EFFICIENT TAG MANAGEMENT IN LARGE-SCALE RFID SYSTEMS

XUAN LIU

Ph.D

The Hong Kong Polytechnic University

2015

THE HONG KONG POLYTECHNIC UNIVERSITY
DEPARTMENT OF COMPUTING

EFFICIENT TAG MANAGEMENT IN LARGE-SCALE RFID SYSTEMS

Xuan Liu

A thesis submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

January, 2015

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

Signature of Xuan Liu

Abstract

Nowadays, Radio Frequency IDentification (RFID) technology is widely used in many applications including supply chain monitoring, warehouse management, inventory control, etc. Unlike the barcode system that has to read the data closely, RFID extends the operation distance from inches to tens of feet (for passive tags) or even hundreds of feet (for active tags). With RFID technology, a system could automatically obtain the products information without manual operations. *Time-efficiency* is one of the most important concern for a modern RFID system. How to efficiently manage and monitor large RFID systems is a practical but challenge research issue.

In the thesis, to quickly obtain required information to improve the management efficiency for large RFID systems, we design efficient tag scanning protocols for three important applications, namely *unknown tag identification*, *tag searching* and *tag stocktaking*. To obtain the information of tags/products, the traditional method is to scan tags and collect the tags' IDs in the system, which is usually referred to as *tag identification*. Since a tag ID is 96-bits long, ID-collection for a large RFID system that contains tens of thousands of tags usually takes too much time. If the system has to identify each tag every time when scanning the system, the scanning time will be obviously unbearable. To shorten the scanning time, we try to avoid tedious ID transmission from tags to readers, which greatly improves the scanning process.

The major contributions of the thesis for efficiently management of large RFID systems are threefold. First, we propose efficient protocols to identify all unknown tag in large RFID systems. The existing work identifies unknown tags with a given probability but cannot identify them completely. Our proposed protocol recognizes known tags without ID transmission and deactivates them to prohibit their further replies. After deactivating all the known tags, the remaining active tags must be unknown tags. Then the reader completely identifies unknown tags. By employing two novel techniques slot pairing and multiple reselections to resolve the known tag collision, the enhanced protocols greatly shorten the time to deactivate the known tags. Second, we propose two efficient tag searching protocols to quickly search tags with given IDs (i.e. wanted tags). The existing work uses bloom filter to search wanted tags. However, it performs badly when the number of wanted tags is large. Our protocols employ the novel technique testing slot to iteratively eliminate nontarget tags without ID transmission round by round, and then quickly obtain the searching result. Third, we present efficient tag stocktaking protocols that quickly take stocking of tags and update the inventory accordingly (i.e., deleting absent tags and adding new tags). Compared to existing tag identification protocols, the proposed stocktaking protocols identify only unknown tags and missing tags, which avoid the time waste of ID re-collection. Meanwhile, these protocols require only very slight modification of the standard protocol, making it easy to be implemented with COTS tags. We also exploit the analog network coding technique to further improve the performance of tag stocktaking protocols.

Publications

Journal Articles

1. **Xuan Liu**, Bin Xiao, Shigeng Zhang, Kai Bu, and Alvin Chan, STEP: A Time-Efficient Tag Searching Protocol in Large RFID Systems, *IEEE Transactions on Computers (**TC**)*, accepted.

2. **Xuan Liu**, Shigeng Zhang, Bin Xiao, and Kai Bu, Flexible and Time-Efficient Tag Scanning with Handheld Readers, *IEEE Transactions on Mobile Computing **(TMC)***, accepted.

3. **Xuan Liu**, Bin Xiao, Shigeng Zhang, and Kai Bu, Unknown Tag Identification in Large RFID Systems: An Efficient and Complete Solution, *IEEE Transactions on Parallel and Distributed Systems **(TPDS)***, accepted.

4. Kai Bu, **Xuan Liu**, and Bin Xiao, Approaching the Time Lower Bound on Cloned-tag Identification for Large RFID Systems, *Ad Hoc Networks*, vol. 13, Part B, no. 0, pp. 271-281, 2014.

5. Kai Bu, **Xuan Liu**, Jiaqing Luo, Bin Xiao, and Guiyi Wei, Unreconciled Collisions Uncover Cloning Attacks in Anonymous RFID Systems, *IEEE Transactions on Information Forensics and Security (**TIFS**)*, vol. 8, no. 3, pp. 429-439, 2013.

Conference Papers

1. **Xuan Liu**, Bin Xiao, Kai Bu, and Shigeng Zhang, LOCK: A Fast and Flexible Tag Scanning Mechanism with Handheld Readers, in *Proc. of the 22th IEEE/ACM International Workshop on Quality of Service (**IWQoS**)*, Hong Kong, China, May. 2014.

2. Kai Bu, Mingjie Xu, **Xuan Liu**, Jiaqing Luo, and Shigeng Zhang, Toward Fast and Deterministic Clone Detection for Large Anonymous RFID Systems, in *Proc. of the 11th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (**MASS**)*, Philadelphia, PA, USA, October 28-30, 2014.

3. Kai Bu, **Xuan Liu**, Jiwei Li, and Bin Xiao Less is More: Efficient RFID-based 3D Localization, in *Proc. of the 10th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (**MASS**)*, Hangzhou, Zhejiang, China, October 14-16, 2013, pp. 86-94.

4. Jiwei Li, Kai Bu, **Xuan Liu**, and Bin Xiao, ENDA: Embracing Network Inconsistency for Dynamic Application Offloading, in *Proc. of The Second Mobile Cloud Computing Workshop (**MCC**)*, Hong Kong, China, August 12, 2013.

5. **Xuan Liu**, Shigeng Zhang, Kai Bu, and Bin Xiao, Complete and Fast Unknown Tag Identification in Large RFID Systems, in *Proc. of the 9th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (**MASS**)*, Las Vegas, Nevada, USA, October 8-11, 2012, pp. 47-55.

6. Kai Bu, **Xuan Liu**, and Bin Xiao, [Poster/Short Paper] Fast Cloned-Tag Identification Protocols for Large-Scale RFID Systems, in *Proc. of the 20th*

*IEEE/ACM International Workshop on Quality of Service (**IWQoS**)*, Coimbra, Portugal, June 4-5, 2012, pp. 1-4.

7. Shigeng Zhang, Jianxin Wang, **Xuan Liu**, and Jiannong Cao, Range-free Selective Multilateration for Anisotropic Wireless Sensor Networks, in *Proc. of the 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (**SECON**)*, June 18-21, 2012, Seoul, Korea.

8. **Xuan Liu**, Shigeng Zhang, Jianxin Wang, Jiannong Cao, and Bin Xiao, Anchor Supervised Distance Estimation in Anisotropic Wireless Sensor Networks, *Proc. of the IEEE Wireless Communications and Networking Conference (**WCNC**)*, Quintana-Roo, Mexico, March 28C31, 2011.

9. Jiannong Cao, Weigang Wu, and **Xuan Liu**, Seamless Mobility support for Adaptive Applications in Heterogeneous Wireless Networks, *Proc.2010 International Workshop on Mobile Cyber-Physical Systems (**MobiCPS**)*, Xian, China. Oct. 26 C 29, 2010.

10. Yuhong Feng, Jiannong Cao, Ivan Chuen Ho Lau, and **Xuan Liu**, A Self-configuring Personal Agent Platform for Pervasive Computing, *Proc. 4th IEEE/IFIP International Conference On Embedded and Ubiquitous Computing (**EUC**)*, Shanghai, China. Dec. 17C20, 2008. pp. 438-444. **Best Paper Award**

Acknowledgements

First and foremost, I would like to especially thank my supervisor, Prof. Bin Xiao. During my whole study period, he always encouraged me to do the research with my own interest. He discussed every research problem and idea with me in detail. I have often benefitted from his insights and invaluable advices. Even my pregnancy affects the progress of research, he gave me the tolerant understanding. In these years, Prof. Bin Xiao support me for not only the study but also my personal life. It is obviously that he is the one who contributes most. I also thank Prof. Jiannong Cao for recommending me to Prof. Bin Xiao as a PhD student. Without his warmly help, I cannot smoothly pursue my research interest. He also gave me many suggestions for my research work.

There are also some other professors who helped me a lot during my study period. I would like to thank my co-supervisor Prof. CHAN Toong Shoon, Prof. Shigang Chen, Prof. Dan Wang, and Prof. Wei Lou for their invaluable advises. I also want to thanks my colleagues Kai bu, Jiwei Li, Fei Wang and Jia Liu. I benefitted a lot from doing the research together with them.

Finally, and most importantly, I would like to gratefully and sincerely thank my parents for their continuous, unconditional love and support. I also thank my husband Shigeng Zhang. Every time I encountered difficulties, he always encouraged me and

helped me to move on. Without his support, I cannot finish this work. At last, I have to mention my baby girl Zhenmiao Zhang. Although she did nothing good for my research work, I still want to say to her: " I love you, my girl. Thank you for being my baby. You are the most amazing gift ever in my life".

Hong Kong S.A.R., China Xuan Liu

May 27, 2015

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 RFID Technologies

Radio Frequency IDentification (RFID) technology has enjoyed significant growth over the past decade. Nowadays, RFID is widely used in many applications including supply chain monitoring, warehouse management, inventory control [Klair et al., 2010, Maneesilp et al., 2013, Shih et al., 2006, Wang et al., 2012], etc. In these applications, physical objects are attached with passive or active RFID tags each containing a unique ID and object related information. The information stored in tags is then scanned by readers through the wireless channel and sent to a backend server for automatic object management. Unlike the barcode system that has to read the data closely, RFID extends the operation distance from inches to tens of feet (for passive tags) or even hundreds of feet (for active tags). With RFID technology, a system could automatically obtain the products information without manual operations. According to a recent forecast report given by IDTechEx [Das and Harrop, 2013], the total RFID market will be worth $30.24 billion by 2024.

Time-efficiency is one of the most important concern for a modern RFID system. How to efficiently manage and monitor large RFID systems is a practical but

challenged research issue. To obtain the information of tags/products, the traditional method is to scan tags and collect the tags' IDs in the system, which is usually referred to as *tag identification*. Since a tag ID is 96-bits long, ID-collection in a large RFID system that contains tens of thousands of tags usually takes too much time. Some applications such as warehouse management that have to periodically scan tags to update the inventory. If the system has to identify each tag every time when scanning the system, the scanning time will be obviously unbearable. Especially, for the system that already stores an inventory of all tags in a backend database, this method will unavoidably re-collect most tags, and causes a huge waste of time. What's more, some applications require the information such as tag number or missing/new-arriving tags rather than tags' IDs. In this case, ID-collection for the whole system is a low efficiency solution. To provide highly efficient management, researchers dedicate significant efforts to scanning operations for different application requirements, such as cardinality estimation [Gong et al., 2014, Han et al., 2010, Kodialam and Nandagopal, 2006, Li et al., 2010a, Qian et al., 2008, Shahzad and Liu, 2012], tag searching [Chen et al., 2013], missing-tag detection and identification [Li et al., 2010b, 2013, Liu et al., 2013, Luo et al., 2012, Tan et al., 2010, Zhang et al., 2011], and information collection [Qiao et al., 2011, Yue et al., 2012].

1.2 Background

1.2.1 Typic RFID Systems

A RFID system usually consists of three components: a back-end server, a set of readers, and a large number of tags. The server usually connects a backend database that stores all tags' IDs in the system. Meanwhile, the server also connects the readers

via wired or wireless networks, and issues orders to schedule their working. Multiple readers are deployed to cover all the tags in the system. They cooperatively scan tags according to the orders issued by the server. Many researchers apply existing reader scheduling algorithms [Tang et al., 2009, 2011, Yang et al., 2011, Zhou et al., 2007] to obtain a conflict-free schedule of multiple readers, by which these readers could be logically treated as one reader that covers the whole system.

1.2.2 Communication Protocol: Frame Slotted ALOHA

Frame slotted ALOHA is the most popular communication protocol for RFID systems. In this protocol, the communication between the reader and tags takes the form of *frame*, which is organized as a number of *slots* synchronized by the reader. When receiving the query, if multiple tags respond at a same slot, the response signals will collide and cannot be received successfully by the reader. Frame slotted ALOHA avoids the collision by arranging tags to respond in different slots.

In ALOHA, tags will respond to the reader after receiving a query from the reader, which is known as the reader-talk-first mode. The reader starts a frame with a query $<f, r>$, where f indicates the number of slots in the frame and r is a random seed used by tags to determine which slot to respond. A tag chooses a slot in [0,f-1] by $S = H(ID||r)$ mod f and responds its ID in the S-th slot, where H is a uniform hash function. When receiving a response successfully, the reader replies an ACK to acknowledge that tag and prevents it from attending the following process until the next protocol execution. If the reader fails to receive the response (e.g. channel error), it will reply a NAK to keep the tag active. At the end of the frame, the reader will issue another query and start a new frame if it detects any active tags. The reader repeats the operation until all the tags are acknowledged.

Since the tags choose the slot randomly with a hash function, the response number in each slot is different and cannot be designed. According to the response number, a slot can be in one of three different states: *empty* (when no tag responds), *singleton* (when only one tag responds) or *collision* (when more than one tags respond). The reader receives responses successfully in only singleton slots.

The duration time of each slot depends on the length of transmitted bit string. Obviously, an empty slot (denoted by t_e) is less than the duration of a non-empty slot. The duration of a non-empty slot depends on the type of responses that tags transmit to the reader. A tag can transmit either its ID or a short response (e.g., a 16-bits random number *RN16* in EPC Class 1 Generation 2 (C1G2) standard [EPCglobal, 2008]) to the reader. We denote by t_{id} and t_s the time needed to transmit a tag ID and a *RN16* short response, respectively. The relationship between duration time of different types of slots is $t_e < t_s \ll t_{id}$.

The setting of the frame size f is critical for the ALOHA-based protocols. To guarantee the proper ratio of singleton slots and obtain the optimal time efficiency, the frame size f is usually set according to the number of tags in the reader's interrogation region. We will discuss the optimal frame sizes for our proposed protocols detailed in the following chapters.

There are many standards developed based on Frame-slotted ALOHA Protocol, such as EPC C1G2 standard [EPCglobal, 2008] and the Philips I-Code specification [Semiconductors, Jan. 2004]. The protocols designed in the thesis adopt the EPC C1G2 standard.

EPC C1G2 standard

EPC C1G2 standard is a widely used tag identification standard and is adopted by many commercial-off-the-shelf tags. In the standard EPC C1G2 [EPCglobal, 2008], the reader issues a series of frames (also called query rounds) to identify tags. The process to identify all the tags is called *an identification operation*. The reader powers down between two identification operations.

To start a frame, the reader broadcasts a *Query* command to tags containing a parameter q, indicating that there are 2^q slots in the frame. Upon receiving the Query command, each tag picks a random value in the range $[0, 2^q - 1]$, and loads the value into its slot counter. A tag decreases its slot counter by 1 every time when it receives a *QueryRep* command from the reader. Meanwhile, every tag generates a 16-bit random number, namely *RN16*, by using a random number generator. If a tag, in response to the Query or QueryRep command, finds that its slot counter value is zero, it backscatters its RN16 field to contend for the channel access.

Fig. 5.1 shows the procedures for three different slots. When only one tag backscatters its RN16 to the reader in a slot, the reader can successfully receive the RN16. In this case, the reader immediately broadcasts an *ACK* command containing the latest received RN16. The corresponding tag, finding that the RN16 piggybacked in the ACK command equals its own RN16, will transmit its ID along with the cyclic redundancy code (CRC) to the reader. The tag then enters the *acknowledged* state and will exit the current identification operation. If more than one tag simultaneously transmit their RN16s in a slot, the reader will detect a collision. Only singleton slots are useful in identifying tag IDs in the EPC C1G2 protocol.

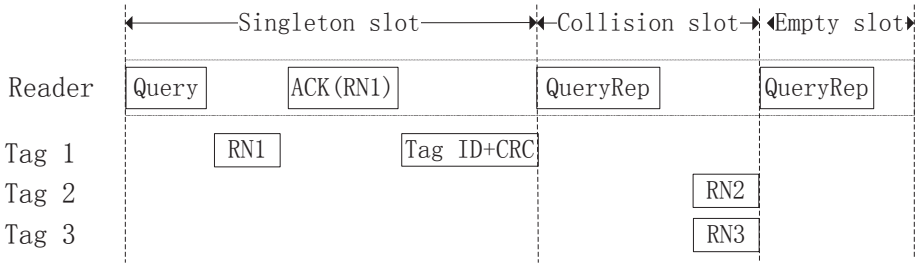


Fig. 1.1: Illustration of the standard EPC C1G2 identification protocol.

1.3 Thesis Contributions

In the thesis, we design efficient tag scanning protocols to quickly obtain required information to improve the management efficiency for large RFID systems. We concentrate on seeking efficient solutions for three important applications in large RFID systems, namely *unknown tag identification*, *tag searching* and *tag stocktaking*. We will provide overview of the proposed protocols in the thesis and highlight their major contributions.

1.3.1 Efficient Unknown Tag Identification

One important research issue in large RFID systems is the identification of unknown tags, i.e., tags that just entered the system but have not been interrogated by reader(s) covering them yet. Unknown tag identification plays a critical role in automatic inventory management and misplaced tag discovery.

In Chapter 3, we propose a series of protocols that can identify all of the unknown tags with high time efficiency. We develop several novel techniques to quickly deactivate already identified tags (i.e., known tags) and prevent them from replying during the interrogation of unknown tags, which avoids re-identification of these tags and consequently improves time efficiency. Meanwhile, we develop two novel techniques,

slot pairing (Section 3.4) and *multiple reselections* (Section 3.5) to resolve the known tag collision. These two techniques can greatly enhance the efficiency in deactivating known tags. They can also be applied as common techniques to improve tag identification efficiency in RFID systems. We build a series of time efficient unknown tag identification protocols on top of these techniques. Furthermore, in order to set optimal frame size in our protocols, we develop a zero-cost algorithm to estimate the number of unknown tags that are currently present in the system and evaluate its accuracy. To our knowledge, our protocols are the first non-trivial solutions that guarantee complete identification of all the unknown tags. We illustrate the effectiveness of our protocols through both rigorous theoretical analysis and extensive simulations. Simulation results show that our protocols can save up to 70 percent time when compared with the best existing solutions.

1.3.2 Efficient Tag Searching

Instead of making an inventory of all the tags in the system, some applications require searching a particular set of tags (called *wanted tags*) with the given tag IDs to confirm which of them are currently present in the system.

In Chapter 4, we study the practically important *tag searching problem*, particularly in large RFID systems that contain multiple readers. we design a novel technique called *testing slot* for a reader to quickly figure out which wanted tags are absent from its interrogation region without tag ID transmissions, which could greatly reduce transmission time during the searching process. Based on this technique, we propose two protocols targeting at time-efficient tag searching in practical large RFID systems containing multiple readers. In our protocols, readers quickly obtain local searching results by employing testing slots to iteratively eliminate wanted tags

that are absent from their interrogation region. The local searching results of all the readers are then combined to form the final searching result. Simulation results show that, our best protocol reduces up to 76 percent time compared with state-of-the-art solutions, and achieves very high precision.

1.3.3 Efficient Tag Stocktaking

An RFID system can greatly improve the efficiency of tagged object inventory setup and update. It is necessary to periodically take stock of tags and update the inventory accordingly (i.e., deleting absent tags and adding new tags) in dynamic scenarios such as warehouses and shopping malls. Fast tag stocktaking is critical for the dynamic RFID system management.

