there are no degrees of freedom left, and T_a, T_s and d_s are fully defined. When comparing Equations 4.19 and 4.20, one can infer that the maximum duty-cycle that can be realized by the $PI - k1^+$-scheme is given by

$$\eta \leq \frac{3\omega + \sqrt{\omega(\omega + 8d_{s,m})}}{8(d_{s,m} - \omega)}. \tag{4.22}$$

Next, we examine parametrizations with $M = 2$.

The PI − k2$^+$ Scheme (M = 2)

Since M has a value of 2, we refer to valuations following this scheme as $PI - k2^+$. As already mentioned, such valuations will result in the situation depicted in Figure 4.1c). As for $M = 1$, k needs to be set to k_{opt} as given by Equation 4.17. However, there are multiple constraints that might prevent setting k to k_{opt}. First, $\eta(d_s)$ needs to be positive, and from Equation 4.16 follows that

$$k > \frac{\eta + 1}{\eta(M+1)} = k_{min}. \tag{4.23}$$

Second, there is a lower limit $d_{s,m}$ on the scan window the hardware can realize, as already described. One can solve Equation 4.16 by k, which results in an upper limit of k as follows.

$$k \leq \frac{d_{s,m}}{3\eta d_{s,m} - (3\eta + 1)\omega} + \frac{1}{3} = k_{max} \tag{4.24}$$

Since k is an integer value, $\lceil (k_{min}) \rceil \leq \lfloor k_{max} \rfloor$ must always be kept. A conservative but analytically solvable form of this inequality is $k_{min} + 1 \leq k_{max} - 1$. By solving this using k_{min} and k_{max} derived from Equations 4.23 and 4.24, we get an upper limit on the duty-cycles that can be realized:

$$\eta \leq \frac{3\omega + \sqrt{\omega(\omega + 8d_{s,m})}}{12(d_{s,m} - \omega)} \tag{4.25}$$

When setting k to round(k_{opt}), or as close to it as allowed by the constraints described above, respectively, all parameters T_a, T_s and d_s are fully defined by η.

4.4 Implementation

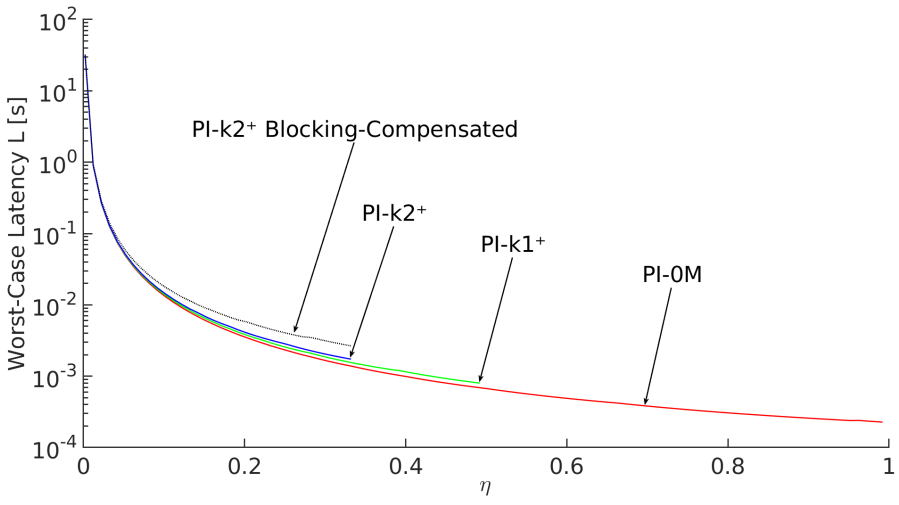


Figure 4.4: Worst-case latencies of our proposed schemes for different duty-cycles ($\omega = 32\mu s$).

This section is dedicated to the implementation of optimized PI-based protocols. Figure 4.4 depicts the worst-case latencies L of the three proposed parametrization schemes $PI-0M$, $PI-k1^+$ and $PI-k2^+$, which are given by $l^* + T_a$. The realization of lower duty-cycles than the ones depicted in Figure 4.4 (i.e., less than 0.2 %) is possible, but the constraints on η, in particular those enforced by Equations 4.9, 4.22 and 4.25, prevent larger duty-cycles from being realized. We assume a packet-length ω of 32 µs, which we will justify in Section 4.6. As can be seen in Figure 4.4, especially for low duty-cycles, all of our proposed parametrizations perform almost identical, with PI-0M providing slightly lower worst case latencies for larger duty-cycles. This difference in performance is due to the following reason: A packet being sent within the last ω time-units of every reception phase cannot be received successfully (since it only partially coincides with the scan window). Nevertheless, also the last ω time-units of every scan window contribute to the duty-cycle. $PI-0M$ concentrates the duty-cycle spent for reception into one large scan window per worst-case latency, whereas $PI-k2^+$ distributes it into 3 relatively smaller ones, which leads to a higher share of “unproductive” reception time. Hence, $PI-0M$ achieves lower relations between duty-cycle and worst-case latency.

In the rest of this chapter, we restrict our considerations to duty-cycles between 0.2 % and 1.55 %, since they lead to a range of worst-case latencies that is relevant in practical applications. In particular, $\eta = 0.2\,\%$ corresponds to a worst-case latency of roughly half a minute, $\eta = 1.55\,\%$ to roughly half a second. Especially for this range of duty-cycles, all of the three parametrization schemes perform nearly identically (cf. Figure 4.4).

Even though PI-based protocols can be implemented easily using hardware timers to schedule T_a and T_s, the hardware properties of the radios, such as turnaround times or clock inaccuracies, impose challenges on the implementation. In the following, we analyze these challenges in detail and propose countermeasures to overcome them.

4.4.1 Implementing the PI − 0M Scheme

In real-world implementations, packet collisions and non-negligible durations to switch from reception to transmission and vice-versa are always present, as shown in Figure 4.5. The rectangle in Figure 4.5a) depicts a scan window and the hatched bars multiple packets of the same device, which are scheduled according to the $PI - 0M$ scheme.

Since $T_a \leq d_s - \omega$, a packet will overlap with every reception window of the remote device. This however also implies that a packet will overlap with every reception window of the same device. In addition, every radio requires a certain amount of time for switching from reception to transmission (d_{rt}) and vice-versa (d_{tr}). For the radio we consider, each of these durations spans approximately 140 µs [53], which makes them large compared to the packet transmission duration of 32 µs. Hence, within d_{rt}, ω and d_{tr} time-units, the radio is unable to receive incoming packets. One can verify from Figure 4.5 that no matter by which amount of time the sequence of transmissions is shifted against the sequence of reception windows on the same device, $d_{rt} + \omega + d_{tr}$ time-units are blocked within every scan window.

In other words, for certain offsets Φ_0 between the first scan window and the first packet of a remote device, a later packet that overlaps with a scan window on the remote device will be sent when the remote radio is unable to receive. Hence, the discovery attempt will fail. The probability of blockage among all possible offsets is

$$p_{blk} = \frac{d_{rt} + \omega + d_{tr}}{d_s - \omega}. \tag{4.26}$$

Within our considered range of duty-cycles between 0.2 % and 1.55 %, the smallest scan window length according to the $PI - 0M$ scheme is 4.225 ms, which corresponds to an

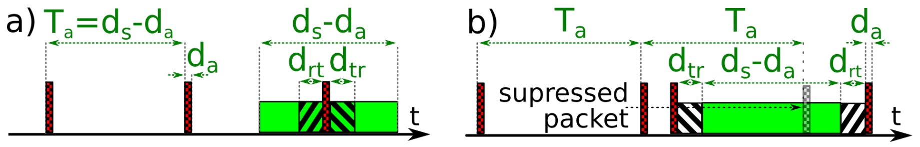


Figure 4.5: a) Blockage of the $PI - 0M$ scheme and b) blockage-compensated $PI - k2^+$ scheme. In the hatched areas, the radio is unable to receive or transmit. In a), these areas lie within the reception window, while they lie outside in b).

unacceptably high blocking probability p_{blk} of 7.4 %. Unfortunately, there is no feasible way of mitigating this for the $PI - 0M$ parametrization scheme.

4.4.2 Implementing the PI – k2$^+$ Scheme

From setting M to 2 in the Equations 4.12 and 4.14, it follows that T_a always has a length of $3 \cdot (d_s - \omega)$ time-units. Hence, only a fraction of the packets need to be sent within a scan window and blocking occurs less frequently. The same holds true for $PI - k1^+$, but since the difference in worst-case latencies between $PI - k1^+$ and $PI - k2^+$ is negligible and the blocking probability of $PI - k2^+$ is lower, we in the following consider the $PI - k2^+$ scheme. In addition, $PI - k2^+$ is eligible for multiple effective countermeasures, which we describe next.

Blocking Mitigation

To reduce the probability of blockage, we propose to suppress the transmission of every packet that lies within a scan window. To compensate for the omitted packets, we send an additional packet by d_{tr} time-units before and another additional packet by d_{rt} time-units after every scan window. The resulting packet flow is shown in Figure 4.5b). The additional packets compensate for the omitted ones for the vast majority of offsets. One can derive that the remaining probability of a failed discovery is given by

$$p_{blk} = \frac{1}{2} \cdot \left(\frac{(d_{tr} + \omega)^2}{T_a T_s} + \frac{(d_{rt} + \omega)^2}{T_a T_s} \right) + \frac{d_{rt} + d_{tr} + 2\omega}{T_s}. \tag{4.27}$$

This equation accounts both for the turnaround times and packet collisions between two devices. Within the considered range of duty-cycles, this reduces the blocking probability to 0.19 % in the worst-case. However, due to the increased duty-cycle for sending two additional packets, the worst-case latency for a given target duty-cycle η is increased by 0.6 % for $\eta = 0.2\,\%$ to 34.7 % for $\eta = 1.5\,\%$.

Clock Quantization Error Correction

Parametrizations following the $PI - k2^+$ scheme will lead to a large number of (short) advertising intervals until discovery is guaranteed. For example, for $\eta = 0.2\,\%$, in the worst-case $999 \cdot T_a$ will pass until discovery occurs. The sleep oscillators of most wireless devices run on very low frequencies to maintain energy-efficiency. For example, for the nRF51822-radio considered in this chapter, the clock frequency f_{clk} is 32 768 Hz [53]. This limits the granularity of T_a and T_s to $1 \cdot T_{clk} \approx 30.5\,\mu s$, and quantization errors can pile up to 30.5 ms after 999 advertising intervals in the worst-case. As a result, the effective value of the offset-shrinkage or growth per scan interval, γ, deviates from its optimal value of $d_s - \omega$ and hence, the deterministic overlap of packet sequences with their corresponding scan windows is not guaranteed anymore. For e.g., $\eta = 0.2\,\%$, $d_s - \omega = 32.2\,\text{ms}$ and hence the maximum accumulated quantization error of 30.5 ms is comparable to this. As our experiments confirmed, this leads to a high number of cases in which the predicted worst-case latencies are significantly exceeded. To overcome this problem, each device must remember the exact values of its advertising- and scan interval in its memory, with a high precision of e.g., 1 ns. Whenever the sleep clock wakes up the CPU, the accumulated quantization error Q is computed by calculating the difference of the time that had passed according to the number of clock ticks and the time that should have passed, based on the exact interval lengths stored in memory. As soon as $|Q|$ exceeds $1/2 \cdot T_{clk}$, the next interval instance is extended or shortened by 1 clock tick, which is again taken into account for computing the next value of Q. With this technique, the resulting effective value of γ cannot deviate by more than $1 \cdot T_{clk}$ from its optimal value. To compensate for the remaining error, d_s needs to be extended by at least $1 \cdot T_{clk}$ beyond its ideal value. We assume an extension of d_s by $5 \cdot T_{clk}$ to account for remaining inaccuracies (i.e., quantization errors and clock skew) in the rest of this chapter. Further, recall that the value of ϵ should be as small as possible. This is given by $\epsilon = 1 \cdot T_{clk} = 32768^{-1}s$ on the nRF51822 radio [53] we consider.

4.5 Configuring Bluetooth Low Energy

Our proposed parametrization schemes can be tuned for parametrizing the BLE protocol, such that discovery latencies and the joint duty-cycle, which is the sum of duty-cycles of both devices, are minimized. Since the specification of BLE does not propose any optimized parameter values, to the best of our knowledge, they form the first known efficient, closed-form parametrization schemes for BLE. We provide a performance analysis of BLE configured using $PI - 0M$ in Section 4.6.7.

4.5.1 Adapting PI – 0M to BLE

Despite the high blocking probability of the implementation of the $PI - 0M$ scheme described in Section 4.3, it has a high practical relevance. The problem of blocking does not occur in one-way scenarios, in which one device broadcasts packets without receiving, whereas the other device only receives without transmitting. Such scenarios are defined by the *non-connectable undirected* advertising procedure of BLE [43]. We in the following first describe how the $PI - 0M$ scheme can be applied to BLE in such one-way scenarios, and then extend it to two-way scenarios.

Non-Connected Undirected Advertising

As already mentioned, BLE adds a random offset between 0 and 10 ms to each advertising interval. The purpose of this delay is to avoid multiple subsequent packet collisions. Further, in each advertising interval, BLE sends up to three consecutive packets on three different channels (viz., channel 37, 38 and 39) in a row. The scanner toggles between these three channels for consecutive scan windows. Hence, 3 packets need to be sent per advertising interval for achieving reliable discovery. Given that these packets are transmitted within a duration of d_e time-units, the effective advertising interval is increased by up to $10\,\text{ms} + d_e$ time-units, as can be seen in Figure 4.6. Here, d_e accounts for the 3-channel discovery, whereas the 10 ms compensate for the largest possible random delay. Recall that $PI - 0M$ guarantees bounded worst-case latencies by requiring the distance between two consecutive packets to be less than or equal to $d_s - \omega$ time-units. Hence, for compensating for the increased effective advertising interval, every scan window needs to be extended by $10\,\mathrm{ms} + d_e$ time-units beyond its optimal value.

However, this extended scan window also increases the duty-cycle, which alters the optimal values for T_a, T_s and d_s. Therefore, this overhead on d_s needs to be accounted for in Equation 4.1. In addition, the three packets sent on different channels per advertising interval incur a certain overhead o_a every T_a time-units. This overhead is given by the time needed to transmit the additional packets, plus the idle times between these transmissions weighted by the reduced power consumption compared to the transmission phases. With this, Equation 4.1 can be written as follows.

$$\eta = \frac{(d_s + 10\,\mathrm{ms} + d_e)}{T_s} + \alpha \cdot \frac{\omega + o_a}{T_a} \tag{4.28}$$

In Equation 4.28, one could further artificially increase T_a by the mean value of the random delay (i.e. 5 ms), which we do not consider further because it only has a negligible impact. From this, T_a, T_s and d_s can be derived as described in Section 4.3.

With the extended scan window, the worst-case latencies for BLE configured by the these values does not differ by more than 10 ms from those of ideal PI-based protocols configured using the same parameter values.

Remaining Advertising Modes

The *non-connectable undirected* advertising mode of BLE implies that a device receiving a packet can neither establish a connection nor request additional data. We next study how the bidirectional modes of BLE, i.e., *connectable* and *scannable* advertising, can be configured using $PI-0M$. Again, our goal is to optimize the latency and joint duty-cycle of both devices.

In such modes, for each advertising packet being sent, the transmitting device needs to listen to the channel for incoming responses 150 µs later. Hence, in addition to o_a, each advertising packet is increased by an effective overhead of o_{a2} time-units that accounts for this reception window, which needs to be accounted for in Equation 4.28. Note that blocking does not occur in such scenarios, since the receiving device only transmits a single packet after it has received a packet from the transmitting device.

4.5.2 Adapting PI – k2$^+$ to BLE

For parametrizations following the $PI - k2^+$ scheme, in general, the same procedure as described above will lead to optimal values of k (and hence, T_a, T_s and d_s) for BLE. However, $PI - k2^+$ guarantees discovery within multiple instances of the scan interval. Recall the distance of any scan window and its neighboring advertising packet on the left is successively reduced by γ time-units after every scan interval T_s. For each such reduction, $n = \lceil T_s/T_s \rceil$ advertising intervals pass. Bounded discovery latencies can be

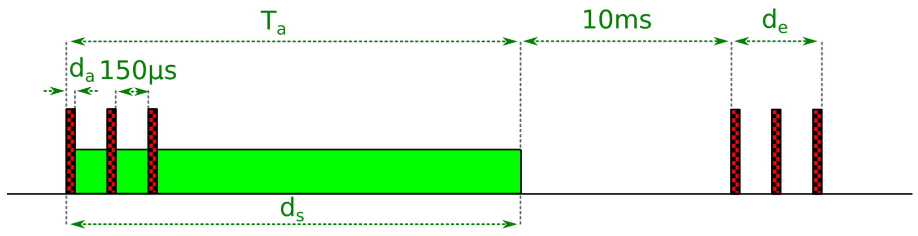


Figure 4.6: The effective advertising interval in BLE is extended by up to $10\,\text{ms} + d_e$ time-units.

guaranteed if $\gamma < d_s - \omega$, and since γ could exceed $d_s - \omega$ because of the random delay, d_s needs to be increased to compensate for this. With a random delay of up to $10\,\text{ms}$ per advertising interval, the sum of these delays can add up to $n \cdot 10\,\text{ms}$ time-units after every n intervals. This sum can easily exceed T_a time-units and hence, a compensation by extending d_s becomes very energy-consuming.

Therefore, the maximum random delay per interval needs to to be reduced to a reasonably low value, e.g., $1/n \cdot 10\,\text{ms}$, compared to the value of $10\,\text{ms}$ suggested by the BLE specification [43]. Though this does not comply to the specification, it remains fully compatible to it.

4.6 Evaluation

In this section, we evaluate the performance (i.e., worst-case latencies, channel utilization and blocking probabilities) of the implementation of the $PI - k2^+$ scheme described in Section 4.4. Whereas we will formally proof that our parametrization schemes achieve theoretically optimal worst-case latencies in Section 4.7, we in this section compare its performance to those of multiple popular previously known protocols and evaluate its behavior in real-world implementations. Towards this, we evaluate the predicted performance by real-world measurements on 560,000 discovery procedures using two nRF51822 [53] radios. Finally, we evaluate the performance of BLE configured using the $PI - 0M$ scheme.

4.6.1 Protocols Considered

In the following, we present a selection of protocols to which we compare our proposed solution. To maintain comparability, we do not consider any indirect approaches, e.g., [82, 22], in which information on future wakeup phases is transmitted along with the packets to accelerate the discovery.

In particular, we consider the following protocols.

