EFFICIENT PROTOCOL DESIGN IN INFRASTRUCTURAL RFID SYSTEMS

By

WEN LUO

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2014

To my parents, my wife and my son

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor, Prof. Shigang Chen, for his great support, guidance and understanding throughout my graduate study. He is an incredible adviser, a passionate scientist, and a terrific person. Without his consistent support and encouragement that helped me to overcome many crisis situations, the work presented in this dissertation would never have existed. For everything you have done for me, Prof. Chen, I want to say thank you from the bottom of my heart.

My special thanks go to Prof. Sartaj Sahni, Prof. Yuguang Fang, Prof. Ahmed Helmy, Prof. Jih-Kwon Peir and Prof. Ye Xia for their advice and support during my study at University of Florida. I would also like to thank all the members in Prof. Chen's group for their help. They are Ming Zhang, Yan Qiao, Zhen Mo, Yian Zhou, Min Chen, and especially for Tao Li, who offers me a lot of suggestions and encouragements.

Finally, and most importantly, I am thankful to each member of my family: my parents, my parents-in-law, my brother, and especially my wife Bobbi Qiuli Lai and my son Peter Haoyu Luo. Thank you all for your support, understanding, encouragement, and love for so many years.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure page

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

EFFICIENT PROTOCOL DESIGN IN INFRASTRUCTURAL RFID SYSTEMS

By

Wen Luo

May 2014

Chair: Shigang Chen
Major: Computer Engineering

RFID (radio-frequency identification) tags are becoming ubiquitously available in warehouse management, object tracking and inventory control. Researchers have been actively studying RFID systems as an emerging pervasive computing platform, which helps create a multi-billion dollar market. Small RFID tags, each with a unique ID, are attached to objects, allowing an RFID reader to quickly access the properties of each individual object or collect statistical information about a large group of objects. Much of the existing work on RFID systems is to design tag identification protocols that read the IDs from tags. Other work designs efficient protocols to estimate the number of tags in a large RFID system, detects missing tags, or collects useful information.

The recent research trend is to improve time efficiency when designing an RFID solution because it is likely to be performed frequently. This is fine for passive tags - the kind used by Wal-Mart, which rely on radio waves emitted from an RFID reader to power their circuit and transmit information back through backscattering. Passive tags have short operational ranges, typically a few meters indoor, which seriously limits their applicability. For an application that covers a large area and involves a large number of tags, battery-powered active tags may be preferred for longer transmission ranges and richer on-tag hardware for more sophisticated functions. When battery is used, energy efficiency becomes a primary concern because it determines the lifetime of these tags. As an example, it may be ok to replace active tags once a couple years,

but it will become a serious burden if they have to be replaced weekly, due to protocol designs that do not take energy efficiency into consideration. Hence, it is also necessary to study the methods of designing energy-efficient RFID solutions and controlling the performance tradeoff between an application's execution time and its energy cost.

In this dissertation, we present two important and interesting problems, RFID missing tag detection and multigroup threshold-based classification. The former is used to detect the missing tag event within a preconfigured confidence interval, with respect to energy efficiency and time efficiency. The later focuses on using the minimum time to classify all above-threshold groups with a prescribed accuracy requirement.

CHAPTER 1
INTRODUCTION

1.1 RFID Technologies

Barcode [3] brought a revolution in the retail industry by speeding up the checkout process with a laser scanner that reads the product ID from barcode printed on each item and retrieves its price automatically from a database. However, there is a serious limiting factor: barcode can only be read in a very close range with direct sight, which makes it impossible to batch-access objects that are piled on store racks or in shopping carts. RFID (radio-frequency identification) technologies remove this limitation by integrating simple communication/storage/computation capacities in attachable tags, whose IDs can be read wirelessly over a distance, even when obstacles exist between tags and the RFID reader.

The RFID technologies promise to revolutionize future inventory management [35, 45, 52]. It is used to tag, identify and track individual items, cases and pallets as they move from the manufacturing floor through the supply chain and into the hands of the buyer or consumer. As the objects move through the supply chain, wireless RFID readers can communicate with an RFID tag on the object, collect information about the object (such as a unique number) and match that number in a database to access a complete record about the object. This real-time technology provides unprecedented speed and accuracy in the supply chain. For example, starting from August 1, 2010, Wal-Mart has begun to embed RFID tags in clothing [50]. If successful, these tags will be rolled out onto other product lines at Wal-Mar's more than 3,750 U.S. stores [5]. That is one step towards cashier-less checkout, where a customer pushes her shopping cart to pass an RFID reader at the checkout, where information in the embedded tags is automatically read and a receipt is printed out. Today, practical RFID systems exist for automatic toll payment, access control to parking garages, object tracking, theft prevention, etc.

RFID tags are attractive because of their simplicity, which places the technical challenge in RFID research. We cannot find any specification on how simple future RFID tags should be, but it is always preferred that a solution can achieve comparable efficiency with less hardware requirement - the simpler and cheaper, the better. Simplicity places constraint on the solution space, often making an otherwise easy problem difficult to solve. For example, information collection is not difficult in a classical wireless network, where each client implements routing/scheduling/MAC protocols [10, 11]. If the MAC protocol is CSMA/CA, the clients will be able to sense the channel and transmit their information when it is idle. In addition, they are able to detect collision and use random backoff to resolve it. Now, suppose we want to use RFID tags - which are deployed on products in a large warehouse, equipment in a hospital, or car plates in a city - for information collection. What if the hardware of tags does not support such an MAC protocol, let alone routing/scheduling protocols? What if their simple antenna cannot sense weak signal from peers for collision avoidance, let alone performing random backoff? Yet, how do we collect information from all tags with the best time and energy efficiency? That becomes a challenge. The same token goes for various other problems that this dissertation will investigate.

The basic technologies for RFID have been around for a long time. Its root can be traced back to an espionage device designed in 1945 by Leon Theremin of the Soviet Union, which retransmitted incident radio waves modulated with audio information. In the past, much research concentrated on two fronts: (1) physical-layer technologies for transmitting IDs from tags to an RFID reader more reliably, over a longer distance, and using less energy; (2) MAC-layer technologies for improving the rate at which a reader can collect IDs from tags. Arguably, these technologies would be sufficient if the only requirement from applications was to read tag IDs as efficiently as possible. However, the future of RFID will go far beyond this.

In recent years, a relatively small number of research groups, have been investigating novel ways in which future RFID systems can be used to solve practical problems [20–22, 25, 27, 28, 31, 37–40, 48, 51, 53–55, 58]. Of course, RFID tags may be embedded in library books, passports, driver licenses, car plates, medical products, etc. In the current application model, tags are treated as ID carriers and they are dealt with individually for the purpose of identifying the object that each tag is attached to. Now, if we make a paradigm shift from this individual view to a collective view, an array of new applications and interesting research problems will emerge. Consider a major distribution center of a large retailer, assuming it applies RFID tags to all its merchandise. These tags, which are pervasively deployed in the center, should not be treated just as ID carriers for individual objects. Collectively, they constitute a new infrastructure, which can be exploited for center-wide applications. If we take one step further, we can make this infrastructure more valuable by augmenting tags with miniaturized sensors [19], such that they report not only static ID information but also dynamic real-time information about their environment or conditions of the tags themselves. If we take another step to consider security or tag mobility, more applications and research problems open up. While these are beyond today's RFID systems, research should take the pioneer role to explore the possibilities and establish the fundamental methods for what are feasible.

1.2 Review of Existing Research in RFID Systems

Below we present a rich set of interesting researches to be explored in the field of RFID technologies. We classify these researches into different categories.

1) Size Measurement. The basic problem is to estimate the size of a system, i.e., the number of tags (or the number of tagged objects) in the system. The solution to this problem can be used to measure the inventory level in a warehouse if goods are tagged, or estimate the number of passengers in a subway train if the subway tickets are embedded with RFID tags. A closely related problem is to precisely determine, instead

of estimating, the number of tags. It is a more difficult operation and will take more time to complete.

Next, suppose objects in a large RFID system are organized into groups. For instance, in a shoe store, each group may correspond to a specific kind of shoes from a specific vendor with a specific property,e.g., certain size, type, or price. The group measurement problem is to estimate the number of tags (i.e., objects) in each group. Consider an arbitrary group. Let k be the actual number of tags in the group and $\hat{k}$ be the estimated number. We have the following accuracy requirement: The probability for k to be in the range $[(1-\beta)\hat{k}, (1+\beta)\hat{k}]$ is at least α, where α and β are two system parameters. A simpler problem is to identify the groups whose sizes are beyond a threshold. We may extend this problem with multiple thresholds, which essentially specify a sequence of consecutive ranges. The problem becomes classifying the groups into different ranges based on their sizes.

In a place where objects may be moved in or out, size measurement should be performed periodically. Between two consecutive measurements, we may want to know how many new objects are moved in, how many existing objects are moved out, and how many objects stay in the warehouse for the period between the measurements. For example, suppose two consecutive measurement results are 10,000 and 20,000 tags, respectively. Can we conclude that 10,000 new objects are moved in? No. In fact, the number of new objects may be any number x between 10,000 and 20,000, and the number of moved-out objects is $x - 10,000$. It is a difficult problem to find out x without keeping track of individual tags.

2) Anomaly Detection. An interesting application of RFID tags is to detect missing items in a large storage. Consider a major warehouse that keeps thousands of apparel, shoes, pallets, cases, appliances, electronics, etc. How to find out if anything is missing? We may have someone walk through the warehouse and count items. This is not only laborious but also error-prone, considering that clothes may be stacked together, goods

on racks may need a ladder to access, and they may be blocked behind columns. If we attach an RFID tag to each item, the whole detection process can be automated with one or multiple RFID readers communicating with tags to find out whether any tags (and their associated objects) are absent.

The problem of missing-tag detection has two versions: *exact detection and probabilistic detection*. The former is to identify exactly *which tags are missing*. The latter is to detect a *missing-tag event* with a certain predefined probability. Exact detection gives much stronger results, but its overhead will be greater than probabilistic detection.

The opposite of missing-tag detection is *unknown-tag detection*, where we want to know whether unexpected tags (their associated objects) are present in the system - for instance, a luggage in an airport is accidentally misplaced in a wrong luggage group. Similarly, it has two versions:*exact detection*, which finds out the exact IDs of the unknown tags, and *probabilistic detection*, which detects the presence of unknown tags with a certain pre-defined probability. When both missing tags and unknown tags occur, we have the mixed versions of these problems. In addition, there is a related problem called *misplacement detection*, which detects objects misplaced from their original placement to a different location. That may occur in a retail store where a customer picks up an item, decides later to not buy it, and drops it off at a different place.

3) Information Collection. The utility of RFID systems will be tremendously expanded if miniaturized sensors are incorporated into tags' circuit, enabling them to collect useful information in real time. A sensor may be designed to monitor the state of the tag itself, for instance, the residual energy of the battery. In another example, consider a large chilled food storage facility, where each food item is attached with an RFID tag that carries a thermal sensor. An RFID reader may periodically collect temperature readings from tags to check whether any area is too hot (or too cold), which may cause food spoil (or energy waste). The problem is how to efficiently collect

sensor-produced data from a large number of tags to an RFID reader, under the constraint of simple communication model adopted by RFID systems; see the System Model in this section for details.

A more general problem is to collect information from a subset of all tags. Consider the previous example of a chilled food storage. Because each area in the facility may be packed with many food items, the temperature readings from these close-by tags are highly redundant. Hence, it is not necessary for the reader to collect information from all tags in the system. The reader may select a subset of tags each time to collect temperature data. It is a harder problem to collect information from a subset of tags than from all tags because the reader has to make sure that tags that are not under query do not transmit - their transmissions will interfere with the transmissions made by tags of interest, particularly when the former outnumbers the latter by far.

Mobility may help information collection. Consider the scenario where car plates are integrated with simple RFID tags that can record a location value and a time value transmitted from an RFID reader and then transmit them back to another reader at a later time. Suppose RFID readers are deployed at chosen locations in a city and their clocks are well synchronized. In order to provide real-time measurement on the latency for traveling from one location x to other locations, the reader at location x may broadcast its location and the current time, which will be recorded by the passing cars. When these cars pass other locations, the readers there will request for that information and learn how long it takes a car to travel from location x to their places. If each tag can only carry one or a limited number of location-time values, the problem is how to coordinate amongst the readers so that they know which passing cars they should write their values and which cars they should only read values from, in order to achieve pre-specified city-wide accuracy requirements on latency measurement.

4) Privacy Preservation. Privacy-preserving authentication is a very important problem. In secure RFID systems, a reader will accept a tag's information only after

it authenticates the tag. A tag may be attached to a pharmaceutical product that carries patient information, a passport that carries a person's identification, or a commercial product that carries information about manufactured date, expiring date, ingredients, etc. Some applications require *privacy-preserving* authentication, in which a tag should not give out any identifying information during authentication process. Suppose a police tries to use a mobile RFID reader to authenticate a driver's license embedded with an RFID tag, and the reader has access to a database of all secret keys that are pre-installed in driver licenses. The reader has to know which key it should use to perform authentication. In a typical authentication protocol, the license transmits an identifying number to the reader, which will use that number to search the database for the right key. However, this leads to a security loophole. The identifying number, transmitted wirelessly, reveals the presence of the carrier. Fake readers may initiate the authentication process at chosen locations and chosen times. They will terminate the process after the identifying number is received. It allows them to monitor the whereabout of the license's carrier. Hence, a research problem is to design an authentication protocol that allows an RFID reader to authenticate a tag without requiring the tag to transmit any identifying data. Ideally, the information transmitted from a tag looks totally different and random each time the tag is authenticated.

Next, we move to privacy-preserving information collection. We have discussed that future car plates may be integrated with RFID tags to enable information collection from moving vehicles. If RFID readers are deployed at intersections, they can collect the tag IDs of passing vehicles. From these collected IDs, we will know not only ”point” trafc volumes at intersections but also ”point-to-point” volumes between any two locations - there must a vehicle traveling between them if both locations record a common tag ID. However, this leads to serious privacy breaching: For each registered vehicle, as its ID is recorded at each intersection that it passes, its entire moving history is revealed, which is against the ”anonymity by design” principle for privacy protection required by IntelliDrive, formerly known as Vehicle Infrastructure Integration or VII, an initiative from US Department of Transportation (USDOT). Hence, the challenge is to allow the collection of ”statistical” point-to-point information, yet protect information about each ”individual” vehicle.

1.3 Scope of Research

In previous section, we have shown that there are numerous problems and solutions in RFID systems, out of which two problems are very interesting, relatively new and under exploration, and hence have a great potential for improvement. These two problems are described as follows:

1) Consider a warehouse that stores a large number of commercial products or a military base that stockpiles a large quantity of guns and ammunition packages. Suppose each object (e.g., a microwave oven, a rifle, or a bullet magazine) is attached with a RFID tag, which is able to communicate with RFID readers. As objects move in and out of the warehouse, a small-range reader at a door will automatically collect the IDs of the new tags that become present and remove the IDs of the tags that are moved out [44]. Now, if some objects are stolen from the warehouse, how to timely detect such events? Without any automatic tools, we have to resort to manual inventory walk-through, which is laborious, expensive, and slow. Such operations cannot be performed frequently, and hence will not help us detect the theft in time in order to catch the thieves. However, if RFID tags are installed, the missing-tag detection can be automated and performed frequently so that any major theft is swiftly reported.

There are two different missing-tag detection problems: *exact detection* and *probabilistic detection*. The objective of exact detection is to identify exactly *which tags are missing*. The objective of probabilistic detection is to detect *a missing-tag event* with a certain predefined probability. An exact detection protocol [24, 42, 44, 56] gives much stronger results, but its overhead is far greater than a probabilistic detection protocol [15, 29, 36, 46]. Hence, they both have their values. In fact, they are complementary to each other, and should be used together. For example, a probabilistic detection protocol may be scheduled to execute frequently, e.g., once every minute, in order to timely catch any loss event such as theft. Once it detects some tags are missing, it may invoke an exact detection protocol to pinpoint which tags are missing. If one execution of a probabilistic detection protocol detects a missing-tag event with 99% probability, five independent executions will detect the event with 99.99999999% probability. If that is not enough, we may schedule an exact detection protocol [24] every five times the probabilistic detection protocol is executed.[1] . In this dissertation, we only consider probabilistic detection.

2) Next, We investigate a different problem. In practice, tags are often attached to objects belonging to different groups, for instance, different brands of shoes in a large shoe store, different titles of books in a bookstore, and goods from different countries or manufacturers in a port. One challenge is to determine whether the number of tags in each group is above or below a prescribed threshold value. The threshold may be set high to identify the populous groups, it may be set to a level that triggers certain actions such as replenishing the stocks, or even multiple thresholds can be used to classify groups based on the range of their population sizes. Solving this *multigroup threshold-based classification* problem gives us a basic tool to access a large population of numerous groups.

1.4 Missing-Tag Detection and Energy-Time Tradeoff in Large-Scale RFID Systems With Unreliable Channels

This work focuses on a particularly challenging problem, the detection of missing tag event in large-scale RFID system. The basic detection method is introduced in the pioneer work [46]: A RFID reader monitors a time frame of f slots. Through a hash

[1] Another exact detection protocol [44] reports which tags are missing, but does not guarantee to report all missing ones. The protocol in [24] guarantees 100% reporting.

function, each tag pseudo-randomly selects a slot in the time frame to transmit. The reader can predict in which slot each known tag will transmit. It detects a missing-tag event if no tag transmits during a slot when there is supposed to be tag(s) transmitting. However, multiple tags may select the same slot to transmit. If a tag is missing, its slot may be kept busy by transmission from another tag. Consequently, the reader cannot guarantee the detection of a missing-tag event.

Furthermore, we observe that time efficiency should not be the sole performance consideration when designing a missing-tag detection protocol. The energy cost may be an even more critical concern if active tags are used. Due to limited operational distances, passive tags are mostly used for small-range applications such as fast checkout. For future automatic inventory management functions that cover a very large area, active tags are likely to be the choice. Active tags use their own power to transmit. A longer range can be achieved by transmitting at a higher power. They are also richer in resources for implementing advanced functions. Their price becomes less of a concern if they are used for expensive merchandizes (such as refrigerators) or reused many times as goods moving in and out of the warehouse. But active tags also have a problem. They are powered by batteries. Recharging batteries for tens of thousands of tags is a laborious operation, considering that the tagged products may stack up, making tags not easily accessible. To prolong the tags' lifetime and reduce the frequency of battery recharge, all functions that involve large-scale transmission by many tags should be made energy-efficient. The protocol in [46] only considers time efficiency, but not energy efficiency. A follow-up work [29] further improves the time efficiency. Firner et.al.[15] design a simple communication protocol, Uni-HB, to detect missing items for fail-safe presence assurance systems and demonstrate it can lead to longer system lifetime and higher communication reliability than several popular protocols. The protocol however does not consider time efficiency and requires all tags to participate and transmit, which will be less efficiency than a sampling-based protocol design that requires only a small fraction of the tags to participate. Similarly, the method in [36] also requires all tags to participate.

The prior work has studied energy-efficient protocols for estimating the number of tags in a RFID system [26], or energy-efficient anti-collision protocols that minimize the energy consumption of a mobile reader when the reader is used to collect the IDs of the tags [20, 34]. We believe this dissertation is the first to study energy-efficient solutions for the missing-tag detection problem.

In addition, much of the existing literature assumes that the communication channel between a RFID reader and tags is reliable, which means that information transmitted is never corrupted. However, in reality, errors may occur due to low signal strength, noise, or interference in the environment. The occurrence of errors usually follows a certain distribution, which is characterized by an error model, describing the statistical properties of underlying error sequences.

To solve the missing tag detection problem, first, we propose two new, more sophisticated protocol designs for missing-tag detection. They take both energy efficiency and time efficiency into consideration. The first one is called EMD, which works in a similar way to The protocol in [46], except that tags are sampled with a small probability. In the second protocol, by introducing multiple hash seeds, our new design provides multiple degrees of freedom for tags to choose which slots they will transmit in. This design drastically reduces the chance of collision, and consequently achieves multiple-fold reduction in both energy cost and execution time. In some cases, the reduction is more than an order of magnitude. Second, with the new designs, we reveal a fundamental energy-time tradeoff in missing-tag detection. Our analysis shows that better energy efficiency can be achieved at the expense of longer execution time, and vice versa. The performance tradeoff can be easily controlled by a couple of system parameters. Through our analytical framework for energy-time tradeoff, we are able to compute the optimal parameter settings that achieve the smallest protocol execution time or the smallest energy cost. The framework also enables us to solve the energy-constrained least-time problem and the time-constrained least-energy problem in missing-tag detection. Third, we extend our protocol design to consider channel error under two different error models. Our protocols can be configured to work under these error conditions.

1.5 An Efficient Protocol for RFID Multigroup Threshold-Based Classification

Most existing work adopts a “flat” RFID system model and performs functions of collecting tag IDs, estimating the number of tags, or detecting the missing tags. However, in practice, tags are often attached to objects of different groups, which may represent a different product type in a warehouse, a different book category in a library, etc. An interesting problem, called *multigroup threshold-based classification*, is to determine whether the number of objects in each group is above or below a prescribed threshold value. Solving this problem is important for inventory tracking applications.

Precise classification requires us to know the precise number of tags in each group. Tag identification protocols [8, 12, 17, 20, 32, 34, 49, 54] can do that, but it takes them significant time to complete if the number of tags is very large. One way to improve efficiency is relaxing the problem from accurate classification to approximate classification [45], where the classification accuracy can be tuned to meet a pre-defined requirement. We may use cardinality estimation protocols [21, 22, 37, 49, 54] to estimate the number of tags in each group, and classify the group based on the estimation. However, those protocols are efficient when estimating a small number of large groups, but they are not efficient when estimating a large number of small groups, because their execution time for each group is largely indifferent in group size, as we will demonstrate shortly. In [45], Sheng et al. apply group testing to approximately detect popular groups. When the number of groups above the threshold is small, their performance is good. However, the performance of the group-testing-based solution degrades quickly (in terms of the execution time) when the number of groups above threshold becomes large.

In this dissertation, we propose a new classification protocol that is scalable to a large number of groups. Its design is drastically different from traditional approaches that measure the size of one group at a time. It measures the sizes of all groups together at once in a mixed fashion. Yet, the new protocol is able to perform threshold-based classification with an accuracy that can be pre-set to any desirable level, allowing tradeoff between time efficiency and accuracy. Our main contributions are summarized as follows:

1. We design a new protocol for threshold-based classification in a multi-group RFID system based on *tag sampling* and *logical bitmaps*, which share time slots uniformly at random among all groups during the process of measuring their populations. We use the maximum likelihood estimation method to extract per-group information from the shared slots. Such slot sharing greatly reduces the amount of time it takes to complete classification. Sampling further improve the performance of the protocol significantly.

2. Given an accuracy requirement, we show analytically how to compute optimal system parameters that minimize the protocol execution time under the constraint of the requirement. Our estimation method based on sampling and logical bitmaps ensures that false positive/false negative ratios are bounded, where *false positive* occurs when a below-threshold group is reported as above-threshold and *false negative* occurs when an above-threshold group is not reported.

3. We comprehensively evaluate the proposed solution and compare it with existing protocols. Our simulation results match well with the analytical results, which demonstrate that the new protocol performs far better in terms of execution time than the best existing work.

1.6 Outline of The Dissertation

The rest of the dissertation is organized as follows: Chapter 2 presents the system models and various performance metrics considered in RFID system. Chapter 3 is the related work of Missing tag detection and RFID multigroup threshold-based classification. Chapter 4 presents an RFID missing-tag detection protocol in large-scale RFID Systems with unreliable channels, and reveal a fundamental energy-time tradeoff. In this section, we propose a novel protocol design that considers both energy efficiency and time efficiency. It achieves multi-fold reduction in both energy cost and execution time when comparing with the best existing work. Chapter 5 proposes an iterative protocol for threshold-based classification in a multi-group RFID system based on *logical bitmaps* that share time slots uniformly at random among all groups during the process of measuring their populations.Chapter 6 presents the implementation of the simple version of our protocols. Chapter 7 draws the conclusions and briefly discusses the potential future work.

CHAPTER 2
SYSTEM MODELS AND PERFORMANCE METRICS

2.1 System Models

There are three types of RFID tags. Passive tags are most widely deployed today. They are cheap, but do not have internal power sources. Passive tags rely on radio waves emitted from an RFID reader to power their circuit and transmit information back to the reader through backscattering. They have short operational ranges, typically a few meters in an indoor environment. To cover a large area, arrays of RFID reader antennas must be installed. Semi-passive tags carry batteries to power their circuit, but still rely on backscattering to transmit information. Active tags use their own battery power to transmit, and consequently do not need any energy supply from the reader. Active tags operate at a much longer distance, making them particularly suitable for applications that cover a large area, where one or a few RFID readers are installed to access all tagged objects and perform management functions automatically. With richer onboard resources, active tags are likely to gain more popularity in the future, particularly when their prices drop over time as manufactural technologies are improved and markets are expanded.