In Chapter 5, we propose HARN, a protocol that can quickly take stock of tags in dynamic RFID systems but use only one more hash in the standard EPC C1G2 protocol. HARN leverages a new hash to generate a 16-bit field to replace the original 16-bit random number. Such replacement enables the transmitted 16-bit string to contain more information to identify present tags, which can save the tedious ID transmission from known tags to readers and greatly improve the inventory process. HARN is compatible with COTS RFID tags and can be easily applied in a real RFID system. To further improve the performance of HARN, we also devise an analog network coding (ANC) based approach to extract useful information from the collided signal when multiple tags transmit simultaneously. Simulation results demonstrate up to 3.8x (when ANC is not used) and 18x (when ANC is used) boosts in stocktaking throughput compared to the state-of-the-art solutions in dynamic RFID systems.

1.4 Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 introduces the related previous research work. Chapter 3 proposes protocols toward efficient unknown tag identification in large RFID systems. The proposed protocols can quickly identify all unknown tags in the systme. Chapter 4 proposes protocols toward efficient tag searching solutions in large RFID systems. The proposed protocols can work in a multi-reader system. Chapter 5 proposes protocols toward efficient tag stocktaking in dynamic RFID systems. The proposed protocols quickly take stock of tags in the system without re-collecting tag IDs that stored in the database, which greatly improve the time efficiency. Finally, Chapter 6 concludes the thesis and indicates future work.

The primary research outputs emerged from the thesis are as follows:

- Xuan Liu, Bin Xiao, Shigeng Zhang, Kai Bu, and Alvin Chan, STEP: A Time-Efficient Tag Searching Protocol in Large RFID Systems, *IEEE Transactions on Computers (TC)*, accepted.

- Xuan Liu, Shigeng Zhang, Bin Xiao, and Kai Bu, Flexible and Time-Efficient Tag Scanning with Handheld Readers, *IEEE Transactions on Mobile Computing (TMC)*, accepted.

- Xuan Liu, Bin Xiao, Shigeng Zhang, and Kai Bu, Unknown Tag Identification in Large RFID Systems: An Efficient and Complete Solution, *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, accepted.

- Xuan Liu, Bin Xiao, Kai Bu, and Shigeng Zhang, LOCK: A Fast and Flexible Tag Scanning Mechanism with Handheld Readers, in *Proc. of the 22th*

IEEE/ACM International Workshop on Quality of Service (IWQoS), Hong Kong, China, May. 2014.

- Xuan Liu, Shigeng Zhang, Kai Bu, and Bin Xiao, Complete and Fast Unknown Tag Identification in Large RFID Systems, *Proc. of the Ninth IEEE International Conference on Mobile Ad-hoc and Sensor Systems (IEEE MASS 2012)*, Las Vegas, Nevada, USA, Oct. 2012, pp. 47-55.

- Xuan Liu, Shigeng Zhang, Bin Xiao, and Kai Bu, One More Hash is Enough: Efficient Tag Identification in Highly Dynamic RFID Systems, under review in *IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*.

Chapter 2

Literature Review

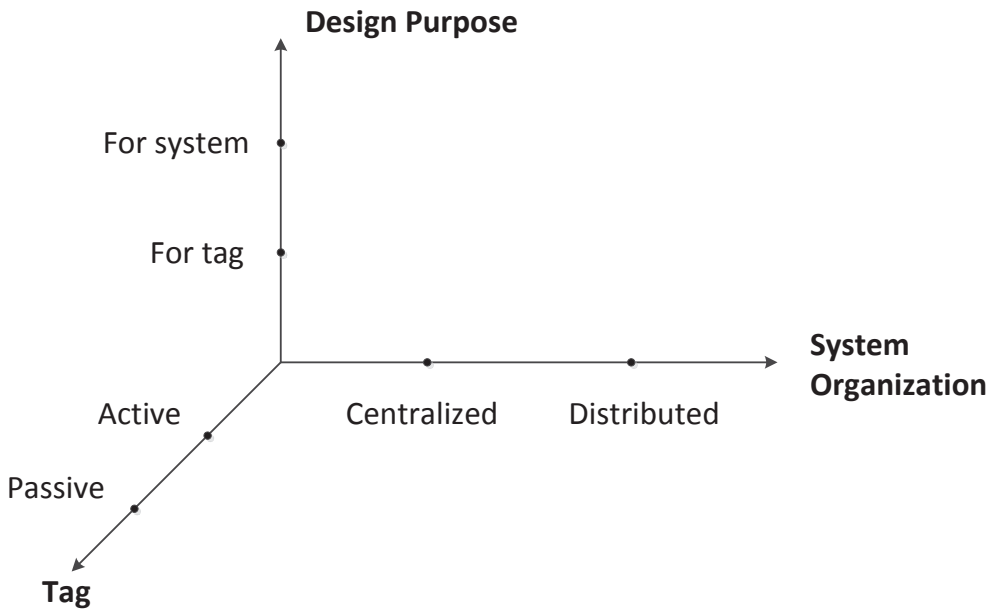


Fig. 2.1: The classifications of scanning protocols from three different dimensions.

The existing RFID scanning protocols could be classified generally from three different dimensions (Fig. 2.1): What's kind of tags the protocol considered (*the tag type*), how many readers in the system (*the system organization*), and what's kind of

information should be obtained (*the design purpose*).

In this Chapter, we will introduce the characteristics of existing scanning protocols from these three dimensions, and also discuss our protocols at last.

2.1 Tag Type

According to whether has the energy support, RFID tags are divided into passive tags and active tags. Passive tags harvest energy from the radio signal from the reader and use such minute amount of energy to deliver information back to the reader. The shortage of energy resources bring passive tags the constraints of the limited reading range and the low computational capability. However, benefit from the low price, passive tags are widely adopted in many systems. On the contrary, with self energy supply, active tags have longer reading range and are capable of doing complex computations. Although they are more expensive than passive tags, they have richer resources for implementing advanced functions. If the goal is to fully automate the warehouse management in a large scale, we believe battery-powered active tags are a better choice.

The researches on these two different kinds of tags have to consider their computing capabilities. Since the passive tags have limited capability, the designed protocols require passive tags to do only very simple calculations. Most passive tag protocols are designed for efficient tag identification: They focus on either how to organize an efficient query tree to identify tags with minimum query times (for tree-based protocols) [Finkenzeller, 2003, Husn and Wood, 1998, Myung et al., 2007, Shahzad and Liu, 2013], or how to set the optimal frame size to shorten the time needed by the identification process (for Aloha-based protocols) [Cha and Kim, 2005, Klair et al.,

2007, Namboodiri and Gao, 2010, Zhen et al., 2005]. In these protocols, tags respond to the reader as in standard and do not require to do any extra job. For the researches considering active tags, the designed protocols usually need the tags provide more information to help select transmission slots and avoid tag collisions. For example, hash multiple times [Chen et al., 2011, Qiao et al., 2011] or count bit number from bit vector sent by the reader [Chen et al., 2011, Li et al., 2010b].

In the thesis, we adopt active tags to obtain the better computer capability. In the rest of the thesis, "active" means the status rather than the type, i.e., active tags are the tags that are not deactivated by the reader.

2.2 System Organization

Researchers design tag scanning protocols considering two kinds of system organizations: the centralized system that has only one reader covering all the tags (i.e. one-reader system), and the distributed system that has multiple readers that cooperatively scan tags and are scheduled by the server (i.e. multi-reader system).

The protocols of one-reader system focus on the efficient communication between the reader and the tags. Although the actual systems usually contain multiple readers, Many researchers apply existing reader scheduling algorithms [Tang et al., 2009, 2011, Yang et al., 2011, Zhou et al., 2007] to obtain a conflict-free schedule of multiple readers, by which these readers could be logically treated as one reader that covers the whole system. The protocols of multi-reader system usually consider more real factors, such as the collisions between readers and so on, that will unavoidably restrict the efficiency of scanning protocols.

2.3 Design Purpose

The design purposes of scanning protocols can be generally divided into two categories: 1. Obtain the information of each individual tag, i.e. collecting the tag ID. 2. Obtain the information for the whole RFID system, e.g. identifying which tags are missing/new arrival in the system, or estimating the cardinality of all tags.

2.3.1 Protocols of Scanning for Tag

Tag identification is the most fundamental research problem that is used to quickly collect tag IDs in the system. Many researches on tag identification aim at enhancing time efficiency. Generally, existing RFID identification protocols can be classified into two categories [Klair et al., 2010, Shih et al., 2006]: tree-based protocols [Finkenzeller, 2003, Husn and Wood, 1998, Myung et al., 2007, Shahzad and Liu, 2013] and ALOHA-based protocols [Cha and Kim, 2005, Klair et al., 2007, Namboodiri and Gao, 2010, Wang et al., 2012, Zhen et al., 2005]. In tree-based protocols, the reader queries tags and splits the tags into two subsets if a transmission collision occurs. The splitting process continues until there is only one tag in each subset, in which case the tag could be successfully identified. In ALOHA-based protocols, the communication between the reader and tags takes a *reader-talks-first* model. The reader issues queries and receives tag responses in a frame of multiple slots. After receiving the query, each tag randomly chooses a time slot in the frame and transmits its ID to the reader. The reader will fail to receive the tag ID if two or more tags transmit at the same time. In this case, the reader will repeat issuing new frames until all the tag IDs have been successfully collected.

Kang et al. [Kang et al., 2012] and Xie et al. [Xie et al., 2013] reported that tag

identification throughput degrades in real deployment. Mobile readers can provide flexible tag identification [Xie et al., 2010, 2013, Zhu et al., 2013] in infrastructure-less (i.e., with no pre-installed readers) RFID systems. Xie et al. [Xie et al., 2013] reported observations on the relationship between identification throughput and the transmitting power of the readers. Based on the observations, they designed algorithms to optimize energy and time efficiency of the reader in large RFID systems containing more than one hundred tags. Zhu et al. [Zhu et al., 2013] discussed how to plan the trajectory of the mobile reader to save energy. Some works also exploit parallel tag identification by using multiple readers in recent years [Kong et al., 2014, Tang et al., 2009, Yang et al., 2014]. RASPberry [Tang et al., 2009] tries to make the system work in a stable way in a long term when the arrival rate of tags is within the capacity region of the readers. In [Yang et al., 2014] the authors proposed an identification protocol that jointly considers reader scheduling and tag identification.

2.3.2 Protocols of Scanning for System

Instead of getting the ID from each individual tag, the protocols of scanning for system is designed, with some specific requirements, to obtain the information of the whole systems. Such as the roughly cardinality of tags, or dynamically track the tag changes in the system.

Tag Estimation

Tag cardinality estimation is a popular research issue for large RFID systems. Instead of counting tags by identifying each of them, Kodialam and Nandagopal [Kodialam and Nandagopal, 2006] propose protocols that quickly estimate the number of tags up to a desired level of accuracy by leveraging probabilistic approach with the

distribution of the number of tag responses in slots. The protocol requires only short responses much shorter than tag IDs, which could greatly shorten the time needed for collecting all tag responses. Compared with tag identification that has linear time complexity with respect to the number of tags, this protocol promises far more time efficiency to large RFID systems. Qian et al. [Qian et al., 2008] then proposes fast cardinality estimation protocols for the multi-reader systems. More recently, considering the RFID systems with active tags, Li et al. [Li et al., 2010a] proposes energy-efficient cardinality estimation protocols that save energy by requiring only a subset of tags to send responses. By leveraging the continuous number of a same slot state, Shahzad, M. et al. [Shahzad and Liu, 2012] further improves the time efficiency to estimate tags' number by probabilistic approach.

Missing/Unknown/Misplaced Tag Identification

In recent years, many researchers pay attention to scanning protocols for obtaining dynamic information brought by tags moving, most of which focus on identifying missing tags. A straightforward way is collecting all tag IDs and then comparing them against the recorded ones to find which tags are missing. Since ID-collection method in large RFID systems is inefficient, Tan et al. [Tan et al., 2010] proposes novel protocols to detect missing-tag event with a given probability when the number of missing tags exceeds a threshold. The proposal requires tags to respond in a number of time slots and leverages the fact that, when a time slot supposed to be occupied by tag responses becomes empty, some tag(s) must be missing. The follow-up work [Li et al., 2010b] assumes that the reader knows all the tag IDs. By comparing the expected replies and the received replies, the reader identifies the missing tags certainly. Considering a collision domain only involves the tags within a readers coverage zone rather than

all the tags in the system, Zhang et al. [Zhang et al., 2011] proposes protocols that identify missing tags with multiple readers. Sheng et al. [Sheng et al., 2010] proposes a continuous scanning scheme , by monitoring singleton and empty slots, to detect the missing tags and unknown tags with a probability. CU first predicts which slot should be empty according to the IDs of known tags, assuming no unknown tags exist. In the predicted non-empty slots, the reader sends ACK to temporarily prohibit replies from known tags. In the predicted empty slots, the reader sends NAK to keep unknown tags active for following identification. It then collects IDs of active unknown tags. Unavoidably, some unknown tags may be prohibited in the predicted non-empty slots and thus cannot be identified. The protocol runs multiple rounds to guarantee that the probability that a required fraction of unknown tags are identified is higher than a threshold. However, CU cannot ensure that all the unknown tags are identified.

The misplaced-tag pinpointing problem focuses on finding misplaced tags that are still in the system rather than missing. Bu et al. [Bu et al., 2011] proposes protocols to detect misplaced tags with reader vectors.

Information Collection

More recently, information collection in sensor-augmented RFID systems is a new research issue that attracts many researchers' interest. Chen et al. [Chen et al., 2011] proposes a protocol to collect sensed data from all tags by arranging the tag transmitting slot based on a multi-hashing scheme. The elimination of ID transmission is critical for guaranteeing time efficiency. Qiao et al. [Qiao et al., 2011] then extend the multi-hashing scheme, for the purpose of energy efficiency, to applications that collect sensed information from only a subset of tags. Since these works are designed for single-reader systems, Yue et al. [Yue et al., 2012] dedicates efforts to efficient

information collection in large RFID systems with multiple readers.

2.4 Our Work

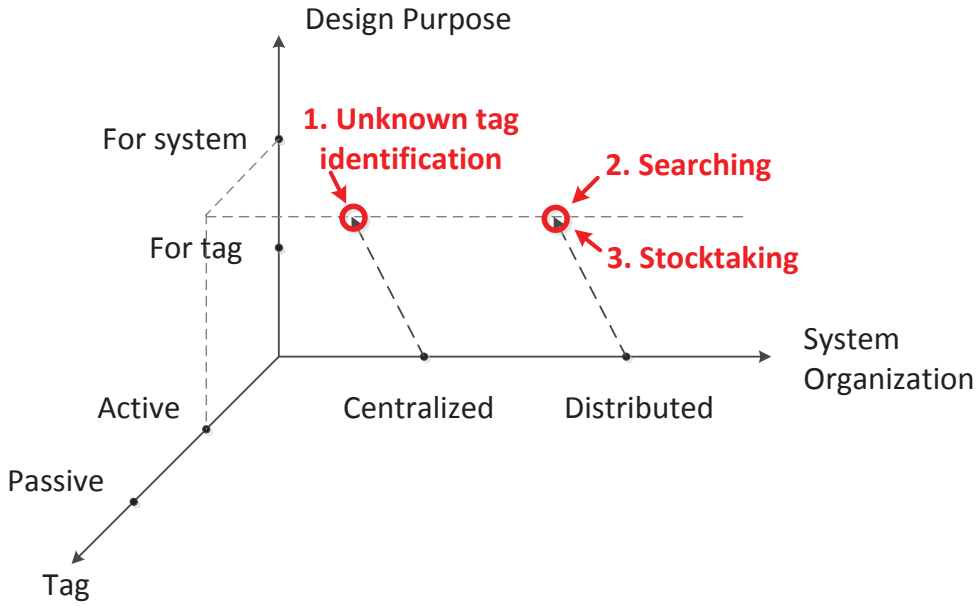


Fig. 2.2: Illustration of the standard EPC C1G2 identification protocol.

The Fig. 2.1 illustrates the positions of our protocols designed in three different dimensions. In the thesis, we focus on three different applications that all solve to obtain the information for systems. Meanwhile, to achieve optimal performance, all protocols apply active tags to obtain advance computational capabilities. The only dimension where these protocols are designed differently is the system organization. For the unknown tag identification, we consider the centralized organization, where all readers are traded as one reader logically, which could greatly simplify the model of the system. On the contrary, For the applications of both the tag searching and the

tag stocktaking, we design protocols considering multiple readers working separately, which is more practically for our daily life.

Chapter 3

Fast Unknown tag identification in Large RFID Systems

One important research issue in large RFID systems is the identification of unknown tags, i.e., tags that just entered the system but have not been interrogated by reader(s) covering them yet. Unknown tag identification plays a critical role in automatic inventory management and misplaced tag discovery, but it is far from thoroughly investigated. Existing solutions either trivially interrogate all the tags in the system and thus are highly time inefficient due to re-identification of already identified tags, or use probabilistic approaches that cannot guarantee complete identification of all the unknown tags. In this chapter, we propose a series of protocols that can identify all of the unknown tags with high time efficiency. We develop several novel techniques to quickly deactivate already identified tags and prevent them from replying during the interrogation of unknown tags, which avoids re-identification of these tags and consequently improves time efficiency. To our knowledge, our protocols are the first non-trivial solutions that guarantee complete identification of all the unknown tags. We illustrate the effectiveness of our protocols through both rigorous theoretical analysis and extensive simulations. Simulation results show that

our protocols can save up to 70 percent time when compared with the best existing solutions.

3.1 Overview

The process of collecting tag IDs is usually referred to as *tag identification*. Efficient tag identification plays a critical role in stimulating innovative application of RFID in various fields, and has attracted a lot of research attention in recent years [EPCglobal, 2008, Klair et al., 2010, Namboodiri and Gao, 2010, Porta et al., 2011, Zhou et al., 2007].

In this chapter, instead of identifying all tags, we focus on an important but not thoroughly investigated problem, *unknown tag identification* in large RFID systems. Unknown tags are those tags that just entered the system but have not been interrogated by the reader(s) covering them yet. For instance, in a large warehouse, the frequent loading of new products introduces unknown tags into the system, which should be interrogated in time to support automatic and efficient product management. Misplaced tags due to misoperation of stevedores could also be treated as unknown tags by reader(s) currently covering them. Timely interrogation of these unknown tags should be carried out to avoid considerable economic profit loss caused by misplacement errors [Bu et al., 2012].

Although many approaches on efficient tag identification have been presented in the past several years [Klair et al., 2010, Namboodiri and Gao, 2010, Porta et al., 2011, Qian et al., 2010, Xie et al., 2010, 2013, Zhen et al., 2005], few of them dedicated to unknown tag identification. Existing work on tag identification mainly focuses on optimizing time/energy efficiency in tag identification [Namboodiri and Gao, 2010,

Zhen et al., 2005], or improving throughput by exploiting parallel working of multiple readers [Tang et al., 2009, 2011, Yang et al., 2011, Zhou et al., 2007], or utilizing mobile reader(s) to facilitate flexible tag reading [Xie et al., 2010, 2013, Zhu et al., 2013]. However, they all target at interrogating *all the tags* in the system from scratch, which would be highly time inefficient if directly adopted to solve the unknown tag identification problem. The low efficiency of these solutions in identifying unknown tags stems from redundant re-identification of already identified tags (hereinafter referred to as *known tags*), which usually constitute the majority of the tag population in a large RFID system.

