Cross layer design for RFID systems

Hou, Yuxiao

2017

Hou, Y. (2017). Cross layer design for RFID systems. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/69453>

<https://doi.org/10.32657/10356/69453>

NANYANG TECHNOLOGICAL UNIVERSITY

Cross Layer Design for RFID Systems

By

Yuxiao Hou

Supervisor: Dr. Mo Li

School of Computer Engineering

This thesis is submitted to Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

July, 2016

Acknowledgments

First, I would like to express my sincere thanks to my advisor Prof. Mo Li for his support, instruction and encouragement. Prof. Li shares with me his valuable experiences and insightful advices. Prof. Li has also extensively participated in research and contributed tremendously to all of my work. Thank you very much.

I would like to thank Prof. Wentong Cai, Prof. Chengzheng Sun, Prof. Xueyan Tang, Prof. Dusit Niyato, Prof. Bingsheng He, Irene Ng-Goh Siew Lai in the previous Parallel and Distributed Computing Center (PDCC), Wong Jun Hao and all new members in the Computer Networks and Communications Graduate Lab (CNCL), for creating great research environments. Special thanks are given to Prof. Chengzheng Sun for playing table tennis with me. I would also like to acknowledge the four-year financial support from research scholarship award.

Im grateful to my dear lab-mates in WANDS: Yuanqing Zheng, Wan Du, Zhengjiang Li, Zhidan Liu, Pengfei Zhou, Jiajue Ou, Cheng Li, Yaxiong Xie, Shiqi Jiang, Jansen Liando, Chu Cao, Huanle Zhang. I will always remember the days we catch paper deadlines, discuss on interesting research topics, do sports and have parties. I give special thanks to Yuanqing and Jiajue, who greatly support my research and advise on my life. You will always be our “elder brother” and “elder sister”.

I am grateful to many friends I met in Singapore. They include but not limited to: Diep Pham, Yi Xu, Yuguang Mu, Tong Liu, Wenbin Ji, Assel Sakanova, Xue Wang, Chiniah Aatish, Yao Chen, Qitian Cheng, Xin Hua, Cheng Yan. Thank you for your company and help. Among these friends, I am most thankful to Diep Pham, who always supports me in my hard times and instructs me to be strong and lives an independent life. I am deeply indebted to you.

Finally, I am thankful to my parents who always support and encourage me, even if they are far from me. I love you.

Abstract

Recently, Radio Frequency IDentification (RFID) systems are widely adopted in real world applications, such as inventory management, item tracking, and access control. Some fundamental RFID operations underlying almost every application, like identification, population estimation, missing tag detection should be executed with high efficiency and accuracy. Although a rich amount of works are devoted to improve the performances of these fundamental operations, most of these works suffer greatly from unconscious and frequent tag collisions – especially in a large tag set – and hence produce limited performance gains. In this thesis, I adopt a cross-layer design principle to facilitate these traditional MAC-layer operations with physical-layer information extracted from each colliding tag response signal, which is largely ignored by previous works. Specifically, two types of RFID operations are considered: population (or cardinality) estimation and identification.

Since pinpointing the exact number of tags in a given set is expensive and unnecessary in most application scenarios, typical methods often take probabilistic models to estimate the cardinality within an accuracy requirement specified by the application. While most of existing probabilistic estimate methods detect and collect only binary states (i.e., empty or colliding) from the series of tag response slots, in this thesis, I propose to detect extra integer states (i.e., the exact number of colliding tags) in each colliding slot, by counting the number of clusters formed in the corresponding constellation map. As each integer state can infer an independent cardinality, I combine estimations based on all states into an optimal one (i.e., error is minimized) as the ultimate result. I name the proposed scheme after PLACE (Physical LAYER Cardinality Estimation). Experiments based on USRP2/WISP testbed and large-scale simulations show that PLACE achieves around $3\sim 4\times$ gains over state-of-the-art cardinality estimation schemes.

RFID identification identify all tags in an unknown set with their binary IDs. Although current tree-based schemes improve the identification speed over classic ALOHA-based ones, their common underlying assumption on uniform ID distribution is not justified in most practical scenarios. In this thesis, I propose to exploit two types of physical layer information in each colliding slot and collect from them the hints on tag ID distribution in the binary tree. First, the RFID reader detects if all colliding tags in the same slot transmit the same bit at each ID index; a positive result at an index indicates no tags reply to a certain prefix pattern and prefixes matching the pattern can thus be skipped for subsequent queries. Second, the reader estimates the number of colliding tags in one slot and compute accordingly the optimal number of bits appended to the current prefix for subsequent queries. I name the proposed scheme after *PHY-Tree*. Experiments based on USRP2/WISP testbed and trace-driven simulations demonstrate that *PHY-Tree* reduce the number of reader queries of state-of-the-art schemes by $1.79\times$, on average.

Contents

Acknowledgments	i
Abstract	ii
List of Figures	vii
List of Tables	ix
1 Introduction	1
1.1 Cardinality Estimation	2
1.2 Identification	4
1.3 Organization of This Thesis	7
2 PLACE: Physical Layer Cardinality Estimation for Large-Scale RFID Systems	8
2.1 Background	8
2.1.1 Problem description	8
2.1.2 Initial observation from our software defined testbed	9
2.2 Slot State Detection Algorithm	12
2.3 Estimation Algorithm	16
2.3.1 Estimation protocol	16
2.3.2 Computing the number of rounds m	18
2.3.3 Two-phase Counting Algorithm	21
2.3.4 Discussion	24
2.4 Impact of SSDA Errors and Enhancement	25
2.4.1 Enhanced PLACE	25
2.4.2 Impact of SNR on q_0	28
2.5 Evaluation	30

2.5.1	SSDA Evaluation	30
2.5.2	PLACE Evaluation	32
2.5.3	The Impact of SSDA Detection Errors on PLACE	35
2.5.4	The Impact of SNR on SSDA Detection Accuracy	37
2.6	Related work	38
2.7	Summary	39
3	Physical Layer Tree-based RFID Identification	41
3.1	Background & Motivation	41
3.1.1	RFID backscatter	41
3.1.2	RFID identification	42
3.1.3	Motivation	43
3.2	Horizontal Information	46
3.2.1	Principle of Horizontal Information	46
3.2.2	Robust Detection Algorithm	48
3.3	Vertical Information	50
3.3.1	Intuition & Definition	50
3.3.2	Basic Observations from Experiment and Simulation Results . . .	51
3.3.3	Appending Rule	54
3.3.4	Modelling EA over k	55
3.4	PHY-Tree Algorithm	56
3.4.1	A Basic Algorithm	56
3.4.2	Error Compensation	59
3.4.3	Further Improvement in Efficiency	59
3.4.4	Ultimate PHY-Tree Algorithm	60
3.5	Discussion on synchronization	61
3.6	Trace-Driven Simulation	62
3.6.1	Microbenchmark	62
3.6.2	Comparison with Existing Identification Schemes	67
3.6.3	Impact of Unsynchronization Rate	69
3.7	Related work	71
3.8	Summary	73

4 Conclusion & Future Work	74
Appendix	76
A Author's Publications	76

List of Figures

1.1	Tag response signals and their corresponding constellation maps.	5
2.1	Our testbed with a GNURadio/USRP2 platform and 4 WISP tags. . . .	10
2.2	An illustrative example of Slot State Detection Algorithm (SSDA). . . .	14
2.3	Detection accuracies of SSDA with different system parameters.	15
2.4	m against k	20
2.5	$g(\lambda)$ against λ	20
2.6	The expectation of $\bar{X}_k$ over load factor λ	22
2.7	Performance of the toy estimation scheme.	26
2.8	Detailed accuracy performance of SSDA.	27
2.9	Illustration on how to utilize the noise before tag response signal to obtain the noise power and reference point.	29
2.10	Performance comparison of DBSCAN and SSDA.	31
2.11	Comparison of operation time to meet different accuracy requirements among 5 schemes: EFNEB, LoF, ZOE, SRC and PLACE.	33
2.12	CDF of estimation results with the same amount of execution time. . . .	34
2.13	Impact of q_0 on estimation accuracy of PLACE.	36
2.14	Impact of SNR of tag response signal on SSDA detection accuracy. . . .	37
3.1	Ideally, 2^k number of clusters appear in the constellation map when k tags collide. However, in practice clusters may overlap.	42
3.2	Illustration on how the proposed <i>PHY-Tree</i> scheme accelerate identifica- tion performance compared to classic Query Tree scheme.	44
3.3	FM0 and Miller-2 coding of a common binary sequence 001011.	46
3.4	Colliding signal of two tags with different IDs using FM0 coding.	47

3.5	Constellation map of 3 tags. Their 10-bit IDs include all 8 bit states, i.e., from 000 to 111.	49
3.6	Tendency between average EA and k . The average EA is computed based on experimental traces.	52
3.7	Tendency between average EA and k . The average EA is computed based on simulated traces.	54
3.8	Searching for optimal range for c in resonant state detection algorithm with different k s.	63
3.9	Extensive study of the performance of the resonant state detection algorithm in obtaining horizontal information when k varies in a wide range and c is fixed to be optimal.	64
3.10	Impact of false positive of horizontal information on overall performance of <i>PHY-Tree</i>	65
3.11	Accuracy comparison in estimating k between EA- k model and SSDA algorithm in [28].	66
3.12	Performance comparison between several schemes under three types of tag ID distribution.	68
3.13	Detailed study on block ID distribution.	69
3.14	The impact of r on the accuracy of horizontal information with initially optimal c for $r = 0$	70
3.15	Impact of r on the average NoQ of <i>PHY-Tree</i>	71

List of Tables

Chapter 1

Introduction

Recently RFID systems (consisting of one or more RFID reader and many RFID tags) [20, 62] have been widely deployed in real-world scenarios, due to several attractive features of RFID tags: tiny form, low power cost, cheap price, etc. RFID tags can thus be attached to most objects to serve multiple purposes such as inventory management [67, 70, 71], access control [5, 26, 30], human-machine interaction [58], localization [41, 57], and mobility tracking [14, 59, 63].

Many fundamental MAC-layer operations are required for almost every RFID system, These operations include identification (identify all tags with their IDs), population estimation (estimate the total number of tags in a set within given accuracy), missing tag detection (identify several missing tags from an existing set), etc. Although a number of works tempt to improve the efficiency of these operations, most of them have limited improvement space, due to unconscious and frequent tag collisions (especially when the tag set population is large).

In this thesis, I want to ask the following question: instead of completely discarding the colliding slots during the operation process, is it possible to extract from them some useful information to facilitate the operation?

The answer is yes. Indeed, some useful information about the number and ID patterns of colliding tags, hide behind the physical-layer colliding tag response signal. By carefully

examining the backscatter nature of RFID tags, these physical-layer information can be extracted and utilized in MAC-layer operations.

The application of this general cross-layer design principle takes different forms in different operations. In this thesis, I study two fundamental operations: population (or cardinality) estimation and identification.

1.1 Cardinality Estimation

Counting the number of tags is a fundamental operation. Knowing the tag cardinality can facilitate many primary operations in RFID systems such as tag identification [68] and tag searching [69]. An estimation with guaranteed accuracy normally suffices for the practical purposes. As such, many probabilistic counting methods [16, 25, 32, 33, 51, 53, 72, 73] trade the estimation accuracy for the execution time. Previous works typically measure the states of f communication slots, where each tag responds in one random slot. The slot state can be binary if we distinguish busy and idle slots, or it can be ternary if we further differentiate singleton and collision slots. Thus, the tag responses in f slots can be represented with a $f \times 1$ binary or ternary sequence, with zeros representing idle slots. Intuitively, when a larger number of tags participate, we expect more tag responses and consequently fewer idle slots in the response sequence. Despite the subtleties in design details, previous methods estimate the tag cardinality by examining the state of each slot in the response frame and following a probability model to derive the cardinality.

For instance, one previous work EFNEB [25] uses the first busy slot to estimate, while ZOE [72] computes the ratio of zero entries over f slots and derives the tag cardinality. The most recent work [16] advocates the importance of two-phase estimation, and approaches theoretical optimal performance with the binary responses. As only 1 bit or slightly more information is extracted from each slot, previous methods need substantial number of slot measurements to guarantee an estimation accuracy.

In this thesis, I present PLACE, a Physical LAYER Cardinality Estimation scheme which extracts more information from each tag response slot, thereby achieving higher estimation efficiency. Unlike previous methods which only distinguish binary or ternary states in each slot, I show that it is possible to detect the number of concurrent tag responses and thus infer integer states from the same slot at RFID physical layer.

To illustrate the possibility of detecting integer slot states, Figure 1.1 plots the received response signals (in complex values) when 0, 1, 2, and 3 tags collide in one slot. Specifically, Figure 1.1 (a),(c),(e),(g) plot the amplitude of the signals over time; Figure 1.1 (b),(d),(f),(h) plot the signals in 2D complex plane (or I-Q plane). Due to certain modulation schemes, the received signal in the I-Q plane will form certain number of clusters, each of which represents one modulation state. Such a set of clusters is called the constellation map of the received signal. These traces are collected through the USRP2/WISP platform (described in later chapters). In cases where more than 1 tag collide together (shown in Figure Fig1:KCluster (f),(h)), individual tags exhibit different signal amplitudes and phases, which is caused by many factors including complex electromagnetic environment, relative positions between tags and between reader and tags, types of tags, antenna polarization, etc. It is noted that the overall shape of the constellation map could be regular sometimes (like the ones in Figure 1.1), but could also be irregular in other times.

From Figure 1.1, we can easily observe that each of the signal states is shown more clearly in the 2D constellation map (i.e., Figure 1.1 (b),(d),(f),(h)), than in the 1D amplitude-time plot (i.e., Figure 1.1 (a),(c),(e),(g)). Specifically, if k tags collide together, we can observe 2^k distinct clusters in the corresponding constellation map. This is because each tag takes one of the two states by either reflecting or absorbing radio waves from the RFID reader. Such observation inspires us to detect the exact number of concurrent tag responses in each slot. Ideally, we can infer the number of responding tags from

the number of clusters formed, and thereby extend the binary or ternary sequence to an integer sequence.

Although simple in concept, the implementation of physical layer estimation entails many practical challenges. (1) Accurate and efficient estimation of the symbol clusters is non-trivial. In particular, the symbol clustering and counting operation has to be accommodated into the time frame of each RFID slot. In this thesis, I design a slot state detection algorithm that divides the I-Q plane into grids and derives the symbol clusters based on the symbol densities of grids. The proposed SSDA algorithm takes only millisecond time in comparison with general clustering algorithms that may take hundreds of seconds. (2) Novel cardinality estimator needs to be designed to make the best use of sequences of integer-state transmission slots instead of previous busy/idle slots. The proposed PLACE scheme combines multiple estimations obtained from each integer slot state and uses an optimal joint estimator such that the overall variance is minimized. (3) Due to noises in practical RFID transmissions, the cluster estimation output inherently contains errors. I run full experiments to understand such errors and analyze the impact on final cardinality estimation accuracy. The analysis and experiments show that the developed probabilistic estimators in PLACE tolerate the error level from practical measurements.

Results from extensive experiments and simulations demonstrate that PLACE achieves approximately $3\sim 4\times$ performance gain over state-of-the-art cardinality estimation schemes.

1.2 Identification

Another fundamental operation is to read tag IDs (a.k.a., RFID identification). Two major types of schemes are ALOHA-based and tree-based. In ALOHA based schemes [50, 68], each tag randomly selects a time slot and responds to reader's query, leading to frequent tag collisions and low communication efficiency. In contrast, tree based schemes

1.1.a: RN-16 signal of noise

1.1.b: noise, 1 cluster

1.1.c: RN-16 signal of 1 tag

1.1.d: 1 tag, 2 clusters

1.1.e: RN-16 signal of 2 tags

1.1.f: 2 tags, 4 clusters

1.1.g: RN-16 signal of 3 tags

1.1.h: 3 tags, 8 clusters

Figure 1.1: Tag response signals and their corresponding constellation maps.

allow readers to actively select a subset of tags with prefix matching. In general, tree-based schemes provide more stable identification performance but involve more reader-tag interactions.

Existing tree-based works [12, 34, 40, 47, 52, 54] reduce the number of interactions assuming uniform ID distribution which is invalid in many practical scenarios. For example, customers randomly pick up items attached with RFID tags in open hours of a shop. At the end of the day, the shop needs to identify the remaining items with their IDs, which follow highly random distribution. While uniform distribution can be uniquely characterized by the same difference in neighbouring tag ID values, random distribution is hard to be predicted. Thus existing schemes degrade their performances under random distribution.

In this thesis, I design a novel tree-based scheme, named *PHY-Tree*, which infers local ID distribution in corresponding subtrees from physical layer hints and thus achieves large gains by adjusting following queries accordingly. In particular, *PHY-Tree* optimizes queries by skipping unnecessary queries and quickly converge to all tags.

When multiple tags collide, we can detect whether all responding tags reply the same bit at each bit index by combining prior knowledge of tag coding scheme and physical layer patterns. For example, if the query prefix is “0” and the all-0 state is detected at index 3, the reader infers that no tags reply with the prefix pattern “0*1” (where * represents any bit value at index 2) and hence skips querying prefixes matching this pattern. From the binary tree view, this is equivalent to pruning the right branch of the layer-2 nodes rooted at “0”. Since this information tells whether left or right branches of certain tree nodes contain any tags, I call it “horizontal information”.