Communications between the reader and the tags are time-slotted. The reader's signal will synchronize the clocks of the tags. In some protocols, the communication is driven by the reader in a request-and-response pattern. The reader issues a request, which is followed by a time frame consisting of f slots, during which the tags may respond by transmitting some information.

A slot is said to be *empty* if no tag responds (transmits) in the slot. It is called a *singleton slot* if exactly one tag responds. It is a *collision slot* if more than one tag responds. A singleton or collision slot is also called a *busy slot*. The EPCglobal Gen-2 standard [14] requires a slot length of 10 bits in order to distinguish singleton slots from collision slots. We define the length of long-response slot as T_{long} so that hte reader can tell whether there is collision in a slot. On the contrary, one bit is enough if we only need to distinguish empty slots from busy slots — ‘0’ means empty and ‘1’ means busy. Hence, tag responses will be much shorter (or consume much less energy) if a protocol only needs to know empty/busy slots, like the one in this dissertation does.

A frame takes $f \times T_{short}$ time, where T_{short} is the time of a slot. When f is large, the time it takes the RFID reader to transmit its request, which is a small constant, can be ignored. For time efficiency, we should minimize the frame size f, subject to the detection requirement in Section 4.1.2. For energy efficiency, we should minimize the total number of responses from all tags. Because there are empty slots and collision slots, the number of responses is not the same as the number of time slots in the frame. As we will see later, reducing the number of responses may require us to increase the number of time slots in order to meet the detection requirement.

Let T_{tag} be the time it takes to transmit a tag ID. Obviously, $T_{tag} > T_{short}$ because it takes a longer time to transmit multiple bits in an ID than one-bit information in a tag response. Based on the specification of the EPCglobal Gen-2 standard [14], we determine that $T_{tag} = 2609.76\ \mu s$ for a 96-bit tag ID and $T_{short} = 290.81\ \mu s$, after the required waiting times (e.g., gap of idle time between transmissions) are included.

Significant asymmetry exists between readers and tags: Unlike tags, the cost for a reader is not a serious concern because it is not needed in large quantities; oftentimes, only one is needed. Therefore, unlike tags, the reader is not limited in storage space, computation power, or energy supply. If necessary, it can be connected to a powerful server for resources. With a high-quality antenna, a reader is able to receive weak signals from tags. With low-quality antennas, although tags can receive strong signals from the reader, they cannot receive each other’s weak signals. They may not even sense whether the channel is busy or idle, i.e., whether another tag is transmitting. Nor can they sense if collision has occurred when two tags transmit simultaneously. But the reader can detect whether the channel is idle or whether collision occurs. Such asymmetry points out a design principle that we should follow: pushing the complexity to the reader while leaving the tags simple.

2.2 Performance Metrics in RFID Systems

There exist four important performance metrics in the protocol design of a RFID system.

2.2.1 Performance Metric 1: Protocol Execution Time

Imagine that a large retailer has a warehouse in its distribution center that regularly stores tens of thousands of electronics, furniture, apparel, shoes, pallets, cases, etc. A missing-tag detection protocol is expected to be executed frequently (e.g., once every 15 minutes) in order to timely raise alarms upon unexpected removal of objects from the warehouse. However, false detection of missing tags may occur if normal operations remove objects during the time when the protocol is executing. In a busy warehouse, as goods constantly move in and out, false alarms may happen even if the protocol's execution time is a number of seconds. Hence, it is highly desirable that the execution time is kept as small as possible in order to minimize the disruption to normal inventory operations. (Our work is able to reduce the execution time to 1 second or less under certain parameter settings. In such a small duration, it is extremely unlikely that some tags are removed by normal operations that are mechanical in nature.)

The execution time is affected by two major factors. The first factor is how stringent the system requirement is. A protocol such as [46] will take far more time to detect one missing tag with 99.9% probability than to detect 100 missing tags with 95%. Hence, in order to control the protocol's execution time, a practical system may be configured with $m = 100$ and $\alpha = 95\%$. It means that a single protocol execution will detect the missing-tag event with 95% if $m = 100$. Because the protocol is executed periodically, after the ith execution, the detection probability becomes $1 - (1 - 95\%)^i$, which rapidly approaches to 100% when i becomes large. Even if the number of missing tags is smaller than m and thus the detection probability of one protocol execution is smaller than α, the missing-tag event will eventually be detected after a sufficient number of executions.

The second factor that has a major impact on execution time is the protocol design. In this dissertation, we show that a better protocol design can reduce the execution time considerably without any significant increase in the complexity of the online operations.

2.2.2 Performance Metric 2: Energy Cost

In order to support advanced management functions that cover a large area, battery-powered active tags are a better choice because they have much longer transmission ranges. If passive tags were used, one would have to take the RFID reader and move around the whole area, collecting information from location to location, or else a dense reader array has to be installed to extend the coverage. Active tags allow one or a few readers to collect information from a large area.

When active tags are used, we must conserve their battery power in order to prolong the tags' lifetime before they have to be recharged. The tagged goods (such as apparel) may stack in piles, and there may be obstacles, such as racks filled with merchandize, between a tag and the reader. We expect the active tags are designed to transmit with significant power that is high enough to ensure reliable information delivery in such a demanding environment. The energy consumed by the RFID reader is of less concern because its batteries can be easily recharged or it may even use an external power source. We assume that the reader transmits at sufficiently high power.

2.2.3 Performance Metric 3: Storage Capacity

It is known that a RFID tag is limited in memory, from 512 bits to 128 kilobytes [6], which is preloaded with a tag ID and other necessary information to identify itself, and hence leaves less memory for protocols installed in the tag. To make the matter even more challenging, some protocols may require to be initialized with a bunch of authentication keys or a great volume of memory to store the intermediate data. In that case, those protocol may have to be installed in some special tags with large memory that is much expensive, which seriously limits their applicability. Hence, our RFID protocol design should be as memory-efficient as possible.

2.2.4 Performance Metric 4: Computation Complexity

The computation complexity is also a very important factor with regards to computation-constraint RFID tags. The limitation of lacking necessary computational resources to support strong cryptographic authentication scheme makes securing RFID system a very challenging task. Therefore, when designing different protocols, we should take good care of the computation part such that the computational price should be low or the major computation work lies on the reader.

CHAPTER 3
RELATED WORK

3.1 Related Work 1: RFID Anti-Collision Protocols

Most existing work on RFID systems is to design the tag-collection protocols, that read the IDs from the tags. They mainly fall into two categories. One is *tree-based* [2, 4, 7, 33, 60] and the other is *ALOHA-based* [9, 23, 41, 47, 49, 57]. The *tree-based protocols* organize all IDs in a tree of ID prefixes [4, 33, 60]. Each in-tree prefix has two child nodes that have one additional bit, '0' or '1'. The tag IDs are leaves of the tree. The RFID reader walks through the tree. As it reaches an in-tree node, it queries for tags with the prefix represented by the node. When multiple tags match the prefix, they will all respond and cause collision. Then the reader moves to a child node by extending the prefix with one more bit. If zero or one tag responds (in the one-tag case, the reader receives an ID), it moves up in the tree and follows the next branch. Another type of tree-based protocols tries to balance the tree by letting the tags randomly pick which branches they belong to [2, 7, 33]. In the best case, when the tree is completely balanced, the reader takes two (or three, depending on the actual design) queries on average for discovering each ID. However, it will incur much more overhead for the average and worst cases.

The *ALOHA-based protocols* work as follows. The reader first broadcasts a query request. Each tag chooses a time slot to transmit its ID. If a tag selects a slot that none of other tags select, it can be successfully identified and will keep silent for the rest of the process. If multiple tags transmit simultaneously, the responses are garbled due to collision and retransmissions are required. The process terminates when all the tags are successfully identified. The enhanced dynamic framed slotted ALOHA (EDFSA) [23] increases the identification probability by adjusting the frame size and restricting the number of responding tags in the frame.

3.2 Related Work 2: Trusted Reader Protocol (TRP)

The most related work we find in the literature is the Trusted Reader Protocol (TRP) by Tan, Sheng and Li [46]. For security reasons, their system consists of a server and a RFID reader. The former stores the tag IDs and performs the computation, while the latter communicates with the tags. We assume the reader is trusted. To simplify the protocol description, we logically combine the server and the reader into a single entity, still called the reader.

A design goal of TRP is to reduce the time of the detection process. To initiate the execution of the protocol, a RFID reader broadcasts a detection reques, asking the tags to respond in a time frame of f slots. The detection request has two parameters, the frame size f and a random number r. Each tag maps itself to a slot in the frame by hashing its ID and r. It then transmits during that slot. The reader records which slots are busy and which are empty. This is binary information where each slot carries either '1' or '0'. Multiple readers may be used to extend the coverage. In this case, all readers will simultaneously monitor the time slots in the frame. A slot is considered to be busy if any reader records that the slot is busy.

Because the reader knows the IDs of all tags, it knows which tags are mapped to which slots. More specifically, it knows which slots are expected to be singletons, i.e., one and only one tag is mapped to each of them. If an expected singleton slot turns out to be empty, the tag that is mapped to this slot must be missing. Not all tags are mapped to singleton slots. When two or more tags are mapped to the same slot (a collision slot), if only one of the tags is missing, the slot will remain busy and thus the missing-tag event will not be detected.

Obviously, if we increase the frame size, collision will be less likely and there will be a larger number of singleton slots, which means the probability for any tag to map to a singleton is greater. In the event of missing tags, the probability that an expected singleton turns out to be empty is also greater, and hence the probability of detecting the missing-tag event is greater. The requirement is that we must detect the missing-tag event with a probability of at least α if m or more tags are missing. TRP is designed to minimize its execution time by using the smallest frame size that ensures a detection probability of α.

A serious limitation of TRP is that it only considers time efficiency. It is not energy-efficient because all tags must be active and transmit during the time frame. B. Firner et al. [15] consider energy cost, but their protocol requires all tags to participate and transmit, which will be less efficient than a sampling-based solution where only a small fraction of tags participate.

In this dissertation, we show TRP are special cases of a much broader protocol design space. Not only are there protocol configurations that perform much better than TRP in terms of both time and energy efficiencies, but we also reveal a fundamental energy-time tradeoff in this design space, which allows us to adapt protocol performance to suit various needs in practical systems.

The recent work in [24] also requires all tags to transmit, and it takes much longer time than TRP. However, it solves a different problem by exactly identifying which tags are missing. Hence, the protocol in [24] complements TRP and our work in this dissertation. The cheaper protocol of TRP or our protocol can be executed frequently to identify the missing-tag event. Once such an event is detected, the expensive protocol in [24] can be executed to find which tags are missing.

3.3 Related Work 3: Cardinality Estimation Protocols and Group Testing

Sheng, Tan and Li studied the multigroup threshold-based classification problem in [45]. They begin with a simple threshold checking scheme (TCS) to approximately answers whether the number of tags exceeds a threshold. Based on TCS, they propose two probabilistic protocols. The first one is based on generic group testing (GT), which consists of multiple rounds. In each round, the reader shuffles all groups into different categories, each of which may contain tags from multiple groups. TCS is then applied to check the number of tags in each category. The categories with sufficient tags are labeled as potential populous categories, which may include above-threshold groups. In the end, the testing history is used to classify all above-threshold groups. The second protocol is a combination of group testing and divide-and-conquer, which ignores the categories that fail to pass the TCS tests in the previous round, divides the remaining categories into multiple sub-categories, and applies TCS to each sub-categories in the remaining rounds.

Another possible solution for the multigroup threshold-based classification problem is to use a reader to collect the actual tag IDs from tags [8, 12, 17, 20, 32, 34, 49, 54], where each ID contains bits that identify the group of the tag. Applying to the problem in this dissertation, these ID-collection protocols do not work well for large-scale RFID systems due to their long identification time.

Many methods were proposed to estimate the whole population of an RFID system. They are essentially single-group estimators. We can use them to first estimate individual group sizes (one group at a time) and then use the sizes for classification purpose. Kodialam and Nandagopal propose the first set of single-group estimators, including the Zero Estimator (ZE), the Collision Estimator (CE), and the Unified Probabilistic Estimator (UPE), which collect information from tags in a series of time frames and estimates the whole population of tags in the system based on the number of empty slots and/or the number of collision slots [21]. A follow-up work by the same authors proposes the Enhanced Zero-Based Estimator (EZB) [22], which is an asymptotically unbiased estimator and makes estimation only based on the number of empty slots. Qian et al. provide a replicate-insensitive estimation algorithm called the Lottery-Frame scheme (LoF) [37]. The Enhanced First Non-Empty slots Based Estimator (Enhanced FNEB) [16] can be used to estimate tag population in both static and dynamic environments by measuring the position of the first non-empty slot in each frame. Li et al. [26] study the estimation problem for large-scale RFID systems from the energy angle based on a Enhanced Maximum Likelihood Estimation Algorithm (EMLEA). They design several energy-efficient probabilistic algorithms that iteratively refine a control parameter to optimize the information carried in the transmissions from tags, such that both the number and the size of the transmissions are minimized. The Average Run based Tag estimation (ART) scheme [43] further reduces the execution time for population estimation, based on the average run length of ones in the bit string received in the standardized frame-slotted Aloha protocol. Finally, the Zero-One Estimator (ZOE) [59] provides fast and reliable cardinality by tuning the system parameters and converging to the optimal settings through a bisection search.

CHAPTER 4
MISSING-TAG DETECTION AND ENERGY-TIME TRADEOFF IN LARGE-SCALE RFID SYSTEMS WITH UNRELIABLE CHANNELS

RFID (radio frequency identification) technologies are poised to revolutionize retail, warehouse and supply chain management. One of their interesting applications is to automatically detect missing tags in a large storage space, which may have to be performed frequently to catch any missing event such as theft in time. Because RFID systems typically work under low-rate channels, past research has focused on reducing execution time of a detection protocol to prevent excessively-long protocol execution from interfering normal inventory operations. However, when active tags are used for a large spatial coverage, energy efficiency becomes critical in prolonging the lifetime of these battery-powered tags. Furthermore, much of existing literature assumes that the channel between a reader and tags is reliable, which is not always true in reality because of noise/interference in the environment. Given these concerns, this dissertation makes three contributions: First, we propose a novel protocol design that considers both energy efficiency and time efficiency. It achieves multi-fold reduction in both energy cost and execution time when comparing with the best existing work. Second, we reveal a fundamental energy-time tradeoff in missing-tag detection, which can be flexibly controlled through a couple of system parameters in order to achieve desirable performance. Third, we extend our protocol design to consider channel error under two different models. We find that energy/time cost will be higher in unreliable channel conditions, but the energy-time tradeoff relation persists.

The rest of this chapter is organized as follows: Section 4.1 gives the system model and problem definition, as well as the prior art. Section 4.2 presents an intermediate protocol that can solve the missing tag detection problem. Section 4.3 is a more efficient protocol called EMD. Section 4.4 proposes our final missing-tag detection protocol. Section 4.5 investigates energy-time tradeoff in protocol configuration. Section 4.6

extends the protocol under two different error models. Section 4.7 evaluates the protocol through simulations. Section 4.8 draws the conclusion.

4.1 Preliminaries

4.1.1 System Model

There are three types of RFID tags. *Passive tags* are most widely deployed today. They are cheap, but do not have internal power sources. Passive tags rely on radio waves emitted from a RFID reader to power their circuit and transmit information back to the reader through backscattering. They have short operational ranges, typically a few meters in an indoor environment, which seriously limits their applicability. *Semi-passive tags* carry batteries to power their circuit, but still rely on backscattering to transmit information. *Active tags* use their own battery power to transmit, and consequently do not need any energy supply from the reader. Active tags operate at a much longer distance, making them particularly suitable for applications that cover a large area, where one or a few RFID readers are installed to access all tagged objects and perform management functions automatically. With richer on-board resources, active tags are likely to gain more popularity in the future, when their prices drop over time as manufacture technologies are improved and markets are expanded. They are particularly attractive for high-valued objects such as luxury bags, laptops, cell phones, TVs, etc., or when the tags are reused over and over again.

Communication between a reader and tags is time-slotted. The reader's signal synchronizes the clocks of tags. There are different types of time slots [24], among which two types are of interest in this disseartation. The first type is called a *tag-ID slot*, whose length is denoted as T_{tag}, during which a reader is able to broadcast a tag ID. The second type is called a *short-response slot*, whose length is denoted as T_{short}, during which a tag is able to transmit one-bit information to the reader, for instance, announcing its presence.

In this dissertation, we consider active tags. Beside communicating with a reader, we assume the tags have the following capability: performing a hash function, carrying a small internal storage to keep a few parameters and a 96-bit seed-selection segment from the reader, being able to check the values in the segment, and having a clock that enables a tag to transmit at a specific slot of a time frame or wait up at a pre-scheduled time.

4.1.2 Missing-Tag Detection Problem

The problem is to design an efficient protocol for a RFID reader to detect whether some tags are missing, subject to a *detection requirement*: A single execution of the protocol should detect a missing-tag event with probability α if m or more tags are missing, where α and m are two system parameters. For example, consider a big shoe store that carries tens of thousands of shoes, and we may set the parameters to be $\alpha = 99\%$ and $m = 10$, so that one execution of the protocol will detect any event of missing 10 or more shoes with 99% probability. If we perform independent executions of the protocol periodically, the detection probability of any missing event will approach to 100%, no matter what the values of α and m are. Furthermore, as we have explained in the introduction, a low-overhead probabilistic detection protocol may be used in conjunction with a high-overhead exact detection protocol (which is scheduled much less frequently) to catch any miss.

We assume that the RFID reader has access to a database that stores the IDs of all tags. This assumption is necessary [46]. Without any prior knowledge of a tag's existence, how can we know that it is missing? The assumption can be easily satisfied if the tag IDs are read into a database when new objects are moved into the system, and they are removed from the database when the objects are moved out — this is what a typical inventory management procedure will do. Even if such information is lost due to a database failure, we can recover the information by executing an ID-collection protocol [4, 18, 33, 41, 57] that reads the IDs from the tags. In this case, we will not

Table 4-1. Notations

Symbols	Descriptions
n	number of RFID tags in the system
m	number of missing tags
α	probability of detecting the missing-tag event
f	number of slots in a time frame
r	random number
T_{tag}	time for transmitting a tag ID
T_{short}	time for transmitting one-bit information
k	number of hash seeds
P_{msmd}	probability for MSMD to detect a missing-tag event
P_{emd}	probability for EMD to detect a missing-tag event
p	sampling probability in MSMD or EMD
$f^*(p)$	frame size that minimizes the execution time under a given sampling probability p
f_{opt}	optimal frame size that achieves the smallest execution time, $f_{opt}=\min_{p}\{f^*(p)\}$
p_t	sampling probability under which f_{opt} is achieved, i.e., $f_{opt} = f^*(p_t)$
p_{opt}	optimal sampling probability that minimizes the energy cost

detect missing-tag events that have already happened. However, once we have the IDs of the remaining tags, we can detect the missing-tag events after this point of time.

Notations (most of which are introduced later) are summarized in Table 4-1 for quick reference.

4.1.3 Performance Metrics

We consider two performance metrics, *execution time* of the protocol and *energy cost* to the tags. First, RFID systems use low-rate communication channels. Low rates, coupled with a large number of tags, often take RFID protocols long time to finish their operations. Hence, in order to apply such protocols in a busy warehouse environment, it is desirable to adopt novel designs to reduce execution time as much as possible.

Second, active tags carry limited battery power. Replacing tags is a tedious, manual operation. One way of saving energy is to minimize the number of tags that are needed to participate in each protocol execution. When a tag participates in a protocol execution, it has to power its circuit during the execution, receive request information from the reader, and transmit back. When a tag does not participate, it goes into the sleep mode and incurs insignificant energy expenditure.

The energy cost to the RFID reader is less of a concern because the reader's battery can be easily replaced or it may be powered by an external source.

4.1.4 Clock Synchronization

For any missing-tag detection protocol that is scheduled to execute at fixed time intervals, there is a need to synchronize the clocks of the tags so that they can wake up at the right moments. To achieve very low energy consumption during sleep, these active RFID tags may use low-power RC oscillators as clock sources, which however have relatively large drift. The drift will become significant if the clock is left unsynchronized for an extended period of time. Following the argument in the introduction, we expect the missing-tag detection protocols to run frequently. One solution to deal with the drift problem is to calibrate all tags' clocks at the beginning of each scheduled protocol execution in order to keep them synchronized. The tags that are not supposed to participate in a round of execution will go back to sleep after clock synchronization. It will add a fixed amount of energy expenditure to all missing-tag detection protocols that require the reader to pull information from tags at fixed time intervals. Because synchronization overhead is common to all such protocols, we will not include it in performance comparison. Also note that this overhead is relatively small when comparing with the energy cost needed to power a tag for receiving, transmitting, and computing in the entire duration of a protocol execution. Another solution is only letting the participating tags wake up, but they need to wake up a little earlier than scheduled to compensate for the clock drift, such that they can receive the requests from the reader. This approach also incurs additional energy overhead because tags have to be powered a little longer for receiving.

The energy overhead for clock synchronization does not exist for a push-based missing-item detection protocol such as Uni-HB [15], where every sensor (attached to an item) transmits its ID and a sequence number to a base station in each epoch. In order to lower the collision probability, Uni-HB spreads sensor transmissions in each epoch, which means the protocol execution continues over each epoch since the IDs may be sent by the sensors at any time. In the introduction, however, we have argued for short protocol execution time to avoid interference from busy warehouse operations that move items in and out. Furthermore, Uni-HB requires all tags to participate by sending their IDs to the base station in each epoch, whereas we prefer an approach that involves only a fraction of tags to save energy. For these reasons, Uni-HB is not suitable for meeting the requirements in this dissertation.

4.1.5 Prior Work

We first describe the Trusted Reader Protocol (TRP) by Tan, Sheng and Li [46]. Given a time frame of f slots, the RFID reader maps each tag to a slot in the frame by hashing its ID and a random number r. After the reader maps all tags to the slots, it classifies slots into three categories: A slot is said to be *empty* if no tag is mapped to the slot. It is called a *singleton slot* if exactly one tag is mapped to the slot. It is a *collision slot* if more than one tag is mapped to the slot. Because the reader knows the IDs of all tags, it knows which tags are mapped to which slots. It knows exactly which slots are empty, which are singletons, and which are collision slots.

To initiate the execution of the protocol, a RFID reader broadcasts a detection request, asking the tags to respond in a time frame of f slots. The detection request has two parameters, the frame size f and the random number r. After receiving the request, each tag maps itself to a slot in the frame through the same hash function. It then transmits during that slot.

Listening to the channel, the reader records the state of each slot, which is either *busy* when one or more tags transmit or *idle* when no tag transmits. This is binary information where each slot carries either ‘1’ or ‘0’. When a tag transmits, it does not have to send any particular information. It only needs to make the channel busy. When no tag is missing, the reader expects all singleton and collision slots are busy. However, if the reader finds an expected busy slot to be actually idle, it knows that the tag(s) that is mapped to this slot must be missing.

TRP is designed to minimize execution time by using the smallest frame size that ensures a detection probability α if m or more tags are missing. Certainly, if fewer tags are missing, the detection probability will be lower. A follow-up work [44] essentially executes TRP iteratively to identify which tags are missing.

A serious limitation of TRP is that it only considers time efficiency. It is not energy-efficient because all tags must be active and transmit during the time frame. B. Firner et al. [15] consider energy cost, but their protocol requires all tags to participate and transmit, which will be less efficient than a sampling-based solution where only a small fraction of tags participate.

In this dissertation, we show TRP is special case of a much broader protocol design space. Not only are there protocol configurations that perform much better than TRP in terms of both time and energy efficiencies, but we also reveal a fundamental energy-time tradeoff in this design space, which allows us to adapt protocol performance to suit various needs in practical systems.

4.2 An Intermediate Protocol

We present an energy-efficient protocol that serves as an intermediate design step towards our final solution in the next section.