The most related work on unknown tag identification is the *continuous scanning* (CU) scheme [Sheng et al., 2010], which is a probabilistic scheme that cannot guarantee complete identification of all the unknown tags. In safety-critical applications like medicine management in hospitals, it is strictly required that all the unknown tags must be identified. The CU scheme cannot meet this requirement. The execution time of CU increases along with the increase of the probability required to interrogate each unknown tag. When the probability is extremely high, the execution time might be comparable to or even longer than the trivial solution that identifies all the tags in the system.

In this chapter, we propose a series of protocols that can identify all the unknown tags with high time efficiency. We find that the key challenge in speeding up unknown tag identification is *how to effectively prohibit the involvement of known tags when interrogating unknown tags*. Our first contribution is to propose a method to recognize known tags and deactivate them to prohibit their further replies. The recognition is realized by comparing expected replies of those known tags with actual received replies

in each slot of a frame.

Our second contribution is to develop two novel techniques, *slot pairing* (Section 3.4) and *multiple reselections* (Section 3.5) to resolve the known tag collision. These two techniques can greatly enhance the efficiency in deactivating known tags. They can also be applied as common techniques to improve tag identification efficiency in RFID systems. We build a series of time efficient unknown tag identification protocols on top of these techniques.

Our third contribution is to conduct extensive simulations to evaluate the performance of our unknown tag identification protocols with various parameter settings. The results show up to 70 percent reduction in execution time when compared to the best existing solutions. Furthermore, in order to set optimal frame size in our protocols, we develop a zero-cost algorithm to estimate the number of unknown tags that are currently present in the system and evaluate its accuracy.

We organize this chapter as follows. In Section 3.2 we describe system model and give problem statement. In Section 3.3 we describe the basic unknown tag identification protocol (BUIP) and point out directions to further improve its performance. The two novel techniques, i.e., slot pairing and multiple reselections, and the two enhanced protocols built on them, i.e., SUIP and MUIP, are described in Section 3.4 and Section 3.5, respectively. Section 3.6 analyzes the performance of SUIP and MUIP. Section 3.7 discusses fault tolerance issues. Section 3.8 presents the zero-cost unknown tag cardinality estimation algorithm. The performance of our protocols are evaluated and compared with state-of-the-art solutions through extensive simulations, and the results are reported in Section 3.9. Finally, Section 3.10 summaries the chapter.

3.2 Background

3.2.1 System Model and Problem Overview

Consider a large RFID system consisting of a reader R and a set of tags $T = \{k_1, \ldots, k_n, u_1, \ldots, u_m\}$, where $k_i (1 \leq i \leq n)$ denotes a know tag that has its ID in the system database, and $u_j (1 \leq j \leq m)$ denotes an unknown tag. All the known tag IDs are recorded in a back-end server. The objective is to *collect all the unknown tag IDs (i.e., $\{u_1, \ldots, u_m\}$) as quickly as possible*. Note that in the current RFID tag identification protocols, the reader has no simple way to differentiate unknown tags from known tags, and thus has to endure the interference from known tags when identifying unknown tags. This work aims to design novel techniques to quickly recognize known tags and prohibit their replying during the identification of unknown tags.

We use the *frame slotted ALOHA* protocol as the underlying MAC layer protocol. The communication between the reader and tags takes the form of *frame*, which is organized as a number of slots synchronized by the reader in which tags transmit information (e.g., tag ID) to the reader. The reader starts a frame with a query $<f, r>$, where f indicates the number of slots in the frame and r is a random seed used by tags to determine which slot to transmit. A tag hashes its ID to an integer in [0,f-1] by $S = H(ID||r)$ mod f and transmits in the S-th slot, where H is a uniform hash function. A tag can transmit either its ID or a short response (e.g., a 16-bits random number *RN16* in EPC C1G2 standard [EPCglobal, 2008]) to the reader. We denote by t_{ID} and t_l the time needed to transmit a tag ID and a short response, respectively.

The protocols proposed in this chapter can also be applied to RFID systems

containing multiple readers. In such cases, we resort to existing reader scheduling algorithms [Tang et al., 2009, 2011, Yang et al., 2011, Zhou et al., 2007] to obtain a conflict-free schedule of readers and run our protocols on every reader. Our protocols could also be tailored for mobile reader scenario(s) by treating tags identified at the previous site as known tags at the current site. The reader(s) can access the back-end server via wired or wireless link(s) to retrieve the known tag list before running the protocols and update the list afterwards.

3.2.2 Assumptions

It is possible that known tags might leave the system, and there are some researches on detecting such missing tags [Li et al., 2010b, Liu et al., 2013, Tan et al., 2010, Zhang et al., 2011]. In this work, we mainly focus on the unknown tag identification problem and assume that no known tags leave the system *during the identification of unknown tags*, which is usually very short, e.g., several minutes. This assumption has been taken in many excellent previous works [Chen et al., 2011, Yue et al., 2012]. To do this, the reader needs to trace which known tags have left the system before executing our protocols, and update the known tag list in the back-end server accordingly. In case that the reader cannot trace which known tags have left, we resort to *missing tag detection* protocols [Li et al., 2010b, Tan et al., 2010, Zhang et al., 2011] to find which known tags have left and update the known tag list accordingly. After unknown tags have been identified, we insert their IDs into the database to keep the known tag list updated.

The reader uses *indicator vectors* [Chen et al., 2011] to send some frame arranging information to tags. An indicator vector is a vector of bits that can be received and interpreted by tags. For example, we can set the bit associated with a slot to “1” to

prevent tags selecting this slot from transmitting. This technique has been used in many existing researches to improve protocol efficiency. Here we adopt the method given in [Chen et al., 2011] to implement indicator vector on top of EPC G1G2 compliant tags. The indicator vector is divided into segments of 96 bits long and each segment is encapsulated into a tag ID. The reader broadcasts segments one after other. Tags need to buffer only one segment in which their corresponding bit resides in. We can also add cyclic-redundancy check (CRC) code to each segment to ensure that the segment could be correctly received. The indicator vector technique based on the method given in [Chen et al., 2011] has been adopted in many excellent existing researches including [Chen et al., 2013, 2011, Li et al., 2010b, Liu et al., 2013, Qiao et al., 2011, Yue et al., 2012, Zheng and Li, 2013b].

3.3 BUIP: The Basic Unknown Tag Identification Protocol

In this section, we propose the Basic Unknown tag Identification Protocol (BUIP) and analyze its performance.

3.3.1 Protocol Design

BUIP consists of two phases: the *known tag deactivation* phase and the *unknown tag collection* phase. In the first phase, the reader recognizes and deactivates all the known tags to prevent them from interfering with the identification of unknown tags in the second phase. In the second phase, the reader completely identifies unknown tags by collecting their IDs.

The known tag deactivation phase consists of multiple rounds. In each round, the

reader collects replies from tags, based on which it recognizes and deactivates known tags (Section 3.3.2). The reader also recognizes and *labels* unknown tags to prevent them from interfering with the recognition of known tags. To shorten the execution time of each round, the reader broadcasts an indicator vector to prohibit tags in collision slots from replying, because collision slots could not be used to recognize known tags or unknown tags.

The number of active known tags decreases after each round. The reader traces how many known tags have been deactivated. When all the known tags have been deactivated, it enters the second phase to identify all the unknown tags. Otherwise, it starts a new round to deactivate the remaining known tags.

3.3.2 Known Tag Deactivation and Unknown Tag Labeling

In this section we develop a technique to recognize known tags and unknown tags by comparing the actually received replies in a slot with the expected relies.

Being aware of known tags' ID, the reader knows what tags will transmit in which slot. Consequently, the reader knows the *expected reply number* in each slot (i.e., the number of known tags transmitting in this slot) if there are no unknown tags. However, replies from unknown tags might make the *actual reply number* in a slot different from the expected value. The matching/mismatching between the two numbers provides us opportunities to recognize known tags and unknown tags: If the actual reply number equals the expected reply number, then all the replying tags *must* be known tags; on the other hand, if the actual reply number is larger than the expected value, there *must be* some unknown tags replying.

We implement the known/unknown tag recognition method as follows. In the ALOHA protocol the reader can obtain only coarse-grained status of a slot: empty

(the reply number is 0), singleton (the reply number is 1), and collision (the reply number is larger than 1). Because the reader cannot know the exact reply number in a collision slot, it cannot determine whether the actual reply number in a collision slot matches its expected value or not. Thus, we use only expected empty slots (whose expected reply number is 0) and expected singleton slots (whose expected reply number is 1) to recognize known and unknown tags:

- **Recognize known tags:** If the reader receives only one reply in an expected singleton slot, it recognizes the replying tag as a known tag.

- **Recognize unknown tags:** If the reader receives one or more replies in an expected empty slot, it recognizes the replying tag(s) as unknown tag(s).

After recognizing known tags and unknown tags, the reader deactivates or labels them accordingly. For recognized known tags, the reader deactivates them in both of the two phases in our protocol. In contrast, for recognized unknown tags, the readers make them *temporally inactive* only in the first phase, which we call *unknown tag labelling*. The purpose of labelling is to prevent unknown tags from interfering the recognition of known tags. All the labelled unknown tags will become active again in the second phase to perform identification.

Known tag deactivation and unknown tag labelling are implemented by sending different types of acknowledgements to the tag. The EPC specification [**?**] provides two different types of acknowledgements: ACK to deactivate a tag and NAK to keep a tag active. Noting that in our protocol we need to differentiate between deactivation of known tags and labelling of unknown tags, we extend ACK to implement the two different purposes: We use ACK_d for known tag deactivation and use ACK_l for

unknown tag labelling. We append one bit to ACK to differentiate ACK_d and ACK_l: $ACK_d = ACK +' 0'$ and $ACK_l = ACK +' 1'$. If a tag receives ACK_d, it will not respond to the reader during the execution of the whole protocol. If a tag receives ACK_l, it enters the labeled status and responds to only the ID-collection command issued by the reader in the second phase.

3.3.3 Protocol Description

BUIP consists of two phases: the known tag deactivation phase and the unknown tag collection phase. In the second phase, the reader employs existing protocols like DFSA [Lee et al., 2005] to collect all unknown tag IDs. We focus on the first phase.

The first phase consists of multiple rounds. In each round, the reader first broadcasts a query $<f, r>$ and an indicator vector $\mathbf{v}_c$ to tags. $\mathbf{v}_c$ is an f-bits long vector in which the S-th bit indicates whether the S-th slot would be colliding or not. The reader constructs $\mathbf{v}_c$ according to the remaining active known tags in the current round. The reader predicts in which slot each known tag will response according to the tag's ID and the query $<f, r>$. It thus knows the expected status of every slot and sets bits in $\mathbf{v}_c$ as follows: The S-th bit is set to '1' if the S-th slot would be colliding, and '0' otherwise.

After receiving the query, every tag first calculates its transmission slot index as $h=H(ID||r) \bmod f$. It then checks the h-th bit in $\mathbf{v}_c$. If the h-th bit is '0', it transmits a short response (e.g., *RN16* in EPC C1G2 standard [**?**]) to the reader in the selected slot; otherwise, it keeps silent because the selected slot must be colliding.

The reader collects replies from tags, recognizes known/unknown tags and deactivates/labels them. In the S-th slot, the reader sends different acknowledgements: 1) If the slot is an expected singleton slot and the reader receives only one reply, it

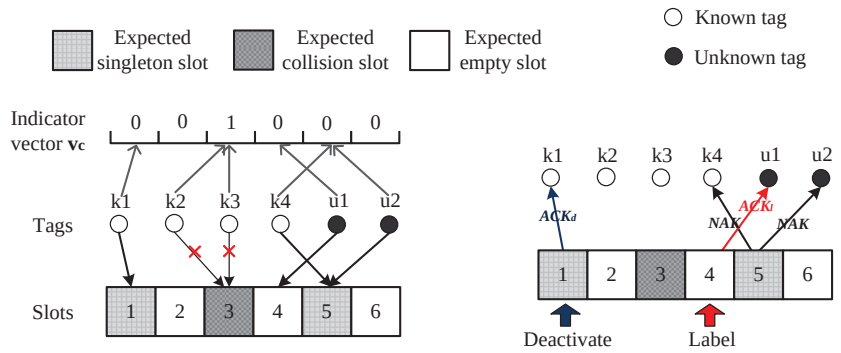


Fig. 3.1: Illustration of BUIP: (a) Prohibiting replies from tags mapped to expected collision slots; (b) Deactivating known tags and labelling unknown tags.

recognizes a known tag and sends an ACK_d to deactivate that tag. 2) If the slot is an expected empty slot and the reader receives some replies, it recognizes unknown tags and sends an ACK_l to label them. 3) In all other cases, the reader sends a NAK to keep tags active. At the end of each round, the reader counts the accumulated number of deactivated known tags up to the current round. If all the known tags have been deactivated, the reader enters the second phase and collects unknown tag IDs. Otherwise, it starts a new round to deactivate the remaining known tags.

Fig. 3.1 illustrates an example of BUIP with four known tags and two unknown tags. The reader first broadcasts $\mathbf{v}_c$ that indicates the third slot is colliding. The two known tags mapped to the third slot, i.e, k_2 and k_3, find $\mathbf{v}_c[2]$ = '1' and do not reply in this round, turning the third slot to an empty slot. Other tags reply in their chosen slots. The reader recognizes k_1 in the first slot and replies ACK_d to deactivate it. Meanwhile, the reader recognizes u_1 in the fourth slot and replies ACK_l to label it. In this example, the reader needs to start a new round to deactivate the rest three known tags.

3.3.4 BUIP Analysis and Discussions

We first derive how to set the frame size to maximize the known tag deactivation efficiency. Consider the i-th round. Let T_i and D_i be the execution time and the number of deactivated known tags in the i-th round, respectively. We define the *amortized cost* to deactivate a known tag as

$$C_i = \frac{T_i}{D_i}. \tag{3.1}$$

C_i represents the average time needed to deactivate a known tag in the i-th round, and thus should be minimized to optimize the known tag deactivation efficiency.

When the frame size is set at f_i, the execution time is $T_i = t_{ID} * \frac{f_i}{96} + f_i * t_l$, where the first part is the time used to broadcast the indicator vector $\mathbf{v}_c$, and the second part is the time used to collect replies from tags. Let P_d be the probability that a tag can be deactivated in an arbitrary slot S. Then the expected number of deactivated known tags is $D_i \approx f_i * P_d$. Thus, the amortized cost to deactivate a known tag when the frame size is f_i is

$$C_i = \frac{T_i}{D_i} = \frac{f_i * (\frac{t_{ID}}{96} + t_l)}{f_i * P_d} \propto \frac{1}{P_d}. \tag{3.2}$$

We can see that C_i is inversely proportional to P_d. Thus, we can reduce the amortized cost by increasing the probability P_d. However, as indicated by Remark 1, P_d cannot be increased arbitrarily by tuning only frame size.

Remark 1. *In BUIP, P_d is maximized when $f_i = m_i + n_i - 1$, and $P_d \leq e^{-1} \approx 0.368$.*

Proof: A slot S can be used to deactivate a known tag only when exactly one known tag selects S and no other tags select it, with probability

$$P_d = \binom{n_i}{1} \frac{1}{f_i} (1 - \frac{1}{f_i})^{n_i + m_i - 1} \approx \frac{n_i}{f_i} e^{-\frac{n_i + m_i - 1}{f_i}}. \tag{3.3}$$

It is easy to derive that P_d is maximized when $f_i = n_i + m_i - 1$, in which case

$$P_d \approx \frac{n_i}{f_i} e^{-1} \leq \frac{1}{e} \approx 0.368. \tag{3.4}$$

Remark 1 shows that we cannot obtain arbitrary large P_d by adjusting only frame size in BUIP. To further improve the deactivation efficiency of BUIP, we need to develop new techniques.

3.4 SUIP: Single-Pairing Unknown Tag Identification Protocol

In this section, we first present a novel technique called *slot pairing* that can greatly increase P_d, then present the Single-pairing Unknown tag Identification Protocol (SUIP).

3.4.1 Protocol Design

The novelty of SUIP is that it uses a novel technique to turn expected collision slots into expected singleton slots and thus improves known tag deactivation efficiency. Recall that in BUIP only expected singleton slots are used to deactivate known tags, and replies in expected collision slots are prohibited. In SUIP, we pair every expected collision slot with an expected empty slot and let tags mapped to the collision slot make a reselection between the two slots. For example, if two known tags k_1 and k_2 are mapped to an expected collision slot S, they are given another chance to reselect one slot from S and its pairing empty slot S'. After the reselection, it is possible that k_1 selects S and k_2 selects S' or vice versa. In this case, both S and S' turn into expected singleton slots and can be used to deactivate known tags. We call this technique *slot pairing*. Note that slot pairing can also resolve collisions when there

are more than two tags colliding. For example, if there are k ($k \geq 3$) known tags selecting the collision slot, then it is possible that only one of them reselects S (S') and all the other $k-1$ tags reselect S' (S), which also generates an expected singleton slot after slot reselection.

3.4.2 Slot Pairing

In slot paring, every *expected collision slot* is paired with an *expected empty slot*. As there are more expected empty slots than expected collision slots when the frame size is optimally set (i.e., $f_i = n_i + m_i - 1$), we can assign a pairing empty slot for *every* expected collision slot. Note that the number of expected empty slots ($f_i * e^{-n_i/f_i}$) is always larger than the number of expected collision slots ($f_i * (1 - \frac{n_i}{f_i} e^{-n_i/f_i} - e^{-n_i/f_i})$) when $n_i/f_i \leq 1$. Meanwhile, when $f_i \geq n_i$, most collisions are two-collisions [Vogt, 2002] (i.e., exactly two known tags select this slot), in which case two expected singleton slots could be generated if the two known tags select different slots after reselection.

We pair the j-th expected collision slot with the j-th expected empty slot. A tag that originally selects the j-th expected collision slot may reselect the j-th expect empty slot in the second chance. The challenge is how to inform a tag of the index of its paired slot in the frame.

In SUIP, we exploit two indicator vectors: the indicator vector $\mathbf{v}_c$ that is used to indicate the expected collision slots, and the indicator vector $\mathbf{v}_e$ that is used to indicate the expected empty slots. Both $\mathbf{v}_c$ and $\mathbf{v}_e$ have f bits. Each bit is associated with a slot at the same index location in the frame. However, these two indicator vectors are constructed in different ways:

- **In $\mathbf{v}_c$**: If the k-th slot is expected to be colliding, $\mathbf{v}_c[k]$ = '1'; otherwise, $\mathbf{v}_c[k]$ = '0'.

- **In $\mathbf{v}_e$**: If the k-th slot is expected to be empty, $\mathbf{v}_e[k]$ = '1'; otherwise, $\mathbf{v}_e[k]$ = '0'.

A tag t mapped to an expected collision slot has two kinds of index values: $I_f(t)$ denotes its slot index in the frame, which indicates how many slots there are before its slot in the frame, and $I_c(t)$ denotes specifically the collision slot index, which indicates how many expected collision slots there are before its slot in the frame (i.e., the number of '1's before its corresponding bit in $\mathbf{v}_c$). It determines the paired expected empty slot in two steps:

- First, t examines the collision index $I_c(t)$ of its slot by counting how many '1's there are before its slot bit in the vector $\mathbf{v}_c$.