In addition, based on collision patterns on physical layer constellation map, we can roughly infer the number of colliding tags in each query. For example, if 4 tags collide when the reader queries prefix 01, it appends two bits to the prefix and queries 4 new

prefixes (0100, 0101, 0110, 0111) to better resolve the collision. From the view of the binary tree, the reader skips two children nodes (i.e, 010 and 011) and directly hops to the 4 grandchildren nodes. I thus call this information “vertical information”. We see many opportunities to safely skip unnecessary query nodes especially when the IDs are not uniformly distributed.

In practice, however, the horizontal and vertical information may not always provide correct hints to guide the query process. While inaccurate vertical information only incurs some unnecessary queries, a false-positive horizontal information may miss some tags. To overcome such problems, I propose an efficient error compensation mechanism to eliminate false positives so that the horizontal information can be safely utilized.

The trace-driven simulation results are summarized as follows. The horizontal information achieves 80% overall accuracy and no more than 1% false positive rate when 20 tags collide. The extra queries incurred by the error compensation scheme is no more than 4% of the total number of queries, on average. The vertical information achieves approximately 80% accuracy for up to 8 colliding tags. Finally, the integrated *PHY-Tree* algorithm outperforms state-of-the-art tree-based identification scheme by $1.79\times$.

1.3 Organization of This Thesis

The rest of this thesis is organized as follows. Chapter 2 and Chapter 3 detail the design of PLACE and *PHY-Tree*, respectively. Chapter 4 concludes this thesis and discusses on future works.

Chapter 2

PLACE: Physical Layer Cardinality Estimation for Large-Scale RFID Systems

2.1 Background

2.1.1 Problem description

We consider a single-reader, multi-tag RFID system, which is the typical scenario for prior works [16, 25, 32, 33, 51, 53, 67–69, 72, 73]. In such a system, passive UHF tags, which operate in the 900MHz band and transmit by backscattering reader’s continuous wave (CW) signal, are assumed in this chapter.

Commercial off-the-shelf (COTS) RFID readers adopt the Frame ALOHA protocol as the MAC layer access protocol to read multiple tags in a set. A time frame consists of many time slots with equal duration. An RFID tag pick up one slot randomly in the whole frame to respond to the reader’s query command. Thus, a time slot can be categorized into 2 general states – idle or busy, depending on whether any tags reply in this slot. In probabilistic-based population estimation works, a tag can respond to the reader with a short (16 bits) binary sequence named RN16, rather than its long (96 bits) EPC [1] (often used in RFID identification [68]), to improve estimation efficiency.

Denote the ground truth of tag population of a given set as t and the corresponding estimation as $\hat{t}$. The desired accuracy requirement, in terms of (ϵ, δ) , is:

$$Pr\{|\hat{t} - t| \leq \epsilon t\} \geq 1 - \delta. \quad (\text{Eq. 2.1})$$

We further explain Eq. 2.1 with a brief example. If t is 2000 and $(\epsilon, \delta) = (10\%, 5\%)$, we expect $\hat{t}$ fall into the range $[1800, 2200]$ with $\geq 95\%$ probability. Given the same accuracy requirement as specified in Eq. 2.1, an algorithm outperforms others if it takes less operation time.

Despite the rich amount of works in designing such an efficient algorithm [16, 25, 32, 33, 51, 53, 72, 73], most of these existing works obtain limited amount of information from each single slot – binary information which only tells whether it is busy or not – regardless of their protocol details. EFNEB [25] focused on the binary index of the first busy slot in the frame. ZOE [72] computed the ratio between the number of busy slots and that of idle ones. ART [53] observed the pattern of the number of continuous busy slots. These protocols [25, 53, 72] have a common point: they first performed the estimation coarsely with a small number of probing slots and further refined it within the specified accuracy with much more slots. Such a style was captured, carefully analyzed, and proved to be a key factor that could affect estimation efficiency, by SRC [16]. Despite all above facts, these works, including SRC, overlooked the potentially rich information from the physical layer, and thus had their performances upper bounded (as suggested in SRC [16]).

2.1.2 Initial observation from our software defined testbed

We reply on our USRP/WISP platform to provide initial insight on how to utilize physical layer information to improve the efficiency of cardinality estimation. The USRP device equipped with an RFX900 daughterboard function as the software defined radio (SDR)

Figure 2.1: Our testbed with a GNURadio/USRP2 platform and 4 WISP tags.

reader, which can send reader command signals and receive tag responses, as it can generate and transmit radio signals in the 900Hz band. This USRP reader connects to the laptop through an Ethernet interface for further processing. We configure the sampling rate of the SDR reader as 4 million samples per seconds, or 4MS/s. In other words, the reader could sample 4000 physical layer symbols in 1ms. It is noted that each physical layer symbol take complex value, whose in-phase (I) and quadrature (Q) components specify a point in the 2D I-Q plane.

The Backscatter Link Frequency (BLF) of an RFID tag, together with the underlying modulation scheme (i.e., FM0-based or Miller-based), control the duration of the RN16 sequence in the air. Typically this duration varies from 0.02ms to 8ms. In our testbed, we configure the WISP tags with $BLF = 64kbps$ and Miller-4 modulation scheme and the corresponding duration is about 2ms. We manage to let one to four WISP tags respond to the SDR reader concurrently and collect the corresponding physical layer symbols for each response (denoted as a trace). In total, we collect more than 500 such traces for analysis purpose.

Figure 1.1 displays the received tag response signals in both time domain and the constellation map (i.e., I-Q plane) for 0 to 3 concurrent tag responses, which are typical instances from our collected traces. It is noted that in Figure 1.1 (a),(c),(e),(g), we

intercept first 600 symbols of about totally 8000 samples in each trace and plot their magnitudes in chronological order, for better visualization. In Figure 1.1(a), the background noise remains when no tag replies. When only one tag replies (as shown in Figure 1.1(c)), the signal amplitude changes according to the binary content. An amplitude threshold can be easily set – with much margin – to distinguish the two levels due to two backscatter states of each tag (i.e., reflecting or absorbing the CW signal from the reader). Figure 1.1(e) shows the 2-tag collision signal, which cannot be easily decoded through threshold-based methods on signal amplitude. When 3 tags collide (as shown in Figure 1.1(g)), it is nearly impossible to find out the signal states solely based on the amplitude of the time domain signal.

The difficulty in observing signal states from time domain drives us to look into the signals from another perspective – I-Q plane. We plot the constellation map of Figure 1.1 (a),(c),(e),(g) in Figure 1.1 (b),(d),(f),(h), respectively. From the new set of subfigures, we surprisingly discover that symbols cluster into regular patterns, even for the case where three tags collide. Let us start with the case with no tag replies, as shown in Figure 1.1(b). We find symbols form a single cluster; this is because of the existence of noise, which typically follows an additive white Gaussian distribution [21, 22, 48]. When only one tag responds, 2 well-separated clusters are shown in Figure 1.1(d), each of which maps to either reflection/absorption or absorption/reflection state. We also observe that a few symbols fall in the line segment between two clusters, which are due to the short transition between state flipping. In Figure 1.1(f), 4 clusters are clearly counted for 2-tag collision; this result is expected: since each tag owns two backscatter states, two tags owns four as each state of each tag combines into a new synthesized state. Similarly, 8 clusters could be clearly told for 3-tag collision, as shown in Figure 1.1(h). To summarize, much richer information could be extracted from the constellation map view of the tag response signal (in our scenario, the extra information is, but not limited to, the number

of colliding tags in a slot), compared to the time domain view. Hence, we can safely infer, from both experimental and theoretical views, that ideally 2^k clusters exist if k tags collide in the same slot, regardless of the specific types of devices for reader and tags.

The overall shape of the constellation map could be affected by many practical factors. First, the number of colliding tags, k , dominates the largest number of observable clusters in the map. Second, received powers, positions, to name a few, of tags would also affect its shape. To accommodate these diverse factors, we conduct dedicated experiments in Section 2.5 to explore the impact of these factors, by varying tag positions and SNRs of tag response signals.

2.2 Slot State Detection Algorithm

According to the experimental observation results in Chapter 2.1, we can infer the number of colliding tags based on the number of symbol clusters observed in the constellation map. It is thus necessary to design an accurate and fast cluster counting algorithm for this purpose. Existing clustering algorithms cannot be directly applied to serve this purpose, for two reasons. First, many schemes (e.g., K-means [38]) demand an information input on the number of clusters, which is in turn the output for the cluster counting algorithm. Second, other schemes that do not need the cluster number as input are generally computation intensive. For example, the computation overhead of DBSCAN [19] is at the scale of $O(l^2)$, where l is the number of symbols in a trace. To pipeline the cluster counting algorithm with the process of collecting RN16 message from replying tags at the reader side, we require the run time of cluster counting algorithm below 2ms (i.e., the duration of RN16 sequence).

In this section, we propose our slot state detection algorithm (short for SSDA), which can accurately estimate the number of concurrent responding tags and achieve linear

computation overhead, i.e., $O(l)$. In the high-level description, we observe that physical layer symbols in any cluster match well with the 2D Gaussian distribution; this is caused by the AWGN noise [21, 22, 48]. As in the 2D Gaussian distribution, the probability at the centroid of this distribution is always higher than that at other positions in the 2D plane, we expect a peak frequency of occurrence of symbols at the center of one cluster. In other words, each local maximum in symbol occurrence frequency indicates each cluster. If we divide the 2D plane into small square grids, we can define the number of symbols occurred in each grid as its grid density. Thus, by finding out the number of grids whose densities are local maximum among their neighbourhood grids, we can estimate the number of clusters in this map. We detail our SSDA algorithm as follows:

Step 1: We detect the smallest rectangular region that covers all physical symbols. This rectangular region is divided into small square grids with equivalent size, which is empirically chosen as 0.01×0.01 to optimize SSDA accuracy performance (while a too small grid size makes it hard to differentiate signal from noise, a too big one may cause false positives in subsequent detection of peak grids). The grid density value of every grid is thus collected according to the definition above. It is noted that instead of being relative, the value of the grid size shares the same unit as the value of the real and imaginary parts of physical symbols.

Step 2: A grid density threshold is set to classify grids into two types: signal grids and noise grids, depending on whether the density of a grid is above or below this threshold. Figure 2.2(a) depicts the original constellation map of a two-tag collision case. Figure 2.2(b) shows the grid density map (or matrix) after Step 1 and Step 2 are done.

It is noted that a static grid density threshold cannot function well in practice, as the absolute grid density values can vary, due to major factors including the number of colliding tags and the number of symbols in a slot. Concerning this issue, we suggest a percentage threshold (PT) – the absolute density threshold is set to be $PT \times l$, where l is the number of symbols in a response slot.

2.2.a: Physical layer symbols.

2.2.b: Filtered grid density matrix.

Figure 2.2: An illustrative example of Slot State Detection Algorithm (SSDA).

Step 3: After filtering out noises in Step 2, we can find out the number of grids whose grid densities are local maximum compared to their nearby neighborhood grids. Ideally 2^k such peak grids should be observed. However in practice, we may observe less than 2^k peak grids and thus need to adjust the ultimate estimated k as $\lceil \log_2 C \rceil$.

There are two critical factors in the SSDA algorithm: PT (defined above) and l . First, a high PT value may miss the real local maximum grids while a low one may reserve the effect from noise grids. Second, l is largely affected by the length of a slot and the radio signal's sampling rate at the receiving port of the RFID reader. As multiple combination options of backscatter link frequency (BLF) and modulation scheme are available in C1G2 standard, the length of a slot can be reduced by using higher BLF and coding rates, in order to reduce the time and computation overhead of SSDA.

We perform trace-driven simulations to study the above two factors. We configure the sampling rate of the USRP reader as 4MS/s and the slot length as 2ms. The WISP tags are enabled for concurrent RN16 transmissions in each slot. To vary the second factor, namely l , we intercept a portion of physical symbols from the beginning of the slot as the input to our SSDA algorithm. We define an interception rate IR to represent the

2.3.a: Detection accuracy with different percentage thresholds (PT).

2.3.b: Detection accuracy with different interception rates (IR).

Figure 2.3: Detection accuracies of SSDA with different system parameters.

amount of intercepted portion. For example, if $IR = 20\%$, then we choose the first 20% of all symbols in this slot as the input for SSDA. After performing SSDA, we measure the detection accuracy of SSDA as the ratio between the number of correctly inferred slots and the total number of slots, since we have the ground truth values of the number of colliding tags.

In Figure 2.3(a), we plot the accuracy of SSDA over PT in the range of $[1\%, 4\%]$. In this case we also provide 5 options for the value of IR . From this subfigure, it is observed that when PT is small (i.e., $PT < 0.5\%$), the SSDA accuracy is low, since multiple noise grids are not well filtered. When PT is in the range of $[0.5\%, 1.5\%]$, SSDA achieves stably high accuracy. When PT is further increased to be larger than 2.0%, the accuracy reduces, as some local maximum grids with globally low grid densities are mistakenly considered as noise grids. From the above observations, we empirically choose PT as 1%.

In Figure 2.3(b), we plot the SSDA accuracy over IR in the range of $[0.1, 1]$. In this case we also provide 3 options for the value of PT . From this subfigure, we discover that SSDA owns high accuracy (i.e., above 95%) when $PT = 1\%$ and $IR = 100\%$. Actually,

the accuracy keeps high when $IR > 50\%$, which indicates that it is sufficient for SSDA to function well as long as half of the physical symbols in one slot are used as the input of SSDA. When $IR > 30\%$, SSDA can also have an accuracy of around 90%. To summarize, it is promising to cut down the transmission time of tags' responses and the processing delay of SSDA without sacrificing the accuracy performance.

2.3 Estimation Algorithm

For each slot state detected through the SSDA algorithm, we can design a cardinality estimator. Intuitively, we can infer the cardinality from the ratio of singleton slots (or double-collision slots, or triple-collision slots) over the total number of slots. After then, we combine results from these estimators so as to minimize the overall estimation error (or variance) and obtain an ultimate result.

2.3.1 Estimation protocol

In each slot, each of t tags generates a random integer r using a uniform hash function. We denote the index of the right-most zero in the binary representation of r as R . As in the previous schemes [51, 72], a tag will respond if $R \geq \theta$, where θ is a parameter specified by the reader. Therefore, the probability p that a tag will respond in a slot is as follows

$$p = 2^{-\theta}. \quad (\text{Eq. 2.2})$$

Suppose X_k ($k = 0, 1, 2, \dots$) is defined as an indicator of k tag responses in a slot, i.e., $X_k = 1$, if k tags are in the slot; $X_k = 0$, otherwise.

For each k , X_k follows the Bernoulli distribution, and the probability of observing k responses in a slot is

$$Pr\{X_k = 1\} = \binom{t}{k} p^k (1-p)^{t-k} \approx \frac{\lambda^k}{k!} e^{-\lambda}, \quad (\text{Eq. 2.3})$$

where $\lambda = pt$ is the load factor.

Thus, the expectation $E[X_k]$ and variance $\sigma_{X_k}^2$ are

$$E[X_k] = \frac{\lambda^k}{k!} e^{-\lambda}, \quad \sigma_{X_k}^2 = \frac{\lambda^k}{k!} e^{-\lambda} \left(1 - \frac{\lambda^k}{k!} e^{-\lambda}\right). \quad (\text{Eq. 2.4})$$

We define $\bar{X}_k = \frac{1}{m} \sum_{i=1}^m X_{k,i}$ as the arithmetic average of m observations. Then, the expectation and variance of $\bar{X}_k$, denoted as $E[\bar{X}_k]$ and $\sigma_{\bar{X}_k}^2$, are as follows

$$E[\bar{X}_k] = E[X_k] = \frac{\lambda^k}{k!} e^{-\lambda}, \quad \sigma_{\bar{X}_k}^2 = \frac{1}{m} \sigma_{X_k}^2 \quad (\text{Eq. 2.5})$$

Since different k values will produce different estimations of t with different variances, we give the following theorem which provides the optimal combination of multiple sub-estimators.

Theorem 1: Suppose $\hat{t}_0, \hat{t}_1, \dots, \hat{t}_k$ are $k+1$ estimations for t with variances $\sigma_0^2, \sigma_1^2, \dots, \sigma_k^2$, respectively. For the weighting scheme $\sum_{i=0}^k w_i = 1$ and $0 \leq w_i \leq 1$, the joint estimator $\hat{t} = \sum_{i=0}^k w_i \hat{t}_i$ has a variance of $\sigma_{\hat{t}}^2 = \sum_{i=0}^k w_i^2 \sigma_i^2$. The optimal weights w_i ($i = 0, 1, \dots, k$) for each sub-estimator that minimizes $\sigma_{\hat{t}}^2$ is

$$w_i^* = \frac{1/\sigma_i^2}{\sum_{j=0}^k 1/\sigma_j^2}, \quad i = 0, 1, \dots, k, \quad (\text{Eq. 2.6})$$

and the minimum variance is

$$\sigma_{\hat{t}, \min}^2 = \frac{1}{\sum_{i=0}^k 1/\sigma_i^2}. \quad (\text{Eq. 2.7})$$

Proof: To minimize $\sigma_{\hat{t}}^2$, we define the following Lagrange multiplier

$$L(w_0, w_1, \dots, w_k, \beta) = \sum_{i=0}^k w_i^2 \sigma_i^2 + \beta \left(\sum_{i=0}^k w_i - 1 \right),$$

where the term $\sum_{i=0}^k w_i - 1$ incorporates the weight constraint $\sum_{i=0}^k w_i = 1$.