4.2.1 Protocol Description

It is well known that a small group of people much fewer than 365 can have a high probability to contain two who celebrate their birthdays on the same day. This is called the birthday paradox. Similarly, two relatively small subsets of the tags in a large RFID system can also have a large probability of sharing a common tag. Let M be the set of m missing tags, and K be a subset of tags that the reader randomly selects from the inventory list of n tags currently in the system. The reader performs a simple operation to verify the presence of these $|K|$ tags, where $|K|$ is the number of the tags in K. It transmits the IDs of these tags one after another. After transmitting an ID, the reader waits for a short period and listens for a response. When a tag receives its ID, it will acknowledge its presence by sendinga response. If the reader does not receive any response back for an ID, it reports the missing tag event. The idea is that if $|K|$ is reasonably large, K and M will have a good chance to share at least one common tag. In other words, the reader will find that the presence of at least one tag in K cannot be positively confirmed. Hence, the missing-tag event is detected. This intermediate protocol is energy-efficient because there are overall $|K|$ tag responses, instead of n responses required by TRP. Note that the energy consumption by the reader for transmitting the IDs is a secondary concern.

4.2.2 Limitations

We observe that the intermediate protocol has much room for improvement. First, it is not time-efficient. Although only a subset of tag IDs is selected, it takes a considerable amount of time to verify the presence of each selected tag. According to the parameters of the EPCglobal Gen-2 standard [14], it takes about 2609.76 μs to transmit a tag ID of 96 bits, and it takes 290.81 μs for a tag to respond with one bit acknowledgment after receiving its ID. In comparison, it takes just 290.81 μs for a RFID reader to identify each empty or busy slot in the frame used by TRP. Second, although only a subset of tags transmit and each of them only transmits once, they have to listen to the channel for their IDs, which means that their circuits have to be continuously powered to receive up to k IDs. It is true that transmitting is likely to consume much more power than receiving if the same number of bits are involved. However, in our intermediate protocol, each selected tag makes just one short transmission, but it has to receive a large number of bits, which makes the aggregate receiving energy significant.

In our next protocol, for better time efficiency, we make sure that no tag IDs are transmitted. For better energy efficiency, we make sure that each tag only needs to receive at most one polling request.

4.3 Efficient Missing-Tag Detection Protocol (EMD)

4.3.1 Protocol Design

We propose our final solution that addresses the limitations of the intermediate protocol in the previous section. The RFID reader initiates the protocol execution by broadcasting a polling request. Upon receipt of the request, each tag decides with a sampling probability p whether to participate in the polling. If it decides to participate, it will randomly select a slot in the subsequent frame to respond. If it decides not to participate, it will simply enter the sleep mode and wake up at the next schedule time for the protocol execution. All decisions are made pseudo-randomly and predictable by the reader.

We first explain how to implement sampling. The polling request consists of three parameters: a frame size f, a random number r, and an integer $x = \lceil p \times X \rceil$, where X is a large, pre-configured constant (e.g., 2^{16}). After receiving the request, a tag performs a hash $H_1(id, r)$, where id is the tag's ID and H_1 is a hash function whose range is $[0, X)$. If $H_1(id, r) < x$, the tag will participate in the polling; otherwise, it won't.

If a tag decides to participate in the polling, it performs another hash $H_2(id, r)$ in the range of $[0, f)$ to determine in which slot of the time frame it will transmit. The tag will then transmit at the $H_2(id, r)$th slot.

During the protocol execution, a tag performs one or two hashes. There are many efficient hash implementation in the literature. In order to keep the tag's circuit simple, we propose to derive the hash values from a pool of pre-stored random bits. We use an offline random number generator with the ID of a tag as the seed to generate a string of random bits, which are then stored in the tag. The bits form a logical ring. $H_1(id, r)$ returns a certain number of bits clockwise after the rth bit in the ring. $H_2(id, r)$ returns a certain number of bits counter-clockwise after the rth bit in the ring. The two hash functions independently pick their 0th bits at random positions in the ring. The number of bits returned by $H_1(id, r)$ is $\log_2 X$. The number of bits returned by $H_2(id, r)$ is $\lceil \log_2 f \rceil$, and the final hash output is the returned value modulo f.

The RFID reader has the IDs of all tags, from which it can derive all information that is needed to know which tags will participate in the polling and, if so, at which slots they will transmit. Hence, it knows exactly which slots in the frame will be empty and which are expected to be busy. At the end of the frame, if the reader finds that a slot that is supposed to be busy turns out to be empty, it knows that the tag(s) that is expected to transmit in the slot must be missing. In this case, the reader reports a missing-tag event. When multiple synchronized readers are used to extend the coverage, we treat a slot as busy if any reader records that the slot is busy.

Clearly, the expected number of tags that will transmit their responses in the frame is $n \times p$. In the following section, we will propose another mechanism that ensures only the tags that transmit in the frame have to receive the polling request from the reader. Other tags that do not transmit in this round of polling will not even have to receive the request. Obviously, it requires some modifications to the protocol. We deliberately delay the description of this mechanism because it only makes sense after we present the necessary analytical result.

The online operation of the above protocol, called EMD (Efficient Missing-tag Detection), is very simple, which we believe is an advantage for a RFID system. Most of the protocol complexity belongs to the offline operations, which determine the values of p and f and reveal a fundamental energy-time tradeoff that has not been investigated before.

4.3.2 Probability of Detecting a Missing-Tag Event

Excluding the m missing tags, there are $n - m$ tags remaining in the system. Consider an arbitrary tag. It decides whether to participate in the polling with a *sampling* probability p. If it decides to participate, it will randomly *map* itself to a slot in the frame. To assist our analysis, we *serially* apply this *sampling-mapping* operation to the tags one after another. Let E_i be the number of empty slots remaining in the frame after i tags are processed. Clearly, $E_0 = f$ and $E_1 = f - 1$. E_{n-m} is the number of empty slots after all $n - m$ tags are processed; it is the number of empty slots that the RFID reader will observe at the end of the frame.

Consider E_i, $2 \leq i \leq n - m$, as random variables. Let e be a constant in the range of $[0, f]$ and $Prob\{E_i = e\}$ be the probability for $E_i = e$. We derive a recursive formula to compute the value of $Prob\{E_i = e\}$. The event $E_i = e$ happens in two cases: i) when $E_{i-1} = e$ and the ith tag is not mapped to an empty slot, or ii) when $E_{i-1} = e + 1$ and the ith tag is mapped to an empty slot. Furthermore, when $E_{i-1} = e$, the conditional probability for the ith tag to not map to an empty slot is $1 - p\frac{e}{f}$. When $E_{i-1} = e + 1$, the conditional probability for the ith tag to map to an empty slot is $p\frac{e+1}{f}$. Hence, we have

$$\begin{aligned}&Prob\{E_i = e\} = \\ &Prob\{E_{i-1} = e\} \cdot (1 - p\frac{e}{f}) + Prob\{E_{i-1} = e + 1\} \cdot p\frac{e+1}{f}\end{aligned}$$

Based on the above recursive formula, we construct a dynamic programming algorithm (see Algorithm 1) to compute $Prob\{E_{n-m} = e\}$, for $0 \leq e \leq f$, which is essentially the probability density function (p.d.f.) of E_{n-m}.

The array $P[e]$, $0 \leq e \leq f$, is initialized with the values of $Prob\{E_0 = e\}$, which are zeros for $0 \leq e \leq f - 1$ and one for $e = f$. Each iteration in the **for** loop over i updates the array with values of $Prob\{E_i = e\}$. At the end of the loop, the array $P[e]$ stores $Prob\{E_{n-m} = e\}$.

Next, we consider the number Ω of missing tags that would have participated in the polling if they had not been missing. Ω is a random variable that following a Binomial distribution with parameters m and p because each of the m missing tags would have a

Algorithm 1 Compute p.d.f. of E_{n-m}

```
INPUT: $n$, $m$, $p$, $f$
OUTPUT: $Prob\{E_{n-m} = e\}$, for $0 \leq e \leq f$

$P[e] = 0$ for $0 \leq e \leq f - 1$, $P[f] = 1$;
$P[f + 1] = 0$; \\ auxiliary variable
for $i$ = 1 to $n - m$ do
  for $e$ = 0 to $f$ do
    $P'[e] = P[e] \cdot (1 - p\frac{e}{f}) + P[e + 1] \cdot p\frac{e+1}{f}$;
  end for
  for $e$ = 0 to $f$ do
    $P[e] = P'[e]$;
  end for
end for
return $P[e]$ for $0 \leq e \leq f$;
```

probability of p to participate. Hence, for any $j \in [0, m]$,

$$Prob\{\Omega = j\} = \binom{m}{j} p^j (1 - p)^{m-j}. \tag{4–1}$$

Because the RFID reader has all the information, it knows the subset of tags in the system that will decide to participate in the polling. This subset may include a number j of missing tags that would have participated if they had not been missing, where $j \in [0, m]$. Each of these j tags is randomly mapped to a slot. If this is an empty slot observed by the reader at the end of the frame, the reader will detect the missing-tag event because it expects a tag to transmit at this slot.

Under the condition that there are e empty slots observed by the reader at the end of the frame, the probability for one missing tag to map to an observed empty slot is $\frac{e}{f}$. *Under the condition that there are j missing tags that would have participated in the polling*, the probability for any of them to map to an observed empty slot is $1 - (1 - \frac{e}{f})^j$. When that happens, the RFID reader detects a missing-tag event.

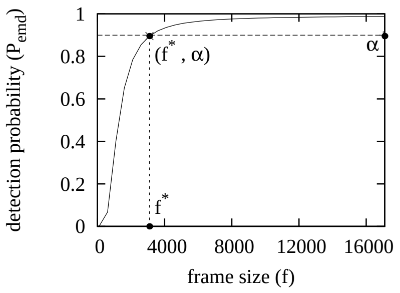

Figure 4-1. Detection probability $P_{emd}(p, f)$ with respect to the frame size f when $n = 50,000$, $m = 100$, and $p = 5\%$.

Summarizing the above reasoning, we give the probability for EMD to successfully detect a missing-tag event, which is a function in p and f, denoted as $P_{emd}(p, f)$.

$$\begin{aligned} &P_{emd}(p, f) \\ &= \sum_{e=0}^{f} Prob\{E_{n-m} = e\} \sum_{j=0}^{m} Prob\{\Omega = j\} \times (1 - (1 - \frac{e}{f})^j) \\ &= \sum_{e=0}^{f} Prob\{E_{n-m} = e\} \sum_{j=0}^{m} \binom{m}{j} p^j (1-p)^{m-j} (1 - (1 - \frac{e}{f})^j) \end{aligned} \quad (4\text{–}2)$$

The first sigma in the formula sums over the conditional probabilities for the number of empty slots. The second sigma in the formula sums over the conditional probabilities for the number of missing tags that would have participated in the polling. The values of $Prob\{E_{n-m} = e\}$ are computed by Algorithm 1.

4.3.3 Energy-Time Tradeoff Curve

A smaller value of f means a shorter protocol execution time, whereas a smaller value of p means a smaller number of transmitting tags, which in turn means a smaller energy cost. We cannot arbitrarily pick the values of f and p. They must satisfy the requirement $P_{emd}(p, f) \geq \alpha$. Subject to this constraint, we show that the values of f and p cannot be minimized simultaneously. The choice of f and p represents an energy-time tradeoff.

If we fix the value of p, $P_{emd}(p, f)$ becomes a function of f. It is an increasing function of f because a larger frame increases the probability for a missing tag to map to a singleton slot. As the tag is missing, the expected singleton becomes an empty slot observed by the reader, resulting in missing-tag detection. The solid line in Fig. 4-9 shows an example of the curve $P_{emd}(p, f)$ with respect to f when $n = 50,000$, $m = 100$, and $p = 5\%$. Because $P_{emd}(p, f)$ is an increasing function, the minimum value of f that satisfies the requirement $P_{emd}(p, f) \geq \alpha$ can be found by solving the equation $P_{emd}(p, f) = \alpha$. The solution is denoted as f^* (see Fig. 4-9 for an illustration).

For each different sampling probability p, we can compute the smallest usable frame size f^* from the equation $P_{emd}(p, f^*) = \alpha$. Hence, f^* can be considered as a function of p.

$$f^*(p) = \min\{f \mid P_{emd}(p, f) \geq \alpha \wedge f \leq U\}, \tag{4–3}$$

where U is an upper bound of the frame size, as a practical RFID system may consider a frame size beyond a certain upper bound to be unacceptable due to excessively long execution time. The algorithm that computes $f^*(p)$ based on bi-section search is given in Algorithm 2.

Algorithm 2 Search for $f^*(p)$

```
INPUT: $n$, $m$, $\alpha$, $p$
OUTPUT: frame size that minimizes execution time under sampling probability $p$

if $P_{emd}(p, U) < \alpha$ then exit; \\ requirement cannot be met
$f_0 = 0$, $f_1 = U$;
while $f_1 - f_0 > 1$ do
    $f_2 = \lceil \frac{f_0 + f_1}{2} \rceil$;
    if $P_{emd}(p, f_2) < \alpha$ then $f_0 = f_2$ else $f_1 = f_2$;
end while
return $f_1$;
```

Fig. 4-2 shows the curve of $f^*(p)$ when $n = 50,000$, $m = 75$ and $\alpha = 95\%$. We call it the *energy-time tradeoff curve*. Each point in the curve, $(p, f^*(p))$, represents a parameter choice whose energy cost is $n \times p$ tag responses and whose time frame consists of $f^*(p)$ slots. The symbols in the plot will be explained later. We plot the relation between the energy cost, $n \times p$, and the execution time, $f^*(p) \times t_s$, in Fig. 4-3.

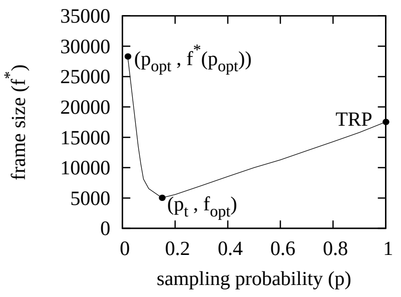

Figure 4-2. Frame size $f^*(p)$ with respect to sampling probability p when $n = 50,000$, $m = 75$, and $\alpha = 95\%$.

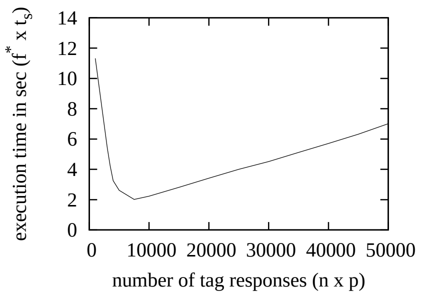

Figure 4-3. Execution time $f^*(p) \times t_s$ with respect to energy cost $n \times p$.

The tradeoff between these two performance metrics is controlled by the sampling probability p. If we decrease the value of p, we decrease the energy cost, but at the mean time the value of $f^*(p)$ may have to increase, which increases the execution time.

4.3.4 Minimum Energy Cost

When the sampling probability p is too small, the detection probability $P_{emd}(p, f)$ will be smaller than α for any value of f. Such a small sampling probability cannot be used by EMD. We design a bi-section search method in Algorithm 3 to find the smallest value of p, denoted as p_{opt}, which can satisfy $P_{emd}(p, f) \geq \alpha$ with a frame size no greater than

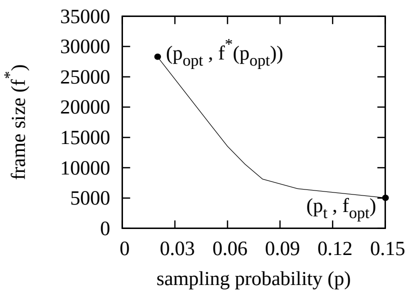

Figure 4-4. Energy-time tradeoff curve in the range $p \in [p_{opt}, p_t]$.

the upper bound U. The sampling probability returned by the algorithm is within an error of δ from the true optimal value p_{opt}, where δ is a parameter of the algorithm that can be set arbitrarily small. When EMD uses p_{opt} and $f^*(p_{opt})$, its energy cost is minimized.

Algorithm 3 Search for p_{opt}

```
INPUT: $n$, $m$, $\alpha$
OUTPUT: optimal sampling probability that minimizes energy cost

$p_0 = 0$, $p_1 = 1$, $\delta = 0.01$, $U$ = 1,000,000; \\400 seconds
while $p_1 - p_0 > \delta$ do
    $p_2 = \lceil \frac{p_0+p_1}{2} \rceil$;
    if $P_{emd}(p_2, U) < \alpha$ then $p_0 = p_2$ else $p_1 = p_2$;
end while
return $p_1$;
```

4.3.5 Minimum Execution Time

From the energy-time tradeoff curve (Fig. 4-2), we can find the smallest value of $f^*(p)$, denoted as f_{opt}, that minimizes the execution time.

$$f_{opt} = \min\{f^*(p) \mid p_{opt} \leq p \leq 1\} \tag{4–4}$$

Let p_t be the corresponding sampling probability. Namely, $P_{emd}(p_t, f_{opt}) = \alpha$. The values of f_{opt} and p_t are determined through bi-section search in Algorithm 4, where the **if** statement essentially uses the local gradient to guide the search direction towards the minimum. When EMD chooses p_t and f_{opt}, its execution time is minimized.

Algorithm 4 Search for f_{opt}

```
INPUT: $n$, $m$, $\alpha$
OUTPUT: optimal frame size and sampling probability that minimize execution time

$p_0 = p_{opt}$, $p_1 = 1$, $\delta = 0.01$;
while $p_1 - p_0 > \delta$ do
    $p_2 = \lceil \frac{p_0+p_1}{2} \rceil$;
    if $f^*(p_2) > f^*(p_2 + \frac{\delta}{2})$ then $p_0 = p_2$ else $p_1 = p_2$;
end while
return $f^*(p_1)$ and $p_1$;
```

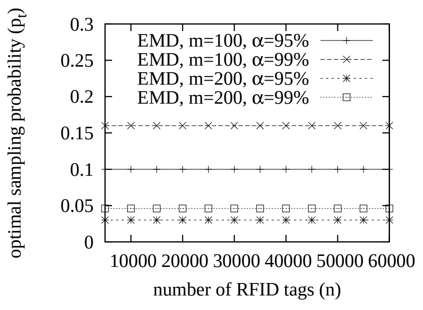

Figure 4-5. The value of p_t with respect to n.

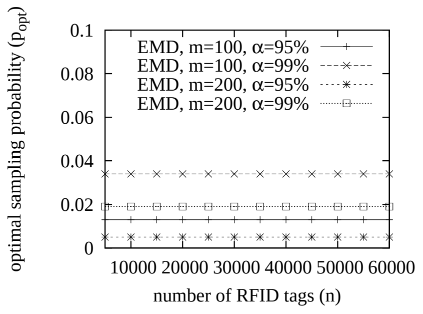

Figure 4-6. The value of p_{opt} with respect to n.

4.3.6 Energy-Time Tradeoff, TRP, and Offline Computation

Fig. 4-2 shows the energy-time tradeoff curve. Let's take a closer look by examining the segment of the curve between point (p_{opt}, $f^*(p_{opt})$) and point (p_t, f_{opt}) in the third plot of Fig. 4-4. When we increase the value of p from p_{opt} to p_t, the energy cost of the protocol is linearly increased, while the execution time of the protocol is decreased. We should not choose $p > p_t$ because both energy cost and execution time will increase when the sampling probability is greater than p_t.

TRP [46] is a special case of EMD when $p = 1$. As we see in Fig. 4-2, $p = 1$ is not a good choice for either energy efficiency or time efficiency. In fact, it is the worst in terms of energy cost.

Because the computation of p_{opt}, $f^*(p_{opt})$, p_t, and f_{opt} relies only on the values of n, m and α, we can calculate them offline in advance. The values of m and α are pre-configured as part of the system requirement. Hence, we can pre-compute p_{opt}, $f^*(p_{opt})$, p_t, and f_{opt} in a table format with respect to different values of n (for instance, from 100 to 100,000 with an increment step of 100), so that these values can be looked up during online operations. It takes two hours for us to build such a table on a Thinkpad T400 laptop.

4.3.7 Further Energy Reduction

By selecting a sampling probability between p_t and p_{opt}, EMD can greatly reduce the energy cost when comparing with TRP. We can further reduce the tags' overall energy consumption such that any tag that does not respond will not even listen for the polling request. We observe that when we change n, the values of p_t and p_{opt} remain largely constants, as shown in Fig. 4-5 and Fig. 4-6. Hence, their values are determined by m and α, which are pre-determined system parameters that specify the detection requirement. It means that we can precompute the values of p_t and p_{opt}. As long as the detection requirement specified by m and α does not change, p_t and p_{opt} can be approximately viewed as constants even though the number of tags in the system changes.

Suppose the detection requirement may be changed only at the beginning of each day. The reader picks a sampling probability p, which is p_t, p_{opt} or a value between them. It then downloads p to all tags and synchronizes their clocks. For the rest of the day, the reader does not have to transmit the sampling probability again. The tags wake up at the times when the protocol is scheduled for execution. Each tag makes a decision with probability p whether to participate in the polling. For those that decide not to, they go back to the sleep mode. The expected number of tags that will participate is $n \times p$. These tags stay awake to receive the polling request, and then respond in randomly-selected slots in the time frame after the request.

The actual implementation of the sampling probability p is slightly different from what's described in Section 4.3.1. At the beginning of the day, the reader downloads an integer $x = p \times X$ and a random number r to all tags, where X is a large constant (say, 2^{16}) and r is relatively prime to the size of the hash-bit ring. At the time of the ith protocol execution during the day, the tag takes a hash value h from the hash-bit ring, starting at the ith bit and taking every rth bit down the ring, until $\log_2 X$ bits are taken. It participates in the polling if $h < x$.

Because the above computation is deterministic (due to its pseudo randomness nature), a tag is able to know in advance which scheduled protocol execution it will participate next. Hence, it can appropriately set a wake-up timer so that it will wake up before the time when it is supposed to participate in the polling.

4.4 Multi-Seed Missing-Tag Detection Protocol (MSMD)

In this section, we propose our final solution that addresses the limitations of the intermediate protocol and EMD in the previous section, and begin our protocol design by assuming a reliable channel. We will then expand the new protocol to work under different error models in the next section.

4.4.1 Motivation

Both TRP [46] and EMD map tags to time slots using a hash function. We derive the probability θ that an arbitrary slot t will become a singleton, which happens when only one tag is sampled and mapped to slot t while all other tags are either not sampled or mapped to other slots. The probability for any given tag to be sampled and mapped to t is $\frac{p}{f}$, where f is the number of slots and p is the sampling probability, which is 100% for TRP. The probability for all other tags to be either not sampled or not mapped to slot t is $(1-\frac{p}{f})^{n-1}$, where n is the number of tags. Hence, we have

$$\theta = n\frac{p}{f}(1-\frac{p}{f})^{n-1} \approx \frac{np}{f}e^{-\frac{np}{f}} \leq \frac{1}{e} \approx 36.8\%,$$

where $\frac{np}{f}e^{-\frac{np}{f}}$ reaches its maximum value when $np = f$. This upper bound for θ is true for both TRP and EMD.

Singletons are important in missing-tag detection. If a missing tag is sampled and mapped to a singleton slot, since no other tag is mapped the same slot, this expected singleton slot will turn out to be idle, which is observed by the reader, resulting in missing-tag detection.

The problem is that the majority of all slots, 63.2% or more of them, are either empty slots or collision slots. They are mostly wasted. Obviously, empty slots do not contribute anything in missing-tag detection. If a collision slot only has missing tags, detection will be successfully made because the reader will find this expected busy slot to be actually idle. However, when the number of missing tags is small when comparing with the total number of tags, the chance for a collision slot to have only missing tags is also small.

Naturally, we want a protocol design that ensures a large value of θ, much larger than 36.8%, because more singleton slots increase detection power. However, the value of θ in TRP is in fact much smaller than 36.8% because TRP minimizes its execution time by using as few time slots as possible, which results in a large percentage of collision slots. The detection probability of TRP is about $1-(1-\theta)^m$ because each of the m missing tags has a probability of θ to map to a singleton slot and thus be detected.[1] As an example, if the requirement is to detect a missing-tag event with 99% probability when 100 tags are missing, TRP will reduce its frame size to such a level that $\theta = 4.5\%$, just enough to ensure 99% detection probability.

This leaves a great room for improvement. We show that a new protocol design, different from that of TRP and EMD, can reduce the frame size to a level that is much smaller than they can do, yet keep θ at a value much greater than 36.8%. Our design, called *Multiple-Seed Missing-tag Detection protocol* (MSMD), turns most empty/collision slots into singletons. There is a compound effect of such a new design when it is coupled with sampling: Suppose $\alpha = 99\%$ and $m = 100$, same as in the previous paragraph. Under sampling, the detection probability is $1 - (1 - p\theta)^m$ because each of the m missing tags has a probability of $p\theta$ to be sampled and mapped to a singleton slot.