- Second, t determines the index of its pairing empty slot in the frame by searching the $(I_c(t)+1)$-th '1' in the vector $\mathbf{v}_e$. Let the index of the $(I_c(t)+1)$-th '1' in $\mathbf{v}_e$ be $I'_f(t)$. It then uses the $I'_f(t)$-th slot as the paired slot.

3.4.3 Protocol Description

The second phase of SUIP is the same as in BUIP, thus we describe only the first phase.

The known tag deactivation phase consists of multiple rounds. At the beginning of each round, the reader computers the slot indexes of expected empty slots, expected singleton slots and expected collision slots in the frame with the known tag IDs, and then constructs two indicator vectors $\mathbf{v}_c$ and $\mathbf{v}_e$. The reader broadcasts a query $<f, r>$

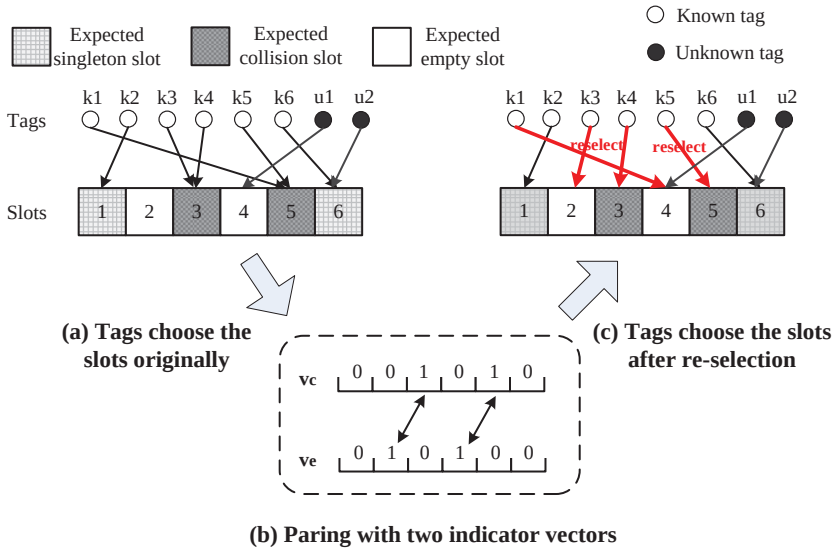


Fig. 3.2: The process of slot paring: (a) before slot reselection (same as BUIP); (b) slot pairing and reselection; (c) after slot reselection.

and two indicator vectors $\mathbf{v}_c$ and $\mathbf{v}_e$. It also broadcasts another seed r' for tags to reselect either of the two pairing slots.

Tags use the received information to determine their transmission slots. Upon receiving f and r, a tag t first calculates its slot index $I_f(t) = H(ID||r) \bmod f$. It then checks the bit value $\mathbf{v}_c[I_f(t)]$ to determine whether it should perform slot reselection or not. If $\mathbf{v}_c[I_f(t)] =$ '0', the tag responses in the original chosen slot $I_f(t)$. If $\mathbf{v}_c[I_f(t)] =$ '1', the tag needs to find its paired slot and perform reselection. Tag t determines its paired slot as previously described and randomly chooses either slot as its final transmitting slot. The reselection result is controlled by r': If $H(ID||r') \bmod 2 = 0$, the tag replies in its original slot; otherwise it replies in the paired slot.

When receiving the responses from tags, the reader deactivates the known tags in expected singleton slots and labels unknown tags in expected empty slots, the same

as in BUIP. Note that with r' and IDs of known tags, the reader exactly knows all reselection results of known tags. Thus the reader knows the new expected singleton slots transformed from the expected collision slots and expected empty slots after reselection.

Fig. 3.2 illustrates an example of SUIP. We can see that before slot pairing and reselection (Fig. 3.2(a), actually as same as BUIP), only one known tag (k_2) can be deactivated. In contrast, after slot pairing and reselection, four known tags (k_2, k_3, k_4, and k_5) can be recognized and deactivated. Fig. 3.2 also illustrates the negative effects of slot pairing on unknown tag recognition. In Fig. 3.2(a) the unknown tag u_1 can be recognized. However, after slot pairing and reselection, there are no expected empty slots and thus no unknown tags can be recognized. As the execution time of our protocols is mainly affected by the efficiency in deactivating known tags, SUIP achieves much better overall performance than BUIP.

3.5 MUIP: Multi-Pairing Unknown Tag Identification Protocol

In this section, we propose a Multi-pairing Unknown tag Identification Protocol (MUIP) that resolves collision slots with a higher probability and further improves the efficiency in deactivating known tags.

3.5.1 Protocol Overview

MUIP is similar to SUIP except that it makes two changes to further resolve the collisions between tags.

First, MUIP provides multiple chances to separate collided known tags between

pairing slots, which could resolve the expected collision slots with a higher probability. Reselecting with one seed r_i as in SUIP, colliding tags could be separated with a probability of only about 50%. We observe that a collision slot that cannot be resolved with r_i may be resolved with another seed r_j. If the reader sends *multiple seeds* and let tags use the most suitable one, more collision slots could be resolved.

Second, in MUIP, unknown tags label themselves without sending responses to the reader, which avoids collisions between unknown tags and known tags in expected empty slots after slot reselection. An expected empty slot may be actually *non-empty* if some unknown tags hash to it. When a known tag reselects such a non-empty slot, it would collide with the unknown tags and cannot be deactivated successfully. For example, as shown in Fig. 3.2, although k_1 reselects the expected empty slot 4 which resolves the collision in slot 5, it collides with u_1 and still cannot be deactivated. If we can guarantee that the paired slots are actually empty, the reader would deactivate more known tags.

3.5.2 Slot Reselection Using Multiple Seeds

The reader will send h seeds $\{r_1, r_2, ..., r_h\}$ to provide multiple reselection chances for known tags that hash to expected collision slots. Then the collided tags have h possible reselection results by hashing with the h seeds. If one of these results separates the collided known tags (i.e., at least one expected singleton slots will be generated with a seed r_i), the expected collision slot could be resolved successfully. Remember that, when reselecting with a seed r_i, the tag calculates $H(ID||r_i)$ mod 2. It replies in its original slot if the value equals 0, and replies in the paired slot otherwise. The process that a tag finds its paired slot is the same as in SUIP.

The reader examines the hash results of the known tags with the h seeds, and

determines which one should be used for each expected collision slot. The problem is how to inform different known tags that which seed is the suitable one.

We exploit a *seed-selection vector* $\mathbf{v}_s$ to indicate the suitable seed assigned for each expected collision slot. $\mathbf{v}_s$ consists of n_c elements, where n_c is the number of expected collision slots (i.e., one element for each expected collision slot). The i-th element of $\mathbf{v}_s$ is constructed as follows:

- If seed r_{l+1} $(0 \leq l \leq h)$ should be used to conduct reselection by tags mapping to the i-th collision slot, we set $\mathbf{v}_s[i]$ as "$\underbrace{0 \ldots 0}_{l}1$".

To obtain the assigned seed, a tag t has to find the $I_c(t)$-th element in $\mathbf{v}_s$. When t finds that it chooses an expected collision slot according to $\mathbf{v}_c$, it first obtains the collision index $I_c(t)$ of its chosen slot. Then it locates the $I_c(t)$-th '1' in the vector $\mathbf{v}_s$, and counts the number of '0's of $I_c(t)$-th element in $\mathbf{v}_s$ (i.e., the number of '0's between the $(I_c(t)-1)$-th '1' and the $I_c(t)$-th '1'). If there are l '0's, then the tag uses the seed r_{l+1}. If all the h seeds cannot resolve the expected collision slot, the corresponding element in $\mathbf{v}_s$ will be set as "1", which indicates that seed r_1 should be used.

Fig. 3.3 illustrates an example of the construction of the indicator vector $\mathbf{v}_s$. We can see that in $\mathbf{v}_s$, each element indicates a seed. The example shows the reselection of the known tags in the first two expected collision slots. Two tags, k_1 and k_3, know that they choose the first expected collision slot after receiving $\mathbf{v}_c$ and $\mathbf{v}_e$. These two tags find their assigned seed r_3 indicated by the first element in $\mathbf{v}_s$. Thus k_1 and k_3 reselect either of the two pairing slots with the hash function $H(ID||r_3)$ mod 2. As the result, k_1 chooses the second slot and k_3 chooses the third slot, which resolves the first expected collision slot successfully. The reselection process of k_4 and k_5 in the

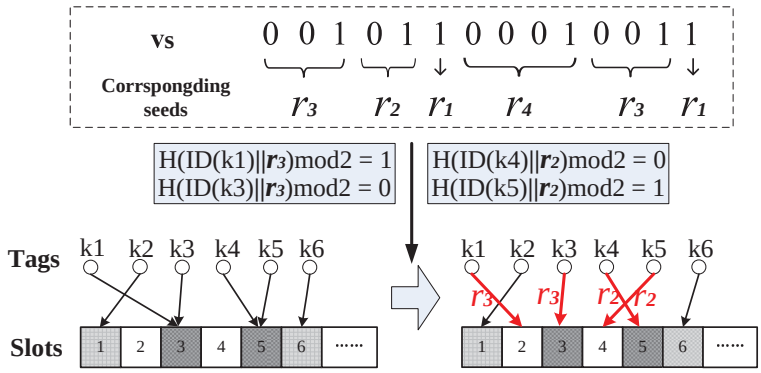


Fig. 3.3: The process of multiple reselections: Tags mapped to different expected collision slots use different seeds in slot reselection.

second expected collision slot is similar.

3.5.3 Automatical Unknown Tag Labeling

Except for reselecting with a suitable seed, to successfully deactivate the known tag in the paired expected empty slot, the other condition is that the slot must be actually empty (i.e., no known tag chooses the slot). Otherwise, the reselecting known tag will collide with unknown tags. As shown in Fig. 3.2, the reselecting known tag k_1 still collides with the unknown tag u_1. However, in SUIP, the reader needs the responses of unknown tags in expected empty slots to label them, which consequently reduces the efficiency of known tag deactivation.

In MUIP, unknown tags label themselves if they choose expected empty slots. Obviously, if a tag chooses an expected empty slot, it must be an unknown tag. In this case, the tag could label itself without receiving the NAK from the reader. A tag could test whether it chooses an expected empty slot by checking corresponding bit in the indicator vector $\mathbf{v}_e$ at the location of its slot index. If the bit value is '1', it labels itself without responding to the reader. Otherwise, it determines its slot and

responds to the reader in the chosen slot.

3.5.4 MUIP Description

The known tag deactivation phase of MUIP consists of multiple rounds. At the beginning of each round, the reader broadcasts a query $< f, r >$ and three indicator vectors $\mathbf{v}_c$, $\mathbf{v}_e$ and $\mathbf{v}_s$. It also broadcasts h reselection seeds $\{r_1, r_2, ..., r_h\}$ for tags to reselect either of the two pairing slots. Upon receiving these parameters, tags determine their transmission slots. Assume that tag t originally selects slot k. It checks the expected status of its chosen slot in both $\mathbf{v}_c$ and $\mathbf{v}_e$:

- If $\mathbf{v}_c[k]$ = '0' and $\mathbf{v}_e[k]$ = '0', which indicates that it chooses an expected singleton slot, the tag responds to the reader in the chosen slot k.

- If $\mathbf{v}_c[k]$ = '0' and $\mathbf{v}_e[k]$ = '1', which indicates that it chooses an expected empty slot, the tag labels itself and keeps silence until receiving the ID-collection command.

- If $\mathbf{v}_c[k]$ = '1', which indicates that it chooses an expected collision slot, the tag needs to reselect its transmission slot. It first examines its paired slot in $\mathbf{v}_e$ and then reselects its slot with the assigned seed indicated in $\mathbf{v}_s$. The tag responds to the reader in the reselected slot.

When receiving responses from tags, the reader deactivates known tags in expected singleton slots and labels unknown tags in expected empty slots as done in SUIP.

3.6 Analyses of SUIP and MUIP

In this section, we analyze the performance of SUIP and MUIP. As shown in Section 3.3.4, the average time to deactivate a known tag is inversely proportional to P_d, thus we mainly focus on analyzing P_d in SUIP and MUIP.

3.6.1 Analysis of SUIP

Without loss of generality, we consider an arbitrary slot in the i-th round. Denote by P_E, P_S, P_C the probability that this slot is an expected empty slot, an expected singleton slot, and an expected collision slot, all before slot pairing and reselection, respectively. Denote by $P\{D|S\}$ the probability that a known tag is deactivated in an expected singleton slot. Similarly, denote by $P\{D|C\}$ and $P\{D|E\}$ the probability that a known tag is deactivated in an expected collision and in an expected empty slot after slot reselection, respectively. Then we have

$$P_d = P_S * P\{D|S\} + P_C * P\{D|C\} + P_E * P\{D|E\}. \tag{3.5}$$

It is easy to calculate P_E, P_S and P_C as

$$P_E = e^{-n_i/f_i}, P_S = \frac{n_i}{f_i} e^{-n_i/f_i}, P_C = 1 - P_E - P_S,$$

where $f = m_i + n_i - 1$. In the following, we discuss how to calculate $P\{D|S\}$, $P\{D|C\}$ and $P\{D|E\}$.

Calculation of P{D|S}: As same as in BUIP, the probability that a known tag can be deactivated in an expected singleton slot equals the probability that no unknown tag selects this slot, i.e.,

$$P\{D|S\} = (1 - \frac{1}{f_i})^{m_i} \approx e^{-m_i/f_i}. \tag{3.6}$$

Calculation of P{D|C}: For an expected collision slot, we denote by $P_C(k_1, k_2)$ the conditional probability that exactly k_1 known tags and k_2 unknown tags select it. Because all the n_i known tags select replying slot independently, k_1 follows a binomial distribution $B(n_i, 1/f_i)$ that can be approximated with a *Poisson distribution* with parameter $\lambda_n = n_i/f_i$. Similarly, k_2 also follows a Poisson distribution with parameter $\lambda_m = m_i/f_i$. Noting $\lambda_n + \lambda_m \approx 1$, we have

$$\begin{aligned} P_C(k_1, k_2) &= \frac{1}{P_C} * e^{-\lambda_n} * \frac{\lambda_n^{k_1}}{k_1!} * e^{-\lambda_m} * \frac{\lambda_m^{k_2}}{k_2!} \\ &\approx \frac{e^{-1}}{P_C} \frac{\lambda_n^{k_1} \lambda_m^{k_2}}{k_1! k_2!}. \end{aligned} \tag{3.7}$$

Let $P\{D|k_1, k_2\}$ be the probability that a known tag is deactivated in this collision slot after reselection. Then $P\{D|k_1, k_2\}$ equals the probability that exactly one of the k_1 known tags selects this slot, and the other $k_1 - 1$ known tag(s) and k_2 unknown tags all select the paired slot after reselection, which is

$$P\{D|k_1, k_2\} = k_1 * \frac{1}{2} * (1 - \frac{1}{2})^{k_1+k_2-1} = k_1 * (\frac{1}{2})^{k_1+k_2}. \tag{3.8}$$

Combining equation 3.7 and equation 3.8, we have (the detailed derivation is given in Appendix)

$$\begin{aligned} P\{D|C\} &= \sum_{k_1=2}^{n_i} \sum_{k_2=0}^{m_i} P_C(k_1, k_2) * P\{D|k_1, k_2\} \\ &\approx \frac{e^{-1/2} \lambda_n}{2 * P_C} (1 - e^{-\lambda_n/2}). \end{aligned} \tag{3.9}$$

Calculation of P{D|E}: An expected empty slot needs to satisfy three conditions to be able to deactivate a known tag: (1) it is a paired slot of some collision slot; (2) no unknown tag selects it; and (3) after reselection, only one known tag reselects it. As there are $f_i * P_E$ expected empty slots and only $f_i * P_C$ expected collision slots, the

probability for an expected empty slot to satisfy the first condition is $(f_i * P_C)/(f_i * P_E) = P_C/P_E$. The probability to satisfy the second condition is approximately e^{-m_i/f_i}. The probability to satisfy the third condition equals $P\{D|C\}$. Thus, we can calculate $P\{D|E\}$ as

$$P\{D|E\} = \frac{P_C}{P_E} * e^{-m_i/f_i} * P\{D|C\}. \tag{3.10}$$

Substituting equations 3.6, 3.9, 3.10 into equation 3.5, we obtain

$$P_d = \lambda_n e^{-1} + \frac{e^{-1/2}\lambda_n}{2}(1 - e^{-\lambda_n/2})(1 + e^{\lambda_m}). \tag{3.11}$$

Remark 2. *In SUIP, P_d is a monotonically decreasing function of λ_m/λ_n, and it achieves maximum value when $\lambda_n = 1$ (i.e., $\lambda_m = 0$), which is*

$$P_d^{max} = e^{-1} + e^{-1/2}(1 - e^{-1/2}) \approx 0.6065. \tag{3.12}$$

Fig. 3.4 plots P_d in SUIP vs. P_d in BUIP when λ_m/λ_n increases from 0 to 1. We observe significant increase of P_d in SUIP compared with in BUIP: P_d in SUIP is 65 percent higher than that in BUIP when $\lambda_m/\lambda_n = 0$ (i.e., there are no unknown tags in the system), and is 29 percent higher when $\lambda_m/\lambda_n = 1$ (i.e., when $n_i = m_i$). The improvement gradually decreases due to interferences caused by unknown tags.

3.6.2 Analysis of MUIP

Impact of h on P_d

In this section we derive $P_d(h)$, the probability that a known tag can be deactivated in an arbitrary slot when h reselection seeds are used. We denote by $P_h\{D|C\}$ and $P_h\{D|E\}$ the probability that a known tag is deactivated in an expected collision slot and in an expected empty slot when h seeds are used, respectively. Using the same notations defined in Section 3.6.1, we have

$$P_d(h) = P_S * P\{D|S\} + P_C * P_h\{D|C\} + P_E * P_h\{D|E\}. \tag{3.13}$$

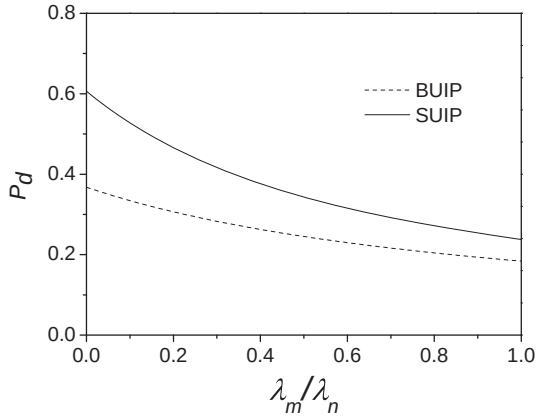


Fig. 3.4: P_d in SUIP vs. P_d in BUIP when λ_m/λ_n varies.