• **Disco** [15] is used as the baseline for comparisons in most related work. Though Disco can only function properly with two prime numbers p_1 and p_2, the equations describing the performance in [15] remain valid if we assume $p_1 = p_2$ and also allow non-prime numbers. This allows for a comparison of all duty-cycles, including those that cannot be realized in practice due to the lack of corresponding prime numbers.

Protocol	$L(\eta)$
Disco	$\frac{4}{\eta^2}I$
U-Connect	$(\sqrt{\frac{1}{2\eta}+\frac{9}{16\eta^2}}+\frac{3}{4\eta})^2 I$
Searchlight	$\left\lceil \frac{\lfloor \frac{1}{\eta} \rfloor}{2} \right\rceil I$
Optimal Diffcodes	$\frac{1}{2\eta^2}I$
G-Nihao	$\left(\frac{I+\omega\gamma}{2\gamma\eta I}+\sqrt{\frac{I+\omega\gamma}{2\gamma\eta I}-\frac{\omega}{I}}\right)^2 \gamma$

Table 4.2: Worst-case discovery latencies of slotted protocols.

- **U-Connect** [23] allows for extremely short slot lengths of 250 µs, which leads to high performances.
- **Searchligh-Striped** [2], which achieves low latencies while being capable of realizing a large set of duty-cycles.
- **Optimal Diffcodes** [55] have been proven to achieve the lowest worst-case latencies (in terms of slots) that any slotted protocol could guarantee [87]. Despite only a very limited set of duty-cycles can be realized, we consider their theoretical performance bound for all duty-cycles.
- **G-Nihao** [61] defines listen-only and transmit-only slots. It provides a parameter γ (which is not related to the shrinkage γ introduced in Chapter 3) to adjust the number of beacons per period, but its optimal value is not clear. We therefore assume $\gamma = 2$, as also assumed in [61].

The corresponding worst-case latencies, based on [23, 2, 55, 61], are given by Table 4.2. Here, I is the slot length.

4.6.2 Protocol Parameters

The performance of all ND protocols depends on the packet- and slot lengths, for which we derive reasonable values next.

Packet Transmission Duration ω

Slotted and PI-based protocols perform best for short packets. We therefore assume a short packet length of 4 bytes, which consists of a 1-byte preamble for synchronization and a 3-byte timestamp to schedule a later data exchange. We assume a nRF51822-radio [53] with a bitrate of 1 Mbit/s, which leads to a transmission duration ω of 32 µs.

Slot Length I

Whereas PI-based protocols guarantee discovery within a certain amount of time, slotted protocols guarantee discovery within a worst-case number of slots. Shorter slot lengths would lead to lower worst-case latencies and hence higher performances for the same duty-cycles. A comparison using the slot lengths assumed in the literature would not be fair, since they are based on “good guesses” and have not been chosen to systematically minimize the discovery latencies or failure probabilities. A systematic reduction of the slot length has not yet been studied. Therefore, we need to identify a slot length for which slotted protocols achieve the same properties as PI-based ones. As already mentioned, for the largest duty-cycle considered, implementations following the $PI-k2^{+}$-parametrization scheme will fail (and hence prevent a successful discovery) in 0.19 % of all cases due to packet collisions and blocking. For slotted protocols, this rate grows for decreasing slot lengths, and we in the following identify the slot lengths that lead to the same fraction of failed discoveries as in $PI-k2^{+}$.

For a successful discovery, two slots from two devices have to overlap in time. This also implies that packets from two devices come into temporal vicinity, which makes slotted schemes prone to collisions even when their channel utilization is low. Consider a slot in which a packet is transmitted at its beginning and end, whereas the devices listen to the channel in between. Two such slots from different devices can only overlap, if the difference of their starting times lies within $[-I, I]$ time-units. For certain offsets within this range, discovery will be prevented due to packet collisions, e.g., for the offset 0. In addition, the radio has to switch from reception to transmission and vice-versa, which blocks d_{rt} and d_{tr} time-units in each slot. One can compute the probability of a failed discovery by summing up all offsets that lead to failures and dividing them by the range of offsets with overlapping slots, which leads to the following failure probability for 2 devices discovering each other:

$$P_{blk,disco} = \frac{2 \cdot (3\omega + d_{rt} + d_{tr})}{2I}. \tag{4.29}$$

This slot design is actually used by Disco [15], for which Equation 4.29 gives the probability of failed discoveries. In contrast, Searchlight [2] and optimal difference codes [55] define overflowing slots, in which at least one packet is sent slightly outside of the slot boundaries. Under the assumption that one packet and turnaround phase lie within the slot, whereas another turnaround phase and packet transmission lie outside of the slot boundaries, a reduced probability of

$$P_{blk,overflow} = \frac{2\omega + d_{tr}}{I} \tag{4.30}$$

can be achieved for these protocols. In G-Nihao, there are always m consecutive listen-only slots, which can be regarded as a large, contiguous reception slot. Since packets are also sent with a period of m slots, always one packet transmission duration ω and a pair of turnaround phases d_{tr} and d_{rt} lie within these m slots. This leads to the following probability:

$$P_{blk,nihao} = \frac{d_{rt} + d_{tr} + 2\omega}{m \cdot I} \tag{4.31}$$

As already mentioned, the $PI-k2^{+}$ scheme achieves a blocking probability of 0.19 % for two devices and for the least beneficial duty-cycle considered (i.e., $\eta = 1.55\,\%$).

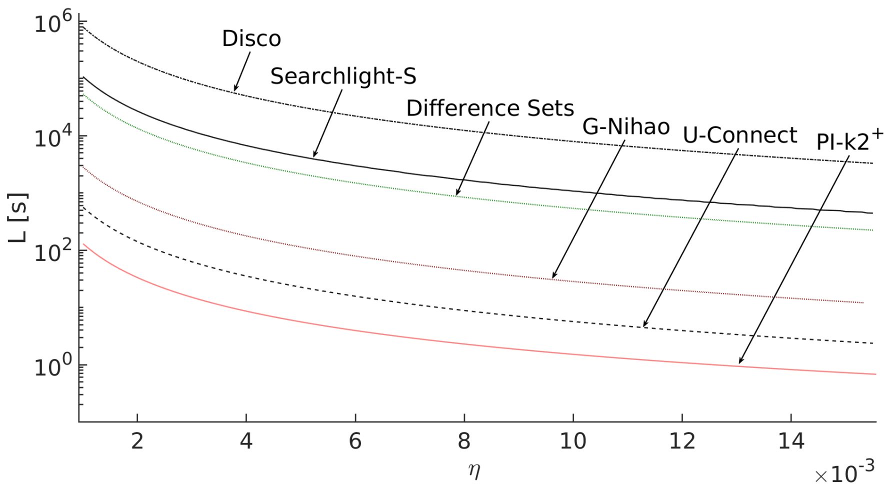


Figure 4.7: Worst case latencies of slotted protocols against the $PI - k2^{+}$ scheme with blocking compensation.

Therefore, in this comparison, we scale the slot length of each slotted protocol, such that a maximum blocking probability of 0.19 % for two devices and for the maximum duty-cycle considered (i.e., 1.55 %) is reached in each of them. This leads to the following slot lengths. Disco: 197.9 ms; Searchlight and Diffcodes: 107.4 ms; G-Nihao: 5.5 ms. Since U-Connect defines special receive-only and transmit-only slots, we here assume $I = 250\,\mu s$, as has been done in [23].

4.6.3 Worst-Case Latencies

In this Section, we evaluate the worst-case discovery latencies of the $PI - k2^{+}$ scheme with all the measures to counteract the hardware non-idealities described in Section 4.4. Further, we compare them to the latencies of previously known protocols, as depicted in Figure 4.7. As can be seen, the slotted protocols Disco, Searchlight and Diffcodes have the highest worst-case latencies. U-Connect achieves lower worst-case latencies due to its separate listen-only and transmit-only slots. The pseudo-slotted protocol Nihao has a significantly lower worst-case latency, but does not reach the performance of the $PI - k2^{+}$ scheme, which provides the lowest worst-case latencies for all duty-cycles. Table 4.3 shows the maximum gains G_m and the mean gains $\overline{G}$ over slotted protocols, defined as $G = \frac{d_{m,slotted}}{d_{m,PI-k2^{+}}}$. For example, for the most beneficial duty-cycle of 0.2 %, in the worst-case, U-Connect would take 4.4 $\times$ longer than the $PI - k2^{+}$ scheme for discovering a neighbor. On the average over all duty-cycles considered, U-Connect would take 4.1 $\times$ times longer. In addition to the results for a maximum rate of failed discoveries of 0.19 %, Table 4.3 shows the results for a rate of 3 % for the highest duty-cycle. This rate is achieved by slotted protocols for two devices by adjusting the slot length accordingly, whereas the $PI - k2^{+}$ scheme fails with this rate when 3 devices come into range. As can be seen, there are still significant gains. These results suggest that the classical slotted protocols Disco, Searchlight and Diffcodes achieve larger worst-case latencies compared to

	$P_{blk} = 0.19\,\%$		$P_{blk} = 3\,\%$	
	G_m	$\overline{G}$	G_m	$\overline{G}$
Disco	6119.1	5663.9	387.5	358.7
Searchlight-S	830.0	768.1	52.6	48.6
Opt. DiffCodes	415.5	384.6	26.8	24.8
G-Nihao	22.0	20.3	1.7	1.6
U-Connect	4.4	4.1	4.4	4.1

Table 4.3: Maximum (G_m) and mean ($\overline{G}$) gains of worst case discovery latencies over slotted protocols.

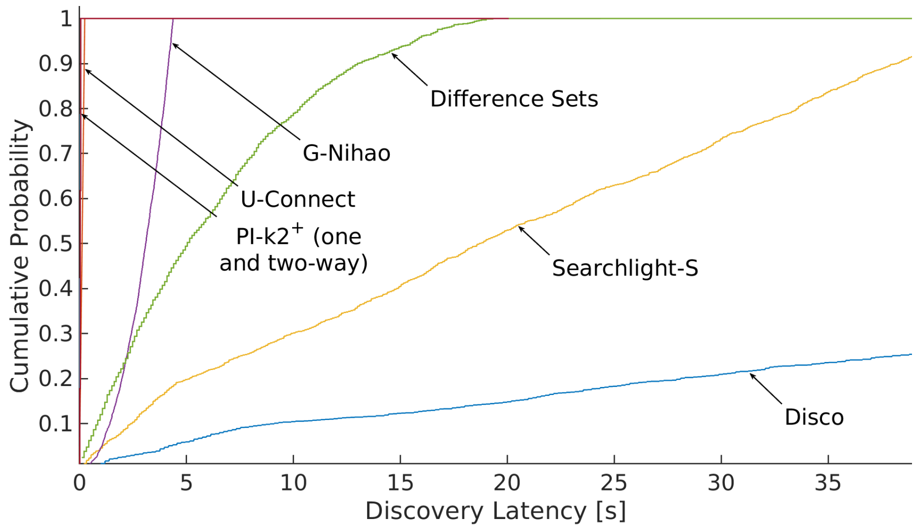


Figure 4.8: CDFs for a duty-cycle of 5 %.

PI-based ones. Further, PI-based ND protocols can achieve significantly lower worst-case latencies by decoupling reception and transmission.

4.6.4 Average-Case Behavior

In what follows, we compare the mean latencies of our proposed scheme against those of previously known protocols. Since the literature does not provide equations on the mean latencies, we have implemented simulation models of all previously known protocols under consideration. We assume a duty-cycle of 5 % for this comparison, which all protocols under consideration can at least approximate closely.

Figure 4.8 shows the cumulative distribution functions (CDFs) of our proposed solution and the previously known ones. We have neglected the small impact on the timing behavior caused by the blocking countermeasures described in Section 4.4. For our proposed scheme, both the CDF-curves for one-way discovery (i.e., a device A receives a packet from device B) and for two-way discovery (i.e., device A receives a packet from

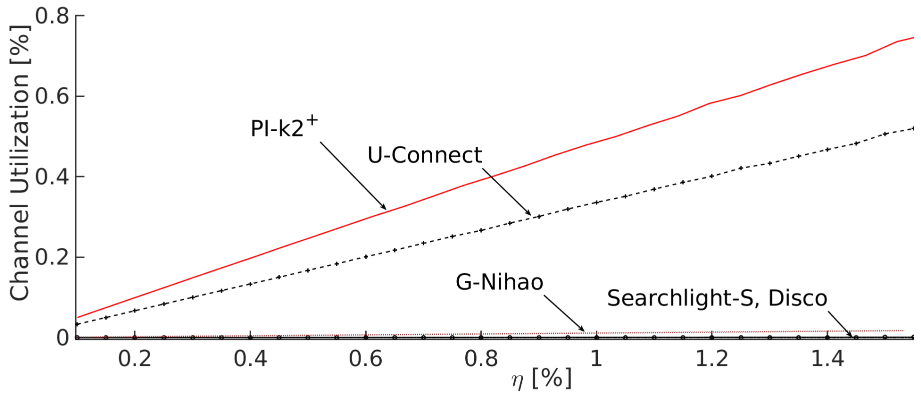


Figure 4.9: Comparison of channel utilizations.

device B and vice-versa) are depicted, but both curves lie in such a close proximity that hardly any difference is visible. The remaining depicted CDF curves represent two-way discoveries.

Our proposed scheme achieves the lowest mean latencies among all protocols considered. For a duty-cycle of 5 %, the mean latencies are as follows: Disco: 100.74 s; Searchlight-Striped: 19.61 s; Difference Sets 6.11 s; G-Nihao 2.93 s; U-Connect: 0.12 s; $PI - k2^{+}$, two-way 0.04 s; $PI - k2^{+}$, one-way: 0.03 s.

4.6.5 Channel Utilization and Collisions

A comparison of channel utilizations is depicted in Figure 4.9. The $PI - k2^{+}$ scheme adjusts the transmission rate for optimal latency-duty-cycle relations, which leads to an increased channel utilization. Recall that our proposed schemes target scenarios with few nodes being in discovery mode simultaneously, and we in the following establish that the resulting collision rates remain low in the scenarios considered. As already described, for 2 devices, the blocking mitigation techniques presented in Section 4.4 lead to a probability of blockage and collisions of up to 0.19 %. This is achieved by keeping all reception phases free of packet transmissions, thereby preventing failures due to collisions. When more than two devices are in range, collisions will occur regardless of this. Since the offsets of packets from different devices are usually distributed uniformly, their collision probabilities are exponentially distributed (cf. [37] for details). Therefore, starting from $nDevices = 3$ devices, the discovery procedure of each device will collide with a probability of

$$p_{col} = 1 - e^{-2(nDevices-1)\cdot(\frac{\omega}{T_a}+2\frac{\omega}{T_s})}. \tag{4.32}$$

For 3 devices, the collision probability is around 0.5 % for $\eta = 0.2\,\%$ and around 3 % for $\eta = 1.55\,\%$, as we had assumed in our comparison against slotted protocols. For 10 devices, the collision probability is around 2 % for $\eta = 0.2\,\%$ and reaches almost 13 % for $\eta = 1.55\,\%$. Therefore, our proposed scheme is well suited for networks with up to 10 devices being in discovery mode simultaneously.

4.6.6 Experimental Latency Measurements

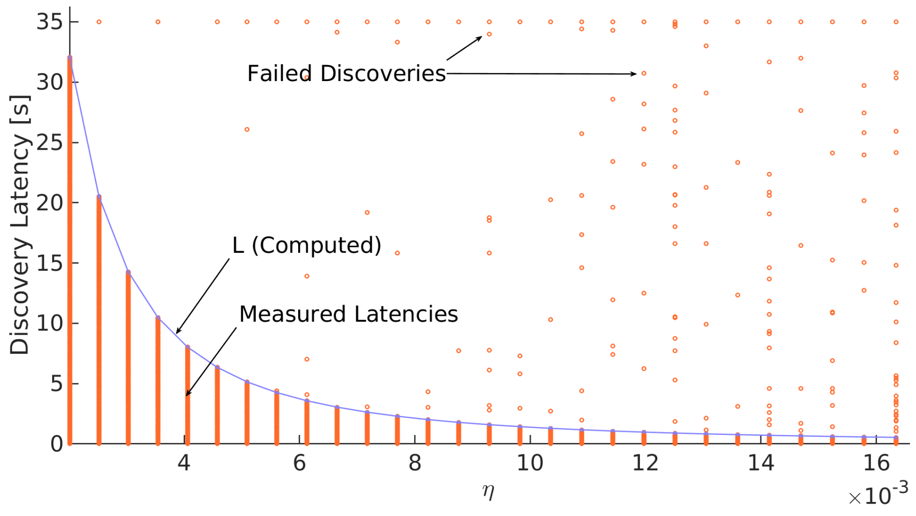


Figure 4.10: Measured discovery latencies (scattered points) and computed upper bounds (solid line).

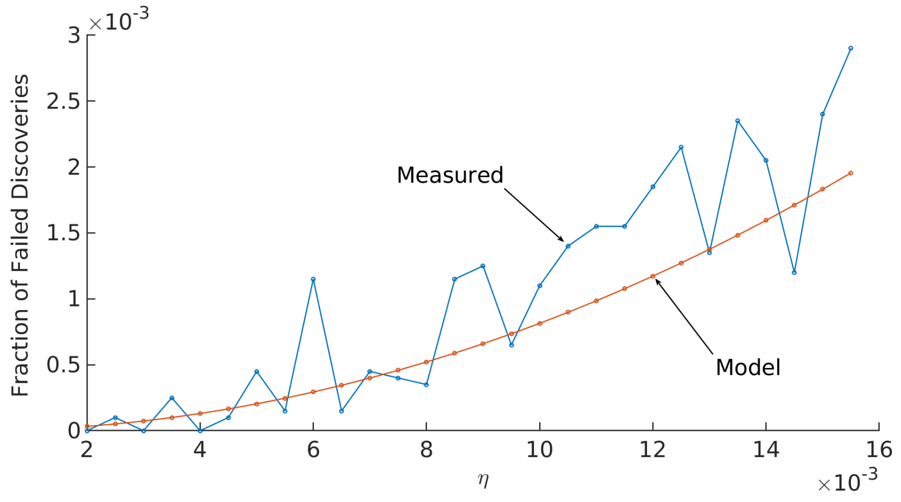


Figure 4.11: Predicted and measured fraction of failed discoveries.