We let the partial derivatives of L over w_i ($i = 0, 1, \dots, k$) and β be 0. Thus, we have

$$\begin{cases} \frac{\partial L}{\partial w_i} = 2w_i \sigma_i^2 + \beta = 0, & i = 0, 1, \dots, k \\ \frac{\partial L}{\partial \beta} = \sum_{i=0}^k w_i - 1 = 0. \end{cases}$$

We solve the equations as follows

$$\begin{cases} w_i^* = \frac{1/\sigma_i^2}{\sum_{j=0}^k 1/\sigma_j^2}, & i = 0, 1, \dots, k. \\ \beta^* = -\frac{2}{\sum_{i=0}^k 1/\sigma_i^2}. \end{cases}$$

Thus, we have the minimum variance of $\hat{t}$ as follows

$$\sigma_{\hat{t},min}^2 = \sum_{i=0}^k w_i^{*2} \sigma_i^2 = \frac{1}{\sum_{i=0}^k 1/\sigma_i^2}.$$

2.3.2 Computing the number of rounds m

In practice, m estimation rounds have to be repeated to further reduce $\sigma_{\hat{t},min}^2$ and meet the requirement in Eq. 2.1. In the following, we analyze the minimum value of m .

Let $Y = \frac{\hat{t} - t}{\sigma_{\hat{t},min}}$. Since t is often a large number, according to the law of large number, Y follows the standard normal distribution. Thus, we can derive

$$Pr\left\{-\frac{\epsilon t}{\sigma_{\hat{t},min}} \leq Y \leq \frac{\epsilon t}{\sigma_{\hat{t},min}}\right\} \geq 1 - \delta. \quad (\text{Eq. 2.8})$$

Eq. 2.8 is equivalent to

$$\frac{\epsilon t}{\sigma_{\hat{t},min}} \geq c, \quad (\text{Eq. 2.9})$$

where c meets the following condition

$$1 - \delta = \text{erf}\left(\frac{c}{\sqrt{2}}\right), \quad (\text{Eq. 2.10})$$

and $\text{erf}()$ represents the Gaussian error function.

Since $\sigma_{\hat{t},min}^2$ is a function of m , we first compute $\sigma_{\hat{t},min}^2$. According to Eq. 2.7, $\sigma_{\hat{t},min}^2$ is a function of σ_k^2 . Because $\hat{t}_k$ is a function of $\bar{X}_k$, the relationship between σ_k^2 and $\sigma_{\hat{X}_k}^2$ can be explicitly expressed.

Suppose $\hat{t}_k = f_k(\bar{X}_k)$ is expressed with the Taylor expansion centered on $E[\bar{X}_k]$:

$$f_k(\bar{X}_k) - f_k(E[\bar{X}_k]) \approx f'_k(E[\bar{X}_k])(\bar{X}_k - E[\bar{X}_k]). \quad (\text{Eq. 2.11})$$

Since we have $f_k(E[\bar{X}_k]) = E[f_k(\bar{X}_k)]$, we derive from Eq. 2.11 that

$$\text{Var}[f_k(\bar{X}_k)] \approx \{f'_k(E[\bar{X}_k])\}^2 \text{Var}[\bar{X}_k]. \quad (\text{Eq. 2.12})$$

We represent Eq. 2.12 as follows

$$\sigma_k^2 = \alpha_k^2 \sigma_{\bar{X}_k}^2, \quad (\text{Eq. 2.13})$$

where

$$\alpha_k = \left. \frac{dt_k}{d\bar{X}_k} \right|_{E[\bar{X}_k]} = \left. \frac{1}{d\bar{X}_k/dt_k} \right|_{E[\bar{X}_k]} = \frac{k!te^\lambda}{\lambda^k(k-\lambda)}. \quad (\text{Eq. 2.14})$$

Combining Eq. 2.4, Eq. 2.5, Eq. 2.7, Eq. 2.13, and Eq. 2.14, we obtain the expression of $\sigma_{t,min}^2$ as follows:

$$\sigma_{t,min}^2 = \frac{t^2}{mg(\lambda)}, \quad (\text{Eq. 2.15})$$

where $g(\lambda)$ is

$$g(\lambda) = \sum_{i=0}^k \frac{(i-\lambda)^2}{i!e^\lambda/\lambda^i - 1} \quad (\text{Eq. 2.16})$$

After obtaining $\sigma_{t,min}^2$, we can derive the number of independent measurements m to meet the accuracy requirement by substituting Eq. 2.15 into Eq. 2.9:

$$m \geq \frac{c^2}{\epsilon^2 g(\lambda)}. \quad (\text{Eq. 2.17})$$

It is noted that k is the number of colliding tags up to which SSDA can detect. We find that $g(\lambda)$ is a function of k . We compute the theoretical m value that can satisfy the accuracy requirement (i.e., ϵ, δ) when k varies. We set ϵ to 0.01 and provide three options for δ : 1%, 10%, and 20%. Figure 2.4 plots the result. In this figure, when k increases, m , the number of estimation rounds, decreases in despite of the accuracy requirement.

Figure 2.4: m against k .

Figure 2.5: $g(\lambda)$ against λ .

This finding demonstrates that the joint estimator in the PLACE algorithm could be enhanced by multiple independent subestimators. Another observation from this figure is that the reduction in m by inferring a large k becomes marginal. To be practically optimal, we combine four subestimators in PLACE.

We further tune λ to maximize $g(\lambda)$ and minimize m . In order to find the optimal λ^* to maximize $g(\lambda)$ and minimize m , we plot $g(\lambda)$ against λ in Figure 2.5. We see that $g(\lambda)$ reaches the maximum value with $\lambda^* \approx 5.2$.

2.3.3 Two-phase Counting Algorithm

We adopt the two-phase estimation design [16]. In the first rough estimation phase, we adjust the threshold θ^* so that the load factor $\hat{\lambda}$ approaches 5.2; in the second phase, we repeat m independent estimation rounds with the optimal threshold θ^* . The weights that are necessary to compute the final estimation $\hat{t}$ are derived from $\hat{\lambda}$ obtained in the first phase.

In the first rough estimation phase, the reader issues a θ value and measures the fraction of each slot state, i.e., $\bar{X}_k$ ($k = 0, 1, 2, 3$). We denote the fraction of slots with more than 3 concurrent responses as $\bar{X}_{4+}$. From Eq. 2.5, we have

$$E[\bar{X}_{4+}] = 1 - \sum_{k=0}^3 E[\bar{X}_k] = 1 - e^{-\lambda} \sum_{k=0}^3 \frac{\lambda^k}{k!}. \quad (\text{Eq. 2.18})$$

Figure 2.6 plots $E[\bar{X}_k]$ ($k = 0, 1, 2, 3, 4+$) with different λ . From this figure we observe that when λ is around 5.2, $E[\bar{X}_k]$ ($k = 0, 1, 2, 3$) can be very small and hence cannot be accurately measured with a small number of slots (e.g., 32 slots). In contrast, $E[\bar{X}_{4+}]$ spans a relatively large range and can be a good indicator of λ . In addition, $E[\bar{X}_{4+}]$ monotonically increases with λ , which allows us to quickly converge to the optimal λ using the binary search method.

In particular, in each query round, we measure $E[\bar{X}_{4+}]$ and compute $\hat{\lambda}$ according to Eq. 2.18. If $\hat{\lambda}$ is smaller than 3, indicating a large θ value, we decrease θ in the next query round; if $\hat{\lambda}$ is larger than 7, indicating a small θ value, we increase θ in the next query round; once $\hat{\lambda}$ is in the range $[3, 7]$, we terminate the rough estimation phase and set θ^* and $\hat{\lambda}$ to the corresponding values in the last query round. We set the optimal range for $\hat{\lambda}$ as $[3, 7]$ and we can always find the θ^* to let $\hat{\lambda}$ fall into this range. Suppose in a certain query round we compute that $\hat{\lambda} > 7$ or $\hat{\lambda} < 3$, in the following round we can increase or decrease θ accordingly until $\hat{\lambda}$ falls into the range $[3, 7]$.

 Figure 2.6: The expectation of $\bar{X}_k$ over load factor λ .

Based on the above rules, the reader adopts a binary search method for θ in the range $[0, 32]$ and starts a query round with $\theta = 16$. Since in practice $2^0 < t < 2^{32}$ almost always holds and $\lambda = t/2^\theta$, we can always find the optimal θ^* in $[0, 32]$ to let λ approaches λ^* , i.e., 5.2.

Algorithm 1 shows the pseudocode of PLACE algorithm for the reader without considering SSDA detection error. This algorithm includes two phases: the rough estimation phase (Line 1-24) and the second phase which performs tag set cardinality within high accuracy (Line 25-38). The goal of the first phase is to tune the optimal θ so that the corresponding λ , with a form of $t/2^\theta$, approaches the optimal λ^* (i.e., 5.2) at the reader side. We adopt a binary search method. In the beginning, the reader sets the initial value of θ to 16 (Line 1), which is the middle of the range $[0, 32]$. Next the reader broadcasts a random seed s and θ to all tags, waits for tag replies in the coming 32 slots, and collect the values of $\bar{X}_k$ ($k = 0, 1, 2, 3$) (Line 2-13). Based on collected $\bar{X}_k$, $\bar{X}_{4+}$ can be deduced (Line 14), which in turn can be used to compute a $\hat{\lambda}$ according to Eq. 2.18 (Line 15). We check whether the obtained $\hat{\lambda}$ falls in the range $[3, 7]$, where λ^* lies. If $\hat{\lambda}$ is not in the range, the reader adjusts θ accordingly and restarts another query round with 32 slots to obtain another $\hat{\lambda}$ (Line 16-19). Otherwise the readers stops the first phase and starts

Algorithm 1 PLACE algorithm for RFID reader

```

1:  $\theta \leftarrow 16$ 
2: while TRUE do
3:    $\bar{X}_0 \leftarrow 0, \bar{X}_1 \leftarrow 0, \bar{X}_2 \leftarrow 0, \bar{X}_3 \leftarrow 0$ 
4:    $i \leftarrow 1$ 
5:   while  $i \leq 32$  do
6:     Generate a random seed  $s$ , broadcast a query command containing  $(\theta, s)$ , and wait for tag responses
7:     if  $k$  tags ( $k = 0, 1, 2, 3$ ) respond in the time slot then
8:        $\bar{X}_k \leftarrow \bar{X}_k + 1/32$ 
9:     else
10:      Do nothing
11:    end if
12:     $i \leftarrow i + 1$ 
13:  end while
14:   $\bar{X}_{4+} \leftarrow 1 - \bar{X}_0 - \bar{X}_1 - \bar{X}_2 - \bar{X}_3$ 
15:  Solve for Eq. 2.18 with  $\bar{X}_{4+}$  and obtain  $\hat{\lambda}$ 
16:  if  $\hat{\lambda} < 3$  then
17:     $\theta \leftarrow \theta/2$ 
18:  else if  $\hat{\lambda} > 7$  then
19:     $\theta \leftarrow (\theta + 32)/2$ 
20:  else
21:    break
22:  end if
23: end while
24: Set  $m \leftarrow c^2/(\epsilon^2 g(\hat{\lambda}))$  based on Eq. 2.16 and Eq. 2.17
25:  $\bar{X}_0 \leftarrow 0, \bar{X}_1 \leftarrow 0, \bar{X}_2 \leftarrow 0, \bar{X}_3 \leftarrow 0$ 
26:  $i \leftarrow 1$ 
27: while  $i \leq m$  do
28:   Generate a random seed  $s$ , broadcast a query command containing the updated  $\theta$  and  $s$ , and wait for tag responses
29:   if  $k$  ( $k = 0, 1, 2, 3$ ) tags respond in the time slot then
30:      $\bar{X}_k \leftarrow \bar{X}_k + 1/m$ 
31:   else
32:     Do nothing
33:   end if
34:    $i \leftarrow i + 1$ 
35: end while
36: For each  $\bar{X}_k$  ( $k = 0, 1, 2, 3$ ), compute corresponding  $\hat{t}_k$  based on Eq. 2.5 and  $\lambda = t/2^\theta$ 
37: Compute the weights  $w_k$  ( $k = 0, 1, 2, 3$ ) based on Eq. 2.5, Eq. 2.13 and Eq. 2.14
38: Estimate  $t$  as  $\hat{t} \leftarrow \sum_{k=0}^3 w_k \hat{t}_k$ 

```

the second phase (Line 20-22).

At the start of the second phase, the needed number of slots m is computed based on accuracy requirements and $\hat{\lambda}$ from the first phase (Line 24). Then the reader issues a query round with the updated θ and a random seed s , waits for tag replies in the following

Algorithm 2 PLACE algorithm for RFID tag

```

1: while TRUE do
2:   Receive reader query command with parameter  $\theta$  and random seed  $s$ 
3:   Generate a random 32-bit sequence  $r$  by hashing  $s$  with a uniform hash function
4:   Compute  $R$ , the index of the right-most zero in  $r$ 
5:   if  $R \geq \theta$  then
6:     Reply to the reader with RN-16 sequence in the coming time slot
7:   else
8:     Do not reply
9:   end if
10: end while

```

m slots, and collects $\bar{X}_k$ values (Line 25-35). For each $\bar{X}_k$, a $\hat{t}_k$ is computed based on Eq. 2.5 (Line 36). The weights w_k ($k = 0, 1, 2, 3$) are computed based on Eq. 2.5, Eq. 2.13 and Eq. 2.14 (Line 37). Finally $\hat{t}$ is obtained through a weighted sum of $\hat{t}_k$ (Line 38).

Algorithm 2 shows the pseudocode for tags. Each time when it receives a reader command with θ and s (Line 2), it creates a random binary sequence with 32 bits r by hashing s with its own uniform hash function (Line 3). Next it computes the index of the right-most zero in r , which is denoted as R (Line 4). If $R \geq \theta$, the tag replies in this slot with a RN-16 sequence. Otherwise it does not reply (Line 5-9).

2.3.4 Discussion

It is possible to utilize physical layer information (specifically, the slot state information) in a different way. We provide a frame-based way as an example. The RFID reader could issue a frame containing f lots. Each RFID tag responds in a random slot in the frame with a probability of p . From the tag response in the whole frame, we can count N_k , the number of slots each of which contains k ($k = 0, 1, 2, 3$) tag replies. Finally, we could also estimate t from multiple N_k values by combining estimations from these N_k s. We adopt a straightforward combination method – $\hat{t} = \sum_{k=0}^3 kN_k/p$. Further optimizations could be achieved by optimally tuning f and p .

This method is quite different from that in PLACE, where each tag responds to each slot with a probability. It seems that this method cuts down the communication overhead

since it does not need tags to reply in each slot as done in PLACE. However, the above frame-based estimation method can hardly produce unbiased results if f is too small; only when f is in the scale of $O(t)$ such estimation could be unbiased.

We perform simulations to compare the accuracies of both above mentioned frame-based method and PLACE. In the frame-based method, we set f to 9000 and vary t in the range of $[10^3, 10^6]$. The estimation results are averaged over 100 independent runs for each t value. Similar configurations are used in PLACE, except that the total number of interaction slots is 9000, rather than the concept of frame size. We plot the comparison results in Figure 2.7(a). In this subfigure, the theoretical estimation curve (i.e., $y = x$) is also plotted. From this subfigure we find that if $t > 10^4 > f$, the frame-based scheme incurs estimation bias, while PLACE owns a nearly ideal curve under the same condition. This subfigure proves the scalability of PLACE over large tag sets. We compare the performance of two from another view in Figure 2.7(b). We set t to a very large value, 50000, and f to 1500. We wonder what the accuracy of two schemes would be, given the same f slot. We run each scheme for 100 times and plot the CDF curves in Figure 2.7(b). From this subfigure, we can easily discover that most of estimations by PLACE are quite near the ground truth 50000, while the median value by the frame-based method is 4500, which is greatly deviated from the ground truth value. This comparison demonstrates that when f is much smaller than t , the frame-based scheme cannot function at all.

2.4 Impact of SSDA Errors and Enhancement

In this section, we analyze how the SSDA detection errors influence the estimation accuracy of PLACE and study the impact of SNR on SSDA detection error.

2.4.1 Enhanced PLACE

We denote q_{ij} as the probability of detecting state i as state j , where $i, j = 1, 2, 3, 4+$. Specifically, if $i = j$, q_{ij} indicates the detection accuracy of state i . As empty slots (state

 2.7.a: Estimation bias with varying t .

 2.7.b: CDF of estimation results when $t = 50000$.

Figure 2.7: Performance of the toy estimation scheme.

0) can be accurately differentiated from busy slots by measuring the signal strength, we only consider the detection accuracy of state i , where $i, j = 1, 2, 3, 4+$. We use a detection rate matrix $Q = [q_{ij}]_{4 \times 4}$ to represent the overall detection performance of SSDA.