Recall that the probability of a known tag t can be deactivated in an original expected collision slot with one seed is $P\{D|C\}$. Thus the probability that t cannot be deactivated with all of the h seeds is $(1 - P\{D|C\})^h$, which leads to

$$P_h\{D|C\} = 1 - (1 - P\{D|C\})^h. \tag{3.14}$$

As in SUIP, we can calculate $P_h\{D|E\}$ as

$$P_h\{D|E\} = \frac{P_C}{P_E} * e^{-m_i/f_i} * P_h\{D|C\}. \tag{3.15}$$

Substituting equations 3.14, 3.15 into equation 3.13, we have

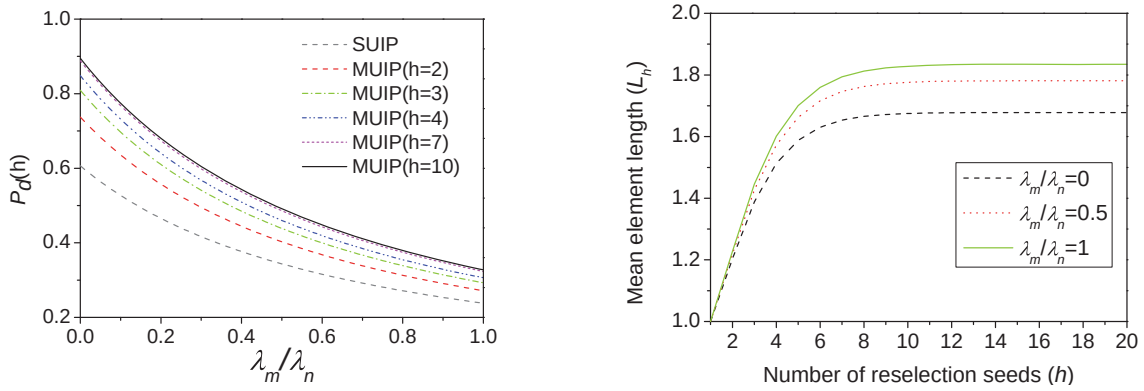
$$P_d(h) \;=\; \lambda_n e^{-1} + P_C * (1 + e^{1-\lambda_n}) * \Psi, \tag{3.16}$$

where

$$\Psi = 1 - \left[1 - \frac{e^{-1/2}\lambda_n}{2 * P_C}(1 - e^{-\lambda_n/2})\right]^h. \tag{3.17}$$

Remark 3. *When λ_m/λ_n is fixed, $P_d(h)$ is a monotonically increasing function of h.*

Proof: From equation 3.17 it is easy to see that Ψ is a monotonically increasing function of h, which immediately implies Remark 3 according to equation 3.16.



(a) n (b) m

Fig. 3.5: (a) $P_d(h)$ in MUIP vs. P_d in SUIP when λ_m/λ_n varies, and (b) Mean segment length when h increases.

Table 3.1: Ratio of singleton slots in MUIP and MIC[Chen et al., 2011] with different h.

h	2	3	4	5	6	7
MIC	0.580	0.696	0.764	0.808	0.839	0.861
MUIP	0.607	0.737	0.809	0.849	0.870	0.881

Fig. 3.5(a) plots $P_d(h)$ for different h when λ_m/λ_n varies from 0 to 1. We observe that when h is small (e.g., when $h = 2, 3, 4$), increasing h can significantly increase $P_d(h)$. On the other hand, when h is large enough, increasing h leads only marginal increase of $P_d(h)$. For example, we can observe that $P_d(10)$ is nearly the same as $P_d(7)$. Actually, the increase of h cannot arbitrarily increase $P_d(h)$: Because $\lim_{h\to\infty} \Psi = 1$, we have

$$\lim_{h\to\infty} P_d(h) = \lambda_n e^{-1} + P_C * (1 + e^{1-\lambda_n}). \tag{3.18}$$

When simulating MUIP in Section 3.9, we set $h = 7$.

Multiple Reselections Technique Efficiency

We compare the performance of our multi-pairing technique with the multiple hashing approach MIC proposed in [Chen et al., 2011] from two aspects: The probability of a slot becoming singleton when h seeds are used, and the total length of the bit vectors sent from the reader to tags.

Table 3.1 lists the ratio of expected singleton slots in MUIP and MIC when different number of seeds are used. It shows that MUIP generates more singleton slots than MIC does with the same number of seeds. In MIC, the colliding tags can reselect any slot that has not been occupied. With one reselection chance, the probability that a colliding tag can reselect an expected singleton slot is about $1/e \approx 0.368$. Thus the efficiency of this technique is affected mainly by *the number of reselection chances*. On the other hand, in MUIP, a colliding tag reselects between two paired slots. For a two-collision slot (i.e. the collision is caused by two tags responding simultaneously), the slot paring technique has a probability of 0.5 to generate two expected singleton slots. For a collision slot in which k tags responding, slot pairing can still generate one expected singleton slot with a probability of $k/2^{k-1}$. The efficiency of MUIP is affected mainly *by the number of colliding tags in a slot*. With the optimal frame size in both MUIP and MIC, most collisions are two-collisions. As a result, MUIP performs better than MIC.

In MIC, for every slot the reader broadcasts three bits to represent the seed index that tags mapped to this slot should use to select replying slot. Thus the total number of bits broadcasted in MIC is $3 * f_i$. In contrast, in MUIP the reader broadcasts two f_i-bits long vectors to notify tags which slots are empty and collision. Besides, in MUIP the reader also broadcasts a seed-selection vector $\mathbf{v}_s$ for *only collision slots*.

Let L_h be the mean length of elements in $\mathbf{v}_s$. The total number of bits broadcasted in MUIP is $2 * f_i + f_i * P_C * L_h$, where P_C is the probability that a slot is collision.

We derive L_h as follows. Recall that in MUIP if the l-th seed should be used in slot reselection, the length of corresponding element is l. Denote by $P_s(l)$ the probability that the l-th seed should be used in slot reselection. Noting that the first seed is used when none of the h seeds can resolve the collision slot, we can calculate the mean element length as

$$L_h = 1 * \left[1 - \sum_{l=2}^{h} P_s(l)\right] + \sum_{l=2}^{h} l * P_s(l). \tag{3.19}$$

$P_s(l)$ equals the probability that the collision slot cannot be resolved by $r_1, \ldots, r_{l-1}$ but can be resolved by r_l, thus

$$P_s(l) = (1 - P\{D|C\})^{l-1} * P\{D|C\}. \tag{3.20}$$

Substituting equation 3.20 into equation 3.19, with a given λ_m/λ_n, we can calculate L_h for different h. Fig. 3.5(b) plots L_h for different λ_m/λ_n when h increases. When $h = 7$ and $\lambda_m/\lambda_n = 0$, i.e., the scenario that MIC addresses, $L_h \approx 1.6533$. Note that $P_C = 1 - e^{-n_i/f_i} - \frac{n_i}{f_i} e^{-n_i/f_i} \leq 1 - 2 * e^{-1} \approx 0.2642$ when $\frac{n_i}{f_i} \leq 1$. Thus the total number of bits broadcasted in MUIP is only $2 * f_i + f_i * P_C * L_h \leq (2 + 0.2642 * 1.6533) * f_i = 2.4369 * f_i$, much less than the $3 * f_i$ bits in MIC.

It can also be observed from Fig. 3.5(b) that L_h is always smaller than 2, which shows that our approach represents seed index more efficiently than MIC does. The small average length of elements owe to the *variable length coding* of the seed index: Most collision slots are resolved by the first or the second seed (see Fig. 3.5(a)), and when none of the h seeds can resolve the collision slot we just use the first seed.

3.7 Fault Tolerance

In this section we discuss how to cope with channel errors to enhance the robustness of our protocols.

3.7.1 Reader to Tag Transmission Error

The indicator vector transmitted from the reader to tags might be corrupted due to channel errors. If the bit corresponding to a tag's replying slot is incorrectly received (i.e., the sent bit is "0" but the received bit is "1" or vice versa), the tag might take incorrect actions. In order to help tags check whether the received information is correct or not, we can add a cyclic-redundancy check (CRC) code in each segment when transmitting the indicator vector. For example, we can divide the indicator vector into 80 bits long segments and add a 16 bits CRC code to each segment. After receiving the indicator vector, the tag first checks whether its indicator bit is correctly received. If the segment is correctly received, the tag takes its action according to its corresponding bit as described in our protocols. Otherwise, it does not participate in the current round, but will keep active to participate in the next round. For SUIP and MUIP, a tag participates in the current round only when it correctly receives all its corresponding bits.

With this mechanism, the correctness of our protocols can be guaranteed even when transmission error happens. The key point to guarantee the correctness of our protocols is that the reader have to correctly trace how many known tags have been deactivated, according to which it determines when to terminate the first phase. Recall that the reader deactivates known tags only when it receives exact one reply in expected singleton slots. If a known tag receives corrupted information, it will

not participate in the current round. However, the other unknown tags selecting the same slot, if there are some, will also not participate in the current round due to transmission error. Thus the reader will not receive any response in the slot. If the reader receives no reply in an expected singleton slot, it skips this slot and does not count the corresponding known tag as being deactivated. In this way, the reader can trace the *correct number of deactivated known tags*, and thus can guarantee the correctness of the protocols.

The time efficiency of the protocols would degrade slightly when we use this scheme. The EPC C1G2 specification [EPCglobal, 2008] provides two types of CRC code: CRC16 that uses 16 bits and CRC5 that uses 5 bits. If we adopt CRC16, according to the time setting given in the specification [EPCglobal, 2008], in each round the execution time of BUIP will be increased by

$$\frac{t_{ID} * (\frac{f_i}{80} - \frac{f_i}{96})}{t_{ID} * \frac{f_i}{96} + f_i * t_l} = \frac{2.4 * (\frac{1}{80} - \frac{1}{96})}{2.4 * \frac{1}{96} + 0.44} = 0.0108, \tag{3.21}$$

which is only about 1.1 percent. Similarly, when CRC16 is used, the execution time of SUIP and MUIP will be increased by no more than 2 percent and 3 percent, respectively. If we use the shorter CRC5 code, the execution time will be increased by only 0.3, 0.6, and 0.8 percent in BUIP, SUIP, and MUIP, respectively.

3.7.2 Tag to Reader Transmission Error

Our protocols can tolerate the tag to reader transmission errors. Note that in our protocols tags transmit 16-bits long short responses (*RN16*) to the reader rather than a single bit. The main purpose of the responses is to help the reader differentiate different status of a slot (e.g., empty, singleton, or collision) to recognize known tags and unknown tags. The transmission error will not affect the status of a slot, and

thus will not break the correctness of our protocols.

3.8 Zero-Cost Estimation of Unknown Tags

Our protocols need to know the number of active unknown tags (m_i) to set the optimal frame size. If we use a separate estimation algorithm [Han et al., 2010, Kodialam and Nandagopal, 2006, Qian et al., 2011, Zheng and Li, 2012] to estimate m_i, it will inevitably increase the execution time. We develop a zero-cost estimation algorithm to estimate m_i by using the information collected in the deactivation phase.

The algorithm is motivated by the observation that the actual status of a slot may differ from its expected status because of unknown tags' interference. For example, if an unknown tag replies in an expected empty (singleton) slot, the actual status of the slot will be singleton (collision). The ratio of slots whose actual status are different from their expected status is related to the number of active unknown tags, i.e., m_i. Thus, by counting how many slots change their status in the previous round, we can estimate m_i in the current round.

Our estimation algorithm works as following. Consider the i-round in which there are n_i known tags and m_i unknown tags, respectively, and assume that the frame size is f_i. Let R_U be the ratio of tags whose actual status differ from expected status. For slot S, its actual status differs from its expected status if and only if there are at least one unknown tags select S, with probability

$$1 - (1 - \frac{1}{f_i})^{m_i} \approx 1 - e^{-m_i/f_i}. \tag{3.22}$$

If we know R_U and f_i, we can estimate m_i by letting

$$R_U = 1 - e^{-m_i/f_i}, \tag{3.23}$$

from which we can derive that

$$m_i = -f_i * \ln(1 - R_U). \tag{3.24}$$

The problem is that f_i also depends on m_i (recall that f_i should be set to $n_i + m_i - 1$). To solve this dilemma, we assume $m_1 = 0$ and set $f_1 = n$ in the first round. At the end of the first round, the reader can obtain R_U by counting the number of slots changed from empty to non-empty or from singleton to collision. It then estimates m_1 with equation 3.24. Note that m_1 is the number of active unknown tags *at the beginning of this round*, but we need to know the number of active known tags *at the end of this round* to set the frame size in the next round. Let Δm_1 be the number of labelled unknown tags in the first round, then we have

$$m_2 = m_1 - \Delta m_1. \tag{3.25}$$

We then can use m_2 to set frame size in the second round. With this method, we can estimate m_{i+1} with the information collected in the i-th round and set the frame size in the $(i+1)$-th round accordingly.

We estimate the number of the labelled unknown tags in the i-th round, i.e., Δm_i, as following. Let r_e be the ratio of expected empty slots in the i-th round. Because unknown tags select their transmission slots uniformly, Δm_i can be estimated as

$$\Delta m_i = m_i * r_e, \tag{3.26}$$

where r_e can be calculated as the ratio of the number of expected empty slots N_e to the total number of slots in the frame, i.e.,

$$r_e = \frac{N_e}{f_i}. \tag{3.27}$$

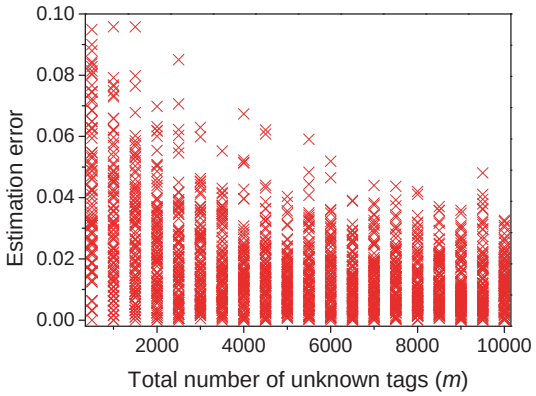


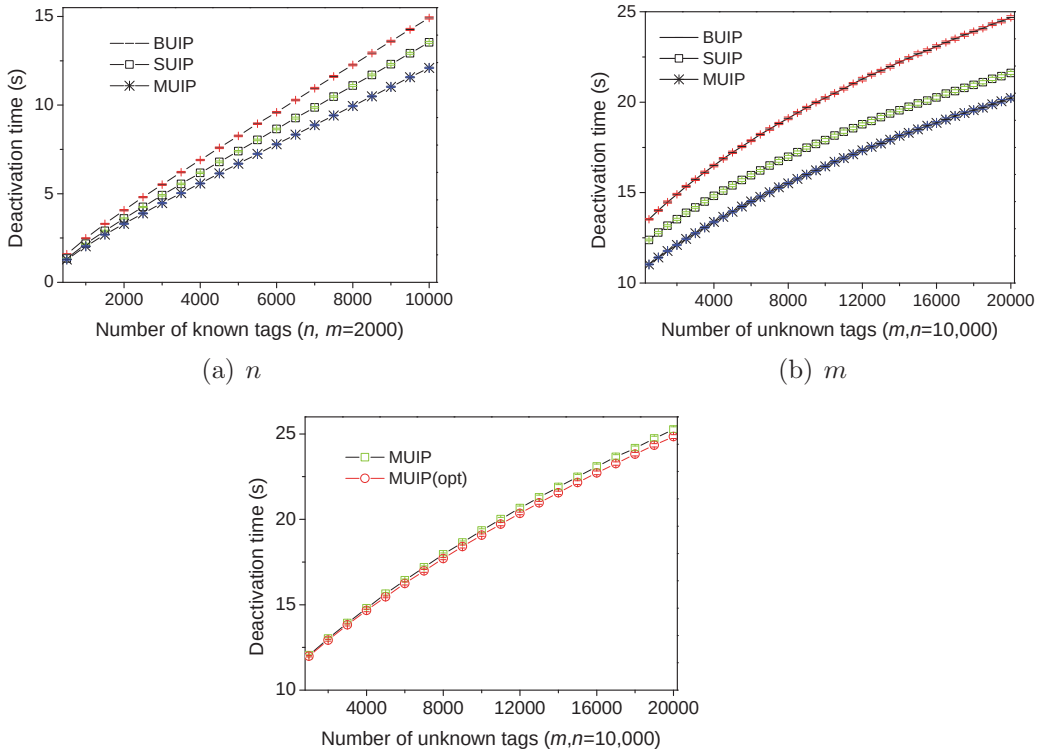
Fig. 3.6: Estimation error when m changes (n=10,000).

Fig. 3.6 plots the estimation error of our algorithm when m varies from 500 to 10000 stepped by 500, assuming $n = 10000$. For each m we plot the estimation error of our algorithm in 100 independent executions of BUIP. It can be seen that the estimation error decreases along with the increase of m, and is smaller than 0.05 in most cases. We point out here that our estimation algorithm incurs no additional cost to our unknown tag identification protocols, because it utilizes the information collected in the i-th round to estimate m_{i+1} in the $(i + 1)$-th round.

3.9 Performance Evaluation

3.9.1 Simulation settings

We develop a simulator with JAVA to evaluate the performance of our protocols. Our protocols are compared with two methods: A *Baseline* method that collects IDs of all the tags in the system, and an *Ideal* method that collects IDs of only unknown tags. The execution time of the Baseline method is an upper bound on the time needed to identify all the unknown tags, and the execution time of the Ideal method represents a lower bound. We also compare our protocols with the CU



(a) n (b) m

(c) m

Fig. 3.7: Deactivation time of different protocols: (a) when n changes; (b) when m changes; (c) Impact of estimation error.

scheme [Sheng et al., 2010], which is the most efficient probabilistic unknown tag identification algorithm up to now.

We adopt the timing scheme specified for the EPC Global Class 1 Generation 2 UHF tags [EPCglobal, 2008] to compute the execution time of different protocols. We set the bidirectional transmission rate between the reader and tags at 62.5 Kbps. With this setting, it takes about 2.4ms to transmit a tag ID and 0.44ms to transmit a 16-bit short response, i.e., $t_{ID} = 2.4ms$ and $t_l = 0.44ms$. An empty slot takes 0.184ms. A slot in CU takes the same time as an empty slot.

The main performance metric is the execution time. We consider two system

parameters, the number of known tags (n) and the number of unknown tags (m), and tune their values to study their impacts on the execution time of different protocols. Their default values are set as $n = 10,000$ and $m = 2,000$, respectively. For MUIP, we set the number of reselection seeds at $h = 7$. For every parameter setting, we run the considered protocols in 100 independent instances and report the average data.

We use the approach proposed in [Liu et al., 2013] to detect missing tags before executing our protocols, and add this overhead to the execution time of our protocols when comparing with other approaches. We implement the DFSA [Lee et al., 2005] protocol to identify unknown tags in CU and our protocols. When implementing DFSA, we use the optimal frame size, which is fair to all the considered approaches. We note that the identification throughput may degrade in real environments [Kang et al., 2012, Xie et al., 2013]. However, as this will affect the performance of the considered protocols in the same way, we do not consider this issue in the chapter.

3.9.2 Deactivation Time

We first investigate how the number of known tags (n) and the number of unknown tags (m) affect the deactivation time (i.e., the execution time of the first phase) in our protocols. Intuitively, the larger n and m are, the longer time needed to deactivate all the known tags in the first phase. Fig. 3.7(a) plots the deactivation time of the three protocols when n changes. We observe that when there are more known tags, the deactivation time in all the three protocols increases. SUIP and MUIP outperform BUIP by using slot pairing and multiple reselections, which greatly increases the probability P_d and consequently decreases the amortized cost to deactivate a known tag. Compared with BUIP, SUIP and MUIP reduce deactivation time by up to 15 percent and 19 percent, respectively.