[1] To quickly get to the point without dealing with too much detail, we ignore the small contribution of collision slots in detection.

If our protocol design can improve θ to 90%, we will be able to set $p = 5\%$. With such a sampling probability, we achieve much better energy efficiency because only 5% of all tags participate in each protocol execution. We also achieve far better time efficiency because, with much fewer tags transmitting, the chance of collision is reduced and a fewer number of time slots are needed to ensure a certain level of singletons.

4.4.2 Hash Function

There exist many efficient hash functions in the literature. In order to keep the tag circuit simple, we build our hash function on top of the simple scheme in [24] using a ring of pre-stored random bits: Before a tag is deployed, an offline random number generator uses the ID of a tag as seed to produce a string of pseudo-random bits, which are stored in the tag. The bits form a logical ring. After deployment, the tag generates a hash value $H(id, s)$ by returning a certain number of bits after the sth bit in the ring, where id is the tag ID and s is a given *hash seed* that can alter the hash output. This hash output is predictable by a RFID reader that knows the tag ID and the seed s.

More sophisticated hash implementations can be designed based on a ring of pseudo-random bits. For example, we may interpret s as a concatenation of a flag x and two random numbers, r_1 and r_2. To produce a hash output, we go clockwise along the ring if $x = 0$ or counterclockwise if $x = 1$. We then output the r_1th bit on the ring, and then output one more bit after every r_2 bits on the ring. If the hash output is required to be in a range $[0, y)$, we first take a sufficient number of hash bits as described above and then perform modulo y.

This hash function is easy to implement in hardware and thus suitable for tags. But it can only produce a limited number of different hash values, depending on the size of the ring. It is not suitable for a protocol whose operations require each tag to produce a large number of different hash values, but it works well for a protocol that only requires each tag to produce a few independent hash values.

4.4.3 Basic Idea

We have known that under a random mapping from tags to slots, an arbitrary slot only has a probability of up to 36.8% to be a singleton. Now, if we separately apply two independent random mappings from tags to slots, a slot will have a probability of up to $1 - (1 - 36.8\%)^2 \approx 60.1\%$ to be a singleton in one of the two mappings. If we separately apply k independent mappings from tags to slots, it has a probability of $1 - (1 - 36.8\%)^k$ to be a singleton in one of the k mappings. The value of $1 - (1 - 36.8\%)^k$ quickly approaches to 100% as we increase k.

It is easy to generate multiple mappings. In the detection request, the RFID reader can broadcast k seeds, s_1, s_2, ..., s_k, to tags. Each seed s_i corresponds to a different mapping, where a tag is mapped to a slot indexed by $H(id, s_i)$, which is a hash function such as [24] that takes an ID and a seed to produce an output (belonging to a required range through modulo operation).

A slot may be a singleton under one mapping, but a collision slot under other mappings. Different slots may be singletons under different mappings. To maximize the number of singletons, the reader — with the knowledge of all tag IDs and all seeds — selects a mapping (i.e., a seed) for each slot, such that the slot can be a singleton. The reader also makes sure that each tag is assigned to a singleton only once. From each slot's point of view, a *specific* seed is used to map tags to it. From the whole system's point of view, multiple seeds are used to map different tags to different slots.

In our protocol, the reader determines system parameters, including the sampling probability p and the frame size f. After selecting k random seeds, the reader chooses a seed for each slot and constructs a *seed-selection vector* V (or selection vector for short), which contains f *selectors*, one for each slot in the time frame. Each selector z has a range of $[0, k]$. If $z > 0$, it means that the zth seed, i.e., s_z, should be used for its corresponding slot. If $z = 0$, it means that the slot is not a singleton under any seed.

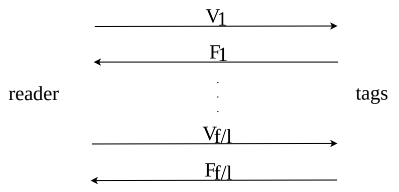

Figure 4-7. In Phase two, the reader broadcasts the seed-selection segments, V_1 through $V_{f/l}$, one at a time. Each segment V_i is immediately followed by a sub-frame F_i of l slots, during which the tags transmit.

Finally, the reader broadcasts the selection vector to the tags. Based on the selectors, each tag determines which slot it should use to respond.

We will address the problems of how to choose the optimal system parameters, p and f, and how the number k of seeds will affect the protocol performance in Section 4.5. Before we describe the operations of the protocol, we introduce the concept of segmentation. In our design, the above idea is actually applied segment by segment.

4.4.4 Segmentation

The seed-selection vector has f selectors, each of which are $\lceil \log_2(k+1) \rceil$ bits long. f may be too large for the whole vector to fit in a single slot. For example, if $k = 7$, each selector is 3 bits long. If we use the same slot T_{tag} for carrying a 96-bit ID to carry the selection vector, it can only accommodate 32 selectors. When f is more than that, we have to divide the selection vector into 96-bit segments, so that they can fit in T_{tag} slots. Each segment contains $l = \frac{96}{\lceil \log_2(k+1) \rceil}$ selectors. The total number of seed-selection segments are $\frac{f}{l}$, and the jth segment is denoted as V_j.

Since we divide the selection vector into segments, we also divide the time frame into sub-frames, each containing l slots accordingly. The jth time sub-frame is denoted as F_j. This allows our protocol to deal with one sub-frame at a time.

4.4.5 Protocol Overview

Our protocol consists of two phases. Phase one performs tag assignment, where the reader identifies the set of sampled tags, and assigns the sample tags to the sub-frames uniformly at random. The subset of sampled tags that are assigned to the

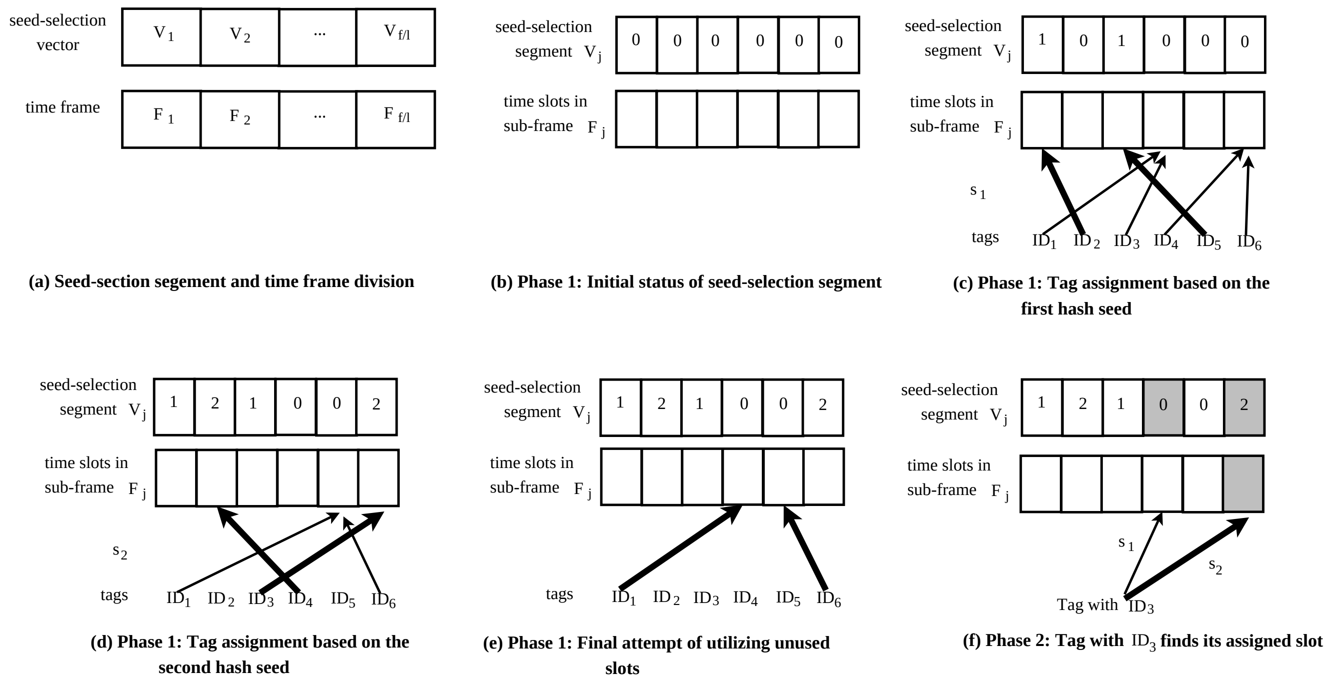
seed-selection vector
V 1
V 2
...
V f/l
time frame
F 1
F 2
...
F f/l
(a) Seed-section segement and time frame division
seed-selection segment V j
0 0 0 0 0 0
time slots in sub-frame F j
(b) Phase 1: Initial status of seed-selection segment
1 0 1 0 0 0
s 1
tags
ID1 ID2 ID3 ID4 ID5 ID6
(c) Phase 1: Tag assignment based on the first hash seed
1 2 1 0 0 2
s2
(d) Phase 1: Tag assignment based on the second hash seed
(e) Phase 1: Final attempt of utilizing unused slots
Tag with ID3
(f) Phase 2: Tag with ID3 finds its assigned slot

Figure 4-8. Arrows represent the mapping from tags to slots based on hash functions. Among them, thick arrows represent the assignment of tags to slots. In this example, $k = 2$.

jth sub-frame is denoted as N_j. For each sub-frame F_j, the reader selects a seed for each of its slots, constructs the seed-select segment V_j, and maps the tags in N_j to slots in F_j using the selected seeds.

Phase two performs missing-tag detection. The reader broadcasts the seed-selection segments one after another, each in a slot of T_{tag}. Each seed-selection segment is followed by a time sub-frame of l slots, each of which is T_{short} long. The tags in N_j will respond in these slots. Each tag only needs to be active during its sub-frame, which conserves energy. The exchange between the reader and tags in Phase two is illustrated in Figure 4-7.

4.4.6 Phase One: Tag Assignment

Phase one consists of three steps, which are explained below. An illustrative example can be found in Fig. 4-8.

4.4.6.1 Determining sampled tags

The reader starts Phase one by uniquely identifying the set of participating tags through sampling. To implement the sampling probability p, the reader broadcasts an integer $x = \lceil pX \rceil$ and a prime number q, where X is a large, pre-configured constant (e.g., 2^{16}). During the ith round of protocol execution, a tag is sampled if and only if the hash result $H(id, qi)$, which is a pseudo-random number in the range of $[0, X)$, is smaller than x, where id is the tag's ID.

After receiving x and q, each tag can predict which rounds of protocol execution it will participate. Since the protocol is scheduled to execute periodically with pre-defined intervals, each tag knows when it should wake to participate. The reader, with the knowledge of all tag information, can predict which tags are sampled for each protocol execution.

4.4.6.2 Assigning sampled tags to sub-frames

When assigning sampled tags to time sub-frames, the reader selects an additional random seed s, which is different from s_1, ..., s_k. For each sampled tag, the reader produces a hash output $H(id, s)$ and assigns the tag to the $H(id, s)$th sub-frame, where id is the tag's ID and the range of $H(id, s)$ is $[0, \frac{f}{l})$. Note that each tag will know which sub-frame it is assigned to, after it receives s in the detection request broadcast by the reader at the beginning of Phase two.

4.4.6.3 Determining seed-selection segments

Each seed-selection segment is determined independently. All selectors in V_j are initialized to zeros as shown in Fig. 4-8 (b). The reader begins by using the first seed s_1 to map tags in N_j to slots in F_j, as shown in Fig. 4-8 (c). For each tag in N_j, the reader produces a hash output $H(id, s_1)$ and maps the tag to the $H(id, s_1)$th slot in F_j, where id is the tag's ID and the range of $H(id, s_1)$ is $[0, l)$. After mapping, the reader finds singleton slots. Each singleton has one and only one tag mapped to it — as an example, the first and third slots in Fig. 4-8 (c). We assign the tag to the slot so that it will transmit in the slot, free of collision, during Phase two. The reader sets the corresponding selector in V_j to be 1, meaning that the first seed s_1 should be used for this slot. The slot is now called a *used* slot, and the sole tag mapped to it will be called an *assigned* tag.

The reader repeats the above process with other seeds, one at a time, for the remaining mappings. For each mapping, we only consider the slots whose selectors have not been determined yet and only consider the tags that have not been assigned to any slots yet, as shown by Fig. 4-8 (d). In other words, the used slots and the assigned tags will not be considered. For a singleton slot that is found using seed s_i, the corresponding selector in V_j will be set to be i.

After all k mappings, if the value of a selector in V_j remains zero, it means that the corresponding slot in F_j is not a singleton under any seed. As a final attempt to utilize these unused slots, if there exist unassigned tags in N_j, the reader randomly assigns the unassigned tags to unused slots. More specifically, it chooses an additional random seed s' and produces a hash output $H(id, s')$ to assign each tag that is not assigned yet to the $H(id, s')$th unused slot, where id is the tag's ID. In case that only one tag is assigned to an unused slot, we will have an extra singleton, as shown in Fig. 4-8 (e). Since the whole tag-to-slot assignment is pseudo-random, the reader knows which unused slots will become singletons. As we will see later in Phase two, after receiving s_1,..., s_k, each tag will know whether it is assigned to a slot. If not, from the received s', it will know which unused slot it is assigned to.

4.4.7 Phase Two: Missing-Tag Detection

At the beginning of this phase, the reader broadcasts a detection request, which is followed by a time frame for sampled tags to respond. The detection request consists of a frame size f and a sequence of seeds, s, s_1, ..., s_k, and s'. The time frame is divided into sub-frames. Before each sub-frame F_j, the reader broadcasts the corresponding seed-selection segment V_j in a single tag-ID slot T_{tag}. It is followed by l short slots (T_{short}) of the sub-frame, during which the tags in N_j can respond. Recall that each selection segment is 96 bits long. If $k = 7$, a segment has $l = \frac{96}{\log_2(7+1)} = 32$ selectors, and thus each time sub-frame has 32 slots.

Consider an arbitrary tag t. It wakes up to participate in a scheduled protocol execution that it is sampled for. After t receives the detection request from the reader, it uses $H(id, s)$ to determine which sub-frame it is assigned to. Without loss of generality, let the sub-frame be F_j. The tag sets timer to wake up before F_j begins. After receiving the seed-selection segment V_j, tag t uses $H(id, s_1)$ to find out which time slot it is mapped by seed s_1. It then checks whether the corresponding selector in V_j is 1. If the selector is 1, according to the construction of V_j in Section 4.4.6.3, t must be the sole tag that is mapped (and assigned) to this slot under s_1. If the selector is not 1, it means that s_1 should not be used to map any tag to this slot. In the latter case, t will move on to other seeds and repeat the same process to determine if it is assigned to a slot. If so, it will transmit during that slot. Otherwise, if t is not assigned to a slot after all k seeds, it will make a final attempt by finding out all unused slots (whose corresponding selectors in V_j are zeros) and using $H(id, s')$ as index to identify an unused slot to transmit.

In summary, after Phase one, the reader knows (1) which sub-frame each sampled tag is assigned to, (2) which slot each sampled tag is expected to transmit, (3) which slots are expected to be singletons, and (4) which slots are expected to be collision slots (due to the final attempt using s'). After Phase two, if an expected singleton/collision slot turns out to be idle, the reader detects a missing-tag event. Because multiple mappings reduce the number of empty/collision slots, both energy efficiency and time efficiency are greatly improved, as we will demonstrate analytically and by simulations in the following sections.

4.5 Energy-Time Tradeoff in Protocol Configuration

We study the energy-time tradeoff of our protocol and show how to compute the system parameters.

4.5.1 Execution Time and Energy Cost

The protocol execution time includes the time for the reader to transmit a detection request, the time for the reader to transmit the seed-selection vector of f selectors, and the time frame of f slots for tags to transmit, where the seed-selection vector is divided into segments and the time frame is divided into subframes. The request only carries a few parameters. Its time is negligible when comparing with the time frame and the seed-selection vector if f is large. Hence, the protocol execution time is roughly proportional to f. To investigate the energy-time tradeoff in relative terms, we characterize the protocol execution time by using the frame size f. A smaller value of f means a shorter protocol execution time. The actual execution time, measured in seconds, will be studied through simulations in Section 4.7.

The computation at each tag is mostly hashing. The hash function can be made very simple, such as [24] where hash output is produced by selecting a certain number of bits from a pre-stored bit ring. Moreover, once the tags receive the hash seeds from the reader's detection request, they can produce the needed hash values ahead of time, and all tags do so in parallel, while the reader is sending its seed-selection vector.

Our protocol design pushes most of its complexity to the reader. The tags' operation is simple: A tag wakes up to participate in a scheduled protocol execution that it is sampled for. It receives the detection request, determines which sub-frame it is assigned to, wakes up again before the sub-frame starts, receives the 96-bit seed-selection segment, determines which slot it is assigned to, and transmit a signal in that slot. Because each participating tag performs similar operation, the energy cost to each participating tag is also similar. The total energy cost among all tags for each protocol execution is proportional to the number of participating tags; note that different tags will be sampled to participate in different protocol executions uniformly at random. Hence, we may characterize the energy cost of a protocol execution by using the expected number of participating tags, pn, which is in turn proportional to the sampling probability p.

4.5.2 Detection Probability

To find the detection probability after one protocol execution, we need to first derive the probability for an arbitrary sampled tag t to be assigned to a singleton slot during Phase one. There are k mappings. Let P_i be the probability that tag t is assigned to a singleton slot after the first i mappings. Let n be the total number of tags and n' be the number of sampled tags that are mapped to the same sub-frame as t does. Assume the hash function assigns sampled tags to sub-frames uniformly at random. n' follows a binomial distribution, $Bino(n, p\frac{l}{f})$, i.e.,

$$Prob\{n' = j\} = \binom{n}{j}(p\frac{l}{f})^j(1 - p\frac{l}{f})^{n-j}. \tag{4–5}$$

$P_0 = 0$. We derive a recursive formula for P_i, $1 \leq i \leq k$. After the first $i-1$ mappings, there are two cases. Case 1: tag t has been assigned to a slot; the probability for this to happen is P_{i-1}. Case 2: tag t has not been assigned to a slot; the probability for this case is $1 - P_{i-1}$. We focus on the second case below.

In the ith mapping, the slot that tag t is mapped to has a probability of $(1 - \frac{n'P_{i-1}}{l})$ to be unused. Each of the other $n' - 1$ tags has a probability $(1 - P_{i-1})$ to be unassigned. If it is unassigned, the tag has a probability of $\frac{1}{l}$ to be mapped to the same slot as t does. Hence, the probability p' for tag t to be the only one that is mapped to an unused slot is

$$p' = (1 - (1 - P_{i-1})\frac{1}{l})^{n'-1}(1 - \frac{n'P_{i-1}}{l}). \tag{4–6}$$

Recall that we are considering Case 2 here. Combining both cases, we have

$$\begin{aligned} P_i &= P_{i-1} + (1 - P_{i-1}) \sum_{j=1}^{n} Prob\{n' = j\} p' \\ &= P_{i-1} + (1 - P_{i-1}) \sum_{j=1}^{n} \binom{n}{j} (p\frac{l}{f})^j (1 - p\frac{l}{f})^{n-j} \\ &\quad (1 - (1 - P_{i-1})\frac{1}{l})^{j-1} (1 - \frac{jP_{i-1}}{l}), \end{aligned} \tag{4–7}$$

where the first item on the right side is the probability for a tag to be assigned to a slot by the first $i - 1$ mappings and the second item is the probability for the tag to be assigned to a slot by the ith mapping. The probability for tag t to be assigned to a slot after all k mappings is P_k.

After the k mapping, we have a final attempt, in which an unassigned tag may be mapped to a singleton slot or a collision slot. If the tag is mapped to a collision slot, it is highly unlikely that all tags in that slot will be missing because the parameter m is typically set far smaller than n. Hence, the contribution of collision slots to missing-tag detection can be ignored. When the tag is mapped to a singleton slot, detection will be made if the tag is missing. Therefore, the final mapping has no difference from the previous mappings. The probability for tag t to transmit in a singleton slot is P_{k+1}, which can be computed recursively from (4–7).

Each of the m missing tags has a probability p to be sampled. When the tag is sampled, it has a probability of P_{k+1} to be assigned a singleton slot. When that happens, since a missing tag cannot transmit, the reader will observe an idle slot instead, resulting in the detection. Therefore, the detection probability of MSMD, denoted as $P_{msmd}(p, f)$, is

$$P_{msmd}(p, f) = 1 - (1 - pP_{k+1})^m. \tag{4–8}$$

The value of $P_{msmd}(p, f)$ not only depends on the choice of p and f, but also depends on n, m and k, which are not included in the notation for simplicity. The values of p and

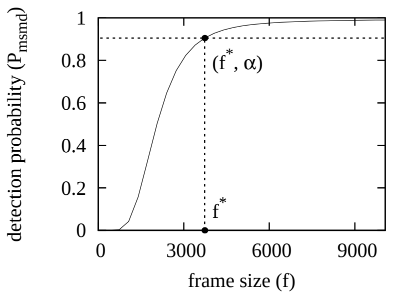

Figure 4-9. Detection probability $P_{msmd}(p, f)$ with respect to the frame size f when $n = 50,000$, $m = 100$, $k = 3$, and $p = 5\%$.

f are determined by the reader and broadcast to tags. They control the energy-time tradeoff as we will reveal shortly. The values of n, m and k are pre-known, where n is known because it is simply the number of tags that the reader expects to be in the system, m is known as a given parameter in the detection requirement, and k is determined before the tags are deployed.

EMD [29] is a special case of MSMD with $k = 1$ and without the final attempt. Hence, the detection probability of EMD, denoted as $P_{emd}(p, f)$, is

$$P_{emd}(p, f) = 1 - (1 - pP_1)^m. \tag{4–9}$$

TRP [46] is a special case of EMD with $p = 1$. Namely, sampling is turned off.

4.5.3 Energy-Time Tradeoff Curve

We cannot arbitrarily pick small values for p and f. They must satisfy the requirement $P_{msmd}(p, f) \geq \alpha$. Subject to this constraint, we show that the values of p and f cannot be minimized simultaneously. The choice of p and f represents an energy-time tradeoff.

If we fix the value of p, $P_{msmd}(p, f)$ becomes a function of f. The solid line in Fig. 4-9 shows an example of the curve $P_{msmd}(p, f)$ with respect to f when $n = 50,000$, $m = 100$, $k = 3$, and $p = 5\%$. Because $P_{msmd}(p, f)$ is an increasing function, the minimum value of f that satisfies the requirement $P_{msmd}(p, f) \geq \alpha$ can be found by solving the following equation,

$$P_{msmd}(p, f) = \alpha.$$

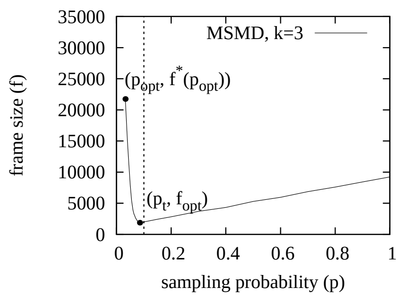

Figure 4-10. Energy-time tradeoff curve, i.e., frame size $f^*(p)$ with respect to sampling probability p, when $n = 50,000$, $m = 75$, $k = 3$, and $\alpha = 95\%$.

The solution is denoted as f^*. See Fig. 4-9 for illustration.

For each different sampling probability p, we can compute the smallest usable frame size f^* that satisfies $P_{msmd}(p, f) \geq \alpha$. Hence, f^* can be considered as a function of p, denoted as $f^*(p)$. A practical RFID system may consider a frame size beyond a certain upper bound U to be unacceptable due to excessively long execution time. In addition, f^* must be an integer. Considering these factors, we give a more accurate definition of f^* below.

$$f^*(p) = \min\{f | P_{msmd}(p, f) \geq \alpha \wedge f \leq U, f \in I^+\}. \tag{4–10}$$

We can find the value of $f^*(p)$ through bi-section search.

The left plot in Fig. 4-10 shows the curve of $f^*(p)$ when $n = 50,000$, $m = 75$, $k = 3$, and $\alpha = 95\%$. We call it the *energy-time tradeoff curve*. Each point, $(p, f^*(p))$, represents an operating point whose energy cost is measured as np participating tags and whose time frame consists of $f^*(p)$ slots. The symbols in the plot will be explained later. The energy-time tradeoff is controlled by the sampling probability p. If we decrease the value of p, we decrease the energy cost, but at the mean time the value of $f^*(p)$ may have to increase, which increases the execution time.