To demonstrate that our proposed ND solution can be realized in practice, we have implemented the $PI - k2^+$ scheme, as described in Section 4.4, on two wireless radios. Based on the open-source BLE stack Blessed [44], we have created a custom firmware for two nRF51822-radios. The radios were connected to an ARM Cortex M4 microcontroller via UART. The purpose of this microcontroller was to start and stop the radios, to send them the appropriate parametrizations, to obtain reports on the received packets and to measure the discovery latencies. Both radios have been located in close proximity within an RF-shielded, anechoic box. In multiple runs, the radios repeatedly discovered each other using the $PI - k2^+$ scheme with blocking compensation. After each discovery attempt, the devices have been desynchronized by a random waiting time. In each measurement, both devices were started such that every two neighboring scan windows on two devices had a random time offset within $[0, T_s]$ time-units between each other. Similarly, any pair of packets on two devices had a random offset within $[0, T_a]$ time-units between each other. After both devices had either discovered each other, or after a timeout of 35 seconds (which exceeded L for all duty-cycles) was reached, the radios were stopped, the measured latencies were logged on a laptop and the next measurement round was initiated. For each duty-cycle, the experiment was repeated $10,000$ times, leading to $20,000$ one-way discoveries. We have considered 28 different duty-cycles between $0.2\,\%$ to $1.55\,\%$, which resulted in $560,000$ measured discovery procedures.

The measured discovery-latencies are shown in Figure 4.10 together with the computed upper limits $L(\eta)$. Each scattered point represents a measured latency of one discovery procedure, whereas the solid line depicts the upper bound predicted by our theory. As can be seen, the measured latencies always lie below the theoretic upper bound, except for a few collided measurements.

Figure 4.11 shows the fraction of discoveries that have exceeded the predicted worst-case bound L by more than $1\,\%$. Deviations below $1\,\%$ have been considered as measurement inaccuracies. In addition, the predicted fraction of failed discoveries from Equation 4.27 is shown. As can be seen, the measurements match the predicted values well. For a duty-cycle of $0.2\,\%$, all $20,000$ discoveries were successful, whereas the maximum number of failures was 58 for $\eta = 1.55\,\%$. This corresponds to the failure rate of $0.29\,\%$ depicted in Figure 4.11. These results show that our proposed $PI - k2^+$ scheme reaches the predicted performance in practice.

4.6.7 Performance of BLE

In Section 4.5, we have described how our proposed parametrization framework can be used to optimize BLE. Recall that BLE incurs a random delay being added to each instance of T_a and requires a reception phase after each transmission in the case of bi-directional discovery. This affects its latency-duty-cycle performance. In the following, we evaluate how the performance of BLE configured using the $PI-0M$ scheme compares to an ideal PI-based protocol configured using the $PI-0M$ scheme (i.e., $PI-0M$ without any overheads of and modifications for BLE).

For this evaluation, we assume the following overheads for BLE: $o_a = 619\,\mu\text{s}$, $o_{a2} = 143\,\mu\text{s}$, $o_s = 11\,\text{ms}$. They result from the following assumptions:

- We assume a packet length of 30 bytes, which is a realistic value for BLE (e.g., for a location beacon).
- We assume that any two consecutive packet transmissions on two different channels are spaced from each other by 150 us. Further, we assume that the radio consumes the same power at the time between two consecutive transmissions as for switching from transmission to reception.
- We assume that the power consumption for transmission is identical to that for reception
- We assume the following values taken from the literature [28] for a BLE radio: Idle-

listening (i.e., the short listening phase after transmitting a packet) takes 74 µs and the quotient of the power consumption for transmission over that for switching from reception to transmission has a value of 0.46.

- We study a range of duty-cycles between 1 % and 10 %.

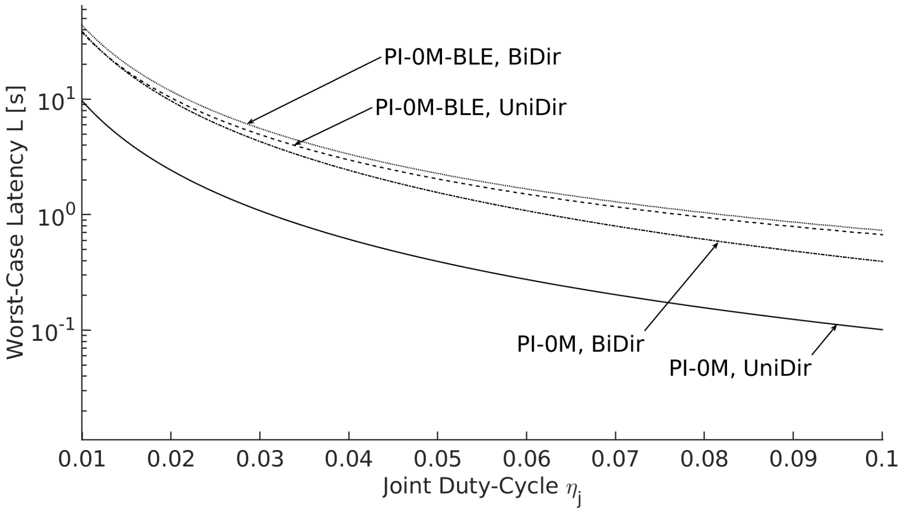


Figure 4.12: Performance of BLE configured using $PI - 0M$.

Figure 4.12 depicts the worst-case latencies of BLE configured using the $PI - 0M$ scheme, both for the unidirectional and bidirectional advertising modes. In addition, the worst-case latencies of $PI - 0M$ without the modifications for BLE are shown. The depicted duty-cycle η_j is the joint duty-cycle of both devices, i.e., the sum of the duty-cycles of two devices that carry out the discovery procedure. For unidirectional discovery, one can see that the worst-case latencies of BLE are increased compared to the latencies obtained by protocols applying the original $PI - 0M$-scheme. On the average, for unidirectional discovery, BLE has a 5.3× larger worst-case latency than the original $PI - 0M$ scheme. For bidirectional discovery, BLE on the average has a 1.5× longer worst-case latency.

4.7 Proof of Optimality

In this section, we formally proof that the discovery latencies achieved by PI-based protocols parametrized using the $PI - 0M$-scheme are actually optimal. In particular, we show that the resulting discovery latencies are identical to the fundamental performance bound for symmetric ND procedures that we have derived in Chapter 2. This implies that no other direct ND protocol - regardless of its underlying principle - can guarantee shorter worst-case latencies than a PI-based one configured using the $PI - 0M$-scheme. Towards this, we apply the same assumptions under which the fundamental bounds from Chapter 2 are valid to the $PI - 0M$ protocol. Under these assumptions, the equation for the worst-case latency of $PI - 0M$ becomes identical to the theoretic optimum, which is given by Equation 2.11. For duty-cycles for which ${}^{2}/_{\eta}$ is an integer and for $\alpha = 1$ (i.e., for the same power consumption for reception and transmission), this bound becomes:

$$L = \frac{4 \cdot \omega}{\eta^2} \tag{4.33}$$

The worst-case latency of $PI - 0M$, as given by Equation 4.2, is equivalent to the latency measured from the first packet that is sent after two devices come into range to the earliest packet that is received by the remote device. This packet-to-packet worst-case latency is relevant when a device is switched on at a random point in time and begins the discovery procedure by immediately sending its first beacon. In contrast, in Chapter 2, we have considered the latency until discovery is guaranteed starting from the point in time at which two already running devices have been brought into range, which is in general before the first packet is sent. Since the first packet might be sent by up to T_a time-units after the devices have come into range, Equation 4.2 becomes:

$$L = \left(\left\lceil \frac{T_s - d_s + \omega}{T_a} \right\rceil + 1 \right) \cdot T_a + \omega \tag{4.34}$$

As for the bounds presented in Chapter 2, we assume that 1) packets that are sent within the last ω time-units of each reception phase are received successfully (i.e., a reception phase takes effectively d_s instead of $d_s - \omega$ time-units) and 2) that the transmission duration of the last, successful packet is neglected. Under these assumptions, we can set $\omega = 0$ in Equation 4.34 and obtain the following worst-case latency of $PI - 0M$:

$$L = \left(\left\lceil \frac{T_s - d_s}{T_a} \right\rceil + 1 \right) \cdot T_a \tag{4.35}$$

Further, the $PI - 0M$-scheme requires that $T_a = d_s$ and $T_s = (M+1)d_s - \Delta$, $M \in \mathbb{R}$ (cf. Equation 4.3), with $\Delta \to 0$. This leads to a worst-case latency L of $\frac{\omega(M+1)^2}{\eta(M+1)-1}$ time-units. The optimal value of M is obtained by solving ${}^{dL}/_{d_M} = 0$ by M, which leads to an optimal value $M_o = {}^{2}/_{\eta} - 1$. Since M must be an integer value (cf. Section 4.3.2), we set $M = \text{round}(M_o)$.

Let us first consider duty-cycles η for which M_o fulfills $M_o = \text{round}(M_o)$. With $M_o = {}^{2}/_{\eta} - 1$, the worst-case latency of $PI - 0M$ L becomes ${}^{4\omega}/_{\eta^2}$. This is identical to the fundamental performance bound from Chapter 2 that no ND protocol can beat (cf. Equation 4.33) and hence, for the considered set of duty-cycles, the $PI - 0M$ scheme is optimal in the latency/duty-cycle metric.

One can easily show that some duty-cycles for which $M_o \neq \text{round}(M_o)$ lead to latencies slightly above the optimum. However, the following modification of PI-$0M$ always results in optimal latencies, also for such duty-cycles. When rounding M from Equation 4.6 up to its next higher integer-value, the following latency is obtained:

$$L_a = \left\lceil \frac{2}{\eta} \right\rceil^2 \cdot \frac{\omega}{\eta \left\lceil \frac{2}{\eta} \right\rceil - 1} \tag{4.36}$$

similarly, when rounding M_o down to its next lower integer value, the latency becomes:

$$L_b = \left\lfloor \frac{2}{\eta} \right\rfloor^2 \cdot \frac{\omega}{\eta \left\lfloor \frac{2}{\eta} \right\rfloor - 1} \tag{4.37}$$

If we now set $M = \lceil M_o \rceil$, if $L_a \leq L_b$, or $M = \lfloor M_o \rfloor$ otherwise, the resulting latency will always be identical to $\min(\upsilon_a, \upsilon_b)$. This worst-case latency is identical to the fundamental bound for symmetric ND, as given by Equation 2.11.

While we have proven the optimality under the assumptions describe above, by how much is the worst-case latency increased in practice when relaxing these assumptions? We consider a radio without switching overheads. When assuming a packet transmission duration of $\omega = 32\,\mu s$ and a range of duty-cycles between 0.1 % and 100 % in steps of 0.1 %, the normalized root mean square error between the fundamental latency bound and the actual performance of $PI - 0M$ as given by Equation 4.2 is 1.2 %.

We do not formally study the optimality of $PI-kM^+$ in detail. However, the negligible difference in performance compared to $PI-0M$ implies that also $PI-kM^+$ is essentially optimal.

4.8 Concluding Remarks

We have introduced a novel parametrization scheme for slotless, PI-based discovery protocols. Since such protocols can make use of more degrees of freedom than slotted ones, they can optimize their packet transmission rate and achieve significantly lower discovery latencies in scenarios with few devices discovering each other simultaneously. In addition, unlike most previously proposed deterministic protocols, PI-based ones can realize practically every specified duty-cycle. Therefore, they are a practical choice for many personal area networks or IoT scenarios. Further, we have not only shown that the resulting parametrizations are optimal, but also that no other ND protocol can guarantee shorter worst-case latencies than PI-based protocols configured using our proposed scheme. Finally, our scheme can also be used for parameterizing BLE, which makes it highly relevant in practical applications.

Whereas we have optimized the worst-case latencies, there is potential to further improve the mean discovery latencies. In the next chapter, we present and analyze a technique for reducing the mean latencies of PI-based protocols.

5 Mutually Assisted Periodic Interval-Based Neighbor Discovery

5.1 Introduction

In the previous chapter, we have established that optimized PI-based protocols can achieve lower worst-case latencies than all previously known protocols. In particular, unlike all slotted protocols, they can perform optimal in the latency/duty-cycle design space. The main reason for this is that, unlike in slotted protocols, sending beacons and listening on the channel are temporally decoupled from each other. Therefore, devices can send the optimal number of beacons at the optimal points in time, whereas in slotted protocols, the number of beacons is limited by the number of active slots.

In slotted protocols, the discovery of device F by E takes place within the same slot as the discovery of E by F. Therefore, the mean latency for two-way discovery is identical to that for one-way discovery. In contrast, in slotless protocols, the mean discovery latency for two-way discovery is significantly larger than for one-way discovery, even though the worst-case bound is equal for both cases. Because both devices follow the same schedule, the discovery of device E by device F occurs independently from the discovery of device F by device E. The points in time both devices come into range, and hence the initial temporal offsets between their packets and scan windows, are random. Since the two-way discovery procedure is only complete if both of the two independent probabilistic processes have finished, the two-way latency is defined by the maximum latency of the two procedures. Therefore, the mean two-way discovery latency is significantly larger than the mean one-way latency.

In this chapter, we propose a technique to decrease the mean two-way discovery latency of symmetric slotless protocols, while still maintaining a similar worst-case latency to the protocols described in the previous chapter. Unlike in slotted protocols, there are two independent chances on a successful packet reception in each period (viz, the reception of a packet from device F by device E and vice-versa). The main insight which we exploit is that the mean two-way discovery latency can be reduced by combining both chances, if both devices carry out the discovery procedure in a cooperative fashion.

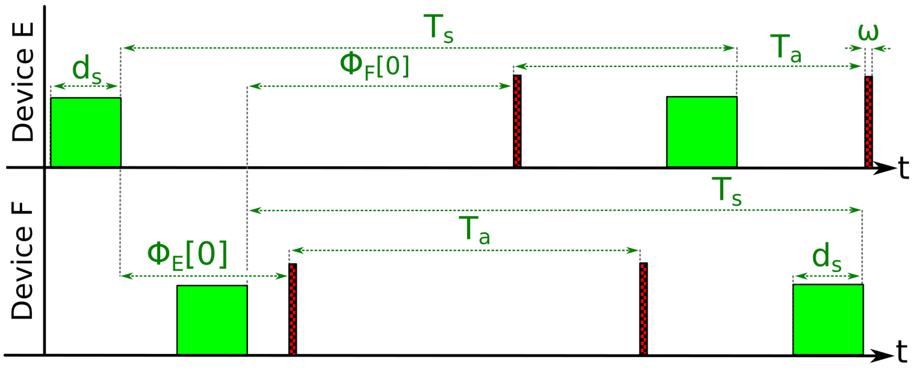


Figure 5.1: Periodic interval-based discovery

Consider two devices E and F, as depicted in Figure 5.1. As already mentioned, the initial temporal distance $\Phi_E[0]$ or $\Phi_F[0]$ between the first scan window of a device E or F and its temporally left neighboring advertising packet of the remote device is a random variable that represents the point in time at which the two devices come into range. Even though one device most likely discovers its neighbor before being discovered by the opposite one, the total latency for two-way discovery remains the maximum of both latencies. If the earlier of both discoveries could be exploited to accelerate the discovery of the second rendezvous, the total mean latency for mutual discovery could be reduced considerably. In this chapter, we propose *Griassdi*[1], a protocol that systematically exploits the available temporal correlations in discovering device F by device E (or vice-versa), given device F has already received a packet from device E. Towards this, every packet sent contains a hint on the amount of time after which the next reception phase of device F takes place. A device receiving such a hint will adapt its beaconing schedule appropriately to send a packet within the reception period of the opposite device. We call this technique to reduce the mean discovery latency *assisted two-way (A2W)* discovery. In slotted protocols, both devices discover each other within the same slot and therefore, the problem of optimizing the two-way latency does not arise.

The technique of mutual assistance has already been described in Chapter 2, where it has been studied for achieving two-way discovery using correlated schedules. In such schedules, two-way discovery could not be achieved without mutual assistance. In contrast, in this chapter, we study mutual assistance in independent schedules, which would also guarantee two-way discovery without this technique. Our goal is not to obtain better worst-case-latencies for a given duty-cycle, but to improve the mean latencies while maintaining a similar worst-case performance. The analysis of such protocols is challenging, since assisted two-way discovery imposes a large number of temporal

[1]The term “Griaß Di” means “hello” in some parts of Southern Germany and Austria, viz., Allgäu and Tyrol.

constellations that need to be accounted for by an appropriate combinatorial model. We present a precise theory for modeling such protocols in this chapter. Based on this theory, we show that the original schedules from the $PI - 0M/PI - k2^{+}$ - schemes described in Chapter 4 are not well-suited for decreasing the mean discovery latencies using A2W techniques. However, by using appropriate adjustments of these schemes, the mean latency can be significantly decreased when allowing a small increase of the worst-case latency. Based on this, we propose optimized A2W-schedules, which reduce the mean discovery latencies by up to 43 % compared to non-assisted solutions, without increasing their worst-case latencies by more than 7 %. The resulting mean latencies for a given duty-cycle are significantly lower than the ones achieved by all known discovery protocols, including all slotted ones. For example, Griassdi achieves by 87 % lower mean latencies than Searchlight [2] for a slot length of 720 µs, while at the same time achieving similar worst-case latencies to the $PI - kM^{+}$-protocol described in Chapter 4, which guarantees the best latency/duty-cycle relations of all known protocols in the worst-case.

Our Contributions: Compared to existing work, we make the following contributions.

- We propose the first slotless, PI-based protocol that systematically exploits mutual assistance to reduce the mean discovery latencies in an optimized manner.
- We provide a theory for analyzing PI-based, slotless A2W protocols, which is capable of computing their mean and worst-case discovery latencies.
- We evaluate the performance of our proposed protocol, compare it to existing solutions and demonstrate its proper functioning using a real-world implementation.

The rest of this chapter is organized as follows. In Section 5.2, we present an overview of related works. In Section 5.3, we propose a high-performance A2W-based protocol. We describe a precise mathematical model for its analysis in Section 5.4. In Section 5.5, we evaluate the performance of this protocol and compare it to existing approaches. We present an implementation on a radio in Section 5.6 and show that the protocol behaves as predicted by the theory using comprehensive experimental data. Finally, we conclude this chapter by outlining some directions for future work in Section 5.7.

As in the previous chapters, unless explicitly stated, we always consider packet-to-packet discovery latencies.

5.2 Related Work

In this section, we give an overview of existing work on ND protocols that apply A2W.

A cooperative approach that broadcasts information on already discovered neighbors is *Acc* [82]. Unlike our approach, Acc is built upon slotted protocols such as e.g., Disco [15], in which mutual discovery between two nodes is always achieved within the same slot. Considering such slotted protocols, Acc can therefore only accelerate the discovery of additional nodes. However, Acc could potentially also be used on top of other protocols, thereby providing speedups also for pairwise discovery.

The recently proposed protocol Lightning [76] applies mutual assistance to achieve two-way discovery, but is not optimized to systematically exploit this for short mean latencies. Further, there is no known real-world implementation of it.

There are other known collaborative approaches that apply mutual assistance, e.g., [60]. However, to the best of our knowledge, none of them studies mutual assistance to improve the mean latencies of an independent, PI-based schedule, as studied in this chapter. In contrast, they instead all consider improving the worst-case latencies. Moreover, most of them consider slotted paradigms.