As we have pointed out in the design of our protocols, replies from unknown tags will disturb the recognition of known tags and thus increase the deactivation time. Fig. 3.7(b) plots the deactivation time in the three protocols when m increases. In all the three protocols, the deactivation time increases when m increases. However, SUIP and MUIP always outperform BUIP due to their ability to deactivate known tags in the original collision/empty slots by slot pairing. Compared with BUIP, the deactivation time in SUIP and MUIP is reduced by 12 percent and 19 percent, respectively.

Fig. 3.7(c) plots the impact of estimation error in the number of unknown tags (m_i) on the deactivation time of MUIP. In this figure, the line marked with MUIP(opt) demonstrates the performance of MUIP when the value of m_i is exactly known, i.e., when the estimation of m_i contains no error. It can be seen that the difference between MUIP and MUIP(opt) is very small. Compared with MUIP(opt), MUIP increases deactivation time by less than 2 percent.

Table 3.2: Execution Time of MUIP and CU When m Changes($n = 10000$).

Alg. Name	Total Execution Time (s)									
	1k	2k	3k	4k	5k	6k	7k	8k	9k	10k
CU(99%)	35.2	41.3	50.4	59.7	68.8	78.3	87.3	96.6	101.0	109.8
CU(95%)	25.7	30.8	39.15	47.5	55.9	64.3	72.8	81.3	89.8	98.3
CU(90%)	19.2	27.1	35.0	42.9	51.1	59.3	67.3	75.2	83.5	**85.9**
MUIP	**18.7**	**26.8**	**34.8**	**42.7**	**50.7**	**58.6**	**66.5**	**74.3**	**82.5**	90.6

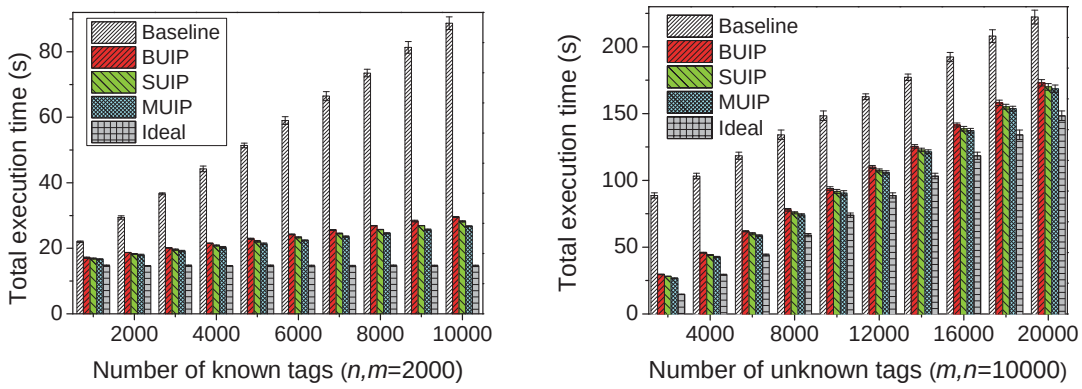
3.9.3 Total Execution Time

Fig. 3.8(a) plots the total execution time of our three protocols and the Baseline and the Ideal method when n changes from 1000 to 10,000, assuming that $m = 2000$.

As n increases, the execution time of our protocols gradually increases, but increases much slower than the Baseline method does. When n increases from 1000 to 10,000, the execution time of the Baseline method increases 66.7s (from 22s to 88.7s). In contrast, the execution time of BUIP, SUIP, MUIP increases only 12.4s, 11.4s, and 10.1s, respectively. Compared with the Baseline method, our best protocol MUIP saves time by up to 70 percent and 55 percent in average.

The gaps between our protocols and the Ideal method increase slightly when n increases. However, when the number of unknown tags is comparable to the number of known tags, the execution time of our best protocol is only slightly longer than the Ideal method. For example, when $n = 1000$, MUIP uses only 2s more time than the Ideal method (16.7s vs. 14.7s), which accounts for only 12 percent of the total execution time of MUIP. When there are much more known tags than unknown tags, our protocols need longer time to deactivate all the known tags. For example, when $n = 10,000$, MUIP uses 26.8s to identify all the unknown tags, while the Ideal method uses only 14.7s. In this case, the deactivation time accounts for 45 percent of the total execution time of MUIP. In cases where there are much less unknown tags than known tags in the system, our best protocol would perform nearly as well as the Ideal method.

Fig. 3.8(b) plots the execution time of different protocols when m increases from 2000 to 20,000, assuming that $n = 10,000$. Compared with the Baseline method, our best protocol MUIP saves 70 percent time when $m = 2000$. Even when m is as large as 20,000, MUIP still saves 24 percent time compared with the Baseline method. Thus our protocols can effectively reduce identification time even when there are two times more unknown tags than known tags in the system.



(a) n (b) m

Fig. 3.8: The total execution time of different protocols: (a) when n changes; (b) when m changes.

3.9.4 Comparison with CU

The CU scheme proposed in [Sheng et al., 2010] can collect a required fraction of unknown tags with probability higher than a specified threshold β. We compare the time needed for CU to collect 90%, 95% and 99% unknown tags (assuming $\beta = 0.95$) with the execution time of MUIP. The parameters in CU are determined by using the methods given in [Sheng et al., 2010]. Table 3.2 lists the results, where CU(90%), CU(95%), and CU(99%) mean the time for CU to collect at least 90%, 95%, and 99% of unknown tags with probability higher than or equal to 0.95, respectively. We can see that the execution time of MUIP to identify all unknown tags is even almost less than the execution time of CU to collect only 90% of unknown tags.

MUIP outperforms CU due to the following reasons. First, MUIP runs much less rounds than CU because it can efficiently deactivate known tags. Recall that in MUIP known tags can be deactivated with a very high probability in each round (up to 0.89 when $h = 7$), while in CU the unknown tags can be collected in each

round with probability only $1 - e^{-1/1.443} = 0.5$. Second, the length of each round in MUIP is significantly shorter than that in CU. In each round of CU, the frame length is $1.443(n + m)$, where n and m are the total number of known tags and the total number of unknown tags, respectively. In contrast, in MUIP the frame length is $n_i + m_i - 1$, where n_i and m_i are the number of known tags and unknown tags in the *current round*, respectively. Third, in MUIP n_i and m_i decreases after every round because some known tags are deactivated and some unknown tags are labeled. In contrast, in CU the number of active known tags are unchanged during the execution because CU does not deactivate known tags.

When m is very large, the execution time of MUIP is dominated by the time spent in the second phase, in which case CU might use shorter time than MUIP does because it will leave a significant number of tags unidentified. For example, as shown in Table 3.2, CU(90%) uses less time than MUIP when $m = 10,000$, in which case it leaves about 1000 unknown tags unidentified. However, even when m is as large as 16,000, MUIP (122.8s) still uses less time than CU(95%)(128.1s) and CU(99%)(149.9s) do.

3.10 Summary

Equally important as the missing tag identification, unknown tag identification deserves more investigation in RFID systems. It is not the reverse way of missing tag identification to completely identify unknown tags. In this chapter, we propose a series of protocols to perform fast and complete unknown tag identification in a large RFID systems. Simulation results show the superior performance of the proposed protocols. While in an ideal unknown identification protocol the execution time should depend

on only the number of unknown tags, in the proposed protocols the execution time is still slightly impacted by known tags to some extent. In the future, we will investigate how to further reduce the impact of known tags on collecting unknown tags.

Chapter 4

Efficient Tag Searching Protocol in Large RFID Systems

The RFID technology are greatly revolutionizing applications such as warehouse management and inventory control in retail industry. Instead of making an inventory of all the tags in the system, some applications need to quickly search a particular set of tags (called *wanted tags*) to confirm which of them are currently present in the system. Since the system may contain an extremely large number of tags, finding the result by collecting all the tag IDs and comparing them with the want tag IDs could be highly time inefficient. In this chapter, we design a novel technique called *testing slot* for a reader to quickly figure out which wanted tags are absent from its interrogation region without tag ID transmissions, which could greatly reduce transmission time during the searching process. Based on this technique, we propose two protocols targeting at time-efficient tag searching in practical large RFID systems containing multiple readers. In our protocols, readers quickly obtain local searching results by employing testing slots to iteratively eliminate wanted tags that are absent from their interrogation region. The local searching results of all the readers are then combined to form the final searching result. Simulation results show that, our best protocol

reduces up to 76 percent time compared with state-of-the-art solutions, and achieves very high precision.

4.1 Overview

Many modern warehouses and supermarkets install RFID systems to efficiently manage their products. Since the interrogation region of one reader is usually very limited, in practice large RFID systems need to deploy multiple readers to cooperatively cover all the tags. These readers identify tags by collecting their IDs, which enables the warehouse to inventory the products automatically.

Instead of making an inventory of all the tags in the system, some applications require searching a particular set of tags (called *wanted tags*) with the given tag IDs to confirm which of them are currently present in the system. Imagine a big warehouse that stores many kinds of products from different manufacturers. Every manufacturer wants to inventory only its own products. In this case, a time-efficient tag searching protocol will be greatly helpful. There are many other important applications of tag searching, such as frequent inventory of a specified kind of products in a shopping mall and finding which products contain flaws in order to recall and fix them.

In this chapter, we study the practically important *tag searching problem*, particularly in large RFID systems that contain multiple readers. We call the wanted tags that exist in the system and should be included in the searching result as *target tags*, and call the wanted tags that do not exist as *nontarget tags*. It may appear that we can collect the IDs of all the tags in the system and find the result by comparing the collected tag IDs with the wanted tag IDs. The tag collection method, however, may incur too long time because it wastes time in collecting IDs of non-wanted tags,

which usually form the majority of tag population in the system. Another intuitive method is to broadcast wanted tag IDs and let tags respond to the reader only when they hear their own IDs. This method's performance heavily depends on the number of wanted tags. As a large RFID system usually deploys multiple readers, all the readers have to broadcast all the wanted tag IDs. When there are many more wanted tags than tags covered by one reader (every reader covers only a small part of tags in the system), although not transmitting any non-wanted tag IDs, this broadcast method may perform even worse than the collection method.

The most related work is the *Compact Approximator based Tag Searching* (CATS) protocol proposed by Zheng and Li [Zheng and Li, 2013b], which adopts Bloom filters to compact the information exchanged between the reader and tags and hence reduces the searching time. CATS consists of two phases. In the first phase, the reader constructs a Bloom filter that represents all the wanted tag IDs and broadcasts the filter to filter part of non-wanted tags in the system. In the second phase, the reader collects replies from tags and constructs the second Bloom filter, and then filters out those wanted tags not belonging to this filter. CATS is highly time-efficient when the number of wanted tags is significantly less than the number of tags in the reader's coverage region. CATS may perform well when the number of wanted tags is far less than the number of tags in one reader's region. When the number of wanted tags is comparable to or even larger than the number of tags in one reader's region, CATS's performance degrades greatly and may even perform worse than the collection method. Furthermore, when there are far more wanted tags than tags in one reader's region, CATS may even do not work. Since there usually are far more wanted tags than tags covered by a single reader, CATS may be inapplicable to solve the tag

searching problem in multiple reader RFID systems.

It is very challenging to quickly search a group of wanted tags in multiple reader systems. In practice the system usually does not record which reader covers exactly which tags, though it may store all the tag IDs in the back-end server. Thus a reader does not know whether there are any wanted tags in its interrogation region and how many there are. Each reader needs to quickly distinguish wanted tags from non-wanted tags. As we have mentioned, collecting all the tag IDs is highly time inefficient. We need to design new approaches for a reader to quickly differentiate between wanted tags and non-wanted tags without transmitting tag IDs.

In this chapter, we design a novel technique called *testing slot* that can quickly test the existence of wanted tags without ID transmission. Based on this technique, we propose a Searching by iterative Testing and Eliminating Protocol (STEP), which can quickly search tags in multiple reader systems with high time efficiency. In STEP, every reader uses the testing slot to quickly eliminate nontarget tags from the wanted tag set and obtains the searching result. Analysis shows that the efficiency of STEP degrades when the ratio of the nontarget tags to the non-wanted tags in the reader's region decreases. To further reduce the searching time, we then propose E-STEP which uses a novel approach to dynamically adjust the ratio of the two types of tags.

The rest of this chapter is organized as follows. In Section 4.2, we give the system model and define the problem. In Section 4.3, we describe STEP and analyze its execution time. In Section 4.4, we propose E-STEP that further improves time efficiency of STEP. Simulation results are reported in Section 4.5. At last, Section 4.7 concludes the chapter.

Table 4.1: Notations used in this chapter.

Notation	Meaning
R/T	Set of readers/tags in the system
$r_i/S(r_i)$	The i-th reader/The local searching result of r_i
X	Set of wanted tags
L_i	The set of local tags of reader r_i
Y_l	The set of candidate target tags in the l-th round
Z_l	The set of nontarget tags in the l-th round
N_l	The number of active local tags in the l-th round
C_l	Average time to eliminate a nontarget tag in round l
N_δ	The threshold of false positive tags
p_{req}	The threshold of precision probability

4.2 Problem Description

In this chapter we describe our system model and define the tag searching problem. Table 4.1 lists the notations used in this chapter.

4.2.1 System Model

We consider a RFID system that consists of three components: a back-end server, a set of readers $R = \{r_1, r_2, \ldots, r_M\}$, and a large number of tags $T = \{t_1, t_2, \ldots, t_N\}$. Here M and N are the total number of readers and the total number of tags in the system, respectively. The server connects the readers via wired or wireless networks, and issues orders to schedule their working. Every reader covers a part of tags that are referred to as the reader's *local tags*. The local tag set of the reader r_i is denoted as L_i, which is a subset of T (i.e., $L_i \subseteq T$). Multiple readers are deployed to cover all the tags in the system, i.e., $T = \bigcup_{i=1}^{M} L_i$.

The system adopts the frame-slotted ALOHA as the basic communication protocol. In this protocol, when receiving a response successfully, the reader replies an ACK to acknowledge that tag and prevents it from attending the following process

until the next protocol execution. Note that, in the rest of the chapter, the local tags of reader r_i denote *the active tags in r_i's region that have not been acknowledged*. The setting of the frame size f is critical for the ALOHA-based protocols. To guarantee the proper ratio of singleton slots and obtain the optimal time efficiency, the frame size f is usually set according to the number of local tags.

4.2.2 Problem Statement

Given a group of wanted tags $X = \{w_1, w_2, \ldots, w_H\}$, the tag searching problem is to *quickly determine which tags in X are currently present in the system, i.e., $X \cap T$*. We call tags in $X \cap T$ as *target tags*, and call tags in $X - T$ as *nontarget tags* and tags in $T - X$ as *non-wanted tags*, respectively. The problem has to be cooperatively solved by multiple readers. The server sends the wanted tag set X to all the readers. Each reader individually finds out target tags in its local region, i.e., $X \cap L_i$. By combining the local results of all the readers, the server can get the final searching result $X \cap T = \bigcup_{i=1}^{M}(X \cap L_i)$. Our goal is to *find the target tags for an individual reader r_i, i.e., $X \cap L_i$, as fast as possible*.

Our protocols are probabilistic, and thus the searching result may contain some false positive tags. The objective is to restrict the number of such false positive tags (i.e., nontarget tags in the searching result) within a small value with high probability. Denote the local searching result of reader r_i as $S(r_i)$, we want to guarantee that

$$Pr\{|S(r_i) - (X \cap L_i)| \leq N_\delta\} \geq p_{req}, \tag{4.1}$$

where N_δ is a small constant and p_{req} is a predefined probability threshold. For example, if $N_\delta = 1$ and $p_{req} = 0.95$, then our protocols can guarantee that the probability that the searching result contains at most one false positive tag is at least

0.95.

We assume that the reader r_i knows the rough number of its local tags (N_i) before executing our protocols. There are many cardinality estimating algorithms that can quickly estimate the number of a reader's local tags [Han et al., 2010, Kodialam and Nandagopal, 2006, Li et al., 2010a, Qian et al., 2008], and the reader can employ these algorithms to estimate N_i before executing the searching task.

We do not pay much attention to multiple reader scheduling in this chapter, which has been studied a lot in recent years [Tang et al., 2009, Zhou et al., 2007]. We mainly focus on designing protocols for a single reader to quickly figure out target tags even when the number of wanted tags is larger than the number of reader's local tags, which is usually the case in practical multiple reader systems. We employ existing reader scheduling algorithms [Tang et al., 2009, Zhou et al., 2007] to find a feasible scheduling of readers when implementing our tag searching protocols.

4.2.3 Slot Timing

Since tags choose slots randomly, a slot may turn out to be empty (no tags chooses it) or non-empty (at least one tags choose it). The duration time of an empty slot (denoted by t_e) is less than the duration of a non-empty slot. The duration of a non-empty slot depends on the type of responses that tags transmit to the reader. A tag can transmit either its ID or a short response (e.g., a 16-bits random number *RN16* in EPC Class 1 Generation 2 (C1G2) standard [EPCglobal, 2008]) to the reader. We denote by t_{id} and t_s the time needed to transmit a tag ID and a *RN16* short response, respectively. The relationship between duration time of different types of slots is $t_e < t_s \ll t_{id}$.

In this chapter, we set the data rate based on the EPC C1G2 specification [EPC-global, 2008], by which the data rate between the reader and tags could be set in a range according to the application environments and physical implementations. The data rate from tag to reader is either 40Kbps to 640Kbps (FM0 encoding format), or 5kbps to 320kbps (Miller modulated subcarrier encoding format). The data rate from reader to tag can be set from 26.7Kbps to 128Kbps. In this chapter, the transmission rate between the reader and tags is set at 62.5Kbps. At this data rate, the duration time for different slots are t_{id}=2.4ms, t_s=0.4ms, and t_e=0.184ms, respectively. Note that our protocol can be implemented with different date rate.

4.3 STEP: Searching by Iterative Testing and Eliminating Protocol

In this section, we first introduce the protocol design guideline, then explain the main idea of our STEP protocol and give detailed description on its implementation. After that, we discuss the optimal frame size setting and analyze how to set the terminal condition. Finally, we discuss some limitations of STEP.

4.3.1 Design Guideline

To reduce the transmission time, our protocol design follows a main guideline: Avoid the ID transmission during the searching process in which a reader communicates with its local tags and confirms the presence of target tags if any. Recall that target tags refer to those wanted tags that should be included in the searching result.

We design a novel technique called *testing slot* for a reader to quickly test the existence of wanted tags. We call a slot testing slot if *at least* one wanted tag maps

to it. In ALOHA-based protocols, with the same hash function $H(ID, s)\ mod\ f$, the reader could compute which slot a tag will select if it knows the tag's ID. As the reader could obtain the IDs of wanted tags in X from the server, it can compute which slots in the frame are testing slots with tag IDs in X. We leverage the testing slots to determine which wanted tags *are not present in the system*: When the reader receives *no response* in a testing slot, it knows that the corresponding wanted tags *must be not present* in its interrogation region. These tags are nontarget tags and should be eliminated from the wanted tag set.

During the testing process, the reader cares only the state of the slot, i.e., empty or non-empty. To do this, we let tags transmit short responses (16 bits), instead of tag IDs (96 bits), to inform the reader of their presence. This can effectively avoid tag ID transmissions and thus consequently improves the time efficiency.