We use a vector $\vec{X} = (\bar{X}_1, \bar{X}_2, \bar{X}_3, \bar{X}_{4+})^T$ to represent the actual fraction of each state. As the detection results of SSDA may contain some errors, we represent the measurement results as $\vec{X}^E = (\bar{X}_1^E, \bar{X}_2^E, \bar{X}_3^E, \bar{X}_{4+}^E)^T$. Based on the definition of Q , we have $\vec{X}^E = Q\vec{X}$. Thus, we can obtain $\vec{X}$, which can be used to generate an accurate estimation of t , as follows:

$$\vec{X} = Q^{-1}\vec{X}^E, \quad (\text{Eq. 2.19})$$

where Q^{-1} is the inverse matrix of Q .

To estimate Q , we perform SSDA with our traces collected from the software defined testbed, which is described in Section 2.1. We set the percentage threshold to be 1% and the interception rate to be 30%. Figure 2.8 plots the state detection accuracy of SSDA. The x-axis of Figure 2.8 is the ground truth of each tag response state, and the y-axis represents the detection results. We represent the measurement results with Q as

Figure 2.8: Detailed accuracy performance of SSDA.

follows:

$$Q = \begin{pmatrix} 0.96 & 0.04 & 0 & 0 \\ 0.08 & 0.84 & 0.09 & 0 \\ 0 & 0.02 & 0.96 & 0.02 \\ 0 & 0 & 0.03 & 0.97 \end{pmatrix}.$$

From the above Q , we find the SSDA method achieves high detection accuracies. For the detection errors, we find that state k is more likely to be mistakenly detected as adjacent states. In practice, Q can vary due to various factors, e.g., reader transmission power, interference to tag responses, etc. To understand the impact of Q on the overall estimation accuracy of PLACE, we approximate Q with Q_0 as follows:

$$Q_0 = \begin{pmatrix} 1 - q_0 & q_0 & 0 & 0 \\ q_0 & 1 - 2q_0 & q_0 & 0 \\ 0 & q_0 & 1 - 2q_0 & q_0 \\ 0 & 0 & q_0 & 1 - q_0 \end{pmatrix},$$

where q_0 can be specified according to empirical measurement results. With Q_0 , we can study how different detection performance of SSDA may impact the overall counting accuracy of PLACE. We can recover $\vec{X}$ from $\vec{X}^E$ according to Eq. 2.19 and use $\vec{X}$ for tag cardinality estimation. We name the enhanced PLACE with the error compensation as EPLACE.

2.4.2 Impact of SNR on q_0

The SNR of the mixed response signal from colliding tags, in practice, has great impacts on q_0 . Here we explain the received signal's SNR from the view of constellation map and explore its effects on q_0 .

2.4.2.1 Definition of SNR of received signal

Physical symbols form multiple clusters in the constellation map and each cluster indicates a signal state plus the impact from noise. We can then naturally define a radius and centroid for each cluster. For instance, the radius could be the average Euclidean distance between individual symbols and the centroid; the centroid could be the mean of the coordination of symbols that belong to the cluster.

Normally, the power of response signal can be computed as the mean amplitude of all symbols. However, a shift in signal states exists, which may be caused by factors like thermal noise of USRP device or channel degradation or multipath. As such, it is necessary to compensate for the signal state shift in the constellation map. If the centroid of a certain cluster is nearest to the origin point compared to that of all other clusters, the corresponding cluster is denoted as the reference cluster. Thus, we compensate for the signal state shift by regarding the centroid of the reference cluster as the new origin point. The following expression characterizes the power of the response signal:

$$P_s = \frac{1}{l} \sum_{i=1}^l \text{dist}(z_i, z_0), \quad (\text{Eq. 2.20})$$

where z_0 is the centroid of the reference cluster, z_i ($i = 1, 2, \dots, l$) represents each data sample of the tag response signal.

The cluster radius indicates the amplitude of noise. To obtain the average noise amplitude or power in a multi-cluster constellation map, we may average over every cluster radius. However, it is too computational intensive. In the following we introduce

2.9.a: Constellation map of the signal

2.9.b: Constellation map of the noise

Figure 2.9: Illustration on how to utilize the noise before tag response signal to obtain the noise power and reference point.

a more efficient method. Generally speaking, we can compute the radius of the reference cluster and directly adopt this radius value as the noise amplitude.

From the above discussions we find SNR is highly dependent on the reference cluster. We use an instance in Figure 2.9 to explain how to efficiently get the information of the reference cluster. Figure 2.9(a) plots the constellation map of the mixed signal of 2 tags. Figure 2.9(b) shows the map of noise which arrives after the reader query and before the actual signal. The trace used for plotting Figure 2.9 is from our testbed (detailed in Section 2.1.2).

We discover that the solely cluster in Figure 2.9(b) occupies similar area as the one at the lower left corner in Figure 2.9(a), which has the shortest distance to the origin compared to the rest of the clusters, i.e., the reference cluster by definition.

The information on the reference cluster (i.e., centroid and radius) can be simply computed from the symbols in the interval between the reader query and the actual response signal. After the reader collects all symbols of the tag responses, the corresponding SNR value could be easily computed as well. On the other hand, it incurs much more significant computational overhead if DBSCAN is performed to compute SNR.

2.4.2.2 Impact of SNR on q_0

It is straightforward that a high SNR value causes long distances between clusters in the map and thus high accuracy of the SSDA algorithm. Since q_0 decreases if the accuracy of SSDA increases, a high SNR results in low q_0 .

Practically, a look-up table between SNR and q_0 could be built through offline training for real-time cardinality operation. An indirect way might be to build the mapping between SNR and SSDA accuracy, as q_0 can be derived from the accuracy (in the following evaluation, we choose the indirect mapping method). Specifically, according to the definition of q_0 in the matrix Q_0 , the SSDA accuracy could be expressed as either $1 - q_0$ or $1 - 2q_0$. It is noted that the number of colliding tags, k , should be known to measure q_0 . We may fix k to 2 in the offline training procedure and the corresponding accuracy is $1 - 2q_0$.

2.5 Evaluation

In this subchapter, SSDA is first compared with a classic clustering algorithm named DBSCAN [19] in the aspects of accuracy and operation time. Next, PLACE is compared against several existing schemes listed as follows: LoF [51], EFNEB [25], SRC [16] and ZOE [72]. Finally, the robustness of SSDA and PLACE with respect to SNR of colliding signals is well evaluated.

2.5.1 SSDA Evaluation

We collect traces using the USRP/WISP testbed as shown in Section 2.1. We configure the WISP tags and make several WISP tags respond concurrently to the USRP reader. The ground truth number of responding tags (from one to four) is known as a priori. We perform both SSDA and DBSCAN on these traces and compare their accuracies and operation times. In SSDA, we configure IR to be 30% and PT to be 1%.

Figure 2.10: Performance comparison of DBSCAN and SSDA.

Some concepts in DBSCAN need to be explained first. A circular region with radius ε and center point p is denoted as p 's ε -neighborhood. If at least T points fall into ε -neighborhood of p , p is a core point; if another point q is in the ε -neighborhood of a core point, q is a border point. If a point is not a core point or a border point, it is a noise point. If p is a core point and q is in its ε -neighborhood, q is *directly density-reachable* from p . If a third point q' is directly density-reachable from q , then q' is *indirectly density-reachable* from p via q , i.e., $p \rightarrow q \rightarrow q'$. The principle of DBSCAN is to put all density-reachable points in the same cluster. To optimize the performance of DBSCAN in our experiment, we optimally set ε to 0.01 and T to $0.01 \times l$. It is noted that the operational overhead of DBSCAN is at the scale of $O(l^2)$; l is the number of symbols in one slot.

Figure 2.10(a) compares the detection accuracy of SSDA and DBSCAN. We present the overall detection accuracy as well as the accuracy for each case with different number of responding tags. Figure 2.10(a) shows the following results. First, the overall accuracy of SSDA is comparable with that of DBSCAN. Specifically, the overall accuracies of SSDA and DBSCAN are 91.2% and 96.7%, respectively. Second, in the case when 4 tags respond

together, SSDA achieves higher accuracy compared with DBSCAN. This is because when 4 tags respond concurrently, the I-Q plane becomes crowded with 16 clusters. As a result, the inter-cluster distances become smaller and the borders between neighboring clusters become blurred. Thus, the border-based DBSCAN may cluster the neighboring clusters together. In contrast, the centroid-based SSDA overcomes this problem and derives the number of clusters by counting the number of local maximums after filtering out noise. Since the local maximums lie in the center of clusters, the distance between the centers of two neighboring clusters tend to be larger than the distance between their borders.

Figure 2.10(b) compares the computational overhead of SSDA and DBSCAN. The physical layer symbols are collected with the USRP reader and the symbols are transferred to a laptop for processing. We execute both algorithms on the laptop and measure the execution time of two algorithms. The laptop is equipped with an Intel quad-core 2.9GHz i7 processor and 15.4GB memory running 64-bit Ubuntu 13.04. In the figure, the x-axis is the trace index and the y-axis is the operation time in seconds, presented in the log scale. We find that SSDA reduces the operation time compared with DBSCAN by orders of magnitude. Specifically, the average operation time of SSDA and DBSCAN is 1.3ms and 84.5s, respectively. SSDA substantially outperforms DBSCAN mainly due to the fact that while DBSCAN incurs $O(l^2)$ computational overhead, SSDA only incurs $O(l)$ overhead. In addition, while DBSCAN has to perform computation-intensive operations such as multiplication and square root calculation to calculate the distance between physical layer symbols, SSDA only needs to perform lightweight operations such as addition and comparison.

2.5.2 PLACE Evaluation

We perform extensive simulations to compare PLACE with previous cardinality estimation schemes. As most of these previous schemes do not tolerate noisy channels, we assume no errors in slot state detection in the performance comparison.

Figure 2.11: Comparison of operation time to meet different accuracy requirements among 5 schemes: EFNEB, LoF, ZOE, SRC and PLACE.

We measure the overall execution time as the performance metric, which counts both communication time and the computation time. The communication time mainly consists of the transmission time of reader's command and tags responses. The computation time is mainly consumed in the execution of SSDA for each slot. We ignore the computation time for benchmark schemes. In practice, as SSDA can be executed in real time, the cluster counting operation (which takes 1.3ms) can be executed in parallel with the signal sampling operation for each RN16 reception (which takes 2ms) at physical layer. Thus, SSDA incurs little extra time overhead.

Figure 2.11 compares the overall operation time to meet different estimation accuracy requirements. The actual tag cardinality is 50000. In Figure 2.11(a), we fix δ to 20% and vary ϵ , ranging from 1% to 5%. From Figure 2.11(a), we find that like benchmark schemes, PLACE takes less time to meet the estimation accuracy requirement of relaxed confidence intervals. Results in Figure 2.11(a) demonstrate that when $\epsilon = 1\%$, PLACE improves the operation time performance over LoF, EFNEB, ZOE and SRC by $18.71\times$, $17.42\times$, $3.78\times$ and $3.19\times$ on average. In Figure 2.11(b), when we fix ϵ to 1% and vary δ

Figure 2.12: CDF of estimation results with the same amount of execution time.

from 1% to 10%, we also find that PLACE substantially outperforms benchmark schemes. In particular, Figure 2.11(b) shows that when $\delta = 1\%$, PLACE improves the operation time over LoF, EFNEB, ZOE and SRC by $23.75\times$, $21.91\times$, $4.62\times$ and $4.02\times$ on average.

We provide each estimation scheme the same amount of execution time to estimate the number of 50000 tags. We repeat the estimation process of each scheme for 100 times. In Figure 2.12, we plot the CDF of estimation results for each scheme. From Figure 2.12, we find that the estimation results of PLACE are more concentrated on the actual tag cardinality. Moreover, the tail of PLACE is much shorter than those of EFNEB, LoF, ZOE and SRC, indicating smaller estimation variance of PLACE. Specifically, according to the estimation results, provided the same amount of operation time, PLACE has 99 estimation results within the confidence interval $[47500, 52500]$, while SRC, which performs best among the benchmarks, has only 91 estimation results within the interval. According to the experiment result, we find that given the same amount of operation time, PLACE can estimate the tag cardinality more precisely and accurately compared with other schemes.

2.5.3 The Impact of SSDA Detection Errors on PLACE

To evaluate the impact of SSDA detection errors on the estimation accuracy of PLACE, we measure the estimation accuracy with the ratio of the estimated tag population $\hat{t}$ over the actual population t as in [51, 72, 73]. Ideally, the accuracy should be 1, indicating a perfect estimation result.

We run the basic PLACE and the Enhanced PLACE (EPLACE), which compensates for the errors and adjusts the estimation results. We use q_0 to represent the slot state detection error. We average over 100 runs to obtain each estimation result.

Figure 2.13(a)-(c) plot the estimation accuracies of PLACE and EPLACE, with q_0 of 5%, 15%, and 25%, respectively. The y-axis and x-axis represent the estimated number and the actual number of tags, respectively. For illustration purposes, we plot the ideal curve $y = x$. From Figure 2.13(a)-(c), we find that without the error compensation, the estimation errors of the basic PLACE increase with both the number of tags and the slot state detection errors. Fortunately, EPLACE is able to compensate for the errors and achieve high accuracy. The experiment results of Figure 2.13(a)-(c) demonstrate that EPLACE is able to leverage the knowledge about the detection errors and adjust the estimation results accordingly.

In Figure 2.13(d), we vary the error rate from 5% to 25% and measure the corresponding estimation accuracy. We fix the tag cardinality to 50000. We specify the $(\epsilon=5\%, \delta=1\%)$ -accuracy requirement, and provide PLACE the corresponding execution time. From Figure 2.13(d), we find that as q_0 increases, the estimation accuracy of basic PLACE decreases dramatically. In contrast, the estimation accuracy of EPLACE remains relatively stable and fluctuates around 1. Although EPLACE cannot achieve the ideal estimation accuracy of 1, the estimation results are all within the targeted accuracy interval of $[0.95, 1.05]$.

2.13.a: $q_0=0.05$

2.13.b: $q_0=0.15$

2.13.c: $q_0=0.25$

2.13.d: Accuracy with varied q_0 .

Figure 2.13: Impact of q_0 on estimation accuracy of PLACE.

Figure 2.14: Impact of SNR of tag response signal on SSDA detection accuracy.

2.5.4 The Impact of SNR on SSDA Detection Accuracy

In this subsection we run experiments to evaluate the impact of SNR on SSDA detection accuracy. For each of all traces, we first compute its SNR. Next, we divide the whole SNR range of all traces into small bins with equivalent length. For each bin, we compute a SSDA detection accuracy as the ratio of the number of correctly detected traces over the total number of traces falling into this bin. We plot the results in Figure 2.14.

In Figure 2.14, the x-axis represents the SNR range. Each SNR value indicates a SNR range, e.g., 13.18dB represents the range [13.18, 16.38]. The y-axis is the SSDA detection accuracy for each bin. To better show the impact of SNR on detection accuracy, we vary the interception rate (IR) of each trace from 0.05, 0.10 to 1. From Figure 2.14 we find that in general, the SSDA detection accuracy increases when SNR increases. This rule is especially obvious when the interception rate is low, while when IR is high, e.g., 1, the increase is not that significant. This is because a higher IR indicates more data samples are used as SSDA input. As a result, more redundancy is available for SSDA and hence it is more robust to low SNR.

2.6 Related work

Many prior efforts [16, 25, 32, 33, 51, 53, 72, 73] have been devoted to improve the efficiency of probabilistic cardinality estimation. Kodialam *et al.* discover that the fraction of empty, singleton and collision slots can be utilized and suggest the classic Unified Probabilistic Estimator (UPE) [32]. Qian *et al.* avoid replication counting and decrease the ALOHA frame size and present the Lottery Frame (LoF) based scheme [51]. Han *et al.* adopt a binary search method to find the index of first busy slot in the response sequence and propose the Enhanced First Non-Zero Based estimator (EFNEB) [25]. Zheng *et al.* organize tags in the form of a binary tree and suggest the Probabilistic Estimating Tree scheme (PET) [73]. Shahzad *et al.* explore the average run size of busy slots and present the Average Run based Tag estimator (ART) [53]. Zheng *et al.* further propose the Zero-One Estimator (ZOE) [72], which utilizes the relative ratio between empty and busy slots. Chen *et al.* propose the theory of two-phase design counting and analyze the upper bound of the efficiency that could be approached [16]. Some other branches also exist. Li *et al.* [35] consider the estimation issue from the view of energy for active RFID tags. Gong *et al.* [23] explore how to count counterfeit tags in an efficient way. Liu *et al.* [37] study how to count the number of key tags in a tag set. We differ from previous works in that we extract extra physical layer information on the number of concurrent responding tags and hence outperform all of these works.