4.5.4 Minimum Energy Cost

When the sampling probability p is too small, the detection probability $P_{msmd}(p, f)$ will be smaller than α for any value of f. Such a small sampling probability cannot be

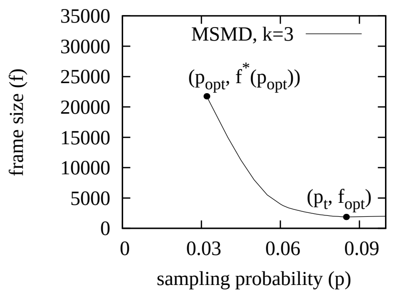

Figure 4-11. Energy-time tradeoff curve in the range $p \in [p_{opt}, p_t]$, which corresponds to the curve segment to the left of the dashed line in Fig. 4-10.

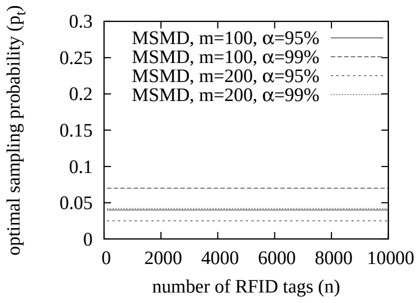

Figure 4-12. The value of p_t with respect to n.

used. We can use bi-section search to find the smallest value of p, denoted as p_{opt}, which can satisfy $P_{msmd}(p, f) \geq \alpha$ with a frame size no greater than the upper bound U. When p_{opt} and $f^*(p_{opt})$ are used, the energy cost is minimized.

4.5.5 Minimum Execution Time

From the energy-time tradeoff curve (the left plot in Fig. 4-10), we can find the smallest value of $f^*(p)$, denoted as f_{opt}, that minimizes the execution time.

$$f_{opt} = \min\{f^*(p) \mid p_{opt} \leq p \leq 1\} \tag{4–11}$$

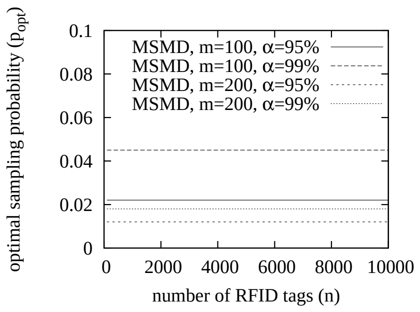

Figure 4-13. The value of p_{opt} with respect to n.

Let p_t be the corresponding sampling probability. Namely,

$P_{msmd}(p_t, f_{opt}) = \alpha$. The values of f_{opt} and p_t can be determined through bi-section search. When p_t and f_{opt} are used, the protocol execution time is minimized.

We amplify the segment of the energy-time tradeoff curve between point (p_{opt}, $f^*(p_{opt})$) and point (p_t, f_{opt}) in the right plot of Fig. 4-10. When we increase the value of p from p_{opt} to p_t, the energy cost of the protocol is linearly increased, while the execution time of the protocol is decreased. We should not choose $p > p_t$ because both energy cost and execution time will increase when the sampling probability is greater than p_t.

4.5.6 Offline Computation

Because the computation of p_{opt}, $f^*(p_{opt})$, p_t, and f_{opt} relies only on the values of n, m, α and k, we can calculate them offline in advance. The values of m and α are pre-configured as part of the system requirement. The value of k is determined before tag deployment. Hence, we can pre-compute p_{opt}, $f^*(p_{opt})$, p_t, and f_{opt} in a table format with respect to different values of n, so that these values can be looked up during online operations.

When performing such computation, we observe that when we change n, the values of p_t and p_{opt} remain largely constants, as shown in Fig. 4-12 and Fig. 4-13. Hence, their values are actually determined by m, α and k. It means that as long as the detection requirement specified by m and α does not change, p_t and p_{opt} can be approximately viewed as constants even though the number of tags in the system changes.

Suppose the values of m and α may be changed only at the beginning of each hour. The reader picks a sampling probability p, which is p_t, p_{opt} or a value between them. It then downloads p to all tags and synchronizes their clocks. For the rest of the hour, the reader does not have to transmit the sampling probability again.

4.5.7 Constrained Least-Time (or Least-Energy) Problem

The *energy-constrained least-time problem* is to minimize the protocol's execution time, subject to a detection requirement specified by m and α and an energy constraint specified by an upper bound u on the expected number of tags that participate in each protocol execution. To minimize execution time, we need to reduce the frame size as much as possible. Our previous analysis has already given the solution to this problem, which is simply $f^*(\frac{u}{n})$, where $\frac{u}{n}$ is the maximum sampling probability that we can use under the energy constraint.

The *time-constrained least-energy problem* is to minimize the number of tags that participate in protocol execution, subject to a detection requirement specified by m and α and an execution time constraint specified by an upper bound u' on the frame size. A solution can be designed by following a similar process as we derive $f^*(p)$ in Section 4.5.3: Starting from (4–8), if we fix $f = u'$, $P_{msmd}(p, f)$ becomes a function of p. We can use bi-section search to find p that meets $P_{msmd}(p, f) = \alpha \wedge p \leq p_t$.

4.5.8 Impact of k

We study how the number k of hash seeds will affect the protocol's performance. Figure 4-14 compares the energy-time tradeoff curves of EMD and MSMD with $k = 3, 7, 15$, respectively. Recall that EMD is a special case of MSMD with one hash seed and TRP is a special case of EMD with $p = 1$, represented by a point on the curve of EMD as shown in the figure. For MSMD, when $k = 3$, each seed selector needs 2 bits; recall that the value zero is reserved for non-singleton slots. When $k = 7$, each selector needs 3 bits. When $k = 15$, each selector needs 4 bits.[2]

In Figure 4-14, a lower curve indicates better performance because, for any sampling probability, its frame size is smaller, i.e., its execution time is smaller. Alternatively, it can be interpreted as, for any frame size, its sampling probability is smaller, i.e., it needs fewer tags to participate in each protocol execution. Clearly, MSMD

[2] One may ask why we do not use $k = 8$ or other values. The reason is that each selector needs 4 bits even when $k = 8$. In that case, we should certainly choose $k = 15$ for better performance.

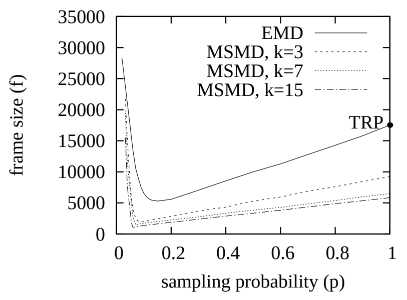

Figure 4-14. Energy-time tradeoff curves of EMD and MSMD under different k values, when $n = 50,000$, $m = 100$, and $p = 5\%$.

significantly outperforms EMD and TRP. As k increases, the performance of MSMD improves. However, the amount of improvement shrinks rapidly, demonstrated by the small gap between $k = 7$ and $k = 15$. When we further increase k to 31 using 5-bit selectors, the improvement becomes negligible. Increasing the value of k does not come for free; larger selectors mean more overhead. For one, it takes more time for the reader to broadcast the seed-selection vector. Therefore, we believe $k = 7$ is a good choice in practice because the performance gain beyond that is very limited.

4.6 MSMD over Unreliable Channels

We now consider unreliable channels. The intuition behind EMD and MSMD is that a slot will be found idle if the tags transmitting in the slot are all missing. It is true if no error occurs in this slot. In reality though, the communication between a reader and tags is, to varying degrees, subject to noise/interference in the environment, which may corrupt slots, for example, turning a would-be empty slot to a busy slot. Table 4-2 gives different possibilities of corrupted slots and their consequences. In the first row of this table, we assume that a tag t is missing. Then, the slot that t is mapped to should become idle during the protocol execution. However, this slot may be corrupted and turn out to be a busy slot. In this case, even if a tag that is supposed to transmit in a slot is missing, the reader can still sense a response induced by channel error, resulting in undetection of a missing tag. If all slots assigned to missing tags happen to be corrupted, the reader will fail to detect the missing-tag event. The second row

Table 4-2. The impact of corructed slots

Original slot	After corruption	Tag status	Reader thinks
idle	busy	missing	not missing
busy	busy	not missing	not missing

illustrates the impact of channel error when t is present in the system. Recall that we only distinguish two states for a slot: idle (no signal detected in the channel) or busy (signal detected). If t is not missing, the original slot should be busy. Since noise or interference in the environment may change the signal but is unlikely to cancel the signal out altogether, the slot should remain a busy slot. In this case, noise/interfere does not cause harm.

We evaluate the impact of channel error under two different models: random error model and burst error model. The former can be characterized by a parameter err, which denotes the probability for each slot to incur error. However, it may happen that channel introduces error in bursts for consecutive slots rather than at random. For example, communications between a reader and tags may be interfered by electromagnetic emission from nearby devices that share the same frequency band. When those devices are transmitting (e.g., sending a packet), their signals keep the channel busy for a small period of time until they stop. As the interfering transmissions are turned on and off, it causes bursts of error to the RFID system, which is characterized by the burst error model.

We stress that the error models are applied only to the tag-to-reader link in order to simplify the analysis. Error on the reader-to-tag link is addressed separately as follows: The transmission from the reader carries a CRC checksum. For example, the 96-bit seed-selection segment may contain a 16-bit CRC checksum and use the remaining 80 bits to encode seed selectors. When a tag receives the transmission from the reader, it computes a CRC based on the received information and then compares the result with the received CRC. If they are the same, the tag performs the operation according to the protocol. Otherwise, the tag will not participate further in the protocol execution, and instead it will transmit its ID to the reader to announce its presence after the protocol execution. This adds additional execution time and energy cost. But if the error ratio is small, the additional overhead will be small.

4.6.1 MSMD under Random Error Model

In the random error model, the impact of channel error is characterized by a parameter err, which is the probability for a slot to incur error. For example, if $err = 5\%$, a would-be idle slot has a chance of 5% to become busy.

MSMD under this model is called MSMD-re. From Section 4.5.2, we know that each of the m missing tags has a probability of pP_{k+1} to be detected in MSMD. Then, the probability for a missing tag to be detected under the random error model is $pP_{k+1}(1 - err)$. Therefore, the detection probability of MSMD-re, denoted as $P_{msmd-re}(p, f, err)$, is

$$P_{msmd-re}(p, f, err) = 1 - (1 - pP_{k+1}(1 - err))^m. \qquad (4–12)$$

Clearly, MSMD is a special case of MSMD-re with $err = 0$. For MSMD-re, the computation of p_{opt}, $f^*(p_{opt})$, p_t, and f_{opt} is similar to that of MSMD except that (4–9) is replaced with (4–12). MSMD-re uses the same offline computation process as described in Section 4.5.6 and follows the same protocol, only with modified parameters that consider the impact of channel error.

4.6.2 MSMD under Burst Error Model

We now consider the burst error model. According to [13], the number of bursts can be approximated as Poisson distribution. We give brief description below and readers are referred to [13] for details. The probability density function for the number of bursts is given by

$$h(x) = \sum_{i=0}^{\infty} \frac{\eta^i}{i!} e^{-\eta} \delta(x - i), \qquad (4–13)$$

where η is the average number of bursts, and $\delta(\cdot)$ is the Dirac Delta Function [1].

According to convolutional codes and trellis code modulations, the probability density function for the number of errors in a burst can be derived by Erlang density of second order. The Erlang distribution of second order is

$$g_c(z) = (2\mu)^2 z e^{-2\mu z}, \tag{4–14}$$

where μ is the rate parameter. Because the random variable for the number of errors in a burst assumes only discrete values [13], the probability density function can be obtained by discretizing the Erlang distribution $g_c(z)$:

$$g(y) = \sum_{w=1}^{\infty} P_E(w)\delta(y-w), \tag{4–15}$$

with

$$\begin{aligned} P_E(w) = P(w-1 < z_c \le w) &= \int_{w-1}^{w} \sum g_c(z)dz \\ &= e^{-2\mu w}[(e^{2\mu}-1)(1+2\mu w) - 2\mu e^{2\mu}]. \end{aligned} \tag{4–16}$$

Here, z_c is the continuous Erlang distributed random variable and $P_E(w)$ represents the probability of having w errors in a burst. Then, the mean value of the distribution given in (4–15) is:

$$E\{g(y)\} = \frac{e^{2\mu}(e^{2\mu}+2\mu-1)}{(e^{2\mu}-1)^2}. \tag{4–17}$$

The probability of having w errors in N bits, $P_N(w)$, depends on the number of bursts in the interval, $P_B(j)$, and on the number of errors in bursts, $P_E(w)$. Therefore, we have

$$P_N(w) = \begin{cases} P_B(0) & \text{for} \quad w = 0 \\ \sum_{j=1}^{\infty} P_e^{(j)}(w) P_B(j) & \text{for} \quad w > 0. \end{cases} \tag{4–18}$$

Here, $P_e^j(w)$ is the probability to have w errors in j bursts in the interval of N bits:

$$P_e^{(j)}(w) = \begin{cases} \sum_{n=1}^{w} P_e^{(j-1)}(w-n)P_E(n) & \text{for } j > 1 \\ P_E(w) & \text{for } j = 1, \end{cases} \tag{4–19}$$

and $P_B(j)$ represents the probability of having j bursts in an interval:

$$P_B(j) = \frac{\eta^j}{j!}e^{-\eta}. \tag{4–20}$$

From (4–16), (4–18), (4–19) and (4–20), we know that the computation of $P_N(w)$ relies on the values of μ and η. Now we should find a way to obtain the values of these two parameters.

We denote the mean value of errors in N bits (i.e., the value of $E\{g(y)\}$ when there are N bits) as N_e. If we know N_e, we can compute the value of μ from (4–17). According to [13], the value of η depends on the probability that a burst occurs and causes errors in the interval of N bits. It is given by:

$$\eta = \frac{Nr}{N_e}, \tag{4–21}$$

where r is a parameter called Bit Error Rate. Finally, the value of N_e is evaluated as follows:

$$N_e = \frac{NN_m}{p_0(N + L_m - 1)}, \tag{4–22}$$

where p_0 is the probability of having at least one error in the considered bits when a burst occurs, and p_0 is given by:

$$p_0 = 1 - (1 - \frac{N_m}{N + L_m - 1})^N. \tag{4–23}$$

Here, N_m is the mean value for number of errors per burst and L_m is the mean value of burst error length.

From (4–17), (4–21) and (4–22), we can see that the computation of μ and η depends on the values of N_m, L_m, r and N, which are input system parameters. After computing the values of μ and η, we can obtain the value of $P_N(w)$. For example, if $N_m = 9.5$, $L_m = 33.5$, $r = 10^{-3}$ and $N = 10$, the value of μ and η are respectively 0.52 and 4.110^{-3}. Then, $P_N(w)$ can be calculated by (4–18).

From Section 4.4, we know that each slot only carries binary information of either '1' or '0'. The information in the time frame therefore represents a bit array of length f. As the time frame is divided into sub-frames, each containing l slots, we also divide the bit array into segments of l bits, which allows our protocol to deal with one segment at a time. Consider the jth sub-frame F_j. Recall from (4–18) that the probability of having $w(0 \leq w \leq l)$ errors in l bits is $P_l(w)$. Then, the probability for a missing tag, which is assigned to F_j, to be detected is $pP_{k+1}P_l(w)(1 - \frac{w}{l})$. The detection probability of MSMD under the burst error model can be computed by summing over all possible values of w.

Hence, the detection probability of MSMD under the burst error model, denoted as $P_{msmd-be}$, is

$$P_{msmd-be}(p, f, N_m, L_m, r) = 1 - (1 - \sum_{w=0}^{l} pP_{k+1}P_l(w) \\ (1 - \frac{w}{l}))^m. \tag{4–24}$$

Under this model, we can obtain the values of p_{opt}, $f^*(p_{opt})$, p_t, and f_{opt} by following the same procedure as described in Section 4.5.6, except that (4–9) is replaced with (4–24).

4.7 Numerical Results

We have performed extensive simulations to study the performance of the proposed MSMD, MSMD-re and MSMD-be, and compare it with EMD and TRP [46]. The design of all fives protocols ensure that the detection requirement specified by m and α is always met. This is indeed what we observe in our simulations.

The performance comparison is made in terms of energy efficiency and time efficiency, given a certain detection requirement. To measure the protocol execution time, we set the transmission parameters based on the typical setting of the EPCglobal Gen-2 standard [14]. Any two consecutive transmissions (from the reader to tags or vice versa) are separated by a waiting time of 266.4 μs. The transmission rate from the reader to tags is 40.97 Kb/sec; it takes 24.41 μs for the reader to transmit one bit. A 96-bit slot that carries a seed-selection segment is 2609.76 μs long, which includes a waiting time before the transmission. The transmission rate from a tag to the reader is also 40.97Kb/sec, so that a single-bit slot T_{short} for a tag to respond (i.e., make the channel busy) is 290.81 μs, also including a waiting time.

For each set of system parameters, including m, α, and n, TRP will compute its optimal frame size. Once the frame size f is determined, the execution time is known, which is fT_{short} plus the time for broadcasting a detection request. MSMD, MSMD-re, MSMD-be and EMD will choose a sampling probability p, and compute the optimal frame size f^* under that sampling probability. Since EMD is a special case of MSMD, we use our analytical framework in the previous section to compute it. For EMD, its execution time is f^*T_{short}, plus the time for a request. For MSMD, MSMD-re and MSMD-be, we need to add the time for transmitting the selection vector.

We cannot find a well-accepted energy model for RFID tags or detailed parameters of energy expenditure for a RFID standard. However, as we have explain in Section 4.5.1, the energy cost can be indirectly measured by the number of participating tags because the former is proportional to the latter. We use this measurement to study the energy-time tradeoff in relative terms and make performance comparison. As part of our future work, we will investigate the exact energy cost (in mJ) of tags when an appropriate energy model for RFID tags becomes available. Although the exact energy consumption of a tag be simulated at this time (which depends on physical-layer implementation), we point out that each participating tag only needs to receive a small amount of data from the reader and transmits one-bit information.

4.7.1 Energy-Time Tradeoff

Let $n = 50,000$, $\alpha = 95\%$, and $m = 50$. For MSMD-be, the required input parameter setting is similar to that in [25]: $N_m = 9.5$, $L_m = 33.5$ and $r = 10^{-3}$. Fig. 4-15 shows the energy-time tradeoff curves produced by simulations. Recall that the energy cost of MSMD or EMD is proportional to p. The point at $p = 1$ on the EMD curve represents TPR. Clearly, MSMD significantly outperforms EMD. MSMD with $k = 7$ uses three-bit elements in the selection vector, while MSMD with $k = 3$ uses two-bit elements. Even though it incurs more overhead in the selection vector, MSMD with $k = 7$ slightly outperforms MSMD with $k = 3$. Further increasing k cannot bring performance gain due to overly large overhead for the selection vector. Furthermore, it is also shown in Fig. 4-15 that when we take the impact of channel error into consideration, the performance of MSMD with $k = 7$ degrades slightly, which can be illustrated by the curves of MSMD-re and MSMD-be. For MSMD-re, increasing err will cause more execution time and energy cost because of large probability for a slot to incur error. Even though channel errors cause more overhead in missing-tag detection (which should be an expected consequence), a key finding is that the general energy-time tradeoff relation stay the same. In Fig. 4-16, we zoom in for a detailed look at the curve segment in the sampling probability range of [0, 0.2]. When $p = 0.08$, the execution time of MSMD with $k = 7$ is 11.2% of the time taken by EMD. When we fix the execution time at 5 seconds, the number of participating tags in MSMD with $k = 7$ is 46.7% of the number in EMD. We do not directly compare MSMD-re and MSMD-de with EMD because the latter does not consider channel error. We vary the values of n, α and m. Similar conclusions can be drawn from the simulation results.

The tradeoff curves in Fig. 4-16 agree with our analytical results in Fig. 4-14 in principle. We want to stress that our simulations do not simply reproduce the analytical

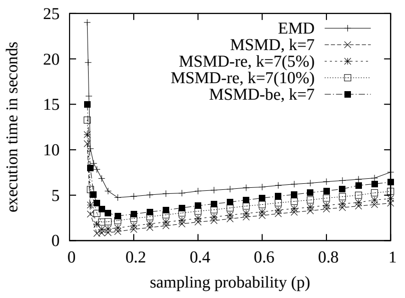

Figure 4-15. Protocol execution time with respect to sampling probability, when $\alpha = 95\%$, $m = 50$, and $n = 50,000$.

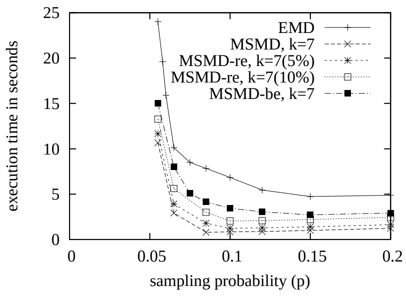

Figure 4-16. Zoom-in view of energy-time tradeoff in Figure 4-15 in the sampling probability range of [0, 0.2].

results. Simulations consider system details by using a real RFID specification. Such details are not captured by analysis. In addition, simulations consider the exact impact of selection vector on execution time (measured in seconds), instead of characterizing time in an indirect way using the frame size.

In Table 5-3, we show the relative performance of MSMD ($k = 7$) with respect to TRP, where $n = 50,000$, $\alpha = 95\%$, and $m = 50, 100$, or 200. MSMD is operated under sampling probability p_t and p_{opt}. For example, when $m = 50$, $p_t = 0.085$ and $p_{opt} = 0.055$. The numbers in the table are ratios of MSMD's energy cost (or execution time) to TRP's energy cost (or execution time). For example, when $m = 200$, the energy cost of MSMD with p_t is 2.1% of what TRP consumes, and its execution time is 4.09% of the time TRP takes. Again we do not directly compare MSMD-re and MSMD-de with TRP because the latter does not consider channel error.

Table 4-3. Relative energy cost and execution time of MSMD ($k = 7$) under p_{opt} and p_t, when $\alpha = 95\%$ and $n = 50,000$

	p_t		p_{opt}	
	energy	time	energy	time
$m = 200$	2.1%	4.09%	1.4%	269.1%
$m = 100$	3.6%	5.61%	2.5%	226.2%
$m = 50$	8.1%	10.84%	5.5%	82.3%

4.7.2 Performance Comparison

Next, we compare the performance of MSMD ($k = 7$), MSMD-re ($err = 5\%$ and $k = 7$), MSMD-be ($k = 7$), EMD, and TRP under different values of m, α and n. MSMD, MSMD-re, MSMD-be and EMD are operated with their optimal sampling probabilities p_t. In Fig. 4-17-4-19, we keep $m = 50$ and vary the value of α. In Fig. 4-17, we let $\alpha = 99.9\%$, meaning that each protocol execution should detect any missing-tag event with probability 99.9%. The left plot compares the energy cost of five protocols with respect to n, and the right plot compares their execution times. MSMD has a smaller energy cost than EMD, which in turn has a much smaller energy cost than TRP. Taking the impact of channel error into consideration, the energy costs by MSMD-be and MSMD-re increase modestly over MSMD. In the meanwhile, MSMD also has a much smaller execution time than EMD and TRP. Taking channel error into consideration, the execution times of MSMD-be and MSMD-re increase modestly over MSMD. Similar results can be drawn from Fig. 4-18 where $\alpha = 99\%$ and Fig. 4-19 where $\alpha = 90\%$. In the latter case, the execution time of MSMD is less than a second.

In Fig. 4-20-4-21, we keep $\alpha = 99\%$ and vary the value of m. In Fig. 4-20, $m = 25$. In Fig. 4-21, $m = 100$. The performance of MSMD remains the best among all five.

4.8 Summary

This dissertation proposes a new protocol design that integrates energy efficiency and time efficiency for missing-tag detection. It uses multiple hash seeds to provide multiple degrees of freedom for the RFID reader to assign tags to singleton slots, during

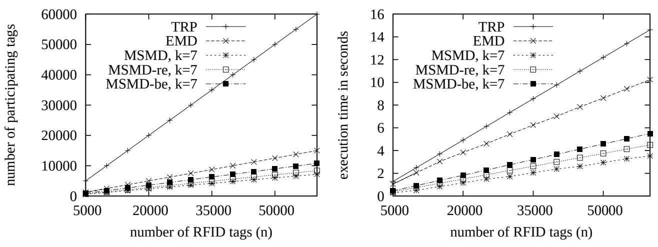

Figure 4-17. A) The number of participating tags with respect to the number of tags, when $m = 50$ and $\alpha = 99.9\%$. B) The protocol execution time with respect to the number of tags, when $m = 50$ and $\alpha = 99.9\%$.