5.3 Mutually Assisted Neighbor Discovery Protocols

In this section, we describe our proposed discovery protocol *Griassdi* and introduce the technique of A2W discovery.

5.3.1 Protocol Definition

PI-Based Discovery

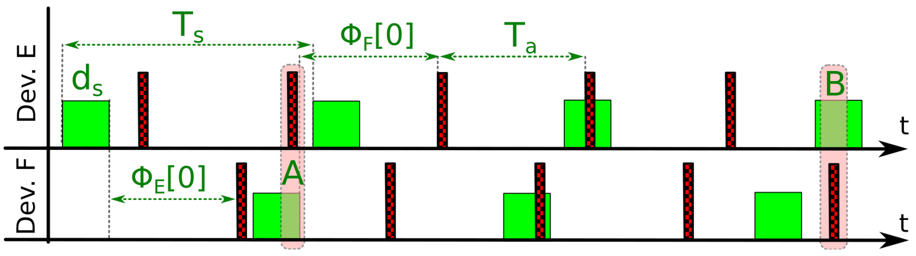


Figure 5.2: Non-assisted PI discovery

Griassdi applies a PI-based scheme, as shown in Figure 5.2. The colored boxes depict the reception windows, whereas the hatched vertical bars show the advertising packets sent. Each device broadcasts packets with an interval T_a. In addition, each device periodically scans the channel with another interval T_s for d_s time-units. If an advertising packet is to be sent while listening, the listening-phase is interrupted for the duration of the transmission, without extending its length.

For the sake of simplicity of exposition, we assume one pair of devices in the rest of the chapter. However, our proposed protocol can also be applied for mutual discovery among a larger number of devices. We further assume that each advertising packet contains some payload (such as device addresses, device types and indicators for the services offered). Two-way discovery occurs once device F has received a packet from device E (marked with A in Figure 5.2) and device E has received a packet from device F (marked with B in the figure). Clearly, the receptions of both packets occur independently from each other. Therefore, the time until both devices have mutually discovered themselves is much larger than the time for one-way discovery. In the example depicted in Figure 5.2, device F discovers device E during its first scan window. The latency for one-way discovery is below $1 \times T_s$, whereas the latency for two-way discovery is almost $4 \times T_s$.

Parametrization

Quick Reference of Symbols

η	Duty-cycle a protocol realizes
k, M	Integers to parametrize T_a and T_s
M_{opt}, k_{opt}	Optimal values of k and M
R	Balancing variable for trading worst-case against mean latencies
$d_{s,m}$	Lower limit of the scan window d_s

A full table of symbols is given in Appendix A.

In what follows, given a certain target duty-cycle η that is to be realized and a packet length ω, we derive efficient values of T_a, T_s and d_s. To exploit mutual assistance, the discovery procedure must take place within multiple scan intervals. Protocols that guarantee discovery within 1 interval cannot benefit from mutual assistance, since the additional packets would be sent within scan windows in which regular packets would have been received, anyway. Unlike the parametrization schemes described in the previous chapter, our proposed parametrizations can realize discovery within a certain, adjustable number of scan intervals, while still providing optimized latencies. This allows for a systematic exploitation of mutual assistance to reduce the mean discovery latencies. Building upon the $PI-kM^+$ scheme presented in Chapter 4, we again consider two integer values k and M. However, in Griassdi, we require that k divides $M+1$. To achieve short worst-case latencies, we choose the scan interval T_s and the advertising interval T_a as

$$T_s = M \cdot (d_s - \omega) - \epsilon, \tag{5.1}$$

$$T_a = \left(\frac{M+1}{k} \cdot (d_s - \omega)\right) - \frac{\epsilon}{k}. \tag{5.2}$$

Equations 5.1 and 5.2 form a configuration in which a certain integer-multiple k of advertising intervals is slightly shorter (viz. by ϵ time-units) than $d_s - \omega$. Therefore, after each scan interval, the temporal distance from any packet to the next scan window is reduced by approximately $\gamma = d_s - \omega$ time-units. This value is optimal, since larger values would lead to misses of the scan window, whereas shorter values would increase the duty-cycle without decreasing the worst-case latency. Further, this parametrization effects that an integer-number of γ-intervals fits into each scan interval, as required by Theorem 2.6. This is efficient, since for a slightly larger, non-integer number of intervals, an additional full γ-interval would be needed to reach the scan window (which incurs multiple additional advertising intervals), while only extending the scan interval (and hence, the duty-cycle) by less than γ time-units.

The configuration $k \cdot T_a = (M+1)(d_s - \omega)$ defines a border-case, since the growth of the temporal distance γ is exactly $d_s - \omega$ time-units, and infinitesimally larger values of γ would lead to misses for some offsets. To achieve a well-defined behavior, T_a and T_s need to be somewhat shorter than their ideal values, which is achieved by subtracting ϵ in the Equations above. In the following sections, we assume a small value of e.g. $\epsilon = 0.1\,\mathrm{ps}$ for our theoretic analysis. In practical implementations, ϵ is not relevant, since d_s will be artificially extended to compensate for clock skew, which also avoids this border-case.

In addition to k and M, we introduce a third integer-parameter $R = \frac{M+1}{k}$. As we describe later, R is used to balance the worst-case discovery latency against the mean discovery latency, since R is identical to the number of scan intervals that pass in the worst-case until rendezvous. If $R = 1$, the discovery procedure will terminate within 1 interval, and no speedup can be achieved by mutual assistance. Though the latency/duty-cycle product is influenced by R, there is no linear relation to it, since the protocol adapts all parameters accordingly to any value of R. R can be chosen by the user, and there is an optimum value for low mean latencies. We will study the impact of R in detail, later.

As already discussed, the main metric for the energy-consumption of ND protocols is the duty-cycle η, which represents the fraction of time the device is active during operation [23]. By computing the duty-cycle (cf. Equation 5.4) and the worst-case latency according to the model presented in Section 5.2, one can show that for realizing a certain duty-cycle η, for a given value of R, the worst-case latency has a local optimum at

$$M_{opt} = \left\lceil \frac{\sqrt{R \cdot \eta + 1}\sqrt{1 - \eta} + 1}{\eta} \right\rceil. \tag{5.3}$$

We further require that R divides $M + 1$ and therefore round M to the next higher value of M_{opt} that is, when incremented by 1, divisible by R. Hence, for a given value of R, the values of M and k are fully determined by the equations above. The only degree of freedom left is d_s. It is defined by the target duty-cycle η the protocol has to realize. As can be verified easily, the duty-cycle of the proposed protocol is

$$\eta = \frac{\omega T_s + d_s T_a}{T_s T_a}. \tag{5.4}$$

We derive from Equations 5.1 and 5.2 that a given target duty-cycle η is realized, if

$$d_s = \omega + \frac{\omega + M\omega(k + 1)}{(M\eta - 1)(M + 1)}. \tag{5.5}$$

In practical implementations, one would limit d_s to a minimum value $d_{s,m}$. While in theory, every value greater than ω is feasible, a certain lower bound, e.g. $d_{s,m} = 10 \cdot \omega$ should be maintained to achieve a high matching probability of additional packets for mutual assistance (see below) and also low collision probabilities (cf. Section 5.5.2).

In summary, using the equations above, given a certain packet length ω, a target-duty-cycle η and a value of R, the equations above provide optimized values of T_a, T_s and d_s. The packet length ω and the target duty-cycle can be chosen freely (within certain feasible bounds). The value of R allows for a trade-off between worst-case and mean latency, which we describe in detail in Section 5.5.

A2W Discovery

To accelerate the two-way discovery, we extend this protocol as follows. Every packet sent by each device contains, as a hint for the remote device, the time-period until the center of its temporally next listening-period. Once a device receives an advertising packet from the

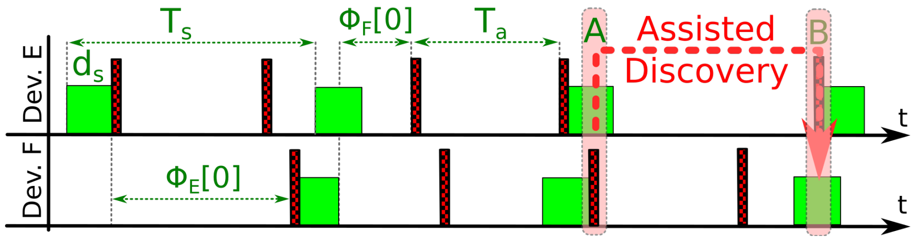


Figure 5.3: Mutually assisted neighbor discovery

remote side, it schedules an additional packet according to the hint received. Therefore, the additional packet directly hits the center of the temporally next scan window of the opposite device. With this scheme, device E assists device F to discover device E and vice-versa. Since the probability of a match between an advertising packet and scan window is greatly increased due to coupling both parallel probabilistic processes, the mean latency is reduced. A discovery-procedure using this technique is shown in Figure 5.3. Device E receives a packet from device F at the point in time marked with A. Based on the information received, it schedules its next packet such that it will meet the next scan window of device F at time B. While the mean latency for the first rendezvous between the two devices remains constant, the second rendezvous takes place guaranteed within one scan interval T_s after the first rendezvous. Therefore, whenever there is more time than T_s time-units between the first and the second rendezvous, the discovery latency is reduced.

5.4 Protocol Analysis

In Chapter 3, we have described a model for computing the one-way discovery latency of PI-based protocols. However, Algorithm 3.1, which computes the discovery latencies of PI-based protocols, is only capable of computing the global mean- and maximum latencies. It cannot compute the latency given a certain offset between an advertising packet and a scan window explicitly. In this Section, we present an modified version of Algorithm 3.1, such that all partition borders and the corresponding latencies are computed explicitly. This will serve as a starting point for mathematically describing mutual assistance techniques.

The theory that is underlying the model from Chapter 3 is based on describing a single probabilistic process. For two-way discovery, two such processes need to be considered in parallel, and both of them are independent from each other. For assisted two-way discovery, these processes are not independent, and therefore the modeling becomes more involved. In this section, we extend the one-way theory to account for two-way discovery, including A2W discovery.

In the following, we first present an algorithm to compute the complete set of partitions and their corresponding one-way latencies, which we call an *explicit model*, because the borders and latencies of all partitions are computed explicitly. We then extend this explicit model towards non-assisted and assisted two-way discovery.

5.4.1 Explicit Latency Modeling

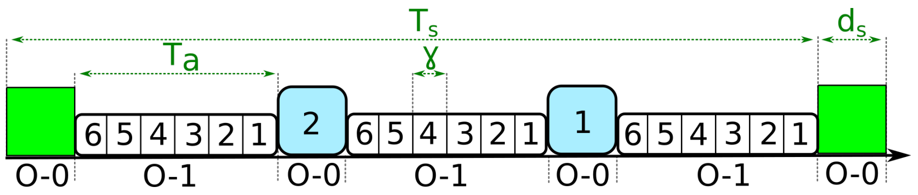


Figure 5.4: Modeling of PI-based protocols

The discovery latency of our proposed protocol is determined by the initial offset $\Phi[0]$ between the first advertising packet and its temporally left neighboring scan window of the remote side. The range of possible initial offsets is $[0, T_s]$, since all other offsets can be described as periodic repetitions of this. As already mentioned in Chapter 3, this range can be subdivided into multiple partitions. The discovery latency is determined entirely by into which partition the initial offset falls, whereas the actual point in time within each partition does not affect the latency (cf. Figure 5.4). In a set of partitions Π, each partition $i \in \Pi$ consists of two borders $t_s[i] < t_e[i]$ and a corresponding discovery latency $l[i]$. The purpose of our proposed model is to compute the complete set of partitions Π for a given triple of parameters (T_a, T_s, d_s). This set of partitions represents the discovery latencies for all possible initial offsets. Note that such a set of partitions Π shares similarities with probability buffers Γ, as introduced in Chapter 3. In contrast to probability buffers, partition sets do not handle (e.g., when merging multiple partitions) probabilities with which a certain offset is reached, but assign a discovery latency to each partition.

We write $[t_s, t_e] \leftarrow l$ to indicate that a new partition with borders t_s and t_e having a latency of l is added to the corresponding set Π. We assume a set of unprocessed partitions $\Pi_{u,i}$ for a given order $i \in \{0, 1\}$ of the corresponding γ-sequence[2] and a set of completely processed partitions Π_c, which concerns all sequence orders. The following algorithm can be used for computing the explicit latencies of all partitions for one-way discovery, which are contained in Π_c after its execution. A formal definition is given in Algorithm 5.1. It works as follows.

1. $\Pi_{u,0}$ is initialized by $[0, T_s - d_s] \leftarrow \omega$, Π_c by $[T_s - d_s, T_s] \leftarrow \omega$

2. For each sequence order $i \in \{0, 1\}$, we compute the appropriate value of γ, as described in Section 5.2. For order-0 sequences, we set $\gamma = T_a$.

3. Then, for each sequence order $i \in \{0, 1\}$, we iterate over all unprocessed partitions. Each examined partition is removed from $\Pi_{u,i}$ and is further subdivided into multiple processed or unprocessed partitions, which are added either to $\Pi_{u,i+1}$ or Π_c. This is done by the *processPartition()*-function, which is described below. All newly added partitions of $\Pi_{u,i+1}$ are processed in the course of the next order.

4. If $\Pi_{u,i+1}$ becomes empty, the algorithm terminates.

[2]Actually, this algorithm also works for the generic case of higher-order sequences, which are not relevant for this chapter.

Algorithm 5.1 Computing Π_c

```
Require: $T_a$, $\omega$, $T_s$, $d_s$
Ensure: $\Pi c$
1: $\Pi_{u,0} : [0, T_s - d_s] \leftarrow \omega$; $\Pi_c : [T_s - d_s, T_s] \leftarrow \omega$
2: Compute $\vec{\gamma} = \gamma_0, \gamma_1, ..., \gamma_n$
3: for i from 0 to 1 do
4:     for each partition $n \in \Pi_{u,i}$ do
5:         Remove partition $n$ from $\Pi_{u,i}$
6:         $(\Pi_{u,i+1}, \Pi_c) = proccessPartition(n, \Pi_{u,i}, \Pi_c)$
7:     end for
8: end for
```

The function *processPartition()* is called for every single partition of $\Pi_{u,i}$ once per sequence order $i \in \{0, 1\}$. After a partition has been processed by this function, this partition is removed from $\Pi_{u,i}$. However, new partitions are added both to $\Pi_{u,i+1}$ and Π_c. In the following, we describe the concept behind this function for growing sequences (i.e., $\Phi[k]$ becomes larger for growing indices k). Shrinking sequences can be handled by a similar scheme.

Based on the borders t_s and t_e of the partition to be processed, we compute a lower number N_l and an upper number N_u of γ-intervals that fit into this partition. It is

$$N_l = \left\lceil \frac{T_s - d_s - t_e}{\gamma} \right\rceil, \quad N_u = \left\lfloor \frac{T_s - d_s - t_s}{\gamma} \right\rfloor. \tag{5.6}$$

Further, similarly to Chapter 3, we define d_{nl} as the absolute distance between $N_l \cdot \gamma$ and the right partition border, and d_{nu} as the distance between the left border and $N_u \cdot \gamma$. Due to space constraints, we restrict our descriptions to the most general case of $N_l < N_u$. Special cases can be handled similarly. Figure 5.5 depicts the basic concept for processing each partition. First, the time between N_l and N_u is separated into parts in which the current sequence hits the scan interval (depicted by h in the figure) after a certain number of γ- or T_a-intervals, and the remaining, missing parts (depicted by m in the figure). As already described, for shrinking or increasing a temporal distance by γ time-units, a certain multiple of T_a-intervals need to pass. For example, for an order-1-sequence, if $T_a < T_s$, $\sigma = \lceil \frac{T_s}{T_a} \rceil \cdot T_a$ time-units take place until the temporal distance between an advertising and a scan window is reduced by γ time-units. Following this scheme, the amount of time until reaching the next scan window or the next partition of a lower-order sequence is computed and added to the latency of this partition. The latency of the partition, before this new addition takes place, is the sum of latencies for all

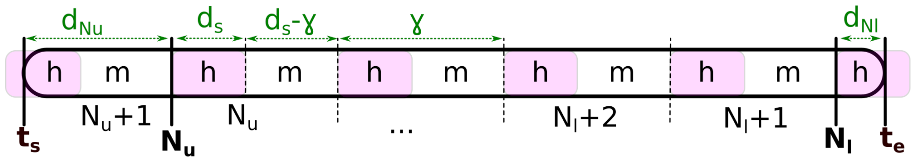


Figure 5.5: Processing a single partition.

previously processed, lower-order sequences. For sequences of order 0, the initial latency is ω time-units (since the latency of one packet-transmission ω occurs in all cases). For each hitting part (marked with h in Figure 5.5), a new partition is added to Π_c. With l being the existing (packet-to-packet) latency of the partition being currently processed, the latencies for these newly-added partitions are $l+(N_l+1)\cdot\sigma$, $(N_l+2)\cdot\sigma$, ..., $N_u\cdot\sigma$ time-units. Similarly, for each missing part (indicated with m in the figure), a new partition $i \in [N_l+1, N_u]$ is added to $\Pi_{u,i+1}$. The latencies l_i for such newly added partitions i in $\Pi_{u,i+1}$ are

$$l_i = \begin{cases} l + i\cdot\sigma & ,\text{if} \quad m_{i+1} = g \\ l + (i+1)\cdot\sigma & ,\text{if} \quad m_{i+1} = s. \end{cases} \tag{5.7}$$

$m_{i+1} \in \{s, g\}$ indicates whether the next higher-order sequence is growing (g) or shrinking (s). For the highest-order sequence, no non-matching areas exist. The time periods within d_{nl} and d_{nu} are processed similarly.

After the completion of Algorithm 5.1, $\Pi_{u,n}$ is empty and Π_c covers the whole range of possible initial offsets $\Phi[0]$. If all special cases are accounted for, Π_c represents the one-way discovery latencies for all possible initial offsets of such a periodic-interval-based protocol.

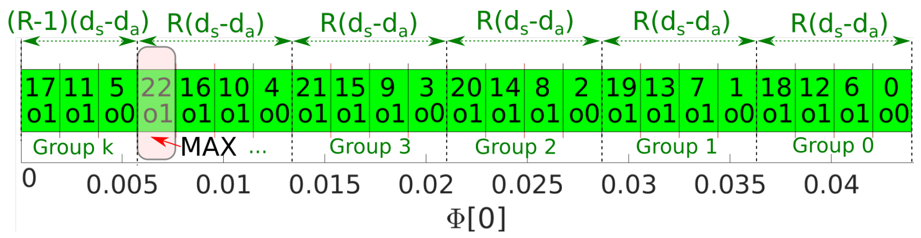


Figure 5.6: Set of partitions for $\eta_t = 10\,\%, R = 4$

Figure 5.6 depicts a complete set of partitions for our proposed protocol, with a target duty-cycle of 10 % and a value of $R = 4$. The upper number in each partition depicts the number of advertising intervals until a successful discovery takes place, whereas the lower number depicts the highest sequence order that occurs. As can be seen, parametrizations as described in Section 5.3.1 lead to a distribution of partitions with equal lengths. As already mentioned, only orders of 0 and 1 are present. We exploit this for computing the latencies for assisted two-way discovery, as described next.