Our work is also related to some collision recovery works, which aim to recover the content of colliding tags from physical layer hints [3, 31, 55]. Shen *et al.* [55] resolves collisions of HF RFID cards with software defined radios. Khasgiwale *et al.* [31] raise the efficiency of tag arbitration by decoding RN16 message from UHF RFID tags. Shahzad *et al.* [82] balance the number of active replying tags among all tags and determine the optimal frame size so that fairness constraint among tags is met when being identified. Although several works [3, 18] could decode limited number of concurrent tag responses,

they are solely deterministic identification schemes and unable to greatly improve the cardinality efficiency. Nevertheless, these works inspire our proposed PLACE scheme to adopt the cross-layer design principle for the cardinality estimation.

Collecting useful data from RFID tags is a fundamental topic. Yue *et al.* [64] adopt a Bloom filter for this purpose. BLINK [66] measures wireless link quality and adapt the transmission rate of tags. Buzz [60] recovers the content of a small number of collision tags in a rateless manner. Zanetti *et al.* [65] distinguish one tag from several other tags through its unique physical layer fingerprints. P-MTI [70] finds out some missing tags from a given set with a prior database. Tagoram [63] uses commercial readers to collect phase information for mobility tracking.

Our work also borrows techniques from the field of data mining, where a plenty of cluster counting algorithms are proposed. These methods can be divided into multiple types: genetic-based [39], validity-index-based [45] and neural network-based [2, 56] methods. However, they require the output of a clustering process while SSDA could run without a related clustering process. Some other schemes, like DBSCAN [19] or VAT-based schemes [4, 46, 61] can operate independently as well, but they incur large computational overhead, i.e., $O(l^2)$. On the other hand, SSDA requires only $O(l)$ overhead to achieve a similar accuracy level.

2.7 Summary

Cardinality estimation is fundamental to RFID systems. In this chapter, we design a physical layer information enhanced estimator which could greatly improve the estimation efficiency compared to prior MAC-only solutions. First, an accurate and efficient slot state detection algorithm is suggested to count the number of clusters in the constellation map and thus the number of colliding tags in a slot. Next, a synthesized estimator combines the estimation results from multiple independent sub-estimators and refines

the accuracy. Comprehensive trace-driven simulations are conducted to demonstrate the efficiency of PLACE over existing solutions. Although the detailed query procedure in PLACE is slightly different from the C1G2 standard, it is much more efficient. Meanwhile, we note that most of the current solutions [16, 25, 51, 53, 72] do not completely follow the standard as well.

Chapter 3

Physical Layer Tree-based RFID Identification

3.1 Background & Motivation

3.1.1 RFID backscatter

Passive tags are of small size and battery-free. They are generally used in UHF RFID systems that operate in the range from 860MHz to 960MHz. RFID reader imposes continuous wave (CW) onto RFID tags and tags can transmit data by either reflecting or absorbing CW. In other words, each tag has two transmission states, HIGH (bit 1) and LOW (bit 0). This operation is called RFID backscatter. RFID tags backscatter CW at frequencies with magnitude of several tens or hundreds kHz.

The power of the backscattered signal from a tag is quite low after it reaches the reader. Hence, the received signal is likely overwhelmed by noise, leading to decoding error at the reader side. To increase the robustness against channel degradation and maintain a high decoding rate on the reader side, certain encoding scheme is imposed to the data transmitted by a tag. EPC Class 1 Gen 2 (EPC-C1G2) standard [1] provides several encoding schemes for tags to choose, including FM0 and Miller coding, which are discussed in Section 3.2.1 in detail.

When k ($k > 1$) tags collide in the same time slot, the two states of each tag (i.e.,

Figure 3.1: Ideally, 2^k number of clusters appear in the constellation map when k tags collide. However, in practice clusters may overlap.

HIGH and LOW) combine linearly in the wireless channel. In principle, there are 2^k states in the mixed signal (in complex values). Among 2^k states, two are worth noting: all-0 and all-1 state. These two states occurs when all k tags transmit bit 0 or bit 1 at the same time. We denote the two states as resonant state, to distinguish them from the mixed state, which occurs when some tags transmit bit 0 and others transmit bit 1.

If we plot the received complex signal in a 2D plane, we obtain the constellation map consisting of many clusters. For each cluster, its centroid represents a combined state of k tags and its radius characterizes the power of additive noise from the wireless channel. Ideally we can observe 2^k clusters (as shown in Figure 3.1(a) and Figure 3.1(b)). However, when k becomes large, some clusters overlap and the observable number of clusters is thus reduced. As shown in Figure 3.1(c), we can see only 15 clusters for $k = 4$.

3.1.2 RFID identification

The task of RFID identification is to collect tag IDs from an unknown set through reader-tag interactions. Since tags backscatter reader's command without knowing if other tags also backscatter, collisions often happen and lead to failure in identifying any

IDs. ALOHA based schemes [50, 68] require tags to reply randomly in any time slot of a frame and hence suffer from collisions. On the other hand, in tree based schemes the reader issues a series of binary prefixes and only tags whose IDs match the issued prefix would reply. Consequently, collisions are gradually reduced during such reader-tag interactions, and finally tag IDs can be identified. In conclusion, tree-based schemes perform more stably than ALOHA-based ones. In the rest of this chapter, we focus on tree-based schemes.

We first introduce some basic concepts in tree-based framework with the binary tree in Figure 3.2(a). Each node represents a binary prefix in the tree and is classified into three types: collision, singleton and empty, depending on the number of responses if it is queried. We find 8 tags in the bottom layer of the tree (in square shape), whose IDs can be read by traversing from the root to itself. For instance, the ID of the 3rd tag from left-most is 0010.

Many tree-based schemes are proposed to reduce the number of needed queries. Query Tree (QT) [34] traverses the tree in depth-first mode. In Smart Trend Traversal (STT) [47], after the reader identifies a tag ID with a node, it queries all subsequent nodes in the same layer until an empty or collision node is encountered. Tree Hopping (TH) [54] estimates the population of the non identified set to compute the optimal series of queried nodes to minimize the total number of queries. One common feature in these works is that they simply append 1 bit to the queried prefix for subsequent queries to resolve a collision node.

3.1.3 Motivation

Despite a plenty of tree-based identification works [12, 40, 47, 52, 54], their underlying assumption is that tag ID distribution is uniform. However this assumption is often violated by practical scenarios and hence the performances of these schemes are seriously

3.2.a: Query Tree takes 20 queries to identify all 8 tag IDs.

 3.2.b: *PHY-Tree* takes only 6 queries to identify all 8 tag IDs.

 Figure 3.2: Illustration on how the proposed *PHY-Tree* scheme accelerate identification performance compared to classic Query Tree scheme.

degraded under non uniform ID distribution. For example, a toy factory attaches RFID tags with continuous IDs to the manufactured toys. A shop imports toys in batches from the factory for 3 times. Though IDs of each batch are continuous, IDs in the whole imported set are intermittent. We refer such ID distribution as block distribution. When the shop sells these toys on the shelf, customers randomly pick up some toys. After several weeks IDs of remaining toys render a random distribution.

In this chapter, we aim to handle the more challenging random ID distribution by looking into physical layer hints. Basically we extract two types of physical layer information from colliding tag response signal, namely horizontal and vertical information. These information are obtained in each query response without extra communication overhead and provide hints on the range where tag IDs may (or may not) locate in the

binary tree. As the hints are accumulated along the identification process, we could gradually refine the range and ultimately converge to exact position of each tag ID.

We briefly explain two types of information as follows. Horizontal information tells us whether all responding tags reply with the same bit 0 (all-0 state) or 1 (all-1 state), or a mix of 0 and 1 (mixed state), in each bit of their tag IDs. It is obtained based on some unique features in each tag's coding scheme as explained in Section 3.2. Vertical information is the estimation of the number of concurrent replying tags for a reader query. It is obtained based on a feature extracted from the constellation map of tag response signal, as shown in Section 3.3.

We use an illustrative example to explain how horizontal and vertical information could be utilized in tree-based identification. In Figure 3.2(b) each of 8 tag IDs is labelled with a square in the bottom layer. The query process in Figure 3.2(b) is illustrated as follows. We start querying with the two nodes in the first layer: "0" and "1". When "0" is queried, the reader obtains the vertical information that 4 tags collide (i.e., 0000, 0001, 0010, 0011). The reader thus appends 2 bits, rather than 1 bit, to "0" to resolve the collision optimally. As such, two nodes, 00 and 01 are skipped and marked as "V skip" (vertical skip). Besides, when "0" is queried, the reader also obtains the horizontal information that no tags reply with prefix 01 since the 4 responding tags have the same bit 0 at index 2. Consequently, the node 01 and its descendants are skipped for query and marked as "H skip" (horizontal skip). It is noted that the skipped node 01 is the result of both skip types and marked as "H+V skip". Similarly, when "1" is queried, the reader obtains both horizontal and vertical information, which respectively indicate that 4 tags collide and no tags reply with the prefix pattern "1*0" (where * represent a bit value at index 2). Thus, the reader further skips the nodes 10 and 11 and two subtrees rooted from node 100 and 110.

Next the reader proceeds with the unskipped nodes in the third layer (i.e., 000, 001, 101, 111). Upon node 000 is queried, the reader learns that only 2 tags collide and

Figure 3.3: FM0 and Miller-2 coding of a common binary sequence 001011.

a mixed state appears at bit index 4. For this combination of horizontal and vertical information, the only possibility is that 0000 and 0001 coexist. Thus the reader can definitely infer two tag IDs without explicitly query 0000 and 0001. Cases are similar when 001,101 and 111 are queried.

Figure 3.2(a) shows that QT takes 20 queries (marked as black solid circles or squares) to identify all 8 tag IDs. In contrast, our *PHY-Tree* scheme takes only 6 queries thanks to “H skip” and “V skip”. To conclude, our *PHY-Tree* scheme improves the performance over existing MAC layer solutions as *PHY-Tree* obtains extra physical layer hints on the range of tag IDs, which are accumulated through query responses.

3.2 Horizontal Information

3.2.1 Principle of Horizontal Information

In this subsection, we explain how to detect resonant states based on some unique features in the coding scheme exposed on each tag.

Figure 3.3 gives an example of FM0 coding of a binary sequence. We can observe a state flipping always occur at the bit boundary between two neighbouring bits. A state flipping always occurs at the boundary between two neighbouring bits. An extra state flipping happens in the middle of bit 0, while the state keeps constant for the whole bit 1. We can infer that if the current bit is 0, the latter half of the current and previous bits have the same state.

Figure 3.4: Colliding signal of two tags with different IDs using FM0 coding.

The above mentioned patterns can be extended to the colliding signal of multiple tags, as shown in Figure 3.4. Specifically, no state flipping occurs in the middle of a bit of all-1 state (e.g., at index 2,3,6). In other words, the first and second half of bit 1 owns the same state. For a bit of all-0 state (e.g., at index 5,8), the latter halves of this bit and its previous bit are the same. A mixed state consisting of bit 0 and 1 does not follow these features and can be distinguished from resonant states. It is noted that detecting the two resonant states does not require the information of the number of replying tags.

Next we examine whether resonant states can be observed in Miller coding as well. Figure 3.3 shows an example of Miller-2 coding waveform, from which we observe for bit 1, the second and third quarters of the signal have the same state. For bit 0, we observe that states of the first and third quarter of bit 0 are the same. These two patterns can be easily generalized for Miller- M coding with $M = 4, 8$. Consequently, we conclude that resonant states can be observed in Miller coding as well. In the rest of this chapter, we focus on FM0 coding.

By detecting resonant states from collision signal in an opportunistic way, the reader can avoid querying unnecessary prefixes. For example, in Figure 3.2(b) the reader queries with a prefix “0” and a collision happens. By detecting that at index 2 all tags reply with bit 0, it infers that prefix 01 and its descendant nodes can be skipped. It is noted that even if a resonant state is detected at an index which is not right after the prefix,

we still can use it to omit some queries. For example, in Figure 3.2(b) if prefix “1” is queried and an all-1 state is detected at index 3, the reader can skip the nodes matching the prefix pattern “1*0”; in this case, the nodes 100 and 110 and their descendant nodes can be skipped.

3.2.2 Robust Detection Algorithm

As detailed in Section 3.2.1, we need to compare whether two half-bit states are the same so as to detect both resonant states. To perform the comparison in the complex tag response signal, we can find the corresponding complex values for the two states and compare their real and imaginary parts. However, the backscatter signal of each tag experiences additive noise in the uplink channel to the reader, which renders the above comparison method invalid.

In this subsection, we design a robust resonant state detection algorithm against the noise. If we plot the noisy signal in the constellation map, we can observe many clusters, as illustrated in Section 3.1.1. Since each cluster represents a combination state from all responding tags, we can know whether states in two half-bit duration are the same by judging whether their corresponding clusters are overlapping with each other.

Figure 3.5(a) shows an example of overlapping clusters for 3 tag replies, where each labelled cluster corresponds to samples in a half bit duration. The signal of each tag is created as illustrated in Section 3.3.2. 3 tags transmit the following binary sequences: 0100001111, 0100110011 and 0101010101, which include all 8 bit states, i.e., from 000 to 111. At the bottom-right corner of Figure 3.5(a), the latter half of state 000 at index 3 (in yellow circle) shares a large overlapping region as the latter of state 111 at index 2 (in green solid square). In fact, we can distinguish between the first and second half of a bit from the tag response signal in the time domain. Moreover, both the frontier and latter half-bit clusters of state 111 at index 1 (in green solid square) and index 10 (in

3.5.a: Half-bit cluster.

3.5.b: Overall cluster.

Figure 3.5: Constellation map of 3 tags. Their 10-bit IDs include all 8 bit states, i.e., from 000 to 111.

black dot) are almost overlapping as well. In contrast, states other than 000 or 111 do not have similar overlapping patterns. Figure 3.5(b) shows the overall constellation map.

To quantify cluster overlapping, we first compute the centroid of two clusters as the average of corresponding samples. Two clusters are judged as overlapping if the distance between their centroids is below a threshold:

$$\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \leq c \times r_0, \quad (\text{Eq. 3.1})$$

where (x_1, y_1) , (x_2, y_2) are the coordinates of two centroids and r_0 is the cluster radius. c is tunable and empirically set to 2.5 in Section 3.6. We can approximate r_0 as the L2-norm of all samples in a cluster:

$$r_0 = \sqrt{\sum_{i=1}^q (x_i - \bar{x})^2 + (y_i - \bar{y})^2}. \quad (\text{Eq. 3.2})$$

In real-time operation, the reader can obtain r_0 by collecting background noise, without any transmitting tag, as only one cluster exists.

3.3 Vertical Information

The number of colliding tags k in a slot can help the reader to find out the optimal number of bits appended to the queried node to resolve a collision.

For example, if prefix “0” triggers 4 replies, the reader appends 2 bits to the current prefix to form 4 longer prefixes for subsequent queries (i.e., from 000 to 011).

Intuitively, the optimal number of appended bits is around $\log_2 k$, which is explained as follows. We can borrow a well-known conclusion from Frame ALOHA [68]: slot efficiency is the highest when the number of provided slots is the same as the number of tags. We can imagine k “slots” (actually k prefix queries) should be provided to resolve a k -tag collision since unknown tag IDs can be regarded as random replies in each of k “slots” (queries). Hence, $\log_2 k$ bits is required should be appended to provide k “slots”(queries).

Many cardinality estimation techniques [16, 51, 53, 72] can estimate k with high accuracy but they consume extra reader queries, which are unacceptable as such estimation is performed for each query in the whole identification process. Other works [3, 28, 31] estimate k by counting the number of visible clusters in the constellation map of the tag response signal. For example, [28] can count k up to 3 accurately. Although these works are lightweight, they are not scalable to large k values. We require scalability in estimating k so that better identification performance can be achieved. In the following, we explore a new metric defined over the signal’s constellation map to estimate k at larger scales than cluster counting based works [28].

3.3.1 Intuition & Definition

As shown in Section 3.1.1, ideally 2^k clusters can be found in the constellation map of k -tag collision signal. However, when k further increases, for example, to 10, it is nearly

impossible to observe 1024 clusters. This renders cluster counting based works [3, 28, 31] unscalable to large k .

We explore another indicator of k that enables us to estimate k at larger scales.

Since an indicator of k only needs to be monotonically increasing or decreasing over k , we can explore different types of indicators. The area occupied by the data samples is a good candidate because: 1) it might increase when k increases; 2) it is possibly robust to overlapping clusters for large k . It is noted that mitigating the effect of noise is necessary when we design an area indicator, as some noisy samples may locate at marginal positions in the constellation map and thus raise the area.

Based on the above observation, we define an indicator of k , named “effective area” (EA), as follows. First we find the smallest rectangular region that contains all data samples in the constellation map and divide it into small square grids. Next we count the number of samples in each grid and set a threshold to differentiate noise grids from signal grids – the grids with the number of samples below the threshold are regarded as noise. EA is computed as the multiplication of the number of signal grids and the area of the grid unit.

Such divide-and-sum approach to compute the area of an irregular shape is common in basic calculus [27]. Both our work and PLACE [28] adopt the notion of “grid” but with different purposes. While [28] uses it to count the number of clusters, we use it to compute the EA metric.