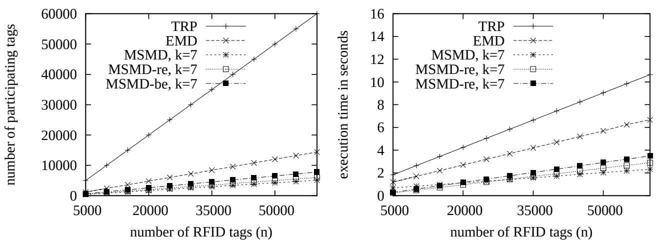

Figure 4-18. Same as the caption of Fig. 4-17 except for $\alpha = 99\%$.

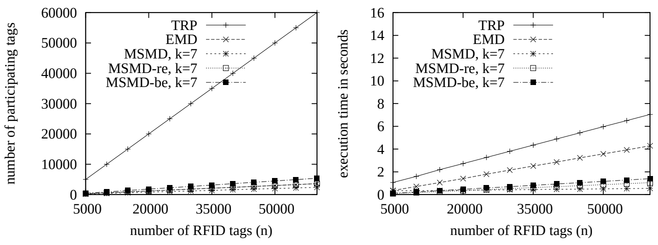

Figure 4-19. Same as the caption of Fig. 4-17 except for $\alpha = 90\%$.

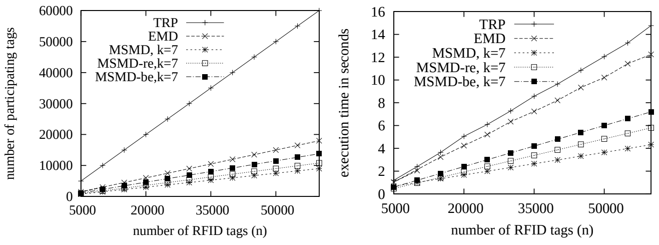

Figure 4-20. A) The number of participating tags with respect to the number of tags, when $m = 25$ and $\alpha = 99\%$. B) The protocol execution time with respect to the number of tags, when $m = 25$ and $\alpha = 99\%$.

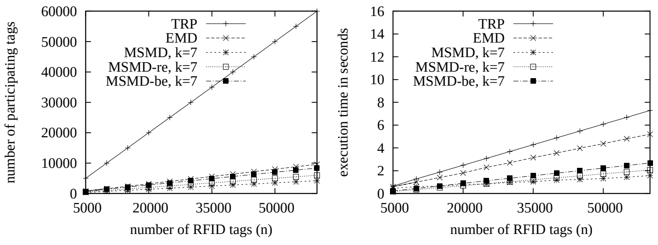

Figure 4-21. Same as the caption of Fig. 4-20 except for $m = 100$.

which the tags announce their presence in the process of missing-tag detection. We first present this new protocol with reliable channels. The result is a multi-fold cut in both energy cost and execution time. Such performance improvement is critical for a protocol that needs to be executed frequently. Then, we extend the protocol to consider two categories of channel errors induced by noise/interference in the environment. The involving of channel errors will make the energy/time gains slightly reduced, but remain significant comparing to EMD and TRP. We also reveal a fundamental energy-time tradeoff in the protocol design. This tradeoff gives flexibility in performance tuning when the protocol is applied in practical environment.

CHAPTER 5
AN EFFICIENT PROTOCOL FOR RFID MULTIGROUP THRESHOLD-BASED CLASSIFICATION

This chapter studies the RFID multigroup threshold-based classification problem. The goal is to determine whether the number of objects in each group is above or below a prescribed threshold value. Solving this problem is important for inventory tracking applications. If the number of groups is very large, it will be inefficient to measure the groups one at a time. The best existing solution for multigroup threshold-based classification is based on generic group testing, whose design is however geared towards detecting a small number of populous groups. Its performance degrades quickly when the number of groups above the threshold become large. In this dissertation, we propose a new classification protocol based on *tag sampling* and *logical bitmaps*. It achieves high efficiency by measuring all groups in a mixed fashion. In the meantime, we show that the new method is able to perform threshold-based classification with an accuracy that can be pre-set to any desirable level, allowing tradeoff between time efficiency and accuracy.

The rest of the chapter is organized as follows. Section 5.1 presents the system model and defines the problem to be solved. Section 5.2 discusses the related work and gives the motivation for our solution. Section 5.3 proposes our two-phase protocol for the RFID threshold-based classification problem. Section 5.4 evaluates the new protocol through simulations. Section 5.5 draws the conclusion.

5.1 Problem Definition and System Model

5.1.1 System Model

There are three types of RFID tags. Passive tags are most widely deployed today. They are cheap, but do not have internal power sources. Passive tags rely on radio waves emitted from an RFID reader to power their circuit and transmit information back to the reader through backscattering. They have short operational ranges, typically a few meters in an indoor environment. To cover a large area, arrays of RFID reader antennas must be installed. Semi-passive tags carry batteries to power their circuit, but still rely on backscattering to transmit information. Active tags use their own battery power to transmit, and consequently do not need any energy supply from the reader. Active tags operate at a much longer distance, making them particularly suitable for applications that cover a large area, where one or a few RFID readers are installed to access all tagged objects and perform management functions automatically. With richer onboard resources, active tags are likely to gain more popularity in the future, particularly when their prices drop over time as manufactural technologies are improved and markets are expanded.

Communication between readers and tags is time-slotted. Readers send out a request, which is followed by a slotted time frame during which tags transmit in their selected slots. The readers may take turns to transmit the request in order to avoid interference, or a more sophisticated scheduling algorithm may be used to allow readers that do not interfere to transmit simultaneously. When a tag transmits, as long as one reader receives the transmission correctly, the transmission will be successful. In our protocol design, we can logically treat all readers as one, which transmits a request and then listens to the tags' responses. We use two types of slots to carry tag responses. The first type is called a long-response slot, whose length is denoted as T_{long}, during which a tag transmits multiple bits, allowing the reader to tell whether there is collision in a slot. The second type is called a short-response slot, whose length is denoted as T_{short}, which carries one-bit information: '0' for an empty slot when no tag transmits, and '1' for an non-empty slot when one or more tags transmit signal to make the channel busy.

5.1.2 Multigroup Threshold-Based Classification Problem

Consider a big warehouse with tens of thousands of items. Each item is attached with an RFID tag for communication with an RFID reader. The items are divided into different groups based on certain properties, which can be the product sub-category, production date, or production place. To support grouping, each tag ID should contain two components: a *group ID*, which identifies the group the tag belongs to, and a *member ID*, which identifies a specific tag in the group. Clearly, all tags in a group must carry the same group ID, while tags in different groups carry different group IDs. We assume that the RFID reader knows the group IDs in the system.

We define the *population* or *size* of a group as the number of tags in this group. As we have explained in the introduction, while it is possible to perform precise multigroup classification at high cost, the focus of this dissertation is to study efficient solutions for approximate multigroup classification. We formally define the problem as follows: Let h be the threshold and α_1 be a large probability value. We require that any group whose population exceeds h should be reported with a probability of at least α_1. Let l be another integer parameter smaller than h and β_1 be a small probability value. We also require that the probability of reporting any group with l or fewer tags should be no more than β_1. Let k_1 be the population of an arbitrary group g. Our performance objectives can be expressed in terms of conditional probabilities as follows:

$$\begin{gathered} Prob\{\text{ group } g \text{ is reported by the reader } |k_1 \geq h\} \geq \alpha_1 \\ Prob\{\text{ group } g \text{ is reported by the reader } |k_1 \leq l\} \leq \beta_1 \end{gathered} \tag{5–1}$$

We treat the report of a group with l or fewer tags as a false positive, and the non-report of a group with h or more tags as a false negative. Hence, the above objectives can also be stated as bounding the false positive ratio by β_1 and the false negative ratio by $1 - \alpha_1$.

5.2 Preliminary

5.2.1 Prior Work

Sheng, Tan and Li studied the multigroup threshold-based classification problem in [45]. They begin with a simple threshold checking scheme (TCS) to approximately answers whether the number of tags exceeds a threshold. Based on TCS, they propose

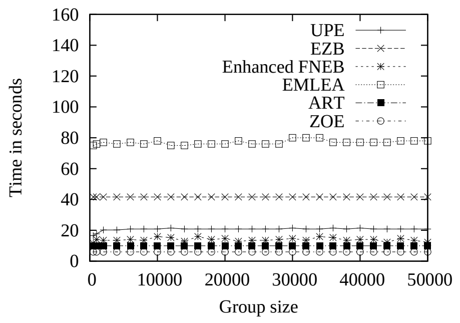

Figure 5-1. The estimation time with respect to the group size for UPE, EZB, Enhanced FNEB, EMLEA and ART, when $\alpha'_1 = 99\%$ and $\beta'_1 = 1\%$.

Table 5-1. Notations

Symbols	Descriptions
$1 - \alpha_1$	upper bound of false negative ratio
β_1	upper bound of false positive ratio
m_1	bit length of logical bitmap
n_1	number of tags
S	number of above-threshold groups
k_1	actual number of tags in an arbitrary group
$\hat{k}_1$	estimated number of tags in an arbitrary group
r_i	random number in the ith polling
f_1	length of the time frame each polling
$H(\cdot)$	hash function whose range is $[0, f_1 - 1]$
m_{id}	a tag's member ID
g_{id}	a tag's group ID
h	a prescribed higher bound threshold value
l	a prescribed lower bound threshold value
w	number of pollings

two probabilistic protocols. The first one is based on generic group testing (GT), which consists of multiple rounds. In each round, the reader shuffles all groups into different categories, each of which may contain tags from multiple groups. TCS is then applied to check the number of tags in each category. The categories with sufficient tags are labeled as potential populous categories, which may include above-threshold groups. In the end, the testing history is used to classify all above-threshold groups. The second protocol is a combination of group testing and divide-and-conquer, which ignores the categories that fail to pass the TCS tests in the previous round, divides the remaining categories into multiple sub-categories, and applies TCS to each sub-categories in the remaining rounds.

Another possible solution for the multigroup threshold-based classification problem is to use a reader to collect the actual tag IDs from tags [8, 12, 17, 20, 32, 34, 49, 54], where each ID contains bits that identify the group of the tag. Applying to the problem in this dissertation, these ID-collection protocols do not work well for large-scale RFID systems due to their long identification time.

Many methods were proposed to estimate the whole population of an RFID system. They are essentially single-group estimators. We can use them to first estimate individual group sizes (one group at a time) and then use the sizes for classification purpose. Kodialam and Nandagopal propose the first set of single-group estimators, including the Zero Estimator (ZE), the Collision Estimator (CE), and the Unified Probabilistic Estimator (UPE), which collect information from tags in a series of time frames and estimates the whole population of tags in the system based on the number of empty slots and/or the number of collision slots [21]. A follow-up work by the same authors proposes the Enhanced Zero-Based Estimator (EZB) [22], which is an asymptotically unbiased estimator and makes estimation only based on the number of empty slots. Qian et al. provide a replicate-insensitive estimation algorithm called the Lottery-Frame scheme (LoF) [37]. The Enhanced First Non-Empty slots Based Estimator (Enhanced FNEB) [16] can be used to estimate tag population in both static and dynamic environments by measuring the position of the first non-empty slot in each frame. Li et al. [26] study the estimation problem for large-scale RFID systems from the energy angle based on a Enhanced Maximum Likelihood Estimation Algorithm (EMLEA). They design several energy-efficient probabilistic algorithms that iteratively refine a control parameter to optimize the information carried in the transmissions from tags, such that both the number and the size of the transmissions are minimized. The Average Run based Tag estimation (ART) scheme [43] further reduces the execution time for population estimation, based on the average run length of ones in the bit string received in the standardized frame-slotted Aloha protocol. Finally, the Zero-One Estimator (ZOE) [59] provides fast and reliable cardinality by tuning the system parameters and converging to the optimal settings through a bisection search.

5.2.2 Motivation

We first show the performance of some existing single-group estimators through simulation. From the simulation results, we argue that these estimators are not time-efficient when they are applied to the multigroup threshold-based classification problem.

Figure 5-1 presents the execution time of five existing single-group estimators [16, 21, 22, 26, 43] with respect to the number of tags in the group; details about the simulation setting and parameters can be found in Section 5.4.1. While these estimators are designed to measure the size of a single group, they may be applied to performing multigroup threshold-based classification by estimating one group at a time. Their estimation accuracy is specified by a confidence interval: The probability for the estimate to deviate from the true group size by β_1' percentage or more should not exceed α_1', where α_1' and β_1' are two pre-specified system parameters. They are set to 99% and 1% respectively in our simulation. From the figure, we observe that the estimation time changes very little with respect to the number of tags. For example, ART takes about 10 seconds to estimate the tag population in the range from 500 to 50,000. If there are two groups of 25,000 tags each, the total estimation time for the two groups will be 20 seconds. However, if there are 100 groups of 500 tags each, the total estimation time will be 1,000 seconds! Hence, these estimators are not suitable when there are numerous small groups.

The group testing method in GT [45] can significantly reduce the execution time for populous group discovery. However, simulation results show that their execution time is approximately proportional to the number of groups above the threshold. Hence, the performance of the protocol will deteriorate if the number of groups above the threshold is large. In addition, the RFID reader must be able to distinguish three types of slots: (1) empty slot, during which no tag transmits; (2) singleton slot, during which only one tag transmits, and (3) collision slot, during which more than one tag transmits.

We follow two general design principles when designing our time-efficient classification protocol. First, we want to minimize the length of each time slot. Based on the parameters of the EPCglobal Gen-2 standard [14], in order to transmit a 96-bit ID from a tag to an RFID reader, we need a slot of 2608.8 μs. However, if the reader is not interested in IDs but wants to distinguish collision slots from singleton or empty slots [45], tags should transmit 10-bit long responses, using slots of 470.5 μs each. Furthermore, if the reader does not need to distinguish collision slots from singleton slots but only wants to know whether the slots are empty or not, tags can transmit one-bit short responses, using slots of 290.8 μs each to carry one bit information (channel busy or idle); this is the type of slots we will use in our protocol design. Note that 266.4 μs waiting time is included in each slot to separate it from neighboring slots.

Second, we want to minimize the number of slots. Figure 5-1 clearly shows that traditional approaches of measuring one group at a time will not work well when there are a large number of groups. We take a new design that is drastically different from traditional ones: measuring the groups all at once and probabilistically sharing each slot by multiple groups. This design has an interesting feature that its execution time is largely insensitive to the number of groups if the total number of tags is about the same. It makes our protocol particularly suitable for situations where there are a large number of small or medium-sized groups.

5.3 An Efficient Threshold-Based Classification Protocol

This section presents an efficient Threshold-Based Classification (TBC) Protocol, which is a combination of *dynamic slot sharing* among groups and *maximum likelihood estimation* of group sizes.

5.3.1 Dynamic Slot Sharing

We share all slots among all groups. Notably, we abandon the approach of applying a single-group estimator [16, 21, 22, 26, 45] to measure one group at a time, but instead measure the sizes of all groups together in one time frame whose slots are shared: Each group ID is pseudo-randomly hashed to a certain number of slots in the time frame. Each tag in the group will probabilistically pick one of these slots to transmit. Listening to the channel, the reader converts the time frame into a bitmap. For each group, it extracts the bits that the group ID is hashed to. Those bits form the *logical bitmap* of the group, from which the group size is estimated. In this approach, each bit and the corresponding slot may be shared by more than one group. This sharing introduces noise; the logical bitmap of one group may carry some bits that are set to ‘1’ not by transmissions of tags in this group, but by transmissions of tags from other groups that happen to be hashed to the same time slots. Fortunately, in a bird’s-eye view, all slots are shared by all groups uniformly at random (through independent hashing), which means the noise is uniformly distributed in the whole time frame. Such uniform noise is measurable. To estimate the size of a group, we will use the logical bitmap of that group, but subtract the noise that tags from other groups introduce.

To further improve performance, the reader repeats the above approach multiple times to gather multiple independent logical bitmaps for each group, and estimation based on multiple logical bitmaps reduces the variance of the result.

Sharing slots saves time. For example, if we share each slot among 30 groups on average (as we observe in a typical simulation of Section 5.4), we will be able to achieve a factor of 30 reduction in execution time. However, sharing slots cause noise among groups during size estimation. Although the noise is statistically uniformly distributed, its variance requires us to repeat for additional logical bitmaps in order to average out the noise variance, which means long execution time. Fortunately, the time saved by sharing outweights the time needed for noise removal as we will demonstrate later.

5.3.2 Overview

Our threshold-based classification (TBC) protocol consists of three phases: the parameter-precomputing phase, the frame phase, and the report phase. The parameter-precomputing phase computes system parameters for optimal performance of the protocol. Using these parameters, the frame phase makes $w(\geq 1)$ polling requests, each of them followed by a time frame, during which tags of all groups transmit in selected slots. The reader converts each time frame into a bitmap, from which logical bitmaps are extracted. Using these logical bitmaps, the report phase employs the Maximum Likelihood Estimation (MLE) method to report the above-threshold groups.

There are four system parameters: 1) w is the number of pollings (or time frames), 2) p_1 is a sampling probability, 3) f_1 is the size of each time frame, i.e., the number of slots in a frame or the number of bits in the bitmap that the frame is converted to, and 4) m_1 is the size of each logical bitmap. Clearly, $m_1 < f_1$. The sampling probability is introduced so that not all tags have to participate in each polling unless $p_1 = 1$; each tag will have a probability of p_1 to be sampled and transmit in a polling. The execution time of TBC is dominated by the w time frames, which have $w \times f_1$ time slots in total. We want to find the optimal values for w, p_1, f_1 and m_1 such that the constraints in (5–1) are met and the value of $w \times f_1$ is minimized.

Before presenting the parameter-precomputing phase for how the optimal system parameters are computed, we will first describe the frame phase and the report phase because deriving the formulas for optimal system parameters relies on the knowledge of how the frame phase works.

5.3.3 Frame Phase

The frame phase is composed of w pollings, which are performed in a similar way: In the ith polling (where $1 \leq i \leq w$), an RFID reader first broadcasts a request message, including a random number r and the system parameters, p_1, f_1 and m_1. The request message also serves the purpose of synchronizing the clocks of all tags for starting a time frame of f_1 slots right after the request.

Consider an arbitrary tag t in an arbitrary group g. The tag decides with a probability p_1 for whether to participate in the current polling. If it decides not to, it will keep silent until the next polling request. If the tag decides to participate, it computes a hash value $H(g_{id} \bigoplus F(r_i, H(t_{id}) \mod m_1))$ as the index of the time slot selected for its transmission, where $H(\cdot)$ is a hash function whose range is $[0, f_1 - 1]$, g_{id} is the tag's group ID, t_{id} is the tag's member ID, and $F(x, y)$ is a pseudo-random number generator that takes two input parameters: x and y. $F(x, y)$ uses x as the seed, generates y random numbers, and outputs the yth number. The transmissions from all participating tags form a bitmap B_i.

Clearly, for tags of group g, the indices of their selected slots in the frame can only be $H(g_{id} \bigoplus F(r_i, 0))$, $H(g_{id} \bigoplus F(r_i, 1))$, ... , $H(g_{id} \bigoplus F(r_i, m_1 - 1))$. These slots or more precisely, the bits converted from these slots, form the logical bitmap of g, denoted as $LB_i(g)$.

Note that the value of $H(t_{id}) \mod m_1$ gives the index of the corresponding bit in the logical bitmap. For example, if a tag selects the $H(g_{id} \bigoplus F(r_i, H(t_{id}) \mod m_1))$th slot to transmit, the $(H(t_{id}) \mod m_1)$th bit in the logical bitmap will be set to '1'. Essentially, we embed the logical bitmaps of all in B_i.

We point out that the complexity of our protocol is mostly placed at the RFID reader, which has to compute the optimal system parameters, initiate the protocol, receive tag transmissions, and perform classification (see the next two subsections). The tag's operation is relatively simple: receiving a request from the reader, performing hash, and transmitting in a time slot. To compute $H(g_{id} \bigoplus F(r_i, H(t_{id}) \mod m_1))$, we expect tags to implement a pseudo-random number generator F as required by [14]. A hash function may be implemented from F by using the hash input as the input to F. There are other simple ways of implementing a hash function for tags, such as [24], which uses a pre-stored bit ring to produce hash output.

Our protocol cannot be directly supported by today's off-the-shelf C1G2-compatible tags because the current standard does not support operations on group IDs (such as hashing based on a group ID for the index of a slot). However, we believe future tags (or standards) may be enhanced to support such a protocol. In our case, a new operational code called Classification needs to be defined, group IDs need to be standardized, and operations based on group IDs (such as hashing) need to be implemented on tags.

5.3.4 Report Phase

After w pollings, the reader obtains w bitmaps, B_i, $1 \leq i \leq w$. It sends the bitmaps to an offline data processing module. There, the logical bitmaps of each group is extracted. For an arbitrary group g, we extract a logical bitmap $LB_i(g)$ from B_i as follows: Set the jth bit of $LB_i(g)$ to be the $H(g_{id} \bigoplus F(r_i, j))$th bit in B_i, i.e., $LB_i(g)[j] = B_i[H(g_{id} \bigoplus F(r_i, j))]$, where $1 \leq i \leq w$ and $0 \leq j \leq m_1 - 1$.

Let x_i be the number of zeros observed in $LB_i(g)$. Let n_1 be the total number of tags in the system and k_1 be the actual population of group g. Below we derive the formula to compute an estimate $\hat{k_1}$ of the population.

Consider the ith polling in the frame phase and an arbitrary bit b in $LB_i(g)$. A tag in group g has a probability of $\frac{p_1}{m_1}$ to select this bit and set it to '1' because the tag is sampled with probability p_1 and if sampled, it only sets one of the m_1 bits in the logical bitmap of g. Any tag in other groups has a probability of $\frac{p_1}{f_1}$ to set this bit to '1' due to dynamic slot sharing across the whole frame. Hence, the probability for b to remain zero is

$$q = (1 - \frac{p_1}{f_1})^{n_1 - k_1}(1 - \frac{p_1}{m_1})^k_1. \tag{5–2}$$

Hence, the likelihood function L_i for us to observe x_i bits of zeros in $LB_i(g)$ is

$$L_i = ((1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})^{k}_{1})^{x_i} \times (1-(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})^{k}_{1})^{m_1-x_i}. \tag{5–3}$$

The likelihood function L for us to observe all x_i values, $1 \le i \le w$, in the w logical bitmaps is $L = \prod_{i=1}^{w} L_i$. That is,

$$L = \prod_{j=1}^{w} \left[((1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})^{k}_{1})^{x_i} \times (1-(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})^{k}_{1})^{m_1-x_i} \right]. \tag{5–4}$$

We want to find an estimate $\hat{k}_1$ that maximizes L, namely,

$$\hat{k}_1 = \underset{k_1}{\text{arg max}}\{L\}. \tag{5–5}$$

Since the maximum is not affected by monotone transformations, we take the logarithm of both sides of (5–4):

$$\ln(L) = \sum_{i=1}^{w} \left[x_i((n_1-k_1)\ln(1-\frac{p_1}{f_1}) + k_1\ln(1-\frac{p_1}{m_1})) + (m_1-x_i)\ln(1-(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})^{k_1})) \right]. \tag{5–6}$$

Differentiating both sides of the above equation, we have

$$\frac{\partial \ln L}{\partial k_1} = \sum_{i=1}^{w} \left[(\frac{x_i - m_1(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})^{k_1}}{1-(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})^{k_1}}) \times (\ln(1-\frac{p_1}{m_1}) - \ln(1-\frac{p_1}{f_1})) \right]. \tag{5–7}$$

After setting the right side to zero and simplifying it, we have the following estimator,

$$\hat{k}_1 = \frac{\ln(\frac{\sum_{i=1}^{w} x_i}{mw(1-\frac{p_1}{f_1})^{n}_{1}})}{\ln(\frac{1-\frac{p_1}{m_1}}{1-\frac{p_1}{f_1}})}. \tag{5–8}$$

In (5–7), m_1 and f_1 are given parameters whose values are pre-computed by the reader. The values of x_i, $1 \leq i \leq m_1$, are obtained from $LB_i(g)$. The total number n_1 of tags can be estimated from the bitmaps, B_i, $1 \leq i \leq w$. Let X_i be the number of zeros in B_i. The probability for each tag to be sampled and set a certain bit in B_i to '1' is $\frac{p_1}{f_1}$. The probability for each bit to remain zero is approximately $(1 - \frac{p_1}{f_1})^n_1$. The likelihood function for us to observe X_i zeros in B_i, $1 \leq i \leq w$, is

$$\mathcal{L} = \prod_{i=1}^{w}(1 - \frac{p_1}{f_1})^{n_1 X_i}(1 - (1 - \frac{p_1}{f_1})^n_1)^{f_1 - X_i} \tag{5–9}$$

Using the maximum likelihood estimation, we take the logarithm of both sides, differentiate it, and then let the right side be zero. We have

$$\sum_{i=1}^{w} X_i - wf_1(1 - \frac{p_1}{f_1})^n_1 = 0$$

$$n_1 = \frac{\ln \frac{\sum_{i=1}^{w} X_i}{wf}}{\ln(1 - \frac{p_1}{f_1})}, \tag{5–10}$$

where X_i, $1 \leq i \leq w$, are obtained from B_i.