4.3.2 Protocol Overview

To obtain the searching result, every reader executes STEP individually to obtain its local searching result $S(r_i)$. Note that the local searching result may contain false positive tags, i.e., $S(r_i) \supseteq (X \cap L_i)$. The back-end server then combines all the local searching results to obtain the final searching result, i.e., $S = \bigcup_{i=1}^{M} S(r_i)$. Here, we consider a reader r_i without loss of generality.

STEP consists of multiple rounds. In each round, the reader uses testing slots to find out nontarget tags and eliminates them from the wanted tag set. In the l-th round, the reader maintains a candidate target tag set Y_l that contains all the possible target tags in its interrogation region. (Note that $Y_1 = X$ because all the wanted tags are possible target tags in the first round.) The reader computers the testing slots with the tags in Y_l. After checking all the testing slots, the reader obtains the updated

candidate target tag set Y_{l+1}. It then checks whether the *termination condition* is met. If the termination condition is not met, the reader issues a new round to further eliminate nontarget tags from Y_{l+1}. Otherwise, it treats all the tags in Y_{l+1} as the local searching result, i.e., $S(r_i) = Y_{l+1}$, and sends $S(r_i)$ to the back-end server to form the final searching result. We will discuss the termination condition in Section 4.3.5

Non-wanted tags in the reader's local region may interfere the elimination of non-target tags from Y_l because they may also select testing slots. Note that only when a testing slot is empty it can help detect and eliminate nontarget tags. If some non-wanted tags also map to testing slots, their responses may disturb the elimination of nontarget tags from Y_l. To alleviate this problem, we use *non-testing slots* to acknowledge non-wanted tags and mitigate their interference. Non-testing slots are those slots that are not testing slots. Since all the target tags respond in only testing slots, the responses received in non-testing slots must be from non-wanted tags. When the reader receives responses in a non-testing slot, it replies an ACK to acknowledge these non-wanted tags and prevents them from attending the next round, which can decrease the interference from those tags when testing and eliminating nontarget tags from the candidate target tag set in the following rounds.

4.3.3 Protocol Description

STEP consists of multiple rounds. In the l-th round, the reader first computes testing slots and non-testing slots according to the candidate target tag set Y_l, then eliminates nontarget tags from Y_l and acknowledges non-wanted tags in the local region. It repeats the process until the termination condition is met.

At the beginning of the l-th round, the reader broadcasts a query $< f_l, s >$ and

computes testing slots according to current candidate target tag set Y_l. After receiving the query, local tags select their slots with the same hash function used for testing slots computing, and then transmit short responses to the reader in their selected slots. The reader scans the frame and eliminates nontarget tags from Y_l according to the actual responses:

- In a testing slot, if the reader receives some responses, it replies a NAK to keep the tag(s) active. If the reader receives no response, it knows that those wanted tags selecting this slot are not present and must be nontarget tags, it eliminates these wanted tags from Y_l.

- In a non-testing slot, if the reader receives some responses, which must be non-wanted tags, it replies an ACK to acknowledge these tag(s) to prevent them from participating in the following rounds.

At the end of the round, the reader obtains the updated candidate target tag set Y_{l+1} and checks whether it should terminate. If it does not satisfy the termination condition, it will start a new round and repeat the testing and eliminating process.

Fig.4.1 illustrates how STEP works. In this figure, dotted arrows represent the mapping between candidate target tags and testing slots, and solid arrows represent the transmissions between local tags and the reader. The initial candidate target tag set is $Y_1 = \{w_1, w_2, ..., w_6\}$. There are five local tags, among which w_2 and w_4 are target tags. Fig.4.1(a) and Fig.4.1(b) illustrate the first round execution of STEP: The reader first computes testing slots for tags in Y_1 and collects responses from local tags (Fig.4.1(a)), then eliminates nontarget tags from Y_1 and acknowledges non-wanted tags in the local region(Fig.4.1(b)). The reader replies NAK in testing slots

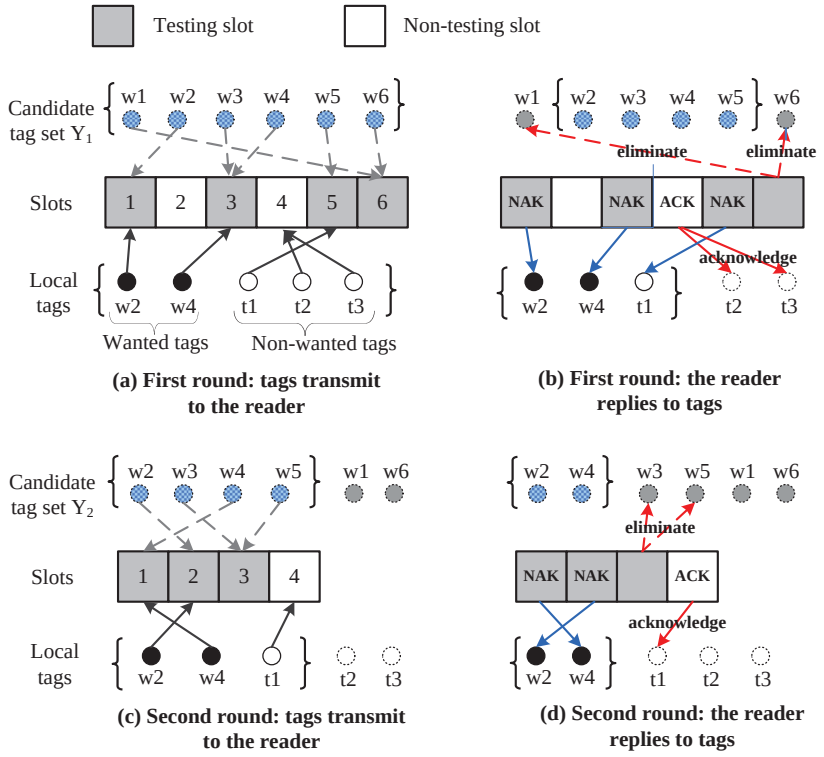


Fig. 4.1: An example of iteratively testing and eliminating nontarget tags from Y_l.

1, 3, and 5 in which it receives responses. The testing slot 6 is empty, so the reader eliminates corresponding nontarget tags w_1 and w_6 from Y_1. Meanwhile, it replies ACK to acknowledge non-wanted tags t_2 and t_3 in slot 4. Fig.4.1(c) and Fig.4.1(d) illustrate the second round execution. The reader re-computes the testing slots with the updated candidate target tag set $Y_2 = \{w_2, w_3, w_4, w_5\}$ and local tags also select new mapping slots. Similar to the first round, the reader eliminates nontarget tags w_3 and w_5 from Y_2 which map to testing slot 3, and acknowledges non-wanted local tag t_1 mapping to the non-testing slot 4. If the protocol terminates after the second round, the local searching result would be $S(r_i)$=$\{w_2, w_4\}$.

4.3.4 Optimal Frame Size Setting

In this section we analyze how to set the frame size to minimize the average time cost C_l to eliminate a nontarget tag from Y_l.

Without loss of generality, we consider the l-th round. Let N_l[1] be the number of active local tags of the reader r_i, and let f_l be the frame size. For any slot in the frame, the probability that it is empty equals the probability that none of local tags select that slot, which is

$$p_e = (1 - \frac{1}{f_l})^{N_l} \approx e^{-N_l/f_l}. \tag{4.2}$$

Obviously, the probability that a slot is non-empty is $(1 - p_e)$. The total duration time of the whole frame is thus

$$\begin{aligned} T_{total} &= f_l * p_e * t_e + f_l * (1 - p_e) * t_s \\ &= f_l * t_s + f_l * e^{-N_l/f_l} * (t_e - t_s). \end{aligned} \tag{4.3}$$

Note that t_e is the duration time of an empty slot, and t_s is the duration time of a short response slot.

Let Z_l be the set of all the nontarget tags in the l-th round, i.e., $Z_l = Y_l - (X \cap L_i)$. Note that $X \cap L_i$ is the target tag set of reader r_i. So the total number of tags that can be eliminated from Y_l is $|Z_l|$. A tag in Z_l will be eliminated if its selected testing slot is actually empty. Because wanted tags select slots uniformly, the expected number of eliminated nontarget tags in this round is approximately

$$N_{eli} \approx |Z_l| * p_e. \tag{4.4}$$

[1]In the first round, $N_1 = N_i$ can be estimated by using estimation algorithms proposed in [Han et al., 2010, Kodialam and Nandagopal, 2006, Li et al., 2010a, Qian et al., 2008]. In the following rounds, as the reader has collected the status of all the slots in the frame, it can estimate the number of acknowledged non-wanted tags as to be discussed in Section 4.3.6 and thus can update N_l accordingly.

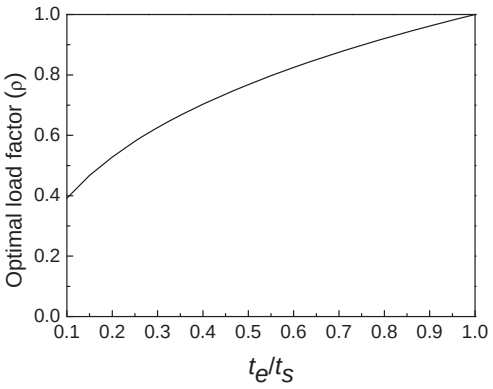


Fig. 4.2: Optimal load factor (ρ) when t_e/t_s varies.

The average time cost to eliminate one nontarget tag from Y_l is

$$C_l = \frac{T_{total}}{N_{eli}} = \frac{t_s}{|Z_l|} * f_l * e^{N_l/f_l} + \frac{1}{|Z_l|} * f_l * (t_e - t_s). \tag{4.5}$$

To minimize C_l, we let

$$\frac{\partial C_l}{\partial f_l} = 0, \tag{4.6}$$

and know that C_l takes minimum value when

$$e^{N_l/f_l}(1 - \frac{N_l}{f_l}) - (1 - \frac{t_e}{t_s}) = 0. \tag{4.7}$$

Define the load factor as $\rho = N_l/f_l$. Then C_l takes the minimum value when the following equation holds:

$$e^{\rho}(1 - \rho) - (1 - \frac{t_e}{t_s}) = 0. \tag{4.8}$$

We can see that the value of the optimal ρ is solely determined by t_e/t_s. In our system model (see Section 4.2.3), $t_e = 0.184ms$ and $t_s = 0.4ms$. In this case, the optimal value of ρ is 0.7432, and the optimal frame size is $f_l = N_l/0.7432 = 1.346 * N_l$.

Since the transmission rate between readers and tags varies in a range, t_e and t_s may vary in different applications. Fig. 4.2 plots the optimal value of ρ when the value of t_e/t_s varies from 0.1 to 1.0.

4.3.5 Termination Condition

We now analyze how to set the termination condition for STEP to achieve the given precision requirement. Note that we have $(X \cap L_i) \subseteq S(r_i)$. The number of false positive tags is $N_\delta = |S(r_i)| - |X \cap L_i|$. Fig 4.3 illustrates the relationship among X, L_i, and $S(r_i)$. Our goal is to guarantee that N_δ does not exceed a threshold with high probability, i.e.,

$$P\{N_\delta \leq \Delta\} \geq p_{req}. \tag{4.9}$$

For example, if $\Delta = 2$ and $p_{req} = 0.95$, satisfying Equation 4.9 means that there are at most two false positive tags in the local searching result with probability higher than or equal to 0.95.

In STEP, we iteratively eliminate nontarget tags from candidate target tag set Y_l round by round. If in a certain round (e.g., the l-th round) no nontarget tag is eliminated from Y_l and no non-wanted tag is acknowledged, we can speculate that the number of false positive tags is small. To prove the speculation and guarantee that the number of false positive tags is within a small value with high probability (i.e., satisfying Equation 4.9), we need to observe k consecutive rounds. If there are no nontarget tags eliminated in all the k rounds, then the reader terminates the protocol. Otherwise, the reader repeats the searching procedure.

We now analyze how to set k to satisfy Equation 4.9 for given Δ and p_{req}. Denote by E_k the event that there are no nontarget tags eliminated in all the k consecutive

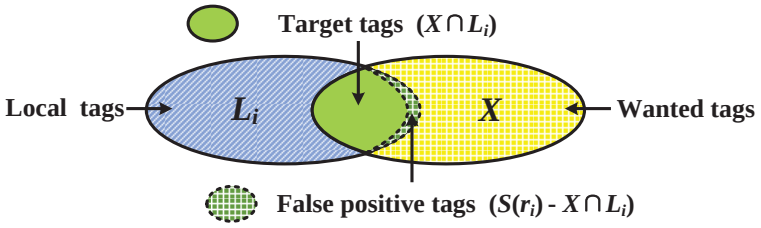


Fig. 4.3: The relationship between X, L_i, and the searching result $S(r_i)$. $S(r_i)$ is the union of target tags $(X \cap L_i)$ and false positive tags $(S(r_i) - X \cap L_i)$.

rounds, and denote by E_v the event that there are exactly v false positive tags in the result. Then we have

$$P\{N_\delta \leq \Delta | E_k\} = \sum_{v=0}^{\Delta} P\{E_v | E_k\}. \tag{4.10}$$

According to Bayes' Theorem, the probability $P\{E_v|E_k\}$ can be calculated as

$$P\{E_v|E_k\} = \frac{P\{E_k|E_v\} * P\{E_v\}}{P\{E_k\}}. \tag{4.11}$$

From the law of total probability it follows that

$$P\{E_k\} = \sum_{v=0}^{\infty} P\{E_k|E_v\}P\{E_v\}. \tag{4.12}$$

Substituting Equation 4.11, 4.12 into Equation 4.10,we have

$$P\{N_\delta \leq \Delta | E_k\} = 1 - \frac{\sum_{v=\Delta+1}^{\infty} P\{E_k|E_v\}P\{E_v\}}{\sum_{v=0}^{\infty} P\{E_k|E_v\}P\{E_v\}}. \tag{4.13}$$

$P\{E_k|E_v\}$ is the probability that there are exactly v false positive tags and none of them are eliminated in all the k consecutive rounds. We know that a nontarget tag cannot be eliminated from the candidate target tag set when it maps to a non-empty testing slot. Thus $P\{E_k|E_v\}$ equals the probability that all the v false positive tags

choose non-empty testing slots in all the k consecutive rounds, which is

$$P\{E_k|E_v\} = ((1-p_e)^v)^k = (1-p_e)^{vk}, \tag{4.14}$$

where $(1-p_e)$ is the probability that a testing slot is non-empty. Since we have no knowledge of the distribution of v, we can assume that v follows the uniform distribution[2], i.e.,

$$P\{E_v\} = \begin{cases} \frac{1}{|S(r_i)|+1}, & v \leq |S(r_i)| \\ 0, & v \geq |S(r_i)|+1 \end{cases}. \tag{4.15}$$

Substituting Equations 4.14 and 4.15 into Equation 4.13, we can derive

$$\begin{aligned} P\{N_\delta \leq \Delta | E_k\} &= 1 - \frac{\sum_{v=\Delta+1}^{|S(r_i)|}(1-p_e)^{vk}}{\sum_{v=0}^{|S(r_i)|}(1-p_e)^{vk}} \\ &\geq 1 - \frac{\sum_{v=\Delta+1}^{\infty}(1-p_e)^{vk}}{\sum_{v=0}^{\infty}(1-p_e)^{vk}} \\ &= 1-(1-p_e)^{(\Delta+1)k}. \end{aligned} \tag{4.16}$$

To satisfy Equation 4.9, we let

$$1-(1-p_e)^{(\Delta+1)k} \geq p_{req}, \tag{4.17}$$

from which we have

$$(\Delta+1)k \geq \log_{1-p_e}(1-p_{req}). \tag{4.18}$$

Then we obtain that k has to satisfy

$$k \geq \frac{\log_{1-p_e}(1-p_{req})}{\Delta+1}. \tag{4.19}$$

As analyzed before, the optimal ρ is 0.7432, with which $p_e = e^{-\rho} \approx 0.4756$. Denote by $\phi = 1-p_e = 0.5244$. In Table 4.2 we list the minimal value of k for different combinations of Δ and p_{req}.

[2]Actually, v has higher probability to be a small value and has lower probability to be a large value. However, the uniform distribution assumption makes the estimation of k tacklable and actually concides well with our simulation results in Section 4.5.

Table 4.2: The minimum k for different combination of Δ and p_{req}.

	$\mathbf{\Delta=0}$	**1**	**2**	**3**	**4**	**5**
$\mathbf{p_{req}=0.95}$	5	3	2	2	1	1
$\mathbf{p_{req}=0.99}$	8	4	3	2	2	2

4.3.6 Discussions

We now discuss how the average time to eliminate a nontarget tag (C_l) varies in different rounds. By substituting the optimal load factor ρ obtained from Equation 4.8 into Equation 4.5, C_l can be expressed as

$$\begin{aligned} C_l &= \frac{t_s}{|Z_l|} * \frac{N_l}{\rho} * e^{\rho} + \frac{1}{|Z_i|} * \frac{N_l}{\rho} * (t_e - t_s) \\ &= \frac{N_l}{|Z_l|}\left(\frac{t_s * e^{\rho}}{\rho} + \frac{t_e - t_s}{\rho}\right). \end{aligned} \tag{4.20}$$

In our system model, $t_e = 0.184ms$, $t_s = 0.4ms$, and $\rho = 0.7432$. Thus we have

$$C_l = \frac{0.841 * N_l}{|Z_l|}. \tag{4.21}$$

Obviously, C_l may vary in different rounds because Z_l and N_l vary round by round. The ratio C_{l-1}/C_l shows the variation trend in two consecutive rounds, which is

$$\frac{C_{l-1}}{C_l} = \frac{N_{l-1}}{N_l} \div \frac{|Z_{l-1}|}{|Z_l|}. \tag{4.22}$$

$|Z_l|$ denotes the number of nontarget tags after some nontarget tags have been eliminated from Z_{l-1} in the (l-1)-th round. Combining Equation 4.2 and Equation 4.4, we have

$$\begin{aligned} |Z_l| &= |Z_{l-1}| - |Z_{l-1}| * p_e \\ &\approx |Z_{l-1}| * (1 - e^{-N_{l-1}/f_{l-1}}). \end{aligned} \tag{4.23}$$

Following a similar process, we can derive the relationship between N_l and N_{l-1}. Different from nontarget tags in Z_{l-1} that will be eliminated in empty slots with probability p_e, the non-wanted tags would be acknowledged in only non-testing slots, i.e., the slots wanted tags do not select. Thus the probability of acknowledging a non-wanted tag is

$$p'_e = (1 - \frac{1}{f_{l-1}})^{|Z_{l-1}|} \approx e^{-|Z_{l-1}|/f_{l-1}}, \tag{4.24}$$

by which we have

$$N_l \approx N_{l-1} * (1 - e^{-|Z_{l-1}|/f_{l-1}}). \tag{4.25}$$

Substituting Equation 4.23 and Equation 4.25 into Equation 4.22, we have

$$\begin{aligned} \frac{C_{l-1}}{C_l} &= \frac{1 - e^{-N_{l-1}/f_{l-1}}}{1 - e^{-|Z_{l-1}|/f_{l-1}}} \\ &= 1 + e^{-|Z_{l-1}|/f_{l-1}} * \frac{1 - e^{(|Z_{l-1}| - N_{l-1})/f_{l-1}}}{1 - e^{-|Z_{l-1}|/f_{l-1}}}. \end{aligned} \tag{4.26}$$

Equation 4.26 shows that $\frac{C_{l-1}}{C_l} < 1$ when $|Z_{l-1}| > N_{l-1}$, which indicates that if the number of nontarget tags in Y_l is larger than the number of non-wanted tags, the time efficiency of STEP decreases. In Fig. 4.4 we plot the value of C_l for the first eight rounds in 500 independent runs of STEP, when wanted tag number $|X| = 5000$, local tag number $|L| = 1000$, and target tag number $|X \cap L| = 200$. Fig. 4.4 shows that the value of C_l gradually increases along with the increase in rounds, which coincides well with our analysis. In the first several rounds, tags in Z_l are eliminated rapidly. Because when the nontarget tags in Z_l is significantly more than local tags, all the slots in the frame might be testing slots (Note that the frame size is set by the number of local tags, i.e., $f_l = N_l/0.7432 = 1.346 * N_l$, which indicates that the nontarget tags is also significantly more than slot number). When receiving responses from local tags that select only part of slots, the reader can quickly decrease $|Z_l|$ by eliminating

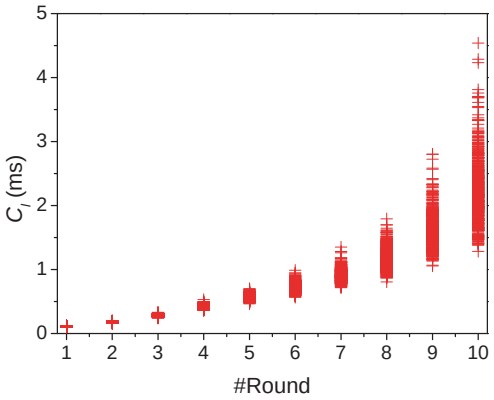


Fig. 4.4: C_l in different rounds.

some nontarget tags in each empty testing slot. On the other hand, it is hard to decrease N_l because non-wanted local tags cannot be acknowledged in testing slots. Since N_l does not decrease a lot, the frame size of the following rounds will be almost the same as in the current round. With the same frame size and much smaller $|Z_l|$, the proportion of empty testing slots will decrease. As a result, it becomes harder to eliminate nontarget tags from Y_l because many testing slots will be selected by local tags. Equation 4.22 indicates that, to keep the tag elimination efficiency high, *we should shrink N_l proportionally when eliminating tags from Z_l.*

We observe that after the sixth round, C_l might be even longer than the time needed to broadcast a tag ID. However, the majority of nontarget tags are eliminated in the first several rounds in which C_l is very small, and this contributes to the high time-efficiency of STEP. When C_l calculated with Equation 4.21 is longer than the time needed to broadcast a tag ID, there should be very few nontarget tags remaining not eliminated. We can further improve the time efficiency by broadcasting remaining candidate target anted tags in Y_l one by one.