5.4.2 Cumulative Distribution Functions

Based on Π_c, we in the following attempt to derive the cumulative distribution functions (CDFs).

One-Way Discovery

To compute the CDF for one-way discoveries, we assign the probability $p_i = \frac{(t_e[i]-t_s[i])}{T_s}$ to each partition i, which corresponds to its fraction of time in the set of initial offsets. The CDF describes the probability $P(l < l_{th})$ of all latencies l being smaller than a certain threshold l_{th}. One can iterate over all partitions i in Π_c to find all partitions with latencies below or equal to $l_{th} = l_i$. The probability $P(l < l_i)$ is the sum $\sum p_i$ of these partitions. If this is carried out repeatedly for all $i \in \Pi_c$, the CDF for one-way discovery is obtained. We denote the resulting CDF as P_{1W}.

Quick Reference of Symbols

P_{1W}	CDF function for one-way discovery
P_{2W}	CDF function for two-way discovery
P_{A2W}	CDF function for mutually assisted two-way discovery
l	Packet-to-packet discovery latency
l_{th}	Latency threshold
$l_{p,A2W}$	Additional latency penalty for A2W discovery after the first rendezvous
Π	Partition of initial offsets
Π_c	Completely processed partition of initial offsets

A full table of symbols is given in Appendix A.

Two-Way Discovery

The CDF for non-assisted two-way discovery can be derived from P_{1W} as follows. Since both device E and device F run the same probabilistic process independently from each other, the two-way discovery latency is the maximum of two individually carried out one-way discovery processes. For the two-way CDF P_{2W}, it is therefore (cf. [79])

$$P_{2W}(l < l_i) = P_{1W}(l < l_i) \cdot P_{1W}(l < l_i) \forall i \in \Pi_c. \tag{5.8}$$

Assisted Two-Way Discovery

The CDF for assisted two-way discovery can be computed as follows. Given a number of N_f partitions in Π_c, a new set of partitions Π_c^* is defined by concatenating N_f periodic repetitions of Π_c. A second set of partitions $\Pi_{c,sh}^*$ is defined by concatenating N_f periodic repetitions of Π_c, in which the partitions of every repetition $i \in [0, N_f]$ are cyclically shifted by $i \cdot (d_s - \omega)$ time-units to the right. For each pair of partitions in Π_c^* and $\Pi_{c,sh}^*$, the minimum latency is derived. The resulting distribution of minimum latencies is denoted by Π_{A2W}^*. Clearly, the scan windows of both devices have a random, uniformly distributed temporal distance from each other, which is determined by the random points in time at which both devices start operating. Since the set of shifted repetitions $\Pi_{c,sh}^*$ considers all possible temporal permutations against Π_c^* and since always the partition having a shorter latency is chosen, Π_{A2W}^* now contains the partitions for the first rendezvous in an A2W protocol. An additional penalty $l_{p,A2W}$ has to be added to the latency of each partition of Π^*_{A2W}, to account for the time until the second discovery takes place.

$\Pi^*_{c,sh}$ has been created by concatenating cyclic shifts of the partitions in Π_c. If a partition is shifted by t time-units to the right, the offsets $\Phi[0]$ that are by t time-units larger than the original offsets in this partition will match the same scan window with the same latency as the unshifted ones. Because of this, the corresponding scan window will also be shifted by t time-units to the right. Hence, shifting every partition by i partition lengths $(d_s - \omega)$ to the right implies that all scan-windows will be i partition lengths later. This also means that for every scan window in Π^*_c, a scan window in $\Pi^*_{c,sh}$ exists $i \cdot (d_s - \omega)$ time-units later. Therefore, if the partition from Π^*_c has a smaller latency than the one from $\Pi^*_{c,sh}$, we add $l_{p,A2W} = i \cdot (d_s - \omega)$ time-units to the latency of the corresponding partition, with i being the number of the repetition of the original pattern Π_c in $\Pi^*_{c,sh}$. Otherwise, the additional penalty $l_{p,A2W}$ has a value of $(N_f - i) \cdot (d_s - \omega)$ time-units. This defines the final distribution $\Pi_{A2W} = \Pi^*_{A2W} + l_{p,A2W}$ for assisted two-way discovery. From Π_{A2W}, the CDF P_{A2W} can be computed equivalently to P_{1W}.

From these considerations, one can derive a simple equation to approximate P_{A2W} in close proximity. When regarding the assisted two-way discovery procedure as the same probabilistic process being carried out two times in parallel, the instance with the shorter latency determines the overall latency. For the first rendezvous, it is

$$P^{1st}_{A2W}(l < L) = 2 \cdot P_{1W}(l < l_i) \cdot (1 - P_{1W}(l < l_i)) + P_{1W}(l < l_i)^2 \;\forall i \in \Pi_c, \tag{5.9}$$

since for a match, either device E needs to have a latency smaller or equal than $l_{th} = l_i$, only, or device F, or both devices. Due to the A2W mechanism, the second rendezvous happens within T_s time units from the first one. To approximate the delay between the two successful receptions, we add $1/2 \cdot T_s$ to all partitions of Π_c and obtain an approximate CDF P'_{A2W}.

Based on Π_c, P_{2W} and P_{A2W}, we can derive the worst-case and mean latencies of our proposed protocol, as described below.

5.4.3 Worst-Case and Mean Latencies

We could derive the worst-case and mean discovery latencies as the maximum/mean values in Π_c, obtained from Algorithm 5.1. However, for mathematical optimizations of the protocol, a closed-form formulation is needed. The worst-case latencies for device E discovering device F and vice-versa are identical to the two-way discovery latencies. They can be derived as follows. When applying parametrizations as described in Section 5.3.1, one can observe that the set of partitions Π_c for one-way discovery always has the following structure (cf. Figure 5.6).

- From Equation 5.1 follows that Π_c consists of M partitions of width $d_s - \omega$.
- From Equations 5.2 and $R = \frac{M+1}{k}$, it follows that there are $k = \frac{M+1}{R}$ partitions in which a match of the order-0 sequence occurs (cf. Figure 5.6).

- It follows that the remaining $M - k$ partitions are related to order-1 - sequences. Since every two subsequent order-0-partitions have a temporal distance of $T_a = {}^1\!/_k \cdot (M+1)(d_s - \omega)$ time-units from each other, it follows that there are ${}^1\!/_k \cdot (M+1) - 1 = R - 1$ order-1-partitions temporally left of each order-0-partition (except for the leftmost one, see below).
- Since we have required $M + 1$ to be divisible by R, M is in general not divisible by R. It is $k \cdot R = M + 1$. Therefore, the leftmost order-0-partition has only $R - 2$ neighboring order-1-partitions. In other words, as shown in Figure 5.6, there are $k - 1$ groups of R partitions with $R-1$ order-1-partitions and one order-0-partition in each of them. In addition, there is one group at the leftmost part of Π_c with $R - 2$ order-1-partitions and 1 order-0-partition.

One needs to recall that whenever shrinking a temporal distance by γ time-units, multiples of T_a time-units of discovery latency are induced. Therefore, the maximum latency can only be reached from partitions in which the largest number of γ-intervals have to pass. This is fulfilled by the leftmost partition of each group (except for the leftmost partition in the shortened group with only $R - 1$ partitions, which is located at the left of Π_c). Among these leftmost partitions, the largest latency is achieved by the group with the largest number of T_a-intervals until reaching the scan window. This is the group $k - 1$, as highlighted with “MAX” in Figure 5.6. It follows that the (packet-to-packet) worst-case discovery latency is as follows

$$\begin{aligned} l^* = \ & ((R-1)k + k - 2)(R(d_s - \omega)) + \omega = \\ & (M-1) \cdot T_a + \omega. \end{aligned} \tag{5.10}$$

Following similar considerations, the mean one-way discovery latency can be derived as

$$\overline{l_{1W}} = \frac{1}{2} \cdot \frac{1}{k}(M-1)(d_s - \omega)(M+1) + \omega. \tag{5.11}$$

One can also observe from Figure 5.6 that each partition of Π_c has a unique latency, i.e., no latency occurs more than once. From the elaborations above, it becomes clear that all latencies from $0 \cdot T_a + \omega$ to $L = (M-1) \cdot T_a + \omega$ time-units occur, and each partition has a probability of $1/M$. Therefore, the CDF function for the on-way discovery can also be written as

$$P_{1W}(l < l_j) = \sum_{k=0}^{j} \frac{1}{M},\ j \in [0, M-1], \tag{5.12}$$

with $l_j = j \cdot T_a + \omega$. Similarly, for the non-assisted two-way-discovery, it is:

$$P_{2W}(l < l_j) = \left(\sum_{k=0}^{j} \frac{1}{M} \right)^2,\ j \in [0, M-1]. \tag{5.13}$$

With Equations 5.12 and 5.9, analytical formulations of the CDF for 2W- and A2W-discovery are available. The expected value $E(X)$ of any random variable X can always be expressed as $E(X) = \int_0^\infty (1 - P(X < x))dx$. Hence, for the non-assisted two-way discovery, the mean latency is

$$\begin{aligned}\overline{l_{2W}} &= \omega + R(d_s - \omega) \sum_{k=0}^{M-1} \left(1 - \sum_{h=0}^{k} \frac{1}{M}\right)^2 \\ &= \frac{1}{6} R(4M - \frac{1}{M} - 3)(d_s - \omega) + \omega. \end{aligned} \tag{5.14}$$

Similarly, one can derive the mean latency for A2W-discovery using Equation 5.9:

$$\overline{l_{A2W}} = \omega + l_{p,A2W} + R(d_s - \omega) \cdot \sum_{k=0}^{M-1} \left(1 - 2\left(\sum_{h=0}^{k} \left(\frac{1}{M}\right) \cdot \left(1 - \sum_{h=0}^{k} \left(\frac{1}{M}\right)\right) + \sum_{h=0}^{k} \left(\frac{1}{M}\right)^2\right)\right). \tag{5.15}$$

In this equation, $l_{p,A2W}$ is the additional time penalty for the second discovery after the first rendezvous has taken place. When assuming that all offsets between two neighboring scan intervals occur with the same likelihood, $l_{p,A2W}$ can be approximated with $1/2 \cdot T_s = 1/2 \cdot M(d_s - \omega)$. Therefore, the mean A2W-latency becomes approximately

$$\overline{l_{A2W}} \approx \omega + (d_s - \omega) \left(\frac{M}{2} + \frac{R(2M^2 - 3M + 1)}{6M} \right). \tag{5.16}$$

5.5 Evaluation

In this section, we first evaluate the behavior of our proposed protocol for different values of R and different duty-cycles η. Next, we compare its performance to previously known protocols. Finally, we evaluate the theories and models presented in this chapter. Using real-world measurements, we show that the predicted performances can be reached in practice.

All estimated discovery latencies presented in this section have been computed using the theory described in Section 5.4.1 (i.e., by applying the explicit model). We have used exact methods rather than the approximations described whenever possible. We have assumed a lower limit of the scan window of $d_{s,m} = 10 \cdot \omega$, as already explained. We assume a packet length of 3 bytes, which is the minimum packet length for which our proposed protocol is feasible: Radios typically need a 1-byte preamble for synchronization. In addition, we need to send 2 bytes for mutual assistance hints. Additional data can be transferred upon success without any significant delay: If a node receives an A2W-hint having the value 0 s (which indicates that the packet is an A2W packet), the receiving node can schedule an additional packet immediately afterwards, whereas the sending node could listen right after the transmission of the additional A2W packet. This packet can be of arbitrary length.

Even though other duty-cycles can also be realized with our proposed protocol, we restrict our evaluations to duty-cycles $\eta \in [0.13\,\%, 1.46\,\%]$ in steps of $0.01\,\%$, since this covers a practical range of worst-case latencies: The value $\eta = 0.13\,\%$ corresponds to worst-case latencies of approximately 1 min, whereas $\eta = 1.46\,\%$ corresponds to worst-case latencies of 0.5 s.

5.5.1 Discovery Latencies

Effect of Mutually Assisted Two-Way Discovery

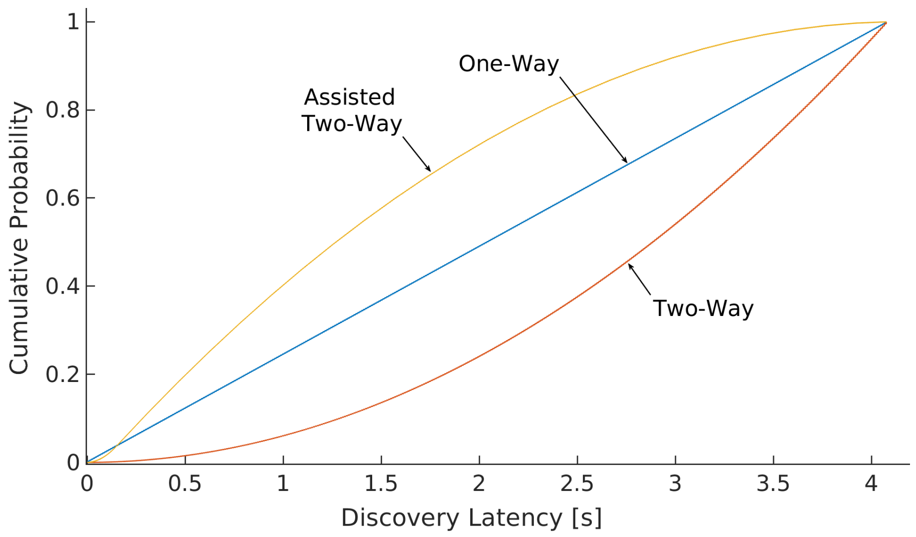


Figure 5.7: CDF for $R = 26$, $\eta = 0.5\,\%$

Figure 5.7 depicts the CDF of Griassdi for $\eta = 0.5\,\%$ and $R = 26$. The three depicted curves show the CDFs for one-way, two-way and mutually assisted two-way discovery. All three curves reach a probability of 100 % after the same latency, which means that the worst-case latency is equal for one-way, two-way and assisted two-way discovery. However, as can be seen, the CDF for two-way discovery is convex, indicating a lower mean discovery latency than half the maximum one. In contrast, the CDF for the assisted two-way discovery has a concave shape, which means that the discovery can be performed faster in the average case. However, the speedup depends on the value of R, which is evaluated next.

Cumulative Distribution Functions

Figure 5.8 depicts multiple CDFs for A2W discovery with different values of R. As can be seen, for a value of $R = 1$, the A2W CDF is convex. In fact, it is identical to the non-assisted two-way discovery curve. When increasing R, the CDF becomes increasingly concave, leading to shorter mean latencies. However, there is an optimal value of R, at which the mean latency becomes minimal. It is referred to as R_{Opt}. Increasing R beyond R_{Opt} does not decrease the mean latency anymore, which can also be observed in the Figure for $R = 50 > R_{Opt}$. R_{Opt} depends on the duty-cycle. It can be computed (approximately) by minimizing the product $\eta \cdot \bar{l}$ using Equations 5.4 and 5.16. One can also observe that larger values than R_{Opt} increase the worst-case latency. Therefore, $R \in [1, R_{Opt}]$ can be used to trade-off worst-case against mean latencies. We study this trade-off more thoroughly, next.

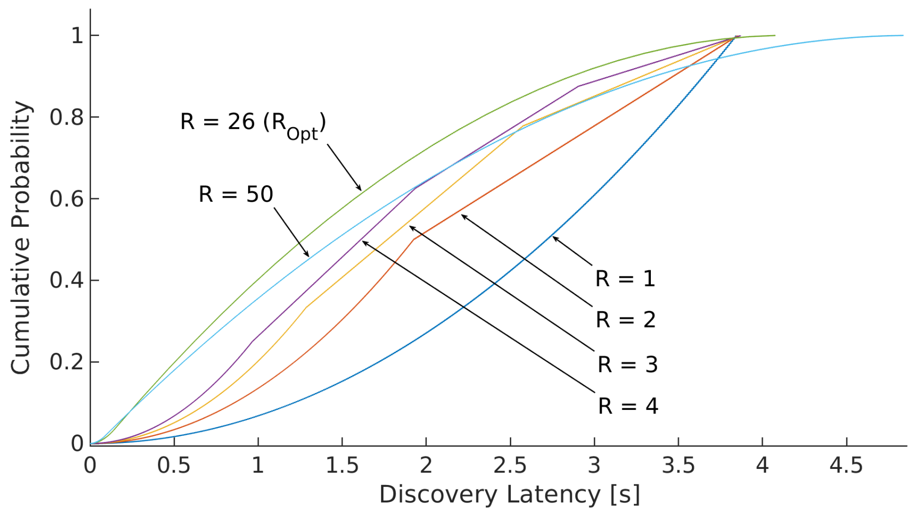


Figure 5.8: CDF for assisted two-way discovery ($\eta = 0.5\,\%$)

Worst-case and Mean Latencies

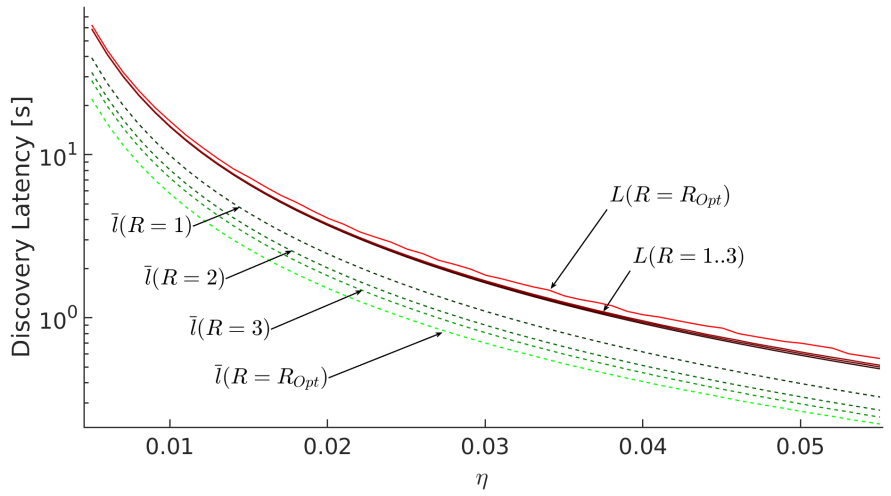


Figure 5.9: Maximum and mean discovery latencies for different values of R

Figure 5.9 depicts the discovery latencies for sweeping duty-cycles of $\eta \in [0.13\,\%, 1.46\,\%]$ in steps of $0.01\,\%$. Multiple curves for different values of R are depicted. In addition, there is one curve in which R has been set to $R_{Opt}(\eta)$ for every given duty-cycle. As can be seen, the maximum latencies are only marginally affected by the parameter R, since all maximum latency curves (L) nearly lie on each-other. However, the mean latencies ($\bar{l}$) are greatly reduced for all duty-cycles. Growing values of R lead to larger reductions of the mean latency $\bar{l}$, as can also be seen from the CDF.