For each group g, after we estimate its population $\hat{k}_1$ based on (5–8), we report the group (as an above-threshold group) if $\hat{k}_1 \geq T$, where T is another system parameter that will be determined in the next subsection based on the probabilistic performance objectives (5–1).

5.3.5 Parameter-Precomputing Phase

We first develop the constraints that the system parameters must satisfy in order to achieve the probabilistic performance objectives. Based on the constraints, we determine the optimal values for the length m_1 of logical bitmaps, the number w of pollings, the frame size f_1, and the parameter T.

A group g whose estimated population is $\hat{k}_1$ will be reported if

$$\hat{k}_1 \geq T. \tag{5–11}$$

That is,

$$\frac{\ln(\frac{\sum_{i=1}^{w} x_i}{mw(1-\frac{p_1}{f_1})_1^n})}{\ln(\frac{1-\frac{p_1}{m_1}}{1-\frac{p_1}{f_1}})} \geq T$$

$$\sum_{i=1}^{w} x_i \leq wm(\frac{1-\frac{p_1}{m_1}}{1-\frac{p_1}{f_1}})^T(1-\frac{p_1}{f_1})_1^n. \tag{5–12}$$

Let $C = wm(\frac{1-\frac{p_1}{m_1}}{1-\frac{p_1}{f_1}})^T(1-\frac{p_1}{f_1})_1^n$. Therefore, the probability for the reader to report a group is $Prob(\hat{k}_1 \geq T) = Prob(\sum_{i=1}^{w} x_i \leq C)$.

From (5–2), we know that x_i follows the binomial distribution with parameters m_1 and $(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k$:

$$x_i \sim Bino(m_1, (1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k). \tag{5–13}$$

Since a binomial distribution $Bino(a, b)$ can be excellently approximated by a normal distribution $Norm(ab, ab(1-b))$ when a is large enough (which is the case for m_1), (5–13) can be approximately written as

$$\begin{aligned} &x_i \sim Norm(m_1(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k, \\ &\quad m_1(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k(1-(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k)). \end{aligned} \tag{5–14}$$

According to (5–14), we know

$$\begin{aligned} &\sum_{i=1}^{w} x_i \sim Norm(mw(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k, \\ &\quad mw(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k(1-(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k)). \end{aligned} \tag{5–15}$$

Let $\mu = mw(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k$ and $\sigma^2 = mw(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k(1-(1-\frac{p_1}{f_1})^{n_1-k_1}(1-\frac{p_1}{m_1})_1^k)$, then we have

$$Prob(\sum_{i=1}^{w} x_i = j) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(j-\mu)^2}{2\sigma^2}} \tag{5–16}$$

Thus,

$$Prob(\hat{k_1} \geq T) = Pro(\sum_{i=1}^{w} x_i \leq C)$$
$$= \sum_{j=0}^{C} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(j-\mu)^2}{2\sigma^2}} \tag{5–17}$$

The first performance objective in (5–1) can be translated into $Prob(\hat{k_1} \geq T | k_1 \geq h) \geq \alpha_1$, which is

$$\sum_{j=0}^{C} \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-(j-\mu)^2}{2\sigma^2}} \geq \alpha_1 \tag{5–18}$$

where $k_1 \geq h$. Since the left side of the inequality is an increasing function of k_1, we can replace the term k_1 with h. Then, we have the first constraint for the system parameters.

$$\sum_{j=0}^{C} \frac{1}{\sqrt{2\pi{\sigma_1}^2}} e^{\frac{-(j-\mu_1)^2}{2{\sigma_1}^2}} \geq \alpha_1, \tag{5–19}$$

where $\mu_1 = mw(1 - \frac{p_1}{f_1})^{n_1-h}(1 - \frac{p_1}{m_1})^h$ and ${\sigma_1}^2 = \mu_1(1 - (1 - \frac{p_1}{f_1})^{n_1-h}(1 - \frac{p_1}{m_1})^h)$.

Similarly, the second performance objective in (5–1) can be translated into the following constraint,

$$\sum_{j=0}^{C} \frac{1}{\sqrt{2\pi{\sigma_2}^2}} e^{\frac{-(j-\mu_2)^2}{2{\sigma_2}^2}} \leq \beta_1, \tag{5–20}$$

where $\mu_2 = mw(1 - \frac{p_1}{f_1})^{n_1-l}(1 - \frac{p_1}{m_1})^l$ and ${\sigma_2}^2 = \mu_2(1 - (1 - \frac{p_1}{f_1})^{n_1-l}(1 - \frac{p_1}{m_1})^l)$.

We want to find optimal system parameters that minimize the execution time required by TBC, i.e., $w \times f_1$, subject to the above two constraints.

$$
\begin{aligned}
&\text{Minimize } w \times f_1, \\
&\text{subject to } \sum_{j=0}^{C} \frac{1}{\sqrt{2\pi{\sigma_1}^2}} e^{\frac{-(j-\mu_1)^2}{2{\sigma_1}^2}} \geq \alpha_1 \\
&\sum_{j=0}^{C} \frac{1}{\sqrt{2\pi{\sigma_2}^2}} e^{\frac{-(j-\mu_2)^2}{2{\sigma_2}^2}} \leq \beta_1 \\
&C = mw\left(\frac{1-\frac{p_1}{m_1}}{1-\frac{p_1}{f_1}}\right)^T\left(1-\frac{p_1}{f_1}\right)_1^n \\
&\mu_1 = mw\left(1-\frac{p_1}{f_1}\right)^{n_1-h}\left(1-\frac{p_1}{m_1}\right)^h \\
&{\sigma_1}^2 = \mu_1\left(1-\left(1-\frac{p_1}{f_1}\right)^{n_1-h}\left(1-\frac{p_1}{m_1}\right)^h\right) \\
&\mu_2 = mw\left(1-\frac{p_1}{f_1}\right)^{n_1-l}\left(1-\frac{p_1}{m_1}\right)^l \\
&{\sigma_2}^2 = \mu_2\left(1-\left(1-\frac{p_1}{f_1}\right)^{n_1-l}\left(1-\frac{p_1}{m_1}\right)^l\right).
\end{aligned}
\tag{5–21}
$$

The parameters h, l, α_1 and β_1 are given by the performance objectives (5–1). To solve the above constrained optimization problem, we need to determine the optimal values of the remaining five system parameters p_1, w, f_1, m_1 and T, such that $w \times f_1$ is minimized. We can approximately solve (5–21) through searching a preset parameter space. See Section 5.4 for the search algorithm used in our simulations and the resulting protocol performance. The value of n_1 can be estimated through an estimation protocol [21, 43, 59]. Moreover, once the performance objectives are decided, we can precompute the system parameters (p_1, w, f_1, m_1 and T) for different values of n_1 (e.g., at steps of 500). After the current number of tags in the whole system is estimated, the system parameters can be quickly looked up from the precomputed results.

5.4 Numerical Results

5.4.1 Setting

We evaluate the performance of TBC and compare it with the Group Testing (GT) [45], the Average Run based Tag estimation (ART) scheme [43] and the Zero-One

Estimator (ZOE) [59]. GT is the most related work. It probabilistically identifies populous groups whose sizes are larger than a threshold. We denote the proposed protocol TBC with the optimal sampling probability p_1^* as TBC(p_1^*), and TBC with $p_1 = 100\%$ as TBC(100%), which is a special case of TBC where sampling is not applied, as is published in the conference version of this dissertation [30]. We want to see how much improvement can be achieved through optimal sampling. ZOE and ART are designed for RFID population estimation, not for satisfying the probability performance objectives in (1). However, the estimation results from these two estimators can be used for classification by reporting those groups whose estimated sizes are above a threshold. More specifically, they estimate the group sizes one group at a time. For each group, they progressively improve the size estimation $\hat{k}_1$. We will reject a group when $\hat{k}_1 \leq \frac{h+l}{2}$ and the probability of $|k_1 - \hat{k}_1| > h - \hat{k}_1$ is no greater than $1 - \alpha_1$. We will accept a group when $\hat{k}_1 \geq \frac{h+l}{2}$ and the probability of $|k_1 - \hat{k}_1| > \hat{k}_1 - l$ is no greater than β_1.

Our simulation parameters are set based on the typical setting of the EPCglobal Gen-2 standard [14]. Any two consecutive transmissions (from a reader to tags or from a tag to the reader) are separated by a waiting time of 266.4 μs. According to the specification, the transmission rate from a tag to the reader is the same as the transmission rate from the reader to a tag. The rate from a tag to the reader is 40.97 Kb/sec; it takes 24.4 μs for a tag to transmit one bit. The length of a slot is calculated as the sum of a waiting time and the time for the tag to transmit a certain number of bits. The type of slots used by TBC, ART and ZOE, denoted as T_{short}, contains only one bit. Its length is 290.8 μs. The type of slots used by GT needs to detect collision and contains 10 bits. Their slot length is $T_{long} = 470.5\mu s$. Comparing with the total amount of time used by all tags to transmit to the reader, the time used by the reader to broadcast information to the tags is negligible in all three protocols.

For TBC(p_1^*), we approximately compute (5–21) with five loops for p_1 from 0 to 1 at steps of 0.001, w from zero to 50, f_1 from 0 to 10n_1, m_1 from 0 to 1000, and T from

l to h. The ranges are set based on our empirical investigation which shows that the optimal parameters persistently fall within these ranges. We find the best value of $w \times f_1$ (together with the corresponding system parameters) in these ranges. Once $w \times f_1$ is determined, the execution time is known, which is $T_{short} \times w \times f_1$ plus the time of estimating the value of n_1 using ZOE; note that even through n_1 is pre-determined for each simulation, we assume that the reader does not know this value before hand and it runs ZOE once to estimate n_1 with $\pm 5\%$ error at 95% confidence level. TBC(100%) works just like TBC(p_1^*), except that it is operated under the sampling probability $p_1 = 1$. GT will also compute its optimal system parameters, including the time frame size f_1, the number R of rounds, and the number of shuffled groups W. The execution time required by GT is $T_{long} \times f_1 \times W \times R$. The parameters of ART and ZOE are set based on their original papers.

In our simulations, $n_1 = 500,000$, the range of group sizes is (0, 500], $h = 250$, and the value of l varies. There are 2,000 groups. The number x of above-threshold groups (whose sizes are greater than h) may vary in some simulations, but its default value is 1,000. Besides randomly choosing the size of each above-threshold group from [250, 500] and that of each below-threshold group from [1, 249] in one simulation, our default way of determining group sizes is given as follows: We first randomly choose the sizes for the above-threshold groups from [250, 500]. After that, we distribute the remaining M tags into the below-threshold groups. For the first below-threshold group, we generate a random number between 1 and $\min\{249, \frac{M}{2000-x}\}$ to be its population, which is denoted as s_1. For the second below-threshold group, we select a random value between 1 and $\min\{249, \frac{M-s_1}{1999-x}\}$ as its population, which is denoted as s_2. Similarly, we assign a random value between 1 and $\min\{249, \frac{M-s_1-s_2}{1998-x}\}$ as the population for the third below-threshold group. This process is repeated for all remaining below-threshold groups. If there are still tags left unassigned, we assign them arbitrarily to below-threshold groups as long as their sizes are below 250.

5.4.2 Execution Time Required with Respect to α_1, β_1 and l/h

We compare TBC(100%), TBC(p_1^*), GT, ART and ZOE in terms of execution time. Tables 5-2 – 5-5 show our simulation results under different values of l/h, α_1 and β_1.

Table 5-2 shows the execution time required when $\alpha_1 = 99.9\%$ and $\beta_1 = 0.1\%$. From the table, we can see that TBC(100%) has a much smaller execution time than GT, ART and ZOE. For example, GT takes $33'43''$ when $l = 0.5h$, which is about triple of the time taken by TBC(100%). When sampling is introduced, the performance of TBC improves, i.e., TBC(p_1^*) takes less time to classify the above-threshold groups under the same setting - only 27.2% of the time taken by GT. ART and ZOE consume an order of magnitude or more time than TBC(p_1^*).

When l/h becomes larger, TBC(100%), TBC(p_1^*), GT, ART and ZOE need more time to classify the above-threshold groups. This is because a larger ratio of l/h means a higher accuracy requirement for classification. The performance gain by TBC(p_1^*) over GT, ART, ZOE and TBC(100%) shrinks as l/h increases, but remains significant. For example, when $l = 0.7h$, the execution time required by TBC(p_1^*) is 31.2% of the time by GT. When $l = 0.9h$, the time by TBC(p_1^*) is 34.1% of that by GT.

GT uses a simple, fast threshold checking scheme to probabilistically identify populous groups with size larger than a threshold. However, it incurs a large variance in its estimated result. To satisfy a high accuracy requirement, a large number of executions are required, which lengthens execution time. In addition, GT has to identify whether a slot is empty, singleton or collision, resulting in longer slots. TBC(100%) estimates all group sizes together and share slots among all groups. In addition, it only needs to know whether each slot is empty or not. Hence, its execution time is shorter. Sampling is turned on in TBC(p_1^*), which requires only a fraction of tags to participate in a protocol execution. When the sampling probability p_1 is chosen to be a small value, the number of participating tags are largely reduced, which in turn reduces the frame size

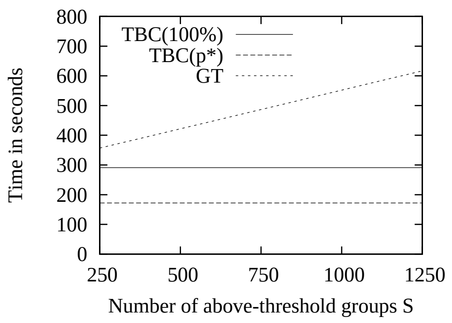

Figure 5-2. Execution time with respect to the number of groups S which are supposed to be reported when $\alpha_1 = 99\%$, $\beta_1 = 1\%$ and $l = 0.7h$. The total number of tags n_1 is fixed to be 500,000 at each point.

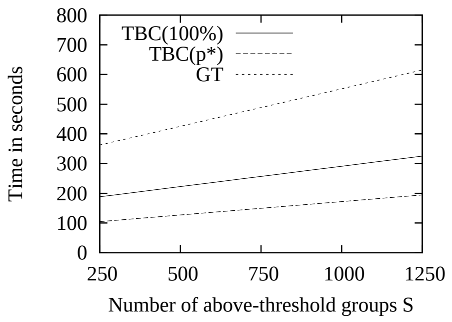

Figure 5-3. Execution time with respect to the number of groups S which are supposed to be reported when $\alpha_1 = 99\%$, $\beta_1 = 1\%$ and $l = 0.7h$. The total number of tags n_1 for all the groups increases along with S.

for each polling. It is not efficient to invoke ART and ZOE to estimate the size of each group one at a time.

Tables 5-3 and 5-5 compare the execution times of the five protocols when $\alpha_1 = 99\%$, $\beta_1 = 1\%$, $\alpha_1 = 95\%$, $\beta_1 = 5\%$, and $\alpha_1 = 90\%$, $\beta_1 = 10\%$, respectively. These three tables show that TBC(p_1^*) outperforms other protocols under different parameter settings. When comparing with Table 5-2, we see that given the same values of h and l, the execution times of all protocols are reduced when α_1 decreases or β_1 increases. But the performance gain by TBC(p_1^*) remains significant.

5.4.3 FPR and FNR with Respect to α_1, β_1 and l/h

We call a group whose size is no more than l (no less than h) as a below-l (above-h) group. The *false positive ratio* (FPR) is defined as the fraction of below-l groups that are mistakenly reported. The *false negative ratio* (FNR) is defined as

Table 5-2. Estimation time comparison when $\alpha_1 = 99.9\%$ and $\beta_1 = 0.1\%$

	Estimation Time in minutes(′), seconds(″)				
	TBC(100%)	TBC(p_1^*)	GT	ART	ZOE
$l = 0.1h$	7′23″	4′43″	20′23″	57′54″	48′10″
$l = 0.3h$	8′59″	6′56″	25′57″	68′18″	57′19″
$l = 0.5h$	10′15″	9′19″	33′43″	81′50″	69′13″
$l = 0.7h$	14′23″	12′12″	42′12″	96′5″	80′30″
$l = 0.9h$	20′34″	17′57″	61′39″	114′19″	95′1″

Table 5-3. Estimation time comparison when $\alpha_1 = 99\%$ and $\beta_1 = 1\%$

	Estimation Time in minutes(′), seconds(″)				
	TBC(100%)	TBC(p_1^*)	GT	ART	ZOE
$l = 0.1h$	1′14″	39″	3′49″	39′25″	26′14″
$l = 0.3h$	1′42″	57″	4′26″	46′31″	28′05″
$l = 0.5h$	3′05″	1′44″	6′15″	61′50″	35′56″
$l = 0.7h$	4′51″	2′52″	9′12″	84′34″	47′15″
$l = 0.9h$	6′21″	4′05″	11′58″	100′24″	59′23″

Table 5-4. Estimation time comparison when $\alpha_1 = 95\%$ and $\beta_1 = 5\%$

	Estimation Time in minutes(′), seconds(″)				
	TBC(100%)	TBC(p_1^*)	GT	ART	ZOE
$l = 0.1h$	44″	18″	1′55″	9′31″	7′47″
$l = 0.3h$	1′6″	29″	2′9″	11′12″	8′23″
$l = 0.5h$	1′47″	1′01″	2′57″	12′10″	9′26″
$l = 0.7h$	2′39″	1′43″	4′11″	14′21″	11′14″
$l = 0.9h$	4′4″	2′56″	6′36″	17′11″	14′32″

Table 5-5. Estimation time comparison when $\alpha_1 = 90\%$ and $\beta_1 = 10\%$

	Estimation Time in minutes(′), seconds(″)				
	TBC(100%)	TBC(p_1^*)	GT	ART	ZOE
$l = 0.1h$	38″	15″	1′33″	5′37″	5′38″
$l = 0.3h$	52″	20″	1′51″	6′19″	5′47″
$l = 0.5h$	1′17″	48″	2′18″	7′56″	6′33″
$l = 0.7h$	1′56″	1′14″	3′6″	8′43″	8′5″
$l = 0.9h$	2′50″	2′25″	4′55″	10′24″	10′11″

Table 5-6. False negative Ratio and False Positive Ratio when $\alpha_1 = 99\%$ and $\beta_1 = 1\%$

	FNR			FPR		
	TBC(100%)	TBC(p_1^*)	GT	TBC(100%)	TBC(p_1^*)	GT
$l = 0.1h$	0.0084	0.0095	0.0098	0.0075	0.0087	0.0086
$l = 0.3h$	0.0084	0.0095	0.0098	0.0075	0.0089	0.0090
$l = 0.5h$	0.0084	0.0095	0.0098	0.0079	0.0089	0.0090
$l = 0.7h$	0.0084	0.0095	0.0098	0.0081	0.0090	0.0090
$l = 0.9h$	0.0084	0.0095	0.0098	0.0089	0.0091	0.0093

Table 5-7. False negative Ratio and False Positive Ratio when $\alpha_1 = 95\%$ and $\beta_1 = 5\%$

	FNR			FPR		
	TBC(100%)	TBC(p_1^*)	GT	TBC(100%)	TBC(p_1^*)	GT
$l = 0.1h$	0.037	0.042	0.046	0.04	0.039	0.041
$l = 0.3h$	0.037	0.042	0.046	0.041	0.042	0.042
$l = 0.5h$	0.037	0.042	0.046	0.043	0.043	0.042
$l = 0.7h$	0.037	0.042	0.046	0.043	0.046	0.042
$l = 0.9h$	0.037	0.042	0.046	0.044	0.047	0.045

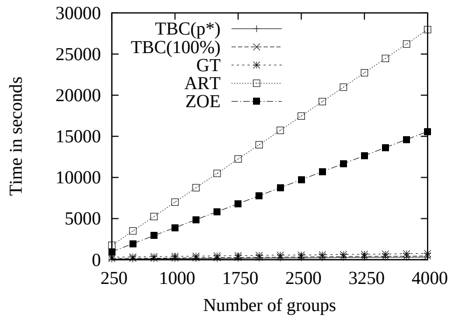

Figure 5-4. Execution time with respect to the number of groups when $\alpha_1 = 99\%$, $\beta_1 = 1\%$, $h = 250$, $l = 0.5h$, and n_1 increases with the number of groups.

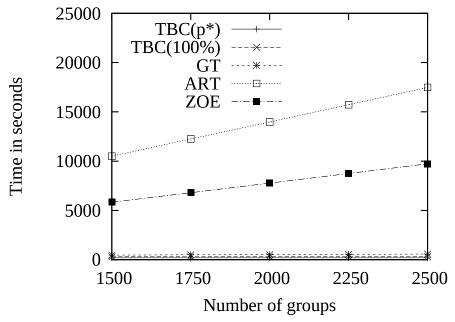

Figure 5-5. Execution time with respect to the number of groups when $\alpha_1 = 99\%$, $\beta_1 = 1\%$, $h = 250$, $l = 0.5h$, and n_1 is fixed at 500000.

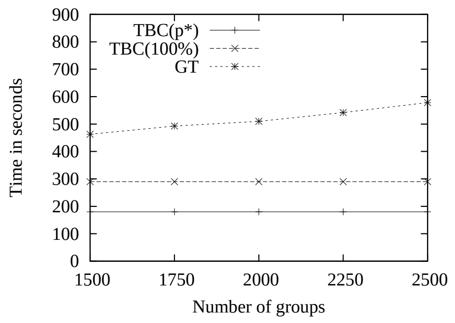

Figure 5-6. Zoom-in of Figure 5-5 for TBC and GT.

the fraction of above-h groups that are not reported. Tables 5-6 – 5-7 present our simulation results of FNR and FPR under different values of α_1, β_1 and l/h. For Table 5-6, $\alpha_1 = 99\%$ and $\beta_1 = 1\%$. The FNR values of TBC(100%), TBC(p_1^*) and GT are consistently smaller than $1 - \alpha_1$ and their FPR values are consistently smaller than β_1, which means that these protocols meet the performance objectives in (1). In addition, we observe that our protocol has smaller FNR and FPR than GT. The results for $\alpha_1 = 95\%$ and $\beta_1 = 5\%$ are Table 5-7, where the values of FPR and FNR also meet the objectives.

5.4.4 Execution Time Required with Respect to Above-Threshold Groups

In the previous comparison, the number of above-threshold groups is set at the default value 1,000. We further compare TBC(100%), TBC(p_1^*) and GT by varying the number of above-threshold groups, denoted as S. Let $\alpha_1 = 99\%$ and $\beta_1 = 1\%$. In Fig. 5-2, we keep $n_1 = 500,000$ and vary the the number of above-threshold groups from 250 to 1,250. From the figure, the execution time of GT is linear in S, but TBC(100%) and TBC(p_1^*) are different. Not only do they outperform GT, but also their execution times are both insensitive to S. As long as the total number of tags in the system is the same, their execution times can be approximately viewed as constants even when the number of groups is different. Such an observation agrees with (5–21), which does not include S in its formulation. Furthermore, thanks to sampling, it takes TBC(p_1^*) less time to classify the above-threshold groups than TBC(100%), for any value of S.

In Fig. 5-3, we fix the number of groups to 2,000, while allowing the total number of tags to change. Each below-threshold group takes a random population in the range of $(0, 250)$ and each above-threshold group takes a random population in the range of $[250, 500]$. From the figure, we observe that the execution times of TBC(100%), TBC(p_1^*), and GT are approximately proportional to the number of above-threshold groups. However, the lines of TBC(p_1^*) and TBC(100%) have somewhat smaller slopes than the line of GT.

5.4.5 Execution Time Required with Respect to The Number of Groups

We let the number of groups in the system to increase from 250 to 2000, with their group sizes randomly selected from $(0, 500]$. Fig. 5-4 shows the execution times of the five protocols with respect to the number of groups. All the protocols take longer time to classify more groups. However, the execution times of TBC(p_1*) and TBC(100%) increase with smaller slopes, and therefore they are more scalable than ZOE, ART and GT. For example, when there are 4,000 groups, the time for ART is 27,974 seconds, the time for ZOE is 15,582 seconds, the time for GT is 753 seconds, and the time for TBC(100%) is 451 seconds, and the time for TBC(p_1*) is 315 seconds.