3.3.2 Basic Observations from Experiment and Simulation Results

We conduct an experiment on our USRP2/WISP testbed (shown in Figure 2.1) to validate whether EA is qualified to be an indicator of k . We configure 1, 2, 3, 4 WISP4.1 tags to reply concurrently to single reader queries. The carrier wave frequency of the RFID

Figure 3.6: Tendency between average EA and k . The average EA is computed based on experimental traces.

reader is set to be 921MHz. The content that each tag replies is a random number (RN) sequence. We vary the relative positions of these tags and collect more than 200 traces of tag response signal from the USRP reader. As it is hard for us to obtain traces of larger k (i.e., $k > 4$) from our testbed, we could easily generate them by adding up available experimental traces of lower k (i.e., $k \leq 4$). For example, we can generate a trace of $k = 6$ by adding up one trace of $k = 2$ and another trace of $k = 4$, or three different traces of $k = 2$. Finally we generate 100 synthesized traces for each k from 5 to 10.

The above trace generation method is reasonable because the backscatter signals from tags to the reader also combine up linearly in the wireless channel. The difference between two types of traces is that the signal summation in experimental traces is weighted by each tag's uplink channel coefficients, while the summation is weighted by 1 in the synthesized traces. In the following we show that such a difference does not impact the overall tendency between the defined EA metric and k .

To compute EA, we empirically choose the grid size to be 0.01×0.01 and the grid density threshold to be 13% of total number of samples in a response slot. For each k , we

compute EA values for corresponding traces and display an average EA in Figure 3.6(a). From this figure, it is observed that EA increases exponentially over k . Figure 3.6(b) plots the standard deviation of EA versus k . We find though the standard deviation of EA also increases when k increases, it is small compared to the corresponding mean value of EA. This ensures that the general tendency between EA and k is stable across diverse traces.

To confirm whether the tendency between EA and k from Figure 3.6 is general or specific to certain environmental settings, we simulate k -tag baseband response signals ($k = 1, 2, \dots, 10$) based on the SNR profile of collected experimental traces, which often falls in the range [10dB, 30dB]. We simulate AWGN noise and fix the noise power N_0 to 0.1. Each tag can randomly pick up an initial signal amplitude in $[1, 10]$ and an initial signal phase in $[0, 2\pi]$. It is noted that each tag keeps its amplitude and phase unchanged in later reader queries, but the noise is random in each query. Since the power scale of simulated traces is much larger than that of experimental ones, we adjust the grid size and the density threshold to better distinguish between signal grids and noise grids and hence compute a more accurate EA. Empirically, we adjust the grid size to be 0.50×0.50 and leave the grid density threshold the same as the setting for Figure 3.6. FM0 scheme is used to encode the RN sequence transmitted by each tag. Each bit corresponds to 60 complex data samples and the total number of samples in a slot, denoted as N_s , is 6000. We add up individual tag responses to produce k -tag collision signal. For each k in $[1, 10]$, we generate 100 different tag response traces, where RN sequence of each tag is randomly generated once but kept unchanged later on.

We plot the average EA versus k for simulated traces in Figure 3.7, from which we observe a similar tendency as in Figure 3.6. The high similarity in two figures not only demonstrates the generality of the tendency between EA and k , but also proves the high credibility of our trace-driven simulations in Section 3.6.

3.7.a: Expectation.

3.7.b: Standard deviation.

Figure 3.7: Tendency between average EA and k . The average EA is computed based on simulated traces.

3.3.3 Appending Rule

The number of observed clusters constellation map hardly reach the theoretical 2^k and always upper bounded by the number of data samples N_s in a response slot. Each of our experimental traces contains 6000 to 8000 samples, which allows us to estimate k up to 12.

In fact, it makes no big difference to infer 8 or more than 8 (but less than 16) colliding replies if k is used for tree-based identification. Specifically, we need to append $\lceil \log_2 k \rceil$ bits to the current prefix for further queries, if the length of the prefix after appending does not exceed the tag ID length. Following this rule 4 bits will be appended if k is detected to be in the range $[9, 16]$. Hence, if we can infer 8 tags at most and the corresponding EA is denoted as a_0 , we would append 4 bits for any measured a satisfying $a > a_0$.

One may argue that more dedicated rules can be applied to optimize the number of appended bits. For $k = 9$, appending 3 bits may resolve most collisions but appending 4 bits may incur some inefficient queries to which no tag replies. Though it seems that

appending 3 bits is more optimal than appending 4 bits, it does not fundamentally improve the identification performance, due to two reasons. First, our mapping model between EA and k is a coarse one originated from experimental observations. Hence, the estimation result of k could be deviated from the ground truth. Second, in general tree-based schemes, an estimation of the magnitude of k for each reader query without incurring extra overhead is suffice to bring promising performance gain for the whole query process.

3.3.4 Modelling EA over k

A straightforward way is to build an exponential model for the relationship between EA and k :

$$EA = f(k) = \alpha e^{\beta k}, \quad (\text{Eq. 3.3})$$

where α and β are model parameters satisfying $\alpha, \beta > 0$. We take the natural logarithm of both sides in Eq. 3.3 and get:

$$\ln f(k) = \beta k + \ln \alpha. \quad (\text{Eq. 3.4})$$

Given $\langle k_i, f(k_i) \rangle$ pairs as inputs ($i = 1, 2, \dots, q$) to Eq. 3.4, we formulate an over-determined linear equation groups and can easily solve it based on the least square error principle.

To achieve the real-time RFID identification of an unknown tag set, we need to train the above model before utilizing EA as a k -indicator. We first isolate a small number of tags k_s ($k_s > 2$) from the target tag set and collect the mixed RN signals from 1 to k_s concurrent tag replies. Next we collect the responses and compute EA for each k in the range $[1, k_s]$. Finally we solve for α and β in Eq. 3.3 using the obtained information as input. To increase the model accuracy, the reader could repeat querying for multiple times and collect large amount of inputs, which does not incur much training overhead as RFID reader can query many times in a second.

The above model is controlled by only two parameters and thus sensitive to computational errors. To handle this issue, we propose a non parametric model. Since inferring k up to 8 is suffice according to Section 3.3.3, we adopt a range based model for each $k \in [1, 8]$. If measured EA is in a certain range, the associated k can be obtained. To train the range based model 8 tags are chosen from the tag set and multiple traces are collected for each k . After then the expectation and standard deviation of EA are computed; the minimum/maximum of the range of EA for k are obtained by subtracting/adding standard deviation from/to expectation. Such model is robust because the range estimation errors for k_1 are not easily propagated to that for k_2 , for $k_1 \neq k_2$.

3.4 PHY-Tree Algorithm

In this section, we propose an efficient and robust algorithm for RFID identification, using both horizontal and vertical information presented in Section 3.2 and Section 3.3.

3.4.1 A Basic Algorithm

Suppose a target set contains M tags, each has an ID with the length of L bits. We denote the ID of the tag j as $ID_j = b_{L-1}b_{L-2}...b_1b_0$, where b_{L-1} is MSB and b_0 is LSB. When the reader queries with an l -bit prefix $Q = q_{l-1}q_{l-2}...q_1q_0$ ($1 \leq l \leq L$), tag j checks if the first l bits of its ID, $b_{L-1}b_{L-2}...b_{L-l}$, match Q . If they match, the tag j replies to the reader with its ID.

We first design a basic identification algorithm. The reader starts the query from the first layer of the binary tree (i.e., $Q = 0$ or $Q = 1$). Upon receiving the colliding physical layer signal from k tags for the prefix Q , the reader can compute an estimation of the number of replying tags $\hat{k}$ (vertical information) and obtain an inference vector $G = g_{L-1}g_{L-2}...g_1g_0$ (horizontal information), where g_i ($i = 0, 1, ..., L - 1$) indicates the

bit state at index i . Specifically, g_i is set to 0 or 1 if an all-0 or all-1 state is detected respectively and -1 otherwise.

Since only tags matching Q respond, we know that $g_{L-1}g_{L-2}\dots g_{L-l} = q_{l-1}q_{l-2}\dots q_1q_0$. If collision happens, the reader resolves the collision as follows. First it uses $\hat{k}$ and the appending rules to derive the optimal number of appended bits h . The reader appends h bits to the current queried Q to obtain the appended prefix query set $\mathcal{Q}'$ for subsequent queries. For example, if h is 3, $\mathcal{Q}'$ includes prefixes from $q_{l-1}q_{l-2}\dots q_1q_0000$ to $q_{l-1}q_{l-2}\dots q_1q_0111$. Second the reader uses G to filter out unnecessary prefixes from $\mathcal{Q}'$. If $g_{L-l-3} = 1$ in the above example, $\mathcal{Q}'$ is reduced to only 4 prefixes: $q_{l-1}q_{l-2}\dots q_1q_0001$, $q_{l-1}q_{l-2}\dots q_1q_0011$, $q_{l-1}q_{l-2}\dots q_1q_0101$ and $q_{l-1}q_{l-2}\dots q_1q_0111$. G is used to filter the unnecessary prefixes from $\mathcal{Q}'$. If $g_{L-l-3} = 1$ in the above example, $\mathcal{Q}'$ is reduced to only 4 prefixes: $q_{l-1}q_{l-2}\dots q_1q_0001$, $q_{l-1}q_{l-2}\dots q_1q_0011$, $q_{l-1}q_{l-2}\dots q_1q_0101$ and $q_{l-1}q_{l-2}\dots q_1q_0111$.

Generally h is computed based on only the appending rule in Section 3.3.3 (i.e., $h = \lceil \log_2 \hat{k} \rceil$). However, if subsequent d bits are detected as resonant states right after the prefix Q , i.e., $g_{L-l-1}\dots g_{L-l-d}$ takes valid values (0 or 1), h is adjusted to be $\max(\lceil \log_2 \hat{k} \rceil, d + 1)$. It is noted that we detect up to 8 colliding tags as we discuss in Section 3.3. For any $\hat{k} > 8$, we set $\hat{k}$ to 9 and $\lceil \log_2 \hat{k} \rceil$ is 4.

We describe the whole query process as follows. In the binary tree representation, the reader queries in the breadth-first mode. Denote $\mathcal{N}$ as the queue of subsequent prefixes for query. $\mathcal{N}$ is initialized with 2 prefixes “0” and “1” and is automatically sorted in increasing prefix length. In the case that two prefixes in $\mathcal{N}$ have the same length, the one with smaller value is queried first. For example, prefix 010 is queried before prefix 011. The reader first obtains the minimum prefix length l_{min} in $\mathcal{N}$ and queries all prefixes with length l_{min} in $\mathcal{N}$. Each time the reader pops out the first prefix in $\mathcal{N}$ for query and waits for tag responses. If less than 3 tag replies, the reader continues to pop out a

Algorithm 3 Basic *PHY-Tree* Algorithm for RFID Reader

```

1:  $\mathcal{N} \leftarrow \{ "0", "1" \}$ ,  $\mathcal{N}$  is always automatically sorted in increasing order.
2: while  $\mathcal{N} \neq \emptyset$  do
3:   Pop out the smallest prefix  $Q$  from  $\mathcal{N}$ .
4:   Send the query command with  $Q$  to tags.
5:   Receive tag responses.
6:   Compute  $G, \hat{k}, d$ .
7:    $h \leftarrow \max(\lceil \log_2 \hat{k} \rceil, d + 1)$ .
8:   if  $\hat{k} > 1$  then
9:     Obtain the appended prefix query set  $\mathcal{Q}'$  (of size  $2^h$ ) by appending  $h$  bits to  $Q$ . Filter  $\mathcal{Q}'$  with  $G$ .
10:    Add  $\mathcal{Q}'$  into  $\mathcal{N}$ .
11:   else if  $\hat{k} = 1$  then
12:     Identify a tag ID.
13:   else
14:     Continue.
15:   end if
16: end while

```

Algorithm 4 Basic *PHY-Tree* Algorithm for RFID Tag

```

1: while TRUE do
2:   Wait for the reader's query command containing query prefix  $Q = q_{l-1}q_{l-2}\dots q_1q_0$ .
3:   Compare its first  $l$ -bit ID ( $b_{L-1}b_{L-2}\dots b_{L-l}$ ) with  $Q$ .
4:   if  $b_{L-1}b_{L-2}\dots b_{L-l} = q_{l-1}q_{l-2}\dots q_1q_0$  then
5:     Reply to the command with its full ID ( $b_{L-1}b_{L-2}\dots b_1b_0$ )
6:   else
7:     Keep silent.
8:   end if
9: end while

```

new Q in $\mathcal{N}$ for query. Specifically if one or two tags reply to Q , they can be uniquely identified. Otherwise, the reader extracts $G, \hat{k}$ and d from the colliding signal to compute h . The reader appends h bits to Q to get the initial $\mathcal{Q}'$, filters out some prefixes in $\mathcal{Q}'$ using G and adds the updated $\mathcal{Q}'$ to $\mathcal{N}$. The reader repeats the above query procedure until $\mathcal{N}$ becomes empty.

Algorithm 3 and Algorithm 4 clearly summarize the behaviours of both reader and tag in the basic *PHY-Tree* scheme.

3.4.2 Error Compensation

The bit state detection algorithm in Section 3.2 may mistake mixed states (i.e., mix of bit 0 and 1) for resonant states and guide the reader to ignore some subtree nodes of the current prefix. Therefore, some tags in the set are missed by the reader. We need to eliminate this problem to meet the basic requirement of RFID identification.

A straightforward solution is to record the list of unqueried prefixes due to the guidance of G for each query in the basic algorithm because the missed tags may reply to these prefixes. After the basic algorithm in Section 3.4.1 stops, the reader adopts the original QT [34] to query prefixes in this list and identifies the remaining tags, without using either horizontal or vertical information. Specifically, it uses depth-first query method to traverse the whole binary tree and always appends one bit for the next queries if collision is encountered. Although it ensures no tag is missed, this solution counters against the benefit brought by horizontal information and degrades the performance gain.

Here we propose a more efficient compensation scheme that could reserve the benefit of horizontal information. The simulation results in Section 3.6.1 show that the accuracy of detecting resonant states is high under tight synchronization among replying tags. This indicates that a majority of tags in the target set are identified using the basic algorithm (which is also supported by the simulation results in Section 3.6.1). This motivates us to design the following compensation scheme. After the basic algorithm stops, we mute those tags which have been already identified so that they will not interfere with the second round, where the reader performs a fresh identification using QT, starting from the first layer. Since only a few unidentified tags are left in the second round, it does not take many reader queries to finish the second round, indicating its efficiency.

3.4.3 Further Improvement in Efficiency

By combining horizontal and vertical information in another perspective, we can identify the IDs of 2 colliding tags without further queries and hence gain extra identification

ability. Basically if we can estimate $\hat{k} = 2$ for one query with high accuracy, we can uniquely identify each of two tags with its partial ID, rather than decode the whole ID. We explain the reason as follows. Since each tag owns a unique ID, IDs of the two tags must take different bit values in at least one bit index (e.g., i). In other words, a mixed bit state exist at index i . Our bit state detection algorithm can find out bits with mixed states and its accuracy is quite high when only two tags reply (reported in Section 3.6.1). The partial IDs for two tags are thus set as the queried prefix concatenated with bit 0 and 1 at index i .

We incorporate this feature in the tree-based algorithm by slightly modifying the appending rule used in Section 3.4.1:

$$h = \max(\lceil \log_2 \hat{k} \rceil - 1, d + 1). \quad (\text{Eq. 3.5})$$

The term $\lceil \log_2 \hat{k} \rceil - 1$ in Eq. 3.5 reflects the change brought by extra identification ability of 2 tags. It is noted that we cannot extend the partial ID identification to the case in which more than 2 tag reply, due to two reasons. First, the estimation error in the EA- k model increases when k increases, leading to the inaccurate estimation results for $k > 2$. Second, even if k is correctly estimated, we cannot identify these tags definitely as there are more than k possible combinations in several bits with mixed states.

3.4.4 Ultimate PHY-Tree Algorithm

We produce the ultimate algorithm by updating the appending rule in Eq. 3.5 and incorporating the error compensation mechanism in Section 3.4.2 into the basic algorithm. We do not display the corresponding pseudo codes for the ultimate algorithms for both reader and tags, as they can be obtained through slight modifications on Algorithm 3 and Algorithm 4.

3.5 Discussion on synchronization

When we illustrate two types of physical layer information in Section 3.2 and 3.3, we assume that the responding tags in a slot are tightly synchronized. In real world scenarios, tags have large diversities in their response delays due to multiple factors such as manufacturers, types, antenna orientations, etc. In this section, we briefly discuss how imperfect synchronization affects the two types of physical layer information and suggest how to mitigate the impact.

On one hand, the imperfect synchronization makes negligible impact on the vertical information. We estimate the number of concurrent replies by processing samples in the entire time slot, rather than samples in individual bit durations. Thus the vertical information does not rely on the internal timings between contiguous bit durations in the response signal.

On the other hand, imperfect synchronization degrades the quality of the horizontal information. Specifically, discovering all-0 and all-1 states in a bit requires bit-level synchronization. Without perfect synchronization, we may accidentally detect a resonant state as a mixed state (false negative), which leads to missed opportunities for saving some reader queries, or detect a mixed state as a resonant state (false positive), which results in some tag IDs being not identified. Unlike Buzz [60], which can calibrate for each tag's clock to adjust its starting time offset so that responding tags achieve tight synchronization, we aim to identify all IDs of a completely unknown tag set. Thus, we should not devote any effort to calibration; otherwise such an effort can afford us to collect all tag IDs one by one.