As already explained, one can choose R freely within 1 and R_{Opt}. $R = 1$ results in the lowest worst-case latencies, whereas $R = R_{Opt}$ results in the lowest mean latencies.

R	1	2	3	4	5	6	7	8	9	10	R_{Opt}
G_m [%]	0	0	-1	-1	-2	-2	-2	-3	-3	-4	-7
$\overline{G}$ [%]	0	19	28	32	35	37	39	40	41	41	43

Table 5.3: Gain of max. (G_m) and mean ($\overline{G}$) latencies for different values of R over the latencies for $R = 1$

For $\eta = 0.5\,\%$, Table 5.3 quantifies this trade-off. It depicts the gain of mean latencies $\overline{G}$, which is defined as

$$\overline{G} = \frac{\bar{l}(1) - \bar{l}(R)}{\bar{l}(1)}. \tag{5.17}$$

Here, $\bar{l}(R)$ are the mean latencies for different values of R, whereas $\bar{l}(1)$ is the mean latency for $R = 1$. In Table 5.3, $\overline{G}$ is averaged over all duty-cycles. In addition, Table 5.3 shows the according gains of the maximum latencies G_m, respectively. In other words, it shows how much the worst-case latency is increased for a certain reduction of the mean latency. Evidently, the largest reduction of $\bar{l}$ on average over all examined duty-cycles is 43 %, thereby increasing the worst-case latency L by 7 %. Hence, mean latencies can be traded against worst-case latencies by choosing the value of $R \in [1, R_{Opt}]$.

5.5.2 Comparison Against other Protocols

We have shown that, for given values of R, A2W discovery can greatly reduce the mean latencies, without significantly affecting the worst-case latencies. In this section, we compare the performance of Griassdi against multiple existing protocols. We have included both slotted and slotless protocols into this comparison. In particular, we have considered the following protocols.

- **DISCO** [15], one of the most popular slotted protocols. While not being realizable in practice (since Disco requires a pair of primes p_1 and p_2), we apply the common assumption of $p_1 = p_2 = p$ [2] in the equations computing the worst-case latencies, as in the previous chapter.

- **Searchlight-Striped** [2], since it is often considered as one of the best-performing symmetric slotted protocols. We assume a slot-overflow δ of 0, indicating that its predicted performance is slightly better than in practice.

- **U-Connect** [23], which is frequently included in performance comparisons.

- **G-Nihao** [61], a pseudo-slotted protocol. We assume $\gamma = 2$, as also done in [61].

- **PI-0M** from the previous chapter, since it provides optimal latency/duty-cycle relations. We assume the following parameter values: $\epsilon = \frac{1}{32768}$, $d_{s,l} = 10 \cdot \omega$. We did not consider any mechanisms to mitigate hardware non-idealities (e.g., radio turnaround times). Note that this parametrization differs from the parameter values assumed in Chapter 4, in which we had assumed e.g., a packet length of 4 bytes instead of 3 bytes or a value of $d_{s,m} = \omega$ instead of $d_{s,m} = 10 \cdot \omega$. As already stated, the performances of the $PI - kM^+$ schemes, in particular, the $PI - k2^+$ scheme, are almost identical to those of the $PI - 0M$ scheme, with the $PI - 0M$ scheme having a slight edge. Therefore, the results for $PI - 0M$ are representative for all parametrization schemes presented in the previous chapter.

In general, slotted protocols perform best using short slot-lengths. However, since a packet is sent at the beginning and/or the end of each slot, discovery fails if these packets overlap, as already described in the previous chapter. For this comparison, we assume a fixed slot length of 30 packet lengths ω. With a packet length of 24 µs, which corresponds to 3 bytes per packet on an $1MBit$ radio, the slot length becomes 720 µs. For the sake of an easier comparison, we assume this slot-length for all slotted protocols except for U-Connect, for which we assume a slot-length of 250 μs, as proposed in [23].

In our comparison in Chapter 4, we have adjusted the slot lengths of all protocols, such that the same probability of failed discoveries has been achieved for all of them. However, in this comparison, we instead assume a fixed slot length. Since most related work assumes a similar, fixed slot length of e.g., 10 ms [61], the comparison in this section can be regarded as a comparison of our proposed solutions against common ND setups. In contrast, we have compared the best achievable performances of slotted protocols (in the latency/duty-cycle metric) with the performances of our proposed solutions in Chapter 4.

All protocols under consideration perform best for short packet lengths, except U-Connect, for which the packet-length has a negligible impact. For determining the minimal required packet length, we assume that the radio needs a one-byte preamble in addition to the payload. For all protocols, we assume that the payload consists of a two-byte node ID. With these considerations, we assume a packet length of 3 bytes for all protocols. We have computed the worst-case latencies of the previously known protocols under consideration according to the equations in Table 4.2. The mean latencies have been obtained by simulations.

Worst-Case Latencies

Figure 5.10 depicts the computed worst-case latencies for duty-cycles between 0.13 % to 1.46 % in steps of 0.01 % with $R = R_{Opt}$. As can be seen, the worst-case latencies of the $PI-0M$-scheme and Griassdi lie almost on the same curve. The worst-case latencies of all other protocols in this comparison are significantly larger for the slot length considered.

Table 5.4 depicts the worst-case latency gains of the protocol proposed in this chapter, defined as $(L - L_{Griassdi})/L$, with L being the worst-case latency of the protocol under consideration and $L_{Griassdi}$ the worst-case latency of the Griassdi protocol. Columns 1 and 3 depict the mean gains over the range of duty-cycles considered, Columns 2 and 4 the maximum gains. The first two columns show the gains for $R = 1$, the last two ones show the values for $R = R_{Opt}$. For $R = R_{Opt}$, the worst-case performance of Griassdi is slightly reduced compared to $PI - kM^+/PI - 0M$, which has the lowest relation of duty-cycles and worst-case latencies among all known protocols. For smaller values of

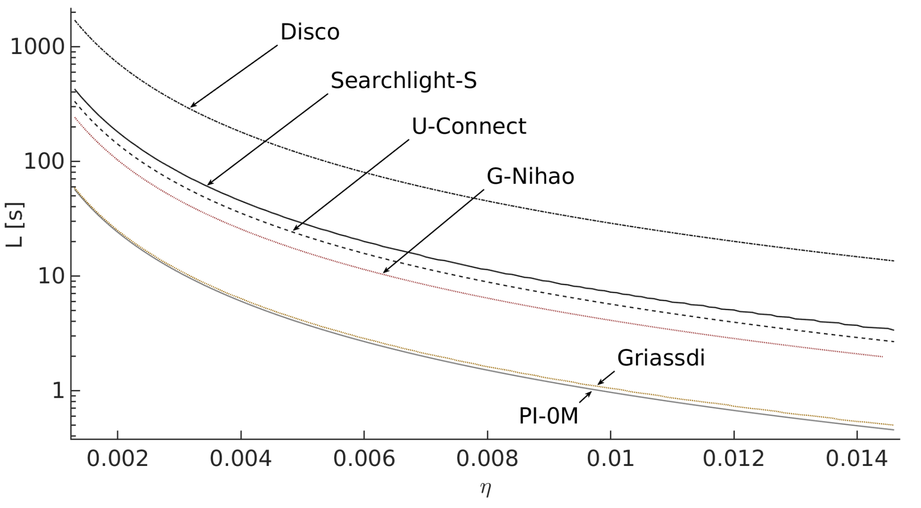


Figure 5.10: Worst-case latencies l^* of the Griassdi protocol (for $R = R_{Opt}$) against other protocols

R, the worst-case performance of both protocols become identical, but no mean latency gains due to mutual assistance can be achieved.

	$R = 1$		$R = R_{Opt}$	
Protocol	Mean	Max	Mean	Max
Disco	0.97	0.97	0.96	0.97
U-Connect	0.83	0.83	0.82	0.82
Searchlight-S	0.87	0.87	0.86	0.86
G-Nihao	0.76	0.77	0.75	0.76
PI-0M	0.0	0.0	-0.07	-0.10

Table 5.4: Gain of worst-case latencies l^* of Griassdi over different protocols

Mean Latencies

Unlike for the worst-case latencies, there are no equations that cover the average behavior of the slotted protocols proposed in the literature [15], [23], [2], [61]. Therefore, we have implemented simulation models for all slotted protocols under consideration. The simulations have been repeated $100,000\times$ for each slotted protocol, with random offsets in each run. For the $PI-0M$-protocol, the model presented in Chapter 3 has been used.

Figure 5.11 shows the simulated CDFs for a duty-cycle of 1.4 %. A duty-cycle of 1.4 % translates to the following protocol-specific parameters: Disco: $p_1 = 137$ and

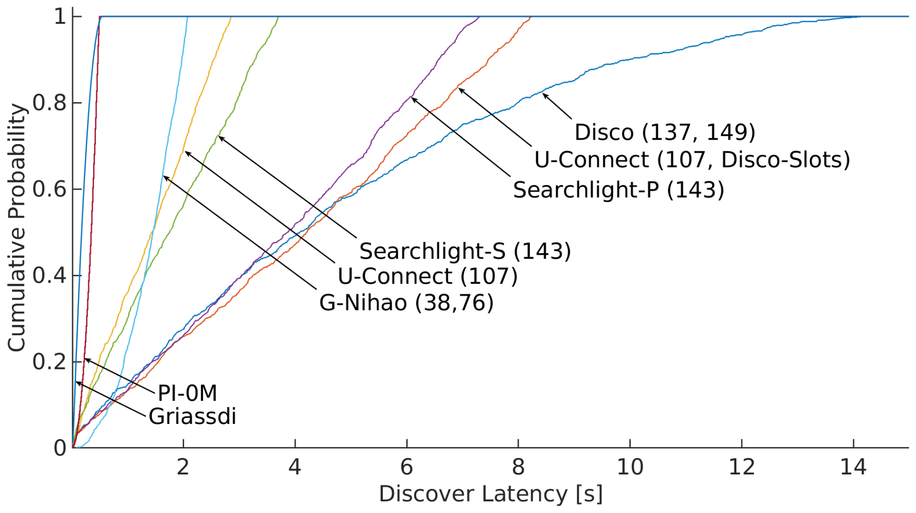


Figure 5.11: CDF of Griassdi compared to existing protocols for $\eta = 1.4\,\%$, $R = R_{Opt}$

$p_2 = 149$; U-Connect: $p = 107$; G-Nihao: $m = 38$ and $n = 76$. For Griassdi, R has been assumed to be R_{Opt}. Clearly, it achieves the lowest mean latencies of all protocols under consideration. In particular, the mean latencies for $\eta = 1.4\,\%$ and - in parenthesis - the mean latency gains $(\bar{l} - \overline{l_{Griassdi}})/\overline{l_{Griassdi}}$, are as follows. Disco 4.74 s (96 %), Searchlight 3.74 s (95 %), Searchlight-Striped 1.76 s (89 %), U-Connect 1.41 s (86 %), G-Nihao 1.38 s (86 %), $PI - 0M$ 0.33 s (40 %), Griassdi 0.20 s. Compared to the best-performing non-assisted slotless protocol $PI - 0M$ (cf. Chapter 4), the reduction of the mean latency by 40 % has a significant impact on practical implementations, since it can be used to reduce the duty-cycle, which allows for increased battery runtimes.

Channel Utilization

In the following, we study the channel utilization of our proposed protocol. Recall that the channel utilization is defined as the percentage of time packets are being transmitted on the channel. When neglecting the A2W packet in case of a successful one-way discovery, a device running Griassdi causes a channel utilization of approximately ${}^{\omega}/{}_{T_a}$. Figure 5.12 depicts this channel utilization. As can be seen, for the considered slot lengths, the channel-utilization of our proposed protocol is larger than the channel utilizations of the slotted ones. However, for the largest duty-cycle considered, it only reaches about 0.7 %.

The number of collided A2W discoveries depends on the chance of more than one device sending their A2W packets at overlapping points in time. When assuming that each device transmits its A2W packets at random points within the scan window of its opposite device, the collision probability can be reduced by increasing the minimum scan window length $d_{s,m}$. However, this adversely affects the achievable worst-case latencies. Studying the optimal trade-off between A2W collisions and worst-case latencies needs to be studied in future research.

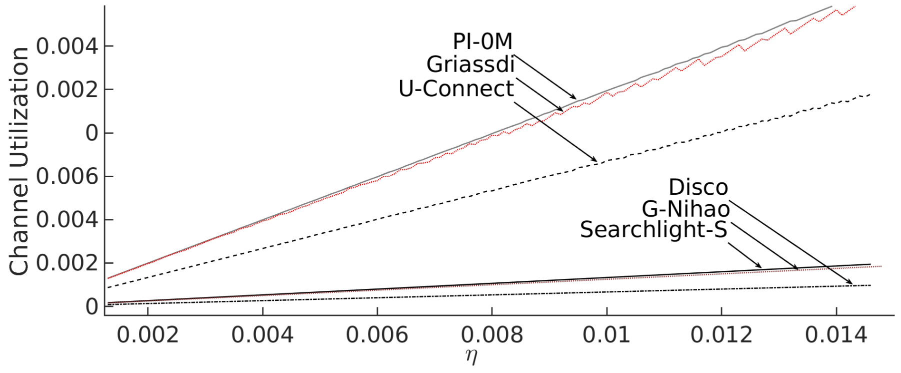


Figure 5.12: Channel Utilizations

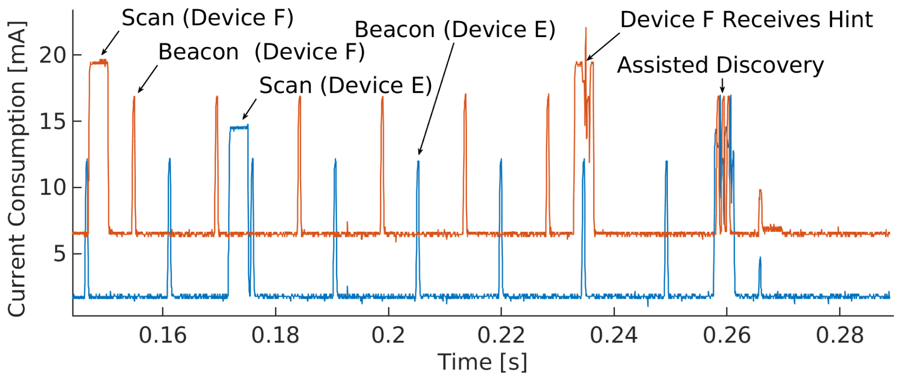


Figure 5.13: Current measurement during A2W discovery

5.6 Implementation

In this section, we demonstrate the proper functioning of Griassdi using a real-world implementation.

5.6.1 Setup

We have implemented Griassdi on two Nordic nRF51822 [53] radios. Based on a modified version of the open-source BLE stack *blessed* [44], we realized the A2W protocol described in this chapter on two PCA10005 evaluation boards [52]. Each board has been connected to a laptop using an UART-to-USB adapter. To ensure a proper functioning and to compensate for clock skew, we have extended each scan window by 80 μs, inducing a duty-cycle error of up to 2 %. The packet length ω has been 368 μs.

The mechanism for assisted two-way discovery has been realized as follows. Whenever a device receives a packet containing a wakeup hint, it schedules three additional packets around the point in time received. One of them is scheduled to begin at the exact received point in time, whereas the other two are sent 1 ms before and after it. This is done to minimize the effect of collisions between the additional A2W-packets and the regular packets of the remote device. No regular packets are sent after a hint has been received from a remote device, until the additional packets have been sent entirely. Afterwards, the device returns to its original schedule. Figure 5.13 depicts the measured current consumption of a successful discovery procedure for two devices E and F. We have used the built-in shunting resistors and measurement amplifiers of the PCA10005 board for this measurement. For a better readability, we have shifted the origin for device F by 5 mA. The broader peaks are related to scanning, whereas the smaller ones indicate the transmission of a packet. As can be seen, after device F receives a regular packet from device E, which includes a hint on the next wakeup time of device E, it schedules three additional packets within the wakeup period of device E. After that, mutual discovery is complete.

5.6.2 Experimental Data

The theory presented in this chapter indicates a high efficiency of our proposed protocol. To evaluate the validity and accuracy of this theory and to show that this performance is reached in practice, we have conducted a comprehensive set of experiments. In each experiment, the laptop has sent start- and stop-commands to the two radios (which are denoted as device E and device F) via the USB/UART connection. The laptop has measured the amounts of time after which the discovery has occurred. The radios have realized each duty-cycle $\eta \in [0.5\,\%, 5.5\,\%]$ in steps of 0.1 %. For each of these duty-cycles, we have carried out 200 two-way discoveries, consisting of 400 one-way discoveries. In total, we thereby measured the latencies of 20, 400 discovery procedures. In each discovery experiment between the two devices, the following steps have been carried out.

1. Device E starts scanning at a point in time t_0.
2. After a random amount of up to T_s time-units from t_0, device F starts scanning.
3. At $t_0 + 2 \cdot T_s + t_{a,1}$, with $t_{a,1}$ being a random amount of time between 0 and T_a time-units, device E starts advertising.
4. At $t_0 + 2 \cdot T_s + t_{a,2}$ time-units, device F starts advertising. $t_{a,2}$ is a random amount of time within $[0, T_a]$.

The discovery latencies have been measured relatively to the points in time at which the first advertising packet has been sent. All random amounts of time were distributed uniformly. The measurements have been aborted if two-way discovery could not been achieved within 70 s, which is longer than the predicted upper bound for all duty-cycles.