Next we show an interesting property of TBC when we fix the total number of tags in the system at 500,000 but vary the number of groups. Fig. 5-5 shows the execution times of the five protocols with respect to the number of groups. The times of ART and ZOE still grow roughly linearly with the number of groups. We replot the curves of GT, TBC(p_1*) and TBC(100%) in Fig. 5-6 for a better view. The execution time of GT grows with respect to the number of groups. But the times of TBC(p_1*) and TBC(100%) stay flat, insensitive to the number of groups when the total number of tags is unchanged. The reason is simple: Each group is assigned a certain number of slots, and each tag in other groups may cause noise in one of those slots due to sharing. For TBC, the overall noise only depends on the total number of tags in other groups, not the number of other groups.

5.5 Summary

This dissertation proposes a new solution for multigroup threshold-based classification in a large RFID system. While much of the prior work focuses on estimating the total number of tags in a system, it is inefficient to apply those solutions to sequentially estimating the size of each tag group and see if it is above a threshold. In this dissertation, we propose a new protocol based on logical bitmaps that allow the sizes of all groups to be estimated together for classification. Slot sharing is exploited to reduce the execution time. Furthermore, sampling is introduced to reduce the number of participating tags (and thus collision), which in turn cuts down the execution time. Reducing the number of participating tags also saves the overall energy expenditure by the tags if battery-powered active tags are used. Our new protocol is able to perform multigroup classification with any pre-set accuracy. We evaluate the proposed solution and demonstrate that it compares favorably with the best existing work. We also present the method of computing the optimal system parameters.

CHAPTER 6
IMPLEMENTATION

6.1 System Setup

We implement a prototype system to validate TRP - a simple missing-tag detection protocol, and a simple multi-group classification protocol, using programmable MOO tags based on the WISP4.1 hardware and firmware, as shown in Fig 6-1. The MOO tag mainly consists of an RFID circuitry and an ultra-low power 16-bit MSP430 microcontroller, which are used to harvest power and backscatter radio signals. In addition, we used an Impinj Speedway RFID reader to interrogate with the MOO, with the reader's transmit power set to +30 dBm and a 6-dBi circularly polarized patch antenna attached. Current MOO firmware has partially implemented the EPCglobal Gen-2 standard operating in the 902-928 MHz industrial-scientific-medical(ISM) band [14]. We extend the EPCglobal C1G2 protocol with the functionality of TRP and cardinality estimation. Actually, the implementation of the TRP protocol and cardinality estimation only requires a slight extension to the EPCglobal C1G2 protocol.

We find that an Aloha-based anti-collision polling scheme has been standardized by EPCglobal where the reader begins each interrogation round by informing all the tags about the frame size. Each tag then chooses a time slot at random and transmits only within that time slot. More specifically, in EPCglobal C1G2 standard [14], a reader initiates each communication round with tags. The reader transmits an operation code (e.g., Query, Write, Select, ACK etc.) indicating the expected operation of tags, the backscatter bit rate, and tag encoding schemes (e.g., FM0 or Miller). For example, the reader may initiate communication by sending tags a Query command, which includes a field that sets the number f_1 of slots in the round. In addition, this command also involves other parameters that can be used to negotiate with the tags about the length of each slot and the waiting time between consecutive slots. Upon receiving the Query, each tag should pick a random value in the range $[0, 2_1^f)$, and load this value into its slot

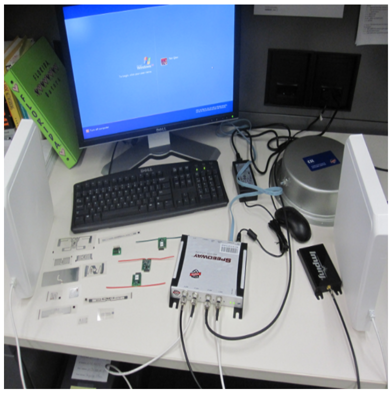

Figure 6-1. RFID System

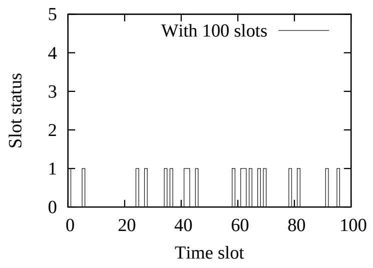

Figure 6-2. Time frame length = 100 slots.

counter. It decrements the slot counter each time when receiving a QueryRep from the reader. If its slot counter is decreased to zero, it replies to the reader; otherwise the tag shall remain silent. The Fig. 6-2 and Fig. 6-3 show an example of replies from tags using anti-collision polling scheme with 20 tags. The horizontal axis is the slot sequence number, and the vertical axis is the magnitude of replies. We normalize the transmission signal of tags, and use '1' to represent the signal strength when there is any reply in a slot. From Fig. 6-2, we can see the 20 tags reply in 20 slots out of 100, resulting in no collision. When the frame length is only 30, the collision happens, which can be observed from Fig. 6-3 where there are 5 collision slots.

6.2 A Simple Missing-tag Detection Porotocol Implementation

We implement the TRP protocol by following the conventional reader-initiated approach. We first add the Detect command into the command set of the standard. To detect the missing tag event, the reader initiates counting procedure by sending a Detect command along with other parameters (r, and encoding scheme, etc). In the case that

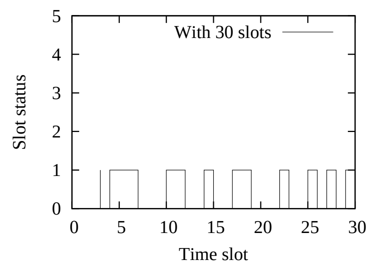

Figure 6-3. Time frame length = 30 slots.

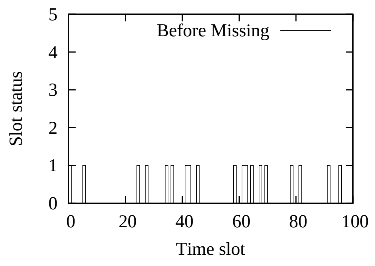

Figure 6-4. Before five tags are Mssing.

the operation code is Detect, each tag computes a hash value with the random number and its tag ID, then transmits a short response according to the encoding scheme. We did the experiment with twenty MOO tags, which is shown in Fig. 6-4 and Fig. 6-5. The former figure shows the slot status in the case that all tags are in the system, while the latter one presents the results when five tags are missing. From Fig. 6-4, we can see the reader receives replies from all twenty tags, which can be identified by twenty singleton slots. In Fig. 6-5, the reader gets only 15 response because 5 tags are missing, which can be observed from the 15 non-empty slots.

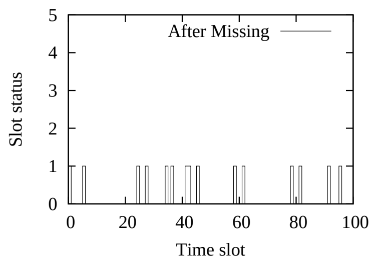

Figure 6-5. After five tags are Missing.

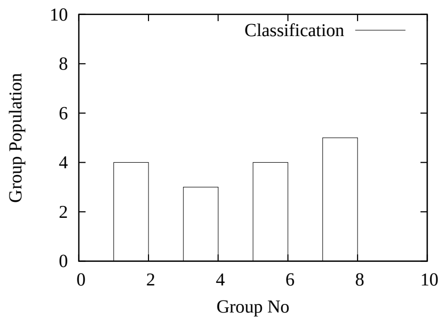

Figure 6-6. The population estimation of four groups.

6.3 A Simple Multi-group Classification Porotocol Implementation

In this section, we want to show the results of our simple multi-group classification protocol implementation. We still use a MOO to simulate five virtual MOOs, and the virtual MOOs simulated on the same physical MOO belongs to the same group (with the same group ID). For each group, an individual time frame is used to estimate its tag population by counting the number of non-empty slots in the frame. The result is shown in Fig. 6-6, where the horizontal axis is the group sequence number, and the vertical axis is the estimated group population. From the figure, we can see that the estimated number of tags in the first group is 4, less than the actual value. The reason is that two tags reply to the reader at the same slot. The second group is estimated to have 3 tags, the third group has four tags and the fourth one has five tags. If we set up the threshold value to be 3, all the gorup excep the second one will be reported as above-threshold groups.

6.4 What's to be expected?

From the above results, we can see it is possible to implement protocols to detect missing-tag event or classify tag groups in the scenario of small-scale systems. The implementation actually can be scaled to work in a large RFID system, just by expanding the time frame to have more time slots. In this case, each tag works in the same way to select a slot randomly and reply. Based on the status of time slots, the missing tag can be detected, or the number of tags in each tag group can be estimated and applied to classify different tag groups.

CHAPTER 7
CONCLUSION AND PROPOSAL FOR FUTURE WORK

7.1 Conclusions

In this dissertation, we first proposes a new protocol design that integrates energy efficiency and time efficiency for missing-tag detection. It uses multiple hash seeds to provide multiple degrees of freedom for the RFID reader to assign tags to singleton slots, during which the tags announce their presence in the process of missing-tag detection. We also present this new protocol with reliable channels. The result is a multi-fold cut in both energy cost and execution time. In addition, we extend the protocol to consider two categories of channel errors induced by noise/interference in the environment. The involving of channel errors will make the energy/time gains slightly reduced, but remain significant comparing to EMD and TRP. We also reveal a fundamental energy-time tradeoff in the protocol design. This tradeoff gives flexibility in performance tuning when the protocol is applied in practical environment.

We then focus on designing a new protocol that can solve the multigroup threshold-based classification problem in a large RFID system. While much of the prior work focuses on estimating the total number of tags in a system, it is inefficient to apply those solutions to sequentially estimating the size of each tag group and see if it is above a threshold. In this dissertation, we propose a new protocol based on logical bitmaps that allow the sizes of all groups to be estimated together for classification. Slot sharing is exploited to reduce the execution time. Furthermore, sampling is introduced to reduce the number of participating tags (and thus collision), which in turn cuts down the execution time. Reducing the number of participating tags also saves the overall energy expenditure by the tags if battery-powered active tags are used. Our new protocol is able to perform multigroup classification with any pre-set accuracy. We evaluate the proposed solution and demonstrate that it compares favorably with the best existing work. We also present the method of computing the optimal system parameters.

In the end, we implement a simple missing-tag detection protocol and multi-group threshold-based classification in the physical devices - RFID MOO tags and speedway RFID readers. The results show that the reader can cooperate with tags to detect missing-tag event based on the simple aloha-based protocol. Futuremore, by estimating the number of tags in each group, the reader is also able to classify tag groups in a small-scale scenario.

7.2 Future Work

As can be seen, the proposed protocol in chapter 4 can be used to solve the problem of RFID multigroup threshold-based classification. In this dissertation, we present how to classify RFID tag groups based on dynamic sloting sharing, and then show analytically how to compute optimal system parameters that minimize the protocol execution time under the constraint of the requirement. We may extend this problem from two different angles:

1) Consider a large warehouse with lots of products, where a reader is not able to cover the whole warehouse to classify all the products. In this scenario, we may need to install multiple readers so that each reader is responsible for the products within its interrogator region. However, the coverage of nearby readers may overlap, and therefore tags in a group may appear in the ranges of different readers. When the readers execute TBC(p^*) or TBC(100%) separately, the group may be classified into different categories by different readers, relying on the number of tags covered. For example, a group is in the interrogator range of two readers, A and B. The reader A is monitoring 300 tags of this group, while the reader B has 100 tags. Suppose the threshold value is 200. Classified by our protocol, the portion covered by A is identified as an above-threshold group, and the other portion is a normal group. Hence, it is necessary to come up with a better solution that can address such disruption classification results generated by by TBC(p^*) or TBC(100%), due to the intersection covered by two or more nearby readers.

2) In practice, different groups may need to be classified based on different thresholds. For example, in a warehouse that stores shoes, computers and other products, the thresholds for shoes and computers may be different because their expected inventory levels may not be the same. The problem becomes classifying the groups into different ranges based on their sizes. When there are multiple thresholds, we first place tag groups in subsets, each of which corresponds to a threshold. We then perform single-threshold classification within each subset. To do so, the reader broadcasts the group IDs in the subset, and the classification is performed among the tags that carry one of those IDs. This solution is easy to implement based on our proposed protocol rather than optimized because we see the performance degrades when there are a large number of thresholds. The reason is that the execution time is insensitive to the the number of groups. When the whole system is divided into a large number of subsets based on different thresholds, the total execution time can be calculated by the product of the number of thresholds and the execution time to classified groups in a subset. For example, we suppose a subset contains groups between 1500 and 2500. Observed from Fig.5-5, it takes TBC(p^*) 180 seconds to classify groups in a subset when $\alpha = 99\%, \beta = 1\%, h = 250, l = 0.5h$, and n is fixed at 500,000. If we have 20 threshold values, the total execution time is $20 \times 180 = 3,600$ seconds, which is an intolerant time duration. Hence, the RFID multigroup threshold-based classification with different thresholds is still under exploration, and the future work should focus on finding a better solution that can cut down the time required.

REFERENCES

[1] "Dirac delta function." *http://en.wikipedia.org/wiki/Dirac_delta_function* (2000).

[2] "Information technology automatic identification and data capture techniques C radio frequency identification for item management air interface - part 6: parameters for air interface communications at 860-960 MHz." *Final Draft International Standard ISO 18000-6* (2003).

[3] Barcode. "Barcode." *http://en.wikipedia.org/wiki/Barcode* (2014).

[4] Bhandari, N., Sahoo, A., and Iyer, S. "Intelligent Query Tree (IQT) Protocol to Improve RFID Tag Read Efficiency." *Proc. of IEEE International Conference on Information Technology* (2006).

[5] Bustillo, M. "Wal-mart Radio Tags to Track Clothing." *http://online.wsj.com/article/SB10001424052748704421304575383213061198090.html* (2010).

[6] Buzzle.com. "Passive Vs. Active RFID Tags." *http://www.buzzle.com/articles/passive-vs-active-rfid-tags.html* (2011).

[7] Capetenakis, J. I. "Tree Algorithms for Packet Broadcast Channels." *IEEE Transactions on Information Theory* 25 (1979).5: 505–515.

[8] Cha, J. and Kim, J. "Novel Anti-collision Algorithms for Fast Object Identification in RFID System." *Proc. IEEE ICPADS* (2005).

[9] Cha, J. R. and Kim, J. H. "Dynamic Framed Slotted ALOHA Algorithms Using Fast Tag Estimation Method for RFID Systems." *Proc. of IEEE Consumer Communications and Networking Conference* (2006).

[10] Chen, S. and Nahrstedt, K. "Maxmin Fair Routing in Connection-Oriented Networks." *Proc. of Euro-Parallel and Distributed Systems Conference* (1998): 163–168.

[11] Chen, S. and Shavitt, Y. "SoMR: A scalable distributed QoS multicast routing protocol." *Journal of Parallel and Distributed Computing* 68 (2008).2: 137–149.

[12] Choi, H., Cha, J., and Kim, J. "Fast Wireless Anti-collision Algorithm in Ubiquitous ID System." *Proc. IEEE VTC* (2004).

[13] Cornaglia, B. and Spini, M. "Letter: New statistical model for burst error distribution." *European Transactions on Telecommunications* 7 (1996).3: 267–272.

[14] EPCglobal. "EPC Radio-Frequency Identity Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz - 960 MHz Version 1.0.9." *http://www.epcglobalinc.org/standards/uhfc1g2/uhfc1g2_1_0_9-standard-20050126.pdf* (2005).

[15] Firner, B., Jadhav, P., Zhang, Y., Howard, R., Trappe, W., and Fenson, E. “Towards Continuous Asset Tracking: Low-Power Communication and Fail-Safe Presence Assurance.” *Proc. of the 6th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks* (2009).

[16] Han, H., Sheng, B., Tan, Chiu C., Li, Q., Mao, W., and Lu, S. “Counting RFID Tags Efficiently and Anonymously.” *Proc. IEEE INFOCOM* (2010).

[17] Hush, D. and Wood, C. “Analysis of Tree Algorithm for RFID Arbitration.” *Proc. of IEEE ISIT* (1998).

[18] Jacobsen, R. M., Nielsen, K. F., Popovski, P., and Larsen, T. “Reliable Identification of RFID Tags using Multiple Independent Reader Sessions.” *Proc. of IEEE RFID* (2009).

[19] Jian, Y., Chen, S., Zhang, Z., and Zhang, L. “A novel scheme for protecting receiver’s location privacy in wireless sensor networks.” *IEEE Transactions on Wireless Communications* 7 (2008).10: 3769–3779.

[20] Klair, D., Chin, K., and Raad, R. “On the Energy Consumption of Pure and Slotted Aloha based RFID Anti-Collision Protocols.” *Computer Communications* (2008).

[21] Kodialam, M. and Nandagopal, T. “Fast and Reliable Estimation Schemes in RFID Systems.” *Proc. of ACM MobiCom* (2006).

[22] Kodialam, M., Nandagopal, T., and Lau, W. “Anonymous Tracking using RFID tags.” *Proc. of IEEE INFOCOM* (2007).

[23] Lee, S. R., Joo, S. D., and Lee, C. W. “An Enhanced Dynamic Framed Slotted ALOHA Algorithm for RFID Tag Identification.” *Proc. of IEEE International Conference on Mobile and Ubiquitous Systems: Networks and Services* (2005).

[24] Li, T., Chen, S., and Ling, Y. “Identifying the Missing Tags in a Large RFID System.” *Proc. of ACM Mobihoc* (2010).

[25] Li, T., Luo, W., Mo, Z., and Chen, S. “Privacy-preserving RFID Authentication based on Cryptographical Encoding.” *Proc. of IEEE INFOCOM* (2012): 2174 – 2182.

[26] Li, T., Wu, S., Chen, S., and Yang, M. “Energy Efficient Algorithms for the RFID Estimation Problem.” *Proc. of IEEE INFOCOM* (2010).

[27] ———. “Generalized Energy-Efficient Algorithms for the RFID Estimation Problem.” *IEEE/ACM Transactions on Networking* 20 (2012): 1978 – 1990.

[28] Liu, X., Li, K., Shen, Y., Min, G., Xiao, B., and Qu, W. “A Fast Approach to Unknown Tag Identification in Large Scale RFID Systems.” *Proc. of ICCCN* (2013).

[29] Luo, W., Chen, S., Li, T., and Chen, S. “Efficient Missing Tag Detection in RFID Systems.” *Proc. of IEEE INFOCOM, mini-conference* (2011).

[30] Luo, W., Qiao, Y., and Chen, S. “An Efficient Protocol for RFID Multigroup Threshold-based Classification.” *Proc. of IEEE INFOCOM* (2013).

[31] M. Chen, Z. Mo, S. Chen W. Luo and Fang, Y. “An Efficient Tag Search Protocol in Large-Scale RFID Systems.” *Proc. of IEEE INFOCOM* (2013).

[32] Myung, J. and Lee, W. “An adaptive memoryless tag anti-collision protocol for RFID networks.” *Proc. IEEE ICC* (2005).

[33] ———. “Adaptive Splitting Protocols for RFID Tag Collision Arbitration.” *Proc. of ACM MOBIHOC* (2006).

[34] Namboodiri, V. and Gao, L. “Energy-Aware Tag Anti-Collision Protocols for RFID Systems.” *Proc. of IEEE PerCom* (2007).

[35] Ni, L. M., Liu, Y., Lau, Y. C., and Patil, A. “LANDMARC: Indoor Location Sensing Using Active RFID.” *Proc. of ACM Wireless Networks* 10 (2004).6.

[36] Popovski, P., Nielsen, K. F., Jacobsen, R. M., and Larsen, T. “Robust Statistical Methods for Detection of Missing RFID Tags.” *IEEE Communications Magazine, Wireless Communications Series* (2010).

[37] Qian, C., Ngan, H., and Liu, Y. “Cardinality Estimation for Large-scale RFID Systems.” *Proc. of IEEE PerCom* (2008).

[38] Qiao, Y., Chen, S., and Li, T. “RFID as an Infrastructure.” *Springer* ISBN: 978-1-4614-5229-4 (2012).

[39] Qiao, Yan, Chen, Shigang, Li, Tao, and Chen, Shiping. “Energy-efficient Polling Protocols in RFID Systems.” *Proc. of ACM MobiHoc* (2011).

[40] S. Chen, M. Zhang and Xiao, B. “Efficient Information Collection Protocols for Sensor-augmented RFID Networks.” *Proc. of IEEE INFOCOM* (2011).

[41] Sarangan, V., Devarapalli, M. R., and Radhakrishnan, S. “A Framework for Fast RFID Tag Reading in Static and Mobile Environments.” *International Journal of Computer and Telecommunications Networking* 52 (2008).5: 1058–1073.

[42] Sato, Y., Igarashi, Y., Mitsugi, J., Nakamura, O., and Murai, J. “Counting RFID Tags Efficiently and Anonymously.” *Proc. of IEEE RFID* (2012).

[43] Shahzad, M. and Liu, A. “Every Bit Counts - Fast and Scalable RFID Estimation.” *Proc. of ACM MOBICOM* (2012).

[44] Sheng, B., Li, Q., and Mao, W. “Efficient Continuous Scanning in RFID Systems.” *Proc. of IEEE INFOCOM* (2010).

[45] Sheng, B., Tan, Chiu C., Li, Q., and Mao, W. “Finding Popular Categories for RFID Tags.” *Proc. of ACM MOBIHOC* (2008).

[46] Tan, C., Sheng, B., and Li, Q. “How to Monitor for Missing RFID Tags.” *Proc. of IEEE ICDCS* (2008).

[47] Telatar, I. E. and Gallager, R. G. “Combining Queuing Theory and Information Theory for Multiaccess.” *IEEE Journal on Selected Areas Communication* 13 (1995).6: 963–969.

[48] Vogt, H. “Efcient Object Identication with Passive RFID Tags.” *Proc. of IEEE PerCom* (2002).

[49] ———. “Efficient Object Identification with Passive RFID Tags.” *Proc. of IEEE PerCom* (2002).

[50] Wolverton, J. “Wal-Mart to Embed RFID Tags in Clothing Beginning August 1.” *http://www.thenewamerican.com/index.php/tech-mainmenu-30/ computers/4157-wal-mart-to-embed-rfid-tags-in-clothing-beginning-august-1* (2010).

[51] Xiao, Q., Xiao, B., and Chen, S. “Differential Estimation in Dynamic RFID Systems.” *Proc. of IEEE INFOCOM, mini-confernece* (2013).

[52] Yao, Q., Qi, Y., Han, J., Zhao, J., Li, X., and Liu, Y. “Randomizing RFID Private Authentication.” *Proc. of IEEE PERCOM* (2009).

[53] Yue, H., Zhang, C., Pan, M., Fang, Y., and Chen, S. “A Time-efficient Information Collection Protocol for Large-scale RFID Systems.” *Proc. of IEEE INFOCOM* (2012).

[54] Zhai, J. and Wang, G. N. “An Anti-collision Algorithm using Two-functioned Estimation for RFID Tags.” *Proc. of ICCSA* (2005).

[55] Zhang, M., Li, T., Chen, S., and Li, B. “Using Analog Network Coding to Improve the RFID Reading Throughput.” *Proc. of IEEE ICDCS* (2010): 547 – 556.

[56] Zhang, R., Liu, Y., Zhang, Y., and Sun, J. “Fast Identification of the Missing Tags in a Large RFID System.” *Proc. of the 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks* (2011).

[57] Zhen, B., Kobayashi, M., and Shimizu, M. “Framed ALOHA for Multiple RFID Object Identification.” *IEICE Transactions on Communications* (2005).

[58] Zheng, Y. and Li, M. “Fast Tag Searching Protocol for Large-Scale RFID Systems.” *Proc. of IEEE ICNP* (2011).

[59] ———. “ZOE: Fast Cardinality Estimation for Large-Scale RFID Systems.” *Proc. of IEEE INFOCOM* (2013).

[60] Zhou, F., Chen, C., Jin, D., Huang, C., and Min, H. “Evaluating and Optimizing Power Consumption of Anti-collision Protocols for Applications in RFID Systems.”

Proc. of ACM International Symposium on Low Power Electronics and Design (ISLPED) (2004).

BIOGRAPHICAL SKETCH

Wen Luo received his B.S. degree in computer science and technology from the University of Science and Technology of China in 2008. After that he joined the University of Florida as a Ph.D. student for the Department of Computer Information and Science Engineering. His advisor is Dr. Shigang Chen, and his research interests include RFID technologies and internet traffic measurement. His email address is wluo@cise.ufl.edu.