We suggest several approaches to mitigate the impact of imperfect synchronization on our scheme. Each tag can use lower frequency for backscattering its ID to reduce the unsynchronization rate, which is the ratio of the maximum starting time offset over the individual bit duration. For example, if 4 symbols are used to encode one bit, the

unsynchronization rate can be reduced by half compared to using FM0 coding. Better circuits are expected to be designed for RFID tags to further reduce their unsynchronization rates [17, 24, 49]. Besides, to completely eliminate false positives for the horizontal information, we design an error compensation mechanism in Section 3.4.2. Additionally, powerful collision recovery schemes like BiGroup [44] (which can decode 4 to 5 collision tags without assuming tight synchronization) can be used independently in our scheme to further improve the overall identification performance, since the partial ID identification of 2 tags (described in Section 3.4.3) contains errors under imperfect synchronization.

3.6 Trace-Driven Simulation

3.6.1 Microbenchmark

In this section, we run microbenchmark simulations to validate the effectiveness of horizontal and vertical information. We generate traces from simulation as described in Section 3.3.2 for k collisions in a slot. For each simulation setting, we repeat for 100 runs to obtain the average performance metric.

3.6.1.1 About Horizontal Information

We first study the performance of the resonant state detection algorithm in obtaining horizontal information. Three important performance metrics are overall accuracy, false positive rate (FPR) and false negative rate (FNR). The overall accuracy refers to the ratio of the number of correct detecting both resonant and mixed states, over the total number of bit states. False positive rate refers to the ratio of the number of detecting mixed state as resonant state, over the number of actual mixed states. False negative rate refers to the ratio of the number of detecting resonant state as mixed state, over the number of actual resonant states.

Figure 3.8: Searching for optimal range for c in resonant state detection algorithm with different k s.

We first find the optimal c which is an important tunable parameter in Eq. 3.1. We vary k from 2 to 8 and design a set of 10-bit tag IDs (which is enough to accommodate 8 different IDs) for each k . We ensure 20%, 20% and 60% of the bit indexes among 10 bits are all-0, all-1 and mix 0/1 states. With random SNR and fixed noise power of each generated trace, we compute an average overall accuracy for each k .

We plot the overall accuracy over c in the range $[0.1, 10]$ in granularity of 0.1 in Figure 3.8. From this figure, we discover that for each k , the overall accuracy first increases sharply with c , maintains high in a certain range and decreases slowly with c in the end. We highlight the 0.95-accuracy line in this figure and find an optimal range $[1.39, 4.07]$ of c for all k s. It is noted that for $k = 2$, the overall accuracy is always 100% for any c , demonstrating the credibility of the partial ID identification of 2 colliding tags in Section 3.4.3.

After obtaining the optimal c range, we study the performance of the resonant state detection algorithm for further increasing k . We set tag ID length to 100 to better accommodate more unique tag IDs and each of the all-0, all-1 and mixed state occupies 33.3% of the total 100 bits. We choose 6 values for c from the optimal range (1.5, 2, 2.5, 3, 3.5 and 4). A slight difference from the setting in Figure 3.8 is that we generate

3.9.a: Overall accuracy.

3.9.b: False positive rate.

3.9.c: False negative rate.

Figure 3.9: Extensive study of the performance of the resonant state detection algorithm in obtaining horizontal information when k varies in a wide range and c is fixed to be optimal.

random set of tag IDs in each run given a value of k . We plot the overall accuracy, FPR and FNR over k from 3 to 100 in Figure 3.9. We start with $k = 3$ because the overall accuracy for $k = 2$ is 100% given a value of c .

The general trend in Figure 3.9(a) is that the overall accuracy in detecting both resonant and mixed states decreases when k increases. This is natural as collision from more tags impose more challenges on the detection. Nevertheless, the algorithm achieves as high as 70% accuracy even when 20 tags collide for $c = 1.5$. From Figure 3.9(b), we find FPR increases when c increases. We emphasize in Section 3.4.2 that high FPR in utilizing horizontal information is particularly not desired and hence should be as low as possible. As such, we can trade off between the overall accuracy and FPR; if we set c to 2.0, we can obtain $\leq 1\%$ FPR and achieves $\approx 80\%$ overall accuracy when k is 20. In Section 3.6.2 when we compare *PHY-Tree* with existing tree-based identification schemes, we fix c to 2 in *PHY-Tree*. In Figure 3.9(c), we can see that FNR increases over k and decreases over c , which matches the trend in Figure 3.9(a).

Any non-zero false positive rate would trigger our error compensation scheme in Section 3.4.2. If the number of tags, which are missed in the basic query round as

3.10.a: Percentage of missing tags in basic query round.

3.10.b: Percentage of NoQ in the error compensation round.

Figure 3.10: Impact of false positive of horizontal information on overall performance of *PHY-Tree*.

described in Section 3.4.1, is large, the follow-up error compensation round incurs large query overhead and renders our proposed *PHY-Tree* inefficient. We thus need to study about such overhead to better evaluate our scheme.

Two metrics are considered: the percentage of missed tags over the tag set population M in the basic query round and the percentage of queries in the error compensation round over the total number of queries. We evaluate the *PHY-Tree* algorithm on M tags with randomly generated unique 15-bit IDs, where M varies from 2^6 to 2^{12} . Figure 3.10 plots the result.

In Figure 3.10(a), we find that for all M values, the average percentage of missed tags is $\leq 2.2\%$. In Figure 3.10(b), we find the percentage of queries in the compensation round is $\leq 4\%$ when $M > 8$. Both figures in Figure 3.10 demonstrate the small overhead and high efficiency of our error compensation scheme.

3.6.1.2 About Vertical Information

Next we study the accuracy of obtained vertical information (i.e., k). As discussed in Section 3.3.3, we only need to infer up to 8 tag replies with reasonable accuracy and for

3.11.a: Inference accuracy of the EA model for wide-range k .

3.11.b: Comparison with SSDA for small-range k .

Figure 3.11: Accuracy comparison in estimating k between EA- k model and SSDA algorithm in [28].

$k > 8$, we only need to judge if $\hat{k} > 8$. We adopt the non parametric model in Section 3.3.4 and train the range for each $k \in [1, 8]$. The training set contains 100 randomly generated traces and the testing set contains another 100 newly generated traces, for each k . Each tag replies with 100-bit RN sequence. In both the training and testing sets, we create 9-tag collision traces for $k > 8$, which is the most challenging scenario for our EA- k model since EA increases over k . We compare the estimation $\hat{k}$ with the ground truth k and compute the ratio of the number of correctly inferred traces over the overall number of traces as the accuracy. Figure 3.11(a) plots the result.

It is observed from Figure 3.11(a) that our EA model achieves $\approx 80\%$ accuracy on average. The corresponding accuracy is somehow low for certain k value like 2, 6, 8. This is because of overlapping ranges incurred in the training test: the upper bound of the range for k is larger than the lower bound of the range for $k + 1$. To alleviate this issue, we can set the middle of the overlapping range as the common value for the upper bound of k and lower bound of $k + 1$.

In Figure 3.11(b), we compare the accuracy between EA- k model and the SSDA

algorithm in [28], which can estimate up to 4 colliding tags. We implement SSDA using the original settings in [28] and set the k range to $[1, 4]$ for fair comparison. We observe from Figure 3.11(b) that SSDA achieves higher accuracy ($\approx 90\%$) than our EA- k model does. Actually we can combine both methods for better accuracy. Specifically, we can use both methods to judge whether both estimation results for k are below 4. If it is so, we apply SSDA for better accuracy; otherwise we apply EA model for better scalability.

3.6.2 Comparison with Existing Identification Schemes

In this subsection, we compare *PHY-Tree* (short for PhT) with three existing tree-based identification works: QT [34], STT [47] and TH [54], which are briefly introduced in Section 3.1.2. It is noted that TH estimates the tag set population using many extra slots before the actual identification process. This estimation contains error and affects its efficiency. We thus provide two options for TH: one with accurate population input (denoted as ATH) and another with rough estimation (denoted as RTH). Similar to [54], we adopt the average number of queries (NoQ) as the performance metric, which is defined as the ratio of total number of reader queries over tag set population M . We also consider three types of tag ID distribution, namely uniform, block and random (described in Section 3.1.3). In block distribution we first set the block size b to 10. Tag ID is of 15 bits and M varies from 2 to 2^{12} . We plot the comparison results under three ID distributions in Figure 3.12.

Our observations from Figure 3.12 are as follows. First, PhT consumes the least amount of queries among all schemes under three ID distributions. In the random ID distribution, which represents the most typical case in real world scenarios, the average number of queries for PhT is around 1 and thus reduces the number of queries of ATH (which is the most efficient scheme among existing ones), by $1.79\times$. Second, existing schemes take more queries to handle non uniform distributions, especially the random

Figure 3.12: Performance comparison between several schemes under three types of tag ID distribution.

distribution, than handle the uniform one. Third, STT performs best under uniform and block ID distributions, due to its assumption on continuous tag ID pattern. Fourth, RTH is the slowest scheme under nearly all circumstances, which is mainly caused by an error-prone estimation of the tag set population M ; in contrast, ATH is much faster. To conclude, we validate the effectiveness of the proposed *PHY-Tree* scheme.

In block ID distribution, one tunable parameter is the block size b . Among three types of tag ID distribution, block distribution lies between uniform distribution and random distribution in terms of ID continuity, depending on b . Random and uniform distribution can be regarded as a block distribution with $b = 1$ and $b \approx M$ (if M is a power of 2) respectively. It is thus interesting to study the performances of these identification schemes with different levels of ID continuity, i.e., b . We fix M to 5000 and vary b in the range $[10, 100]$ in granularity of 10 and $[100, 1000]$ in granularity of 100. For each b we create 100 random sets of tag IDs and execute identification schemes on these sets. We plot the average NoQ over b in Figure 3.13.

From this figure we find that for all schemes, especially STT, the number of reader queries decreases when b increases. When b is large (e.g., 100), the performance of STT nearly approaches that of PhT. This, once again, validates that STT is efficient

3.13.a: Small block size.

3.13.b: Large block size.

Figure 3.13: Detailed study on block ID distribution.

under block distribution with large b as it has nearly perfect knowledge on the tag ID distribution. However when b is small, *PHY-Tree* is more efficient.

3.6.3 Impact of Unsynchronization Rate

Above performance comparison results are obtained by assuming that concurrent replying tags are perfectly synchronized with each other. Nevertheless, unsynchronized scenarios are common in practice [60]. In this subsection, we simulate the scenario of unsynchronized tag replies and study the impact of unsynchronization rate, defined in Section 3.5, on the performance of *PHY-Tree*.

We first describe how we generate unsynchronized tag response traces. We denote the unsynchronization rate as r and the number of samples in a bit as s . Given r and s , we could compute the number of samples g as $r \times s$, which should be offset from the starting transmission time. Before a tag replies, it delays for a random number of samples from the range $[0, g]$, such that any two replying tags reply at different starting time. An unsynchronized trace is generated by adding these signals together.

Since the vertical information is not likely affected by imperfect synchronization

3.14.a: Overall accuracy.

3.14.b: False positive rate.

3.14.c: False negative rate.

Figure 3.14: The impact of r on the accuracy of horizontal information with initially optimal c for $r = 0$.

among replying tags, we focus on the impact of the unsynchronization rate r on the accuracy of horizontal information, when $k = 2, 4, 6, 8$. We use the same settings for the length and distribution of tag ID set given k as in Figure 3.9. We fix the parameter c in Eq. 3.1 to be 2.5, which falls into the optimal range according to Figure 3.8. We vary r in the range $[0.1, 5]$ and compute the average overall accuracy, FPR and FNR for each r . Results are plotted in Figure 3.14.

From Figure 3.14, we find that both FPR and FNR increase over r and the overall accuracy decreases accordingly over r in general. Since FPR is around 0.05 when $r = 0.5$ and $k = 2$, our error compensation scheme is still efficient under typical scenarios ($r \leq 0.5$, reported in Buzz [60]). Besides, the partial ID identification of 2 colliding tags is still possible when $r \leq 0.5$, as long as the tag ID is long (which increases the probability of detecting at least one mixed state in any bit index).

We could also conclude that high FNR is the main reason for the performance degradation of *PHY-Tree*. Nevertheless, the horizontal information is not rendered useless under unsynchronized scenarios as it can be utilized opportunistically in *PHY-Tree*.

We also study the impact of r on the overall identification performance. We run the standard *PHY-Tree* algorithm on a tag set with random ID distribution. We fix

Figure 3.15: Impact of r on the average NoQ of *PHY-Tree*.

the population M to 2^{12} and vary r in the range $[0.1, 5]$. We set c to be 2.5. Figure 3.15 plots the result. For better comparison, we also plot the performance under perfect synchronization (i.e., $r = 0$) with a blue horizontal line.

From Figure 3.15, we observe that the performance for $r \in [0.1, 0.4]$ is approaching that for $r = 0$ (with an average NoQ of around 1.20). This demonstrates that the horizontal information is still correctly utilized and brings gain when unsynchronization is not severe. When r further increases, the performance degrades due to the malfunctioning of both horizontal information and related extended identification ability for $k = 2$. Nevertheless, the average NoQ under such case is around 2.05, which is $1.08\times$ smaller than that under ATH. Despite the degraded horizontal information, *PHY-Tree* still achieves performance gain thanks to the unaffected vertical information.

3.7 Related work

Many research efforts are devoted to design fast RFID identification methods. In the classic ALOHA [68] scheme, tags reply in a random slot and only singleton slots can be identified. By optimizing the frame size as the number of tags in a given set, the slot utilization efficiency is maximized. Tree-based schemes [15, 34, 40, 47, 54] aim to reduce

the number of collision replies through reader-tag interactions. Hybrid approaches like [50] combine binary tree search and ALOHA for tag identification. Unlike these works, we utilize physical layer information from tag replies to increase the identification speed.

While identification schemes requires multiple slots to resolve a collided response, collision recovery methods aim at decoding concurrent replies from multiple tags. Similar to our work, Buzz [60] also assumes bit-level synchronization among responding tags. After then it could treat multiple replying tags as a whole and decode in a rateless way. [3, 55] analyze the constellation map and could separate 2 tag collisions. BiGroup [44] can decode up to 5 concurrent tag replies with reasonable accuracy by extracting temporal-spatial features from the received signal. Although our proposed scheme can identify 2-tag collisions, we do not target at improving the collision recovery capacity in one slot. Instead, we focus on improving the overall query process to identify all tags with minimum number of queries. Most of the current collision recovery schemes like [44] are orthogonal to our work and hence could be easily integrated into our framework for better performance.

Existing works [28, 42, 63, 77] have extensively explored how to utilize physical layer information to support other operations. PLACE [28] counts the number of clusters in constellation map to enhance RFID cardinality estimation. Tagoram [63] utilizes phase information from RFID tags attached to mobile objects in order to locate and track them. Tagyro [77] attaches tag arrays on a rotating object and infers its 3D orientation from the relative phase offset of these tags over a reference tag in the array. Tagyro considers and handles many practical challenges including tag coupling effect, tag antenna polarization and frequency hopping discontinuity for COTS readers. By examining the constellation map of received MIMO symbol in wireless access network, [42] can decode transmitted symbols from multiple users and significantly reduce the computation overhead.

3.8 Summary

Traditional tree-based RFID identification methods are only MAC-layer solutions and perform poorly especially when the tag ID distribution is highly random. In this chapter, a novel tree-based RFID identification scheme is proposed, which utilizes physical layer information to improve the identification performance. Two types of physical layer information (horizontal and vertical) are explored and extracted from the colliding tag response signal. First, a subset of tree nodes, rooted from the queried prefix, can be pruned, by opportunistically detecting resonant states for each bit index. Second, the number of colliding tags can be inferred from a metric defined over the constellation map. By obtaining both information from each tag response and accumulating them in the long-run identification process, the proposed *PHY-Tree* algorithm converge to exact positions of tag IDs in the binary tree more quickly than existing MAC-only tree-based identification schemes. Trace-driven simulations demonstrate significant performance gain of *PHY-Tree*, in terms of the number of reader queries, over existing tree-based works.

Chapter 4

Conclusion & Future Work

Most of works on improving the efficiency of fundamental RFID operations are purely MAC-layer solutions and suffer greatly from unconscious tag collisions. In this thesis, I suggest that physical layer information from colliding tag responses can be extracted and utilized to further improve operation efficiencies of two scenarios: cardinality estimation and identification. In cardinality estimation, the number of colliding tags (up to 3) can be accurately inferred from the signal's constellation map and be utilized to improve estimation accuracy. In tree-based identification, I can estimate the number of colliding tags in larger scales with a novel metric. Moreover, special bit states of the colliding signal are accurately detected and utilized as an hint to prune the search space of the binary tree. Both USRP/WISP testbed and extensive simulations demonstrate the performance gain of the proposed schemes.

In the future, I would apply the cross-layer design principle to more RFID MAC-layer operations, such as tag searching [69], missing tag identification [70], or tag grouping [36]. A further step might be to integrate these PHY-enhanced operations together and design a multifunctional and integrated RFID database systems.