Worst-Case Latencies

The solid line in Figure 5.14 depicts the modeled worst-case latencies for assisted two-way discovery. It has been computed using the theory presented in Section 5.4. Each circle depicts the measured A2W latency of one experiment. As can be seen, the predicted upper bound is a very accurate estimate of the worst-case latency. The measured points in the top of the figure, with latencies around 70 s, depict failed discoveries. In addition, especially for larger duty-cycles, some measurements exceeded the predicted worst-case latency, but did not reach the maximum latency of 70 s. The reasons for these failures are collisions of regular and A2W packets, as well as hardware issues, such as radio turnaround

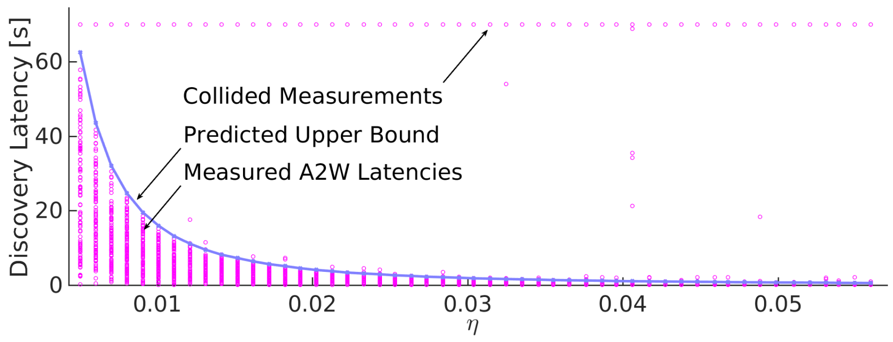


Figure 5.14: Fit of the model for worst-case latencies

times (cf. Chapter 4). The total (direct and A2W) rate of all measurements exceeding the predicted worst-case bound has been 7.3 %, which is mainly caused by frequent collisions (due to the large packet length of 368 µs). The total number of one-way measurements that exceeded the predicted bound but did not reach 70 s has been 145 (0.7 %). It needs to be mentioned that collisions are not a problem specific to Griassdi. They have also been reported for slotted [23], pseudo-slotted [61] and slotless (cf. Chapter 4) protocols. The A2W mechanism has failed 45 times in total. In the rest of this section, we discard all measurements exceeding the predicted upper bound by more than 1 %, in order to evaluate the statistical properties of the non-collided discoveries.

Mean Latencies

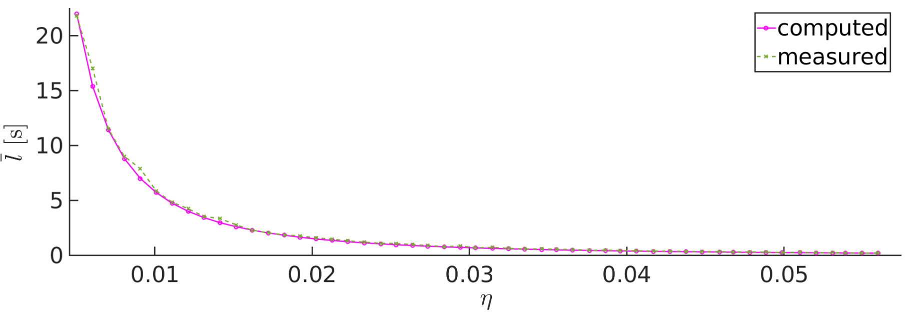


Figure 5.15: Fit of the model for mean latencies

Figure 5.15 depicts the measured mean latencies for A2W discovery. In addition, the mean curve predicted by our proposed theory is shown. Failed discoveries have been excluded. As can be seen, the predicted mean latencies are very accurate estimates for the experimental ones. The NRMSE κ considering all (non-discarded) measurements has been 1.3 %

Statistical Behavior

Our proposed theory cannot only model the overall behavior, such as mean- and worst-case latencies, but can also accurately describe the complete statistical behavior of our proposed protocol. Figure 5.16 depicts the modeled CDFs (solid lines) together with the

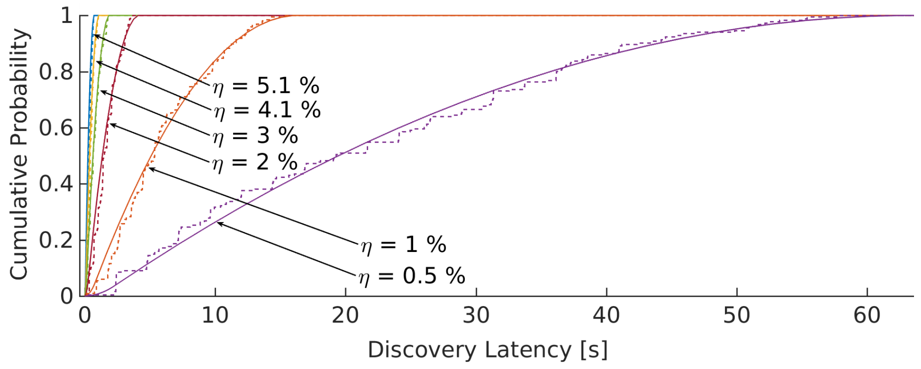


Figure 5.16: Computed (solid) and measured (dotted) CDFs

measured distributions (dotted lines) for multiple duty-cycles. Measurements exceeding the predicted upper bound l^* by more than 1 % have been discarded, as already described. As can be seen, our predictions lie in close proximity with the experimental data. Overall, our experiments fully confirm our theory. They show that our proposed protocol can achieve the high predicted performances in practice. The low remaining errors are caused by the loopback latency of the USB connection, the granularity of the random amounts of time that form the initial offsets, the compensation measures against clock inaccuracies and, in some rare cases, failed A2W discoveries (e.g., caused by collisions), or implementation issues.

5.7 Concluding Remarks

We have presented a technique that exploits mutual assistance to improve the mean latencies for two-way discovery. Based on this technique, we have proposed the protocol *Griassdi*, which achieves similar worst-case latencies to the best known discovery protocols, but provides up to 43 % shorter mean latencies. Therefore, our proposed protocol is, to the best of our knowledge, the one with the lowest mean-latency/duty-cycle relation of all known ones.

6 Conclusion and Future Work

6.1 Conclusion

In this thesis, we have studied optimal neighbor discovery protocols. We have considered optimality in two different metrics. As the first metric, we considered optimal relations between worst-case latencies and duty-cycles. Achieving high performances in this metric has been the main goal of the vast majority of works on ND, including the popular protocols Disco [15], U-Connect [23], Searchlight [2] and difference set-based protocols [87, 88, 55, 56]. We have shown that optimality in this metric can only be reached by slotless protocols in practice, and have presented a slotless protocol that actually achieves near optimal latency/duty-cycle relations in Chapter 4. It is worth recalling that the theoretical performance of this protocol, as proven in Section 4.7, coincides entirely with the Pareto curve in the latency/duty-cycle design space. Further, besides worst-case latencies, the mean latencies determine the performance experienced in practice. We have shown a technique to reduce the mean latencies of PI-based protocols in Chapter 5, while the near-optimal worst-case latencies are only marginally affected.

Besides using correlated schedules in conjunction with mutual assistance (cf. Section 2.5.3), which induces certain practical disadvantages, there is no further potential to improve the relations between duty-cycles and worst-case latencies. However, future protocols can potentially further reduce the mean latencies.

The latency/duty-cycle metric is relevant for sparse networks, in which collisions do not play a major role. For example, it is relevant when connecting a gadget to a smartphone, or in indoor-localization scenarios relying on battery-powered beacons, which are spaced sufficiently to avoid collisions. However, in dense networks with large numbers of devices carrying out the ND procedure simultaneously, collisions cannot be neglected. Here, a metric that accounts for collision probabilities is required. Such a metric is given by considering jointly the duty-cycle, the channel utilization and the worst-case latency. In this metric, as can be seen in Figure 6.1, the $PI - 0M$ parametrization scheme presented in Chapter 4 is also optimal. However, it only covers a small fraction of the 3-dimensional Pareto curve, as depicted by Figure 6.1. The grid in Figure 6.1 depicts the bound for collision-constrained ND, as given by Theorem 2.9. Slotless protocols applying the parametrizations defined by the $PI - kM^{+}/PI - 0M$ schemes form a curve on the 3-dimensional surface, which is located at the points with the optimal channel utilization. Further, as described in Section 2.5.2, slotted protocols can cover parts of the Pareto surface with lower channel utilizations, but due to collisions and radio turnaround times, they become increasingly unreliable for shorter slot lengths and hence higher channel utilizations. The parts of the Pareto-front that can be covered using slotted protocols lie in the left of Figure 6.1.

As can be seen in the figure, some parts of the Pareto curve, which lie in between the performance of the $PI - 0M$ scheme and that of slotted protocols, cannot be realized by any known protocol. This underpins the need for future research, as outlined next.

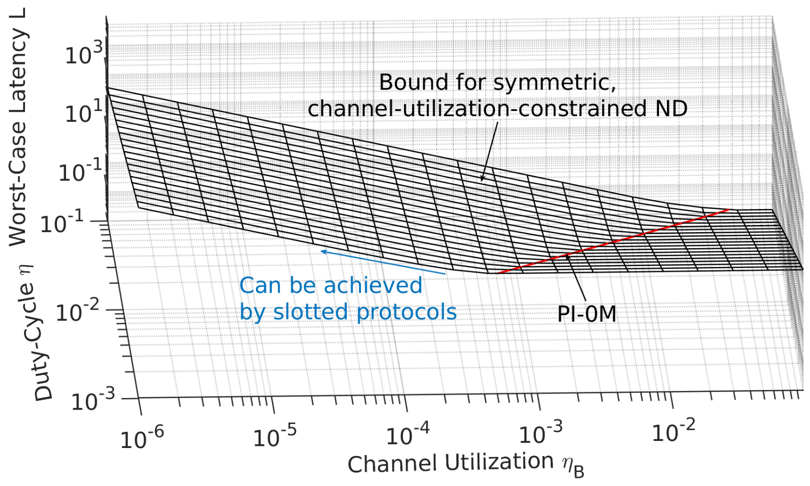


Figure 6.1: Parts of the Pareto front that can be covered using existing protocols.

6.2 Future Work

As described in Chapter 2, slotted protocols can cover a large part of the Pareto surface by scaling the slot length, but cannot closely approach the optimal (regarding relation between latencies and duty-cycles) channel-utilization. Therefore, a slotless framework that scales from the optimal channel utilization towards arbitrary lower ones could potentially cover the entire Pareto surface. It seems feasible to extend the parametrization schemes presented in Chapter 4, such that they can translate a certain target duty-cycle η_t and a certain constraint on the channel-utilization β into a set of parameters using which a PI-based protocol performs optimal. If such a scheme could also support asymmetry, it could potentially replace all existing pairwise, deterministic ND protocols without any disadvantages.

Another important aspect, which has received little attention in the literature, is increasing the resilience against collisions. In particular, if two packets from different devices collide, the extent to which the discovery latency is increased might vary for different protocols, because a later pair of packets could achieve discovery after different amounts of additional time. In such scenarios, it seems feasible to sacrifice a certain percentage of the worst-case latencies achieved in small networks. Because of this sacrifice, some redundancy can be added to antedate the second encounter after a collided one. Future research needs to address this aspect. Ideally, a protocol that optimizes the discovery latencies given an estimated number of devices, their duty-cycles and a maximum allowed failure rate can be developed in the future.

Due to their practical disadvantages, we have not exploited mutually exclusive beacon- and reception window schedules, as described in Section 2.5.3, in the protocols studied in this thesis. We have avoided such techniques mainly because they rely on mutual assistance, which potentially increases the packet collision rate. However, such correlated schedules need to be studied further and solutions that mitigate the collision problem need to be found. If the practical problems could be overcome, such correlated schedules could further improve the worst-case latencies in both metrics.

Another important property of ND protocols is duty-cycle asymmetry. Currently, to the best of our knowledge, there is no known asymmetric PI-based protocol. Though designing such an asymmetric protocol can be done in a straight-forward manner, e.g., by allowing other values of k than the optimal ones in the $PI-kM^{+}$ schemes from Chapter 4, the design of an optimal one, which utilizes the bound anticipated by Theorem 2.11, remains a challenge for future research. In particular, a protocol that achieves optimal relations between duty-cycles, channel-utilizations and worst-case latencies, while at the same time allowing each device to choose its duty-cycle freely, would be of high practical relevance.

Summarizing all the above-mentioned aspects, the main goal of future research should be developing a slotless parametrization scheme, which defines all parameters of a slotless protocol, based on the desired duty-cycle, the expected number of devices in range and the accepted rate of failed discoveries. The resulting parameters should lead to the optimum latency for the required properties. Further, the scheme should support asymmetric duty-cycles. Such a scheme would always perform optimally and therefore be a beneficial replacement of the ND procedures in existing protocol stacks of MANETs, e.g., in BLE. Further, the expected number of devices could also be estimated online by the devices themselves, leading to short latencies but higher channel utilizations in sparse networks and longer latencies with lower channel-utilizations in dense networks. We believe that this work puts a solid foundation for such a development. Finally, we have shown that some existing protocols actually achieve optimal relations between channel utilizations, duty-cycles and worst-case latencies, whereas others are not optimal in this metric. Examples for such optimal protocols are e.g., cyclic difference sets with overflowing slots [55]. However, we have not studied the underlying mechanisms that lead to optimal performances, which non-optimal protocols are obviously lacking. Our theory from Chapter 2 suggests that only protocols in which every deterministic partial sequence $B \in B_\infty$ covers every possible initial offset exactly once can be optimal. Therefore, it appears to be reasonable to classify existing protocols by their *redundancy*, which is a measure of how much unnecessary coverage Λ a deterministic sequence induces on the average.

Bibliography

[1] N. M. Abramson. The aloha system: Another alternative for computer communications. In *ACM fall joint computer conference*, 1970.

[2] M. Bakht, M. Trower, and R. H. Kravets. Searchlight: Won't you be my neighbor? In *Annual International Conference on Mobile Computing and Networking (MOBICOM)*, pages 185–196, 2012.

[3] L. Barenboim, S. Dolev, and R. Ostrovsky. Deterministic and energy-optimal wireless synchronization. *ACM Transactions on Sensor Networks (TOSN)*, 11(1):13:1–13:25, 2014.

[4] S. A. Borbash, A. Ephremides, and M. J. McGlynn. An asynchronous neighbor discovery algorithm for wireless sensor networks. *Ad Hoc Networks*, 5(7):998–1016, 2007.

[5] M. Bradonjic, E. Kohler, and R. Ostrovsky. Near-optimal radio use for wireless network synchronization. *Theoretical Computer Science*, 453:14–28, 2012.

[6] Z. Cao, Z. Gu, Y. Wang, and H. Cui. Panacea: A low-latency, energy-efficient neighbor discovery protocol for wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, 2018.

[7] H. Chen, W. Lou, Z. Wang, and F. Xia. On achieving asynchronous energy-efficient neighbor discovery for mobile sensor networks. *IEEE Transactions on Emerging Topics in Computing (TETC)*, 2017.

[8] L. Chen and K. Bian. Neighbor discovery in mobile sensing applications. *Ad Hoc Networks*, 48(C):38–52, 2016.

[9] L. Chen, K. Bian, and M. Zheng. Never live without neighbors: From single- to multi-channel neighbor discovery for mobile sensing applications. *IEEE/ACM Transactions on Networking (TON)*, 24(5):3148–3161, 2016.

[10] L. Chen, R. Fan, K. Bian, L. Chen, M. Gerla, T. Wang, and X. Li. On heterogeneous neighbor discovery in wireless sensor networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 693–701, 2015.

[11] T. Chen, J. Ghaderi, D. Rubenstein, and G. Zussman. Maximizing broadcast throughput under ultra-low-power constraints. *IEEE/ACM Transactions on Networking (TON)*, 26(2):779–792, 2018.

[12] K. Cho, G. Park, W. Cho, J. Seo, and K. Han. Performance analysis of device discovery of Bluetooth Low Energy (BLE) networks. *Computer Communications*, 2015.

[13] K. Cho, W. Park, M. Hong, G. Park, W. Cho, J. Seo, and K. Han. Analysis of latency performance of Bluetooth Low Energy (BLE) networks. *Sensors*, 15(1):59–78, 2015.

[14] R. Cohen and B. Kapchits. Continuous neighbor discovery in asynchronous sensor networks. *IEEE/ACM Transactions on Networking (TON)*, 19(1):69–79, 2011.

[15] P. Dutta and D. E. Culler. Practical asynchronous neighbor discovery and rendezvous for mobile sensing applications. In *ACM Conference on Embedded Network Sensor Systems (SenSys)*, pages 71–84, 2008.

[16] P. Dutta, D. E. Culler, and S. Shenker. Asynchronous neighbor discovery: Finding needles of connectivity in haystacks of time. In *International Conference on Information Processing in Sensor Networks (IPSN)*, 2008.

[17] D. M. Gordon. La Jolla difference set repository, 2018. `www.ccrwest.org/diffsets.html`.

[18] X. Guo, B. B. Chen, and M. C. Chan. Analysis and design of low-duty protocol for smartphone neighbor discovery. *IEEE Transactions on Mobile Computing (TMC)*, 16(12):3294–3307, 2017.

[19] A. Hess, E. Hyytiä, and J. Ott. Efficient neighbor discovery in mobile opportunistic networking using mobility awareness. In *International Conference on Communication Systems and Networks (COMSNETS)*, 2014.

[20] G. Jakllari, W. Luo, and S. V. Krishnamurthy. An integrated neighbor discovery and MAC protocol for ad hoc networks using directional antennas. *IEEE Transactions on Wireless Communications (TWC)*, 6(3):1114–1024, 2007.

[21] W. S. Jeon, M. H. Dwijaksara, and D. G. Jeong. Performance analysis of neighbor discovery process in Bluetooth Low-Energy networks. *IEEE Transactions on Vehicular Technology (TVT)*, 66(2):1865–1871, 2017.

[22] C. Julien, C. Liu, A. L. Murphy, and G. P. Picco. Blend: Practical continuous neighbor discovery for bluetooth low energy. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, pages 105–116, 2017.

[23] A. Kandhalu, K. Lakshmanan, and R. Rajkumar. U-connect: A low-latency energy-efficient asynchronous neighbor discovery protocol. In *International Conference on Information Processing in Sensor Networks (IPSN)*, pages 350–361, 2010.

[24] A. Kandhalu, A. E. Xhafa, and S. Hosur. Towards bounded-latency bluetooth low energy for in-vehicle network cable replacement. In *International Conference on Connected Vehicles and Expo (ICCVE)*, pages 635–640, 2013.

[25] N. Karowski, A. C. Viana, and A. Wolisz. Optimized asynchronous multi-channel neighbor discovery. In *IEEE Conference on Computer Communications (INFOCOM)*, 2011.

[26] K. Kim, H. Roh, W. Lee, S. Lee, and D.-Z. Du. PND: a p-persistent neighbor discovery protocol in wireless networks. *Wireless Communications and Mobile Computing*, 13(7):650–662, 2013.

[27] P. H. Kindt, S. Park, and S. Chakraborty. Fast collision simulation for cyclic wireless protocols. *arXiv preprint arXiv:1807.01645*, 2018.

[28] P. H. Kindt, D. Yunge, R. Diemer, and S. Chakraborty. Precise energy modeling for the bluetooth low energy protocol. *CoRR*, abs/1403.2919, 2014.

[29] M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *International Conference on Computer Aided Verification (CAV)*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

[30] H. Lee, D. Ok, J. Han, I. Hwang, and K. Kim. Performance anomaly of neighbor discovery in Bluetooth Low Energy. In *IEEE International Conference on Consumer Electronics (ICCE)*, 2016.

[31] W. Lee, S. Choi, N. Kim, J.-H. Youn, and D. Moore. Block combination selection scheme for neighbor discovery protocol. In *International Conference on Communication Systems and Network Technologies (CSNT)*, 2015.

[32] W.-S. Lee, J.-H. Youn, and T.-S. Song. Prime-number-assisted block-based neighbor discovery protocol in wireless sensor networks. *International Journal of Distributed Sensor Networks*, 15(1):1550147719826240, 2019.

[33] B. Li, W. Feng, L. Zhang, and C. J. Spanos. DEPEND: Density adaptive power efficient neighbor discovery for wearable body sensors. In *IEEE International Conference on Automation Science and Engineering (CASE)*, 2013.

[34] J. Liu and C. Chen. Energy analysis of neighbor discovery in bluetooth low energy networks. Technical report, 07/2012 2012.

[35] J. Liu, C. Chen, and Y. Ma. Modeling and performance analysis of device discovery in bluetooth low energy networks. In *2012 IEEE Global Communications Conference (GLOBECOM)*, pages 1538–1543, 2012.

[36] J. Liu, C. Chen, and Y. Ma. Modeling neighbor discovery in bluetooth low energy networks. *IEEE Communications Letters*, 16(9):1439–1441, 2012.

[37] J. Liu, C. Chen, Y. Ma, and Y. Xu. Adaptive device discovery in bluetooth low energy networks. In *IEEE Vehicular Technology Conference (VTC Spring)*, pages 1–5, June 2013.

[38] M. Magno, V. Jelicic, B. Srbinovski, V. Bilas, E. M. Popovici, and L. Benini. Design, implementation, and performance evaluation of a flexible low-latency nanowatt wake-up radio receiver. *IEEE Transactions on Industrial Informatics*, 12(2):633–644, 2016.

[39] R. Margolies, G. Grebla, T. Chen, D. Rubenstein, and G. Zussman. Panda: Neighbor discovery on a power harvesting budget. *IEEE Journal on Selected Areas in Communications*, 34(12):3606–3619, 2016.

[40] Y. Matsuo, D. Umehara, H. Murata, and S. Denno. Performance evaluation of device discovery in Bluetooth LE with probabilistic model checking. In *Asia-Pacific Symposium on Information and Telecommunication Technologies (APSITT)*, 2015.

[41] Bluetooth SIG. Specification of the Bluetooth system 4.0, June 2010. Volume 0, available via `bluetooth.org`.

[42] Bluetooth SIG. Find me profile specificiation, June 2011. Revision V10r00, available via `bluetooth.org`.

[43] Bluetooth SIG. Specification of the Bluetooth system 5.0, December 2016. Volume 0, available via `bluetooth.org`.

[44] Borges, Paulo et Al. Bluetooth Low Energy Software Stack for Embedded Devices (BLESSED), 2015. `https://github.com/pauloborges/blessed`.

[45] Dynastream Innovations Inc. ANT AN11: ANT channel search and background scanning channel, 2009. Revision 2.1, available via `thisisant.com`.

[46] Dynastream Innovations Inc. ANT message protocol and usage, 2014. Revision 5.1, available via `thisisant.com`.

[47] Dynastream Innovations Inc. Why ANT+: The facts, 2015. `www.thisisant.com/business/why-ant/facts`.

[48] IEEE. 802.11 tutorial - 802.11 MAC entity: MAC basic access mechanism privacy and access control, 1996.

[49] IEEE Computer Society. IEEE Standard 802.11-2012, 2012. Document No. IEEE P802.11-96/49C, available via `www.ieee802.org/11/Tutorial/MAC.pdf`.

[50] IEEE Computer Society. IEEE standard for low-rate wireless networks, 2016.

[51] Mathworks, Inc. Documentation of the tic function, 2015. available via `http://de.mathworks.com/help/matlab/ref/tic.html`.

[52] Nordic Semiconductor ASA. nRF51822 development kit user guide v 1.3, July 2013. available via `nordicsemi.com`.

[53] Nordic Semiconductor ASA. nRF51822 product spec. v3.1, 2014. Available via `nordicsemi.com`.

[54] M. J. McGlynn and S. A. Borbash. Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks. In *ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc)*, 2001.

[55] T. Meng, F. Wu, and G. Chen. On designing neighbor discovery protocols: A code-based approach. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1689–1697, 2014.

[56] T. Meng, F. Wu, and G. Chen. Code-based neighbor discovery protocols in mobile wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 24(2):806–819, 2016.

[57] O. Oystein. *Number Theory and its History*. McGraw-Hill, 1948.

[58] D. Perez Diaz de Cerio, Hernández-Solana, Ángela, J. L. Valenzuela, and A. Valdovinos. Analytical and experimental performance evaluation of BLE neighbor discovery process including non-idealities of real chipsets. *Sensors*, 17(3):499, 2017.

[59] S. T. S. Portal. Bluetooth Low Energy (BLE) enabled devices market volume worldwide, from 2013 to 2020 (in million units)., 2018. `www.statista.com/statistics/750569/worldwide-bluetooth-low-energy-device-market-volume`.

[60] A. Purohit, B. Priyantha, and J. Liu. Wiflock: Collaborative group discovery and maintenance in mobile sensor networks. In *ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2011.

[61] Y. Qiu, S. Li, X. Xu, and Z. Li. Talk more listen less: Energy-efficient neighbor discovery in wireless sensor networks. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1–9, 2016.

[62] C. Röhl, H. Woesner, and A. Wolisz. A short look on power saving mechanisms in the wireless lan standard draft ieee 802.11. In *WINLAB Workshop on Third Generation Wireless Systems*, 1997.

[63] C. Schurgers, V. T. Tsiatsis, S. Ganeriwal, and M. B. Srivastava. Optimizing sensor networks in the energy-latency-density design space. *IEEE Transactions on Mobile Computing (TMC)*, 1(1):70–80, 2002.

[64] M. Senior. Novartis signs up for google smart lens. *Nature Biotechnology*, 32:856, 2014.

[65] D. Spenza, M. Magno, S. Basagni, L. Benini, M. Paoli, and C. Petroli. Beyond duty cycling: Wake-up radio with selective awakening for long-lived wireless sensing systems. In *IEEE Conference on Computer Communications (INFOCOM)*, 2015.

[66] Statista. Anzahl der personen in deutschland, für die eine lange akkulaufzeit ein besonders wichtiges kriterium beim kauf eines handys oder smartphones ist, von 2013 bis 2015 (in millionen).

[67] W. Sun, Z. Yang, K. Wang, and Y. Liu. Hello: A generic flexible protocol for neighbor discovery. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 540–548, 2014.

[68] Y.-C. Tseng, C.-S. Hsu, and T.-Y. Hsieh. Power-saving protocols for IEEE 802.11 based multi-hop ad hoc networks. In *IEEE Conference on Computer Communications (INFOCOM)*, 2002.

[69] S. Vasudevan, M. Adler, D. Goeckel, and D. Towsley. Efficient algorithms for neighbor discovery in wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 21(1):69–83, 2013.

[70] S. Vasudevan, J. F. Kurose, and D. F. Towsley. On neighbor discovery in wireless networks with directional antennas. In *IEEE Conference on Computer Communications (INFOCOM)*, volume 4, 2005.

[71] S. Vasudevan, D. F. Towsley, D. Goeckel, and R. Khalili. Neighbor discovery in wireless networks and the coupon collector's problem. In *Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2009.

[72] H. Wang, J. Ma, Y. Liu, W. Liu, and L. Wang. Bi-directional probing for neighbor discovery. In *IEEE International Conference on Computational Science and Engineering (CSE)*, 2014.

[73] K. Wang, X. Mao, and Y. Liu. BlindDate: A neighbor discovery protocol. In *International Conference on Parallel Processing (ICCP)*, pages 120–129, 2013.

[74] K. Wang, X. Mao, and Y. Liu. BlindDate: A neighbor discovery protocol. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 26(4):949–959, 2015.

[75] R. Want, B. Schilit, and D. Laskowski. Bluetooth LE finds its niche. *IEEE Pervasive Computing Magazine*, 12(4):12–16, 2013.

[76] 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*, 20(5):966–969, 2016.

[77] D. Yang, S. Jongmin, J. Kim, and C. Kim. An energy-optimal scheme for neighbor discovery in opportunistic networking. In *IEEE Consumer Communications and Networking Conference (CCNC)*, 2009.

[78] D. Yang, J. Shin, J. Kim, and G.-H. Kim. Opeed: Optimal energy-efficient neighbor discovery scheme in opportunistic networks. *Journal of Communications and Networks*, 17(1):34–39, 2015.

[79] R. D. Yates and D. J. Goodman. *Probability and Stochastic Processes*. John Wiley & Sons, Inc, second edition, 2005.

[80] L. You, Z. Yuan, P. Yang, and G. Chen. Aloha-like neighbor discovery in low-duty-cycle wireless sensor networks. In *IEEE Wireless Communications and Networking Conference (WCNC)*, 2011.

[81] W. Zeng, S. Vasudevan, X. Chen, B. Wang, A. Russell, and W. Wei. Neighbor discovery in wireless networks with multipacket reception. In *Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2011.

[82] 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 *ACM Conference on Embedded Network Sensor Systems, (SenSys)*, 2012.

[83] D. Zhang, T. He, F. Ye, R. K. Ganti, and H. Lei. Neighbor discovery and rendezvous maintenance with extended quorum systems for mobile applications. *IEEE Transactions on Mobile Computing (TMC)*, 16(7):1967–1980, 2017.

[84] M. Zhang, L. Zhang, P. Yang, and Y. Yan. McDisc: A reliable neighbor discovery protocol in low duty cycle and multi-channel wireless networks. In *IEEE International Conference on Networking, Architecture and Storage (NAS)*, 2013.

[85] Y. Zhang, K. Bian, L. Chen, P. Zhou, and X. Li. Dynamic slot-length control for reducing neighbor discovery latency in wireless sensor networks. In *IEEE Global Communications Conference (GLOBECOM)*, 2017.

[86] Z. Zhang and B. Li. Neighbor discovery in mobile ad hoc self-configuring networks with directional antennas: algorithms and comparisons. *IEEE Transactions on Wireless Communications (TWC)*, 7(5):1540–1549, 2008.

[87] R. Zheng, J. C. Hou, and L. Sha. Asynchronous wakeup for ad hoc networks. In *ACM International Symposium on Mobile Ad Hoc Networking & Computing (MobiHoc)*, 2003.

[88] R. Zheng, J. C. Hou, and L. Sha. Optimal block design for asynchronous wake-up schedules and its applications in multihop wireless networks. *IEEE Transactions on Mobile Computing (TMC)*, 5(9):1228–1241, 2006.

A
Symbols and Acronyms

Table of Symbols

Chapter 2 - Fundamental Bounds

α	Ratio of the power spent for transmission over the power spent for reception
η	Duty-cycle
$\eta_{B,m}$	Specified maximum channel utilization
η_B	Duty-cycle for transmission, which is equivalent to the channel utilization
η_C	Duty-cycle for reception
Λ	Coverage of a beacon sequence B' given an infinite reception window sequence C_∞
$\Lambda^*(\Phi_1)$	Number of beacons that cover the offset Φ_1
λ_i	Gap between beacon i and beacon $i+1$
μ	Constant ratio of the reception or beaconing duty-cycles of two devices
Ω_i	Set of offsets Φ_1 covered by beacon i
ω_i	Transmission duration of beacon i
Φ_i	Offset of the i'th beacon of a beacon sequence from the coordinate offset in C
τ_i	Time at which beacon i is sent
ζ	Fixed temporal distance of a certain beacon from a certain reception window on the same device in every period T_C
B/B_∞	Finite/infinite beacon sequence
b_i	Beacon i
C/C_∞	Finite/infinite reception window sequence
c_i	Reception window i
d_i	Time duration of reception window i
d_{oRxTx}	Effective additional active time for switching from reception to transmission
d_{oRx}	Effective additional active time for switching from sleep to reception and vice-versa
d_{oTxRx}	Effective additional active time for switching from transmission to reception
d_{oTx}	Effective additional active time for switching from sleep to transmission and vice-versa
I	Slot length in a slotted protocol

L	Worst-case latency
l^*	Beacon-to-beacon latency: Worst-case latency measured from the first beacon in range to the last, successfully received one.
L_i	Ideal worst-case discovery latency
L_r	Worst-case discovery latency when relaxing all simplifying assumptions
M	Minimum number of beacons needed to deterministically match an infinite reception window sequence C_∞
m_B	Period of a repetitive beacon sequence (in terms of number of beacons)
n_C	Number of reception windows contained in a finite length reception window sequence C, whose concatenations form an infinite sequence C_∞
P_c	Collision probability
P_f	Probability of a failed discovery
P_{Tx}, P_{Rx}	Power consumption of a radio for transmission or reception, respectively
S	Number of transmitting devices
T_B	Time-period of a repetitive, infinite beacon sequence
T_C	Time between the ends of two consecutive instances of the finite reception window sequence C, whose concatenations form an infinite sequence C_∞
t_i	Point in time the reception window i begins at

Chapter 3 to 5 - PI-based Protocols

a_k	Coefficient of a linear combination
d_s	Scan Window
d_t	Distance left to travel
d_{Nl}, d_f, d_{Nu}	Length of the part of a segment that has $N_l - 1/N_l ... N_u/N_u + 1$ intervals to travel until reaching a target area
d_e	Duration for sending three advertising packets in a row on different channels in BLE
d_{loh}	Hitting duration of a leftover partition
d_{lom}	Missing duration of a leftover partition
d_{lo}	Leftover duration (Duration of a partition which does not fit entirely in the partition of its previous-order sequence)
d_{rt}	Duration a radio takes to switch from reception to transmission
$d_{s,m}$	Lower limit of the scan window d_s the hardware can realize
d_{tr}	Duration a radio takes to switch from transmission to reception
ϵ	Arbitrarily small value close to zero
η	Duty-cycle
η_t	Target duty-cycle a protocol should realize
f_{clk}	Frequency of the sleep clock of a radio
γ	Growth/Shrinkage per interval
$\vec{\gamma}$	Vector of γ-parameters for all orders
γ_s,γ_g	γ-parameter given the sequence is shrinking/growing
γ_{wc}	Worst-case value of γ
Γ	Probability buffer
$\|\Gamma\|$	Number of segments in the probability buffer Γ
$\Gamma[k]$	Probability density of segment $k \in \Gamma$
$\overline{G}$	Mean gain over a protocol under comparison

G_m	Maximum gain over a protocol under comparison
h, i, j	Generic indices / integer multiples
$i_n,\ j_n$	Multiplier of advertising (i_n) and scan intervals (j_n)
I	Slot length (in slotted protocols)
$\overline{\kappa}$, κ_m	Normalized root mean square error for mean/maximum latencies
k, M	Integer numbers
$\overline{K}$, K_m	Root mean square error (RMSE) for mean/maximum latencies
$\overline{l_p}$	Weighted partial latency of a partition
$\overline{l}$	Mean packet-to-packet ND latency
l	Packet-to-packet worst-case neighbor discovery latency
l^*	Packet-to-packet neighbor discovery latency
l_{1W}	Packet-to-packet latency for one-way discovery
l_{2W}	Packet-to-packet latency for non-assisted two-way discovery
l_{A2W}	Packet-to-packet latency for assisted two-way discovery
l_{comp}, l_{sim}	Computed/Simulated latency
l_{th}	Threshold discovery latency
L	Worst-case ND latency
L_p	Maximum partial latency for a part of a segment
m	Mode of a sequence (s $\leftarrow$ shrinking, g $\leftarrow$ growing, c $\leftarrow$ coupling). In addition, m is also used as a parameter in G-Nihao [61].
n	Index for the order of a sequence
n_m	Maximum sequence order
n_{Nodes}	Number of nodes
$\mathbb{N}_0$	Set of all non-negative integers
N_f	Number of periodic repetitions of the one-way set of partitions Π
N_i	Number of intervals until reaching the next target area
N_l, N_u	Minimum/Maximum number of γ-intervals until reaching a target area for a segment of Γ
N_{sh}	Number of intervals to shrink a certain distance
$\overline{MD}$, MD_m	Maximum deviation ($\max(\lvert l_{comp} - l_{sim}\rvert)$) for mean/maximum latencies
o	Overhead (additional active time)
p	Probability density
p_i, l_i	Probability density/Latency of a segment i of a probability buffer or of a partition of initial offsets
p_{blk}	Blocking probability (probability of failed discoveries because of the turnaround times d_{rt} and d_{tr} and because of collisions)
p_{col}	Probability of collisions
Π	Partition of initial offsets
P	CDF function
P_{1W}	CDF function for one-way discovery
P_{2W}	CDF function for two-way discovery
P_{A2W}	CDF function for mutually assisted two-way discovery
Φ	Offset between an advertising packet and the closest neighboring scan window on its left side
Φ'	Offset between an advertising packet and the closest neighboring scan window on its right side

$\Phi[0]$	Initial offset
$\Phi_E[0]$	Initial offset between a scan window of device E an advertising packet of its remote device
Q	Number of γ intervals that fit into d_t.
QE	Accumulated clock quantization error
ρ	Amount of time d_s is extended by beyond its theoretically optimal value
R	Balancing variable for trading worst-case against mean latencies in A2W protocols
R_{opt}	Value of R (see above) that leads to the shortest mean latencies of A2W protocols
σ, σ_s	Penalty related to γ / Penalty related to d_t
S	Largest sum of penalties $\sum \sigma$
t/t'	Original/Transformed point in time
$t_e[k]$	Ending time of a segment $k \in \Gamma$ or of a partition k in Π
$t_s[k]$	Starting time of a segment $k \in \Gamma$ or of a partition k in Π
T_a	Advertising Interval
T_s	Scan Interval
υ_{γ_n}	Linear combination $\sum_{k=0}^{n} a_k \cdot \gamma_k$
$\zeta[k]$	Largest sum of penalties for a segment $k \in \Gamma$
ω	Duration of an advertising packet

Table of Acronyms

ND neighbor discovery

MANET mobile ad-hoc network

BLE Bluetooth Low Energy

CSMA carrier-sense multiple access

CA collision avoidance

IoT Internet of Things

PI periodic interval

CDF cumulative distribution function

A2W assisted two-way

RMSE root mean square error

NRMSE normalized root mean square error