Besides RFID MAC layer operation, I am also interested in applying the principle to more fantasy RFID application-layer scenarios. Currently there exists many interesting works [74–76, 78–80, 83] that leverage available RSSI and phase data of the tag

response signals in COTS RFID readers to enable ubiquitous human-machine interactions. Shangguan *et al.* [75] use phase values over time from COTS readers to fingerprint diverse human behaviors and infer important customer purchasing features when they are shopping. Ding *et al.* [74] compute Doppler shifts from raw phase data in COTS readers and monitor the body training progress of an exerciser by attaching RFID tags to dumbbells that he uses. Specific body movement patterns can be detected and many professional metrics be extracted from the series of Doppler samples, so as to better evaluate the training performance and guide the subsequent training plans. Lin *et al.* [80] propose Tagball, a controllable ball as a “mouse”, for 3D human-computer interaction. By attaching multiple RFID tags on a ball object, its translation and orientation could be accurately tracked.

Beyond the above mentioned works in the RFID field, works adopting the similar principle exist in other domains like wireless networks [13, 42, 43, 78, 79, 83], MIMO systems [6–11], cellular networks [29], etc. For example, in the wireless domain, Kellogg *et al.* [78] propose AllSee, a gesture recognition system for low-end battery-free devices. AllSee utilizes ambient signals in the air (e.g., TV tower) as a natural carrier wave for amplitude-based gesture modulation. It designs special hardware to extract such information and make inference about the gesture performed by extracting time-domain features of the gestures. Pu *et al.* [79] present WiSee, a system that utilizes wireless signals for human gesture recognition. In WiSee, no sensing devices are mounted on the human body and different gestures are recognized and classified, according to their impacts of movements on the Doppler shift profiles of wireless signals. Parate *et al.* [83] attach multiple Inertial Measurement Units (IMU), which consists of accelerometer, gyroscope, compass and provides 3D orientation info about the target moving object, on the wrist of a person to recognize his repetitive smoking gestures. I think it would be better to refer to these works in these domain and obtain hints by comparing and contrasting them to the RFID counterparts.

Appendix A

Author's Publications

- (i) Yuxiao Hou, Yuanqing Zheng and Mo Li. “Fair QoS Multi-Resource Allocation for Wireless LAN”, in *IEEE/ACM International Symposium on Quality of Service (IWQoS)*, Hong Kong, China, 2014.
- (ii) Yuxiao Hou, Yuanqing Zheng and Mo Li. “Fair QoS Multi-Resource Allocation for Uplink Traffic in WLAN”, *Springer Wireless Networks (WINET)*, 1-20, 2015.
- (iii) Yuxiao Hou, Jiajue Ou, Yuanqing Zheng, and Mo Li. “PLACE: Physical Layer Cardinality Estimation for Large-Scale RFID Systems”, in *IEEE International Conference on Computer Communications (INFOCOM)*, Hong Kong, China, 2015.
- (iv) Yuxiao Hou, Jiajue Ou, Yuanqing Zheng, and Mo Li. “PLACE: Physical Layer Cardinality Estimation for Large-Scale RFID Systems”, *IEEE/ACM Transactions on Networking (ToN)*, 99(10):pp, 2016.
- (v) Yuxiao Hou, Yuanqing Zheng, and Mo Li. “PHY Assisted Tree-based RFID Identification”, in *IEEE International Conference on Computer Communications (INFOCOM)*, Atlanta, USA, 2017.

References

- [1] EPCglobal C1G2 UHF RFID protocol at 860MHz-960MHz, <http://www.epcglobalinc.org/standards/uuhfc1g2>, July 2014.
- [2] B. Abidi, S. Yahia, and A. Bouzeghoub. A new algorithm for fuzzy clustering able to find the optimal number of clusters. In *IEEE International Conference on Tools with Artificial Intelligence*, 2012.
- [3] C. Angerer, R. Langwieser, and M. Rupp. RFID reader receivers for physical layer collision recovery. *IEEE Transactions on Communications*, 58(12):3526-3537, 2010.
- [4] S. Asadi and O. Obulesu. Cluster the unlabeled datasets using extended dark block extraction. In *IEEE INCOCCI*, 2010.
- [5] G. Avoine and P. Oechslin. A scalable and provably secure hash-based RFID protocol. In *IEEE PerCom Workshops*, 2005.
- [6] S. Kumar, D. Cifuentes, S. Gollakota, and D. Katabi. Bringing cross-layer MIMO to today's wireless LANs. In *ACM SIGCOMM*, 2013.
- [7] S. Gollakota, S. Perli, and D. Katabi. Interference alignment and cancellation. In *ACM SIGCOMM*, 2009.
- [8] K. Lin, S. Gollakota, and D. Katabi. Random access heterogeneous MIMO networks. In *ACM SIGCOMM*, 2011.
- [9] H. Rahul, S. Kumar, and D. Katabi. MegaMIMO: scaling wireless capacity with

- user demand. In *ACM SIGCOMM*, 2012.
- [10] E. Aryafar, N. Anand, T. Salonidis, and E. Knightly. Design and experimental evaluation of multi-user beamforming in wireless LANs. In *ACM MobiCom*, 2010.
- [11] O. Bejarano, E. Knightly, and M. Park. IEEE 802.11ac: from channelization to multi-user MIMO. In *IEEE Communication Magazine*, 51(10):84-90, 2013.
- [12] N. Bhandari, A. Sahoo, and S. Iyer. Intelligent query tree (IQT) protocol to improve RFID tag read efficiency. In *IEEE ICIT*, 2006.
- [13] D. Bharadia, E. McMillin, and S. Katti. Full duplex radios. In *ACM SIGCOMM*, 2013.
- [14] Z. Cao, C. Sutton, Y. Diao, and P. Shenoy. Distributed inference and query processing for RFID tracking and monitoring. *Proceedings of the VLDB Endowment*, 4(5):326-337, 2011.
- [15] J. Capetanakis. Tree algorithms for packet broadcast channels. *IEEE Transactions on Information Theory*, 58(12):3526-3537, 1979.
- [16] B. Chen, Z. Zhou, and H. Yu. Understanding RFID counting protocols. In *ACM MobiCom*, 2013.
- [17] N. Cho, S. Song, S. Kim, S. Kim, and H.J. Yoo. A 5.1- μ W UHF RFID tag chip integrated with sensors for wireless environmental monitoring. In *IEEE ESSCIRC*, 2005.
- [18] M. Delgado, C. Angerer, J. V. Alonso, and M. Rupp. Estimation of the tag population with physical layer collision recovery. In *The Third International EURASIP Workshop on RFID Technology*, 2010.
- [19] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *ACM KDD*, 1996.
- [20] K. Finkenzeller. RFID handbook: radio-frequency identification fundamentals and applications. John Wiley & Sons, 2000.

- [21] A. Gamal, M. Mohseni, and S. Zahedi. Bounds on capacity and minimum energy-per-bit for AWGN relay channels. *IEEE Transactions on Information Theory*, 52(4):1545-1561, 2006.
- [22] A. Goldsmith. Wireless communications. Cambridge University Press, 2005.
- [23] W. Gong, K. Liu, X. Miao, Q. Ma, Z. Yang, and Y. Liu. Informative counting: fine-grained batch authentication for large-scale RFID systems. In *ACM MobiHoc*, 2013.
- [24] R. Greeff, F. Smith, and D. Ovard. RFID device time synchronization. Patent US7889083, 2006.
- [25] H. Han, B. Sheng, C. Tan, Q. Li, W. Mao, and S. Lu. Counting RFID tags efficiently and anonymously. In *IEEE INFOCOM*, 2010.
- [26] H. Hassanieh, J. Wang, D. Katabi, and T. Kohno. Securing RFIDs by randomizing the modulation and channel. In *USENIX NSDI*, 2015.
- [27] J. Heinbockel. Introduction to calculus. Old Dominion University, 2012.
- [28] Y. Hou, J. Ou, Y. Zheng, and M. Li. PLACE: physical layer cardinality estimation for large-scale RFID systems. In *IEEE INFOCOM*, 2015.
- [29] J. Huang, F. Qian, Y. Guo, Y. Zhou, Q. Xu, Z. Mao, S. Sen, and O. Spatscheck. An in-depth study of LTE: effect of network protocol and application behavior on performance. In *ACM SIGCOMM*, 2013.
- [30] A. Juels. RFID security and privacy: a research survey. *IEEE Journal on Selected Areas in Communications*, 24(2):381-394, 2006.
- [31] R. Khasgiwale, R. Adyanthaya, and D. Engels. Extracting information from tag collisions. In *IEEE International Conference on RFID*, 2009.
- [32] M. Kodialam and T. Nandagopal. Fast and reliable estimation schemes in RFID systems. In *ACM MobiCom*, 2006.
- [33] M. Kodialam, T. Nandagopal, and W. Lau. Anonymous tracking using RFID

- tags. In *IEEE INFOCOM*, 2007.
- [34] C. Law, K. Lee, and K. Siu. Efficient memoryless protocol for tag identification. In *ACM DIALM*, 2000.
- [35] T. Li, S. Wu, S. Schen, and M. Yang. Generalized energy-efficient algorithms for the RFID estimation problem. *IEEE/ACM Transactions on Networking*, 20(6):1978-1990, 2012.
- [36] J. Liu, B. Xiao, S. Chen, F. Zhu, and L. Chen. Fast RFID grouping protocols. In *IEEE INFOCOM*, 2015.
- [37] X. Liu, K. Li, H. Qi, B. Xiao, and X. Xie. Fast counting the key tags in anonymous RFID systems. In *IEEE ICNP*, 2014.
- [38] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, 1967.
- [39] U. Maulik and S. Bandyopadhyay. Fuzzy partitioning using a real-coded variable length genetic algorithm for pixel classification. *IEEE Transactions on Geoscience and Remote Sensing*, 41(5):1075-1081, 2003.
- [40] V. Namboodiri and L. Gao. Energy-aware tag anticollision protocols for RFID systems. In *IEEE PerCom*, 2007.
- [41] L. Ni, Y. Liu, Y. Lau, and A. Patil. LANDMARC: indoor location sensing using active RFID. *ACM Wireless Networks*, 10(6):701-710, 2004.
- [42] K. Nikitopoulos, J. Zhou, B. Congdon, and K. Jamieson. Geosphere: consistently turning MIMO capacity into throughput. In *ACM SIGCOMM*, 2014.
- [43] J. Ou, Y. Zheng, and M. Li. MISC: Merging incorrect symbols using constellation diversity for 802.11 retransmission. In *IEEE INFOCOM*, 2014.
- [44] J. Ou, M. Li, and Y. Zheng. Come and be served: parallel decoding for COTS RFID tags. In *ACM MobiCom*, 2015.

- [45] M. Pakhira, S. Bandyopadhyay and U. Maulik. Validity index for crisp and fuzzy clusters. *Pattern Recognition*, 37(3):487-501, 2004.
- [46] M. Pakhira and A. Dutta. Finding number of clusters using VAT image, PBM index and genetic algorithms. In *ACM IITM*, 2010.
- [47] L. Pan and H. Wu. Smart trend-traversal: a low delay and energy tag arbitration protocol for large RFID systems. In *IEEE INFOCOM*, 2009.
- [48] D. Pauluzzi and N. Beaulieu. A comparison of SNR estimation techniques for the AWGN channel. *IEEE Transactions on Communications*, 48(10):1681-1691, 2000.
- [49] J. Posamentier. RFID tag clock synchronization. Patent US20070205871, 2006.
- [50] C. Qian, Y. Liu, H. Ngan, and L. Ni. ASAP: scalable identification and counting for contactless RFID systems. In *IEEE ICDCS*, 2010.
- [51] C. Qian, H. Ngan, and Y. Liu. Cardinality estimation for large-scale RFID systems. In *IEEE PerCom*, 2008.
- [52] J. Ryu, H. Lee, Y. Seok, T. Kwon, and Y. Choi. A hybrid query tree protocol for tag collision arbitration in RFID systems. In *IEEE ICC*, 2007.
- [53] M. Shahzad and A. Liu. Every bit counts - fast and scalable RFID estimation. In *ACM MobiCom*, 2012.
- [54] M. Shahzad and A. Liu. Probabilistic optimal tree hopping for RFID identification. In *ACM SIGMETRICS*, 2013.
- [55] D. Shen, G. Woo, D. Reed, and A. Lippman. Separation of multiple passive RFID signals using software defined radio. In *IEEE RFID*, 2009.
- [56] J. Vesanto and E. Alhoniemi. Clustering of the self-organizing map. *IEEE Transactions on Neural Networks*, 11(3):586-600, 2000.
- [57] J. Wang and D. Katabi. Dude, where's my card? RFID positioning that works with multipath and non-line of sight. In *ACM SIGCOMM*, 2013.

- [58] J. Wang, D. Vasisht, and D. Katabi. RF-IDraw: virtual touch screen in the air using RF signals. In *ACM SIGCOMM*, 2014.
- [59] J. Wang, F. Adib, R. Knepper, D. Katabi, and D. Rus. RF-Compass: robot object manipulation using RFIDs. In *ACM MobiCom*, 2013.
- [60] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk. Efficient and reliable low-power backscatter networks. In *ACM SIGCOMM*, 2012.
- [61] L. Wang, C. Leckie, K. Ramamohanarao, and J. Bezdek. Automatically determining the number of clusters in unlabeled data sets. *IEEE Transactions on Knowledge and Data Engineering*, 21(3):335-350, 2009.
- [62] R. Want. An introduction to RFID technology. *IEEE Pervasive Computing*, 5(1):25-33, 2005.
- [63] L. Yang, Y. Chen, X. Li, C. Xiao, M. Li, and Y. Liu. Tagoram: real-time tracking of mobile RFID tags to high precision using COTS devices. In *ACM MobiCom*, 2014.
- [64] H. Yue, C. Zhang, M. Pan, Y. Fang, and S. Chen. A time-efficient information collection protocol for large-scale RFID systems. In *IEEE INFOCOM*, 2012.
- [65] D. Zanetti, B. Danev, and S. Capkun. Physical-layer identification of UHF RFID tags. In *ACM MobiCom*, 2010.
- [66] P. Zhang, J. Gummesson, and D. Ganesan. BLINK: a high throughput link layer for backscatter communication. In *ACM MobiSys*, 2012.
- [67] R. Zhang, Y. Liu, Y. Zhang, and J. Sun. Fast identification of the missing tags in a large RFID system. In *IEEE SECON*, 2011.
- [68] B. Zhen, M. Kobayashi, and M. Shimizu. Framed ALOHA for multiple RFID objects identification. *IEICE Transactions on Communications*, 88(3):991-999, 2005.
- [69] Y. Zheng and M. Li. Fast tag searching protocol for large-scale RFID systems.

- In *IEEE ICNP*, 2011.
- [70] Y. Zheng and M. Li. P-MTI: Physical-layer missing tag identification via compressive sensing. In *IEEE INFOCOM*, 2013.
- [71] Y. Zheng and M. Li. Read bulk data from computational RFIDs. In *IEEE INFOCOM*, 2014.
- [72] Y. Zheng and M. Li. ZOE: fast cardinality estimation for large-scale RFID systems. In *IEEE INFOCOM*, 2013.
- [73] Y. Zheng, M. Li, and C. Qian. PET: probabilistic estimating tree for large-scale RFID estimation. In *IEEE ICDCS*, 2011.
- [74] H. Ding, L. , Z. Yang, J. Han, Z. Zhou, P. Yang, W. Xi, and J. Zhao. FEMO: a platform for free-weight exercise monitoring with RFIDs. In *ACM SenSys*, 2015.
- [75] L. Shangguan, Z. Zhou, X. Zheng, L. Yang, Y. Liu, and J. Han. ShopMiner: mining customer shopping behavior in physical clothing stores with COTS RFID devices. In *ACM SenSys*, 2015.
- [76] L. Shangguan and K. Jamieson. The design and implementation of a mobile RFID tag sorting robot. In *ACM MobiSys*, 2016.
- [77] T. Wei and X. Zhang. Gyro in the air: tracking 3D orientation of batteryless internet-of-things. In *ACM MobiCom*, 2016.
- [78] B. Kellogg, V. Talla, and S. Gollakota. Bringing gesture recognition to all devices. In *USENIX NSDI*, 2014.
- [79] Q. Pu, S. Gupta, S. Gollakota, and S. Patel. Whole-home gesture recognition using wireless signals. In *ACM MobiCom*, 2013.
- [80] Q. Lin, L. Yang, Y. Sun, T. Liu, X. Li, and Y. Liu. Beyond one-dollar mouse: a battery-free device for 3D human-computer interaction via RFID tags. In *IEEE INFOCOM*, 2015.
- [81] E. Miluzzo, A. Varshavsky, S. Balakrishnan, and R. Choudhury. TapPrints: your

REFERENCES

- finger taps have fingerprints. In *ACM MobiSys*, 2012.
- [82] M. Shahzad and A. Liu. Fairness matters: identification of active RFID tags with statistically guaranteed fairness. In *IEEE ICNP*, 2015.
- [83] A. Parate, M. Chiu, C. Chadowitz, D. Ganesan, E. Kalogerakis. RisQ: recognizing smoking gestures with inertial sensors on a wristband. In *ACM MobiSys* 2014.