4.4 Enhanced Protocol

In this section, we introduce an enhanced tag searching protocol *E-STEP*: Enhanced STEP. This protocol further enhances the time efficiency of STEP by exploiting a novel technique to automatically acknowledge non-wanted tags in order to shrink N_l proportionally when eliminating tags from Z_l.

4.4.1 Design Guideline

A straightforward method to shrink N_l is to increase the proportion of non-testing slots in the l-th round, which requires increasing the frame size f_l. However, if f_l is longer than the optimal frame size, it will also lead to lower time-efficiency.

To decrease N_l, we use indicator vector [Li et al., 2010b] to suppress non-wanted tags and prevent them from participating in the following procedure. In the l-th round, before broadcasting the query $< f_l, s >$ and eliminating nontarget tags from Y_l as in STEP, the reader first broadcasts a query $< b_l, s >$ and a b_l-bit indicator vector. Each bit in the vector corresponds to a slot in the frame at the same index location. If the slot is a testing slot, the bit value is '1'; otherwise, the bit value is '0'. The reader divides the vector into segments of 96 bits (i.e., equal to the length of a tag ID) and transmits each segment in the duration of t_{id}. When receiving the query from the reader, local tags select slots in the frame of b_l slots. Then local tags will check the bit value at the indexes of their selected slots. If the bit value is '0', the tags suppress themselves and will not participant in the following procedure. Note that in this phase *tags actually do not transmit responses to the reader*, they only receive the indicator vector and suppress themselves accordingly. After broadcasting the indicator vector to suppress some non-wanted tags, the reader broadcast a new

query $< f_l, s >$ and scans the responses to eliminate nontarget tags from Y_l and acknowledge non-wanted tags as does in STEP.

The transmission of the indicator vector brings extra time cost. For some rounds, using the indicator vector may not effectively improve the time-efficiency. We have to determine when we should use an indicator vector, and how to set the optimal length of the vector if we should use it.

4.4.2 Protocol Overview

We should determine whether to use an indicator vector or not for each individual round. Consider an arbitrary round l. We analyze whether the vector could decrease the average time to eliminate a nontarget tag in the l-th round. Our solution is to analyze the difference $\Delta C = C_l^1 - C_l^2$, where C_l^1 is the average time to eliminate a nontarget tag when the indicator vector is not used, and C_l^2 is the average time when the indicator vector is used. If $\Delta C > 0$, the reader will broadcast the indicator vector to suppress non-wanted tags.

For the l-th round, there are two cases:

1) **The reader does not broadcast the indicator vector**. In this case, with the optimal frame size $f_l = N_l/\rho$, the average time to eliminate a nontarget tag from Y_l is (Equation 4.20):

$$C_l^1 = \frac{N_l}{|Z_l|}\left(\frac{t_s * e^{\rho}}{\rho} + \frac{t_e - t_s}{\rho}\right). \tag{4.27}$$

2) **The reader broadcasts the indicator vector**. In this case, we derive the average time to eliminate a nontarget tag from Y_l into two parts:

- For the first part, the reader uses a b_l-bits indicator vector to suppress non-wanted tags. The time cost of the first part is $T_{p1} = \lceil \frac{b_l}{96} \rceil * t_{id}$. A local tag

would be suppressed if it maps to a slot whose bit value is '0'. The probability that a local tag maps to a slot whose bit value is '0' equals the probability that no tags in Y_l choose this slot, which is $p_0 = (1 - \frac{1}{b_l})^{|Y_l|} \approx e^{-|Y_l|/b_l}$. Thus the expected number of suppressed tags is approximately $N_l * e^{-|Y_l|/b_l}$.

- For the second part, the reader tests and eliminates nontarget tags from Y_l as in STEP. We can derive that the number of remaining local tags after suppressing tags with indicator vector would be $N_l' \approx N_l * (1 - e^{-|Y_l|/b_l})$. Thus the optimal frame size should be $f_l = N_l'/\rho$. From Equation 4.3, we can derive the time cost of the second part is $T_{p2} = \frac{N_l'}{\rho} * t_s + \frac{N_l'}{\rho} * e^{-\rho} * (t_e - t_s)$.

The total time of the two parts is

$$\begin{aligned} T_{total} &= T_{p1} + T_{p2} \\ &= \lceil \frac{b_l}{96} \rceil * t_{id} + \frac{N_l'}{\rho} * t_s + \frac{N_l'}{\rho} * e^{-\rho} * (t_e - t_s). \end{aligned} \tag{4.28}$$

To calculate C_l^2, we need to know how many nontarget tags would be eliminated from Y_l when using the indicator vector. As we have pointed out in the analysis for STEP (refer to Equation 4.3.4), the expected number of nontarget tags that would be eliminated is

$$N_{eli} \approx |Z_l| * e^{-N_l'/f_l} = |Z_l| * e^{-\rho}, \tag{4.29}$$

where $Z_l = Y_l - (X \cap L_i)$. In the first several rounds, $|Y_l|$ is usually much larger than $|X \cap L_i|$, thus we can assume $|Z_l| \approx |Y_l|$.

With Equation 4.28 and Equation 4.29, we can derive the average time to eliminate a nontarget tag from Y_l in the second case, which is

$$
\begin{aligned}
C_l^2 &= \frac{T_{total}}{N_{eli}} \\
&= \frac{\lceil \frac{b_l}{96} \rceil t_{id} + \frac{N_l'}{\rho} t_s + \frac{N_l'}{\rho} e^{-\rho} * (t_e - t_s)}{|Z_l| e^{-\rho}} \\
&= \frac{1}{|Z_l|} \left(e^{\rho} \lceil \frac{b_l}{96} \rceil t_{id} + \frac{N_l'}{\rho} t_s e^{\rho} + \frac{N_l'}{\rho} (t_e - t_s) \right) \\
&\approx \frac{1}{|Y_l|} \left(e^{\rho} \lceil \frac{b_l}{96} \rceil t_{id} + (\frac{e^{\rho} t_s}{\rho} + \frac{t_e - t_s}{\rho})(1 - e^{-|Y_l|/b_l}) N_l \right).
\end{aligned}
\tag{4.30}
$$

To compare the time-efficiency in the two cases, we calculate their difference $\Delta C = C_l^1 - C_l^2$, which is

$$
\Delta C = \frac{1}{|Y_l|} \left[(\frac{t_s e^{\rho}}{\rho} + \frac{t_e - t_s}{\rho}) N_l e^{-|Y_l|/b_l} - e^{\rho} \lceil \frac{b_l}{96} \rceil t_{id} \right]. \tag{4.31}
$$

In case of the EPC C1G2 tag specification [EPCglobal, 2008], we have $t_{id} = 2.4ms$, $t_e = 0.184ms$, and $t_s = 0.4ms$. Meanwhile, as have been derived in the analysis for STEP, the optimal load factor under this timing scheme is $\rho = 0.7432$. Thus we have

$$
\Delta C = \frac{1}{|Y_l|} \left[0.841 * N_l * e^{-|Y_l|/b_l} - 5.0464 * \lceil \frac{b_l}{96} \rceil \right]. \tag{4.32}
$$

The value of ΔC represents the expected time reduction when we use an indicator vector. The larger ΔC is, the more time reduced. In l-th round, with the given number N_l and $|Y_l|$, the value of ΔC depends only on the length of the indicator vector b_l. Let b_l^* denote the optimal b_l that maximizes the value of ΔC, i.e.,

$$
b_l^* = \arg_{b_l} \max \Delta C, \tag{4.33}
$$

and let ΔC_{max} denote the value of ΔC when $b_l = b_l^*$. If ΔC_{max} is still smaller than 0, which means that using an indicator vector cannot improve the time efficiency, we

should not use the indicator vector in the l-th round. Otherwise, we should use an indicator vector of length b_l^* to further improve time efficiency.

4.4.3 Protocol Description

E-STEP also consists of multiple rounds. Every round is divided into two phases: the suppressing phase and the eliminating phase. Consider the l-th round without loss of generality.

In the suppressing phase, with the number of candidate target tags $|Y_l|$ and the number of local tags N_l, the reader first calculates the value of b_l^* and ΔC_{max}.

- If $\Delta C_{max} \leq 0$, the reader does noting, and enters the second phase directly.
- If $\Delta C_{max} > 0$, the reader broadcasts a query $<b_l^*, s>$, and broadcasts a b_l^*-bits indicator vector. The reader constructs the vector by mapping candidate target tags in Y_l to a frame containing b_l^* slots: If a slot is a testing slot, i.e., some tags in Y_l select it, the reader will set the bit at the same index location to '1'; otherwise, it sets the bit to '0'.

Tags do not respond to the reader in this phase. When a local tag receives the query and the vector, it first checks the bit in the indicator vector at the same index location of its selected slot. If the bit is '0' (i.e., the tag maps to a non-testing slot), the tag will be suppressed and not participant in the following procedure. If the bit is '1', the tag will continue to receive the query from the reader in the second phase.

In the eliminating phase, the reader issues a new query $<f_l, s>$ and receives responses from local tags. Note that f_l is calculated according to number of remaining local tags as discussed in the Section 4.4.2. With these responses, the reader eliminates nontarget tags from Y_l and acknowledges non-wanted tags as in STEP.

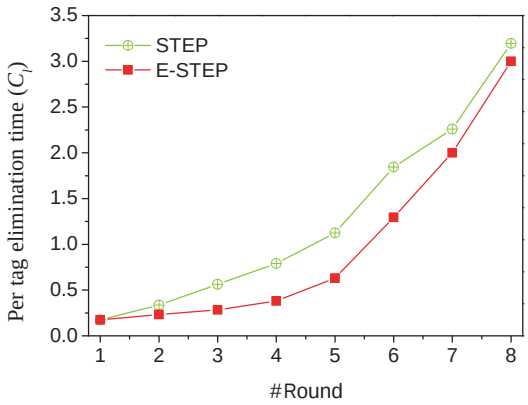


Fig. 4.5: Comparison of C_l in STEP and E-STEP in one example run.

At the end of this round, the reader checks whether it should terminate or not. It starts a new round if the termination condition is not met. The termination condition of E-STEP is the same as of STEP.

The indicator vector can effectively reduce per tag elimination time in the first several rounds. Fig. 4.5 plots C_l (the average time cost to eliminate a nontarget tag) in STEP and E-STEP in the first eight rounds in one random chosen example, when wanted tag number $|X|$=5000, local tag number $|L|$=1000, and target tag number $|X \cap L|$=200. We observe significant reduction of C_l in E-STEP compared with STEP, especially in the first several rounds. For example, in rounds 2 to 4, E-STEP reduces C_l by almost a half. As most nontarget tags are eliminated in the first several rounds, E-STEP effectively reduces the total execution time.

4.4.4 Fault Tolerance

The indicator vector transmitted from the reader to tags might be corrupted due to channel errors. In this case, if a tag receives wrong bit, it may take incorrect action.

Recall that when a tag finds that its corresponding bit is '0', it will be suppressed and will not participate in the following rounds. If the actually sent bit is '1' but the received bit is '0', the tag will exit the protocol incorrectly. If this tag is actually a target tag, then it is possible that this tag is incorrectly eliminated from Y_l and thus will not be included in the searching result, which will affect the correctness of E-STEP.

To fix this problem, we can add a cyclic-redundancy check (CRC) code in each segment when transmitting the indicator vector. The tag can thus check whether the received information is correct or not. If the corresponding bit is correctly received, the tag takes its action accordingly as described in our protocol. Otherwise, it will reply to the reader in its selected slot and ignore the corresponding bit in the vector. For example, if the tag finds that the segment containing its bit is ruined, it will reply to the reader in the selected slot, even though the corresponding bit is '0'. With this mechanism, the correctness of E-STEP can be guaranteed in the meaning that there will be no false negative results.

The time efficiency of E-STEP would degrade slightly when we use this scheme. The EPC C1G2 specification [EPCglobal, 2008] provides two types of CRC code: CRC16 which uses 16 bits and CRC5 which uses 5 bits. If we adopt CRC16, we need to divide the indicator vector into segments of 80 bits long. If we use CRC5, we need to divide the indicator vector into segments of 91 bits long. As the time used to broadcast the indicator vector takes only a small part of the total time in each round, the execution time of E-STEP will increase only slightly.

4.5 Performance Evaluation

We developed a simulator with JAVA and implemented five protocols to compare their performance: STEP, E-STEP, CATS [Zheng and Li, 2013b], ITSP [Chen et al., 2013], and the baseline Collection solution. Two metrics are used to compare the performance of different protocols: (a) The *execution time* measured in seconds, and (b) the number of false positive tags in the searching result (N_δ). We consider three parameters that may affect the performance of different protocols: (a) The total number of wanted tags ($|X|$), (b) the ratio of target tags to the wanted tags, which is defined as $\eta = |X \bigcap T|/|X|$, and (c) the number of readers in the system (M). For STEP and E-STEP, we also investigate the impact of Δ, i.e., the threshold of false positive tags, on their performances. For each parameter setting, we run the considered protocols 100 times and report the average result.

4.5.1 Simulation Scenarios and Time Setting

Simulation scenarios: We consider two different simulation scenarios. The first scenario is a single reader RFID system, in which 2000 tags are deployed. This scenario is used to investigate how different parameters affect the performance of STEP and E-STEP, meanwhile compare their execution time and precision with CATS and the state-of-the-art ITSP protocol. The second scenario considers multiple reader RFID systems. In default, we deploy 64 readers and 50000 tags in the system. The readers are deployed in a grid pattern with distance between adjacent readers set at $\sqrt{2}r$, where r is the interrogation radius of a reader. This results in about 1500 local tags for each reader. The default number of wanted tags ($|X|$) is set at 10000, and the default ratio of target tags (η) is set at 0.2. For multiple reader scenarios, we use

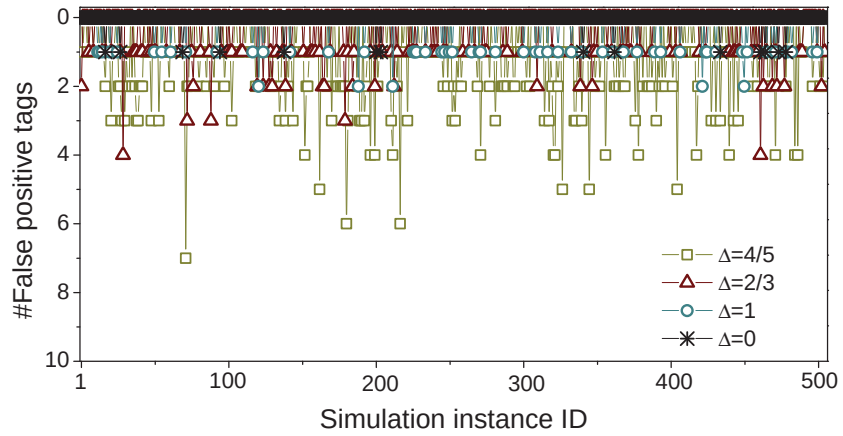


Fig. 4.6: N_δ of E-STEP in 500 runs when Δ changes.

the graph coloring algorithm developed in [Brélaz, 1979] to find a feasible scheduling of readers.

Time setting: We set the time duration of different slots according to the time specification of the EPC C1G2 UHF RFID tags [EPCglobal, 2008]. The data rate between the reader and tags is set at 62.5Kbps. Under this data rate, the time of different slots are as follows: $t_{id} = 2.42$ms, $t_e = 0.184$ms, and $t_s = 0.2$ms. We note that when the data rate changes, the absolute metric data might be different, but similar conclusions could be reached.

4.5.2 Single Reader Scenario

Impact of False Positive Tag Number Threshold

The false positive tag number threshold Δ affects the performance of STEP and E-STEP. A smaller Δ results less false positive tags in the searching result, but it also requires the reader to observe more rounds before it terminates and consequently increases execution time. Fig. 4.6 plots the number of false positive tags of E-STEP

in 500 runs when Δ varies from 0 to 5. The results validate the speculation that larger Δ results in more false positive tags. Meanwhile, the proposed protocol guarantee the searching precision well. The ratios of runs in which $N_\delta > \Delta$ to the total runs are 0.972, 0.99, 0.99, 0.996, 0.986, and 0.994 when Δ changes from 0 to 5, respectively, all larger than the required probability 0.95.

Fig. 4.7 plots the average execution time of E-STEP when Δ changes. The execution time decreases when Δ increases, because more false positive tags could be tolerated when Δ is larger. When Δ increases from 0 to 5, the execution time decrease from 1.2s to 0.9s, approximately 25 percent reduction. The execution time of STEP shows the similar trend. Thus Δ could be used to make tradeoff between time efficiency and searching precision: If the application prefers high time efficiency and can tolerate false positive tags, it can set Δ to a relatively large value. In contrast, if the application requires very precise searching result, it should use a small Δ. In the following experiments, we set $\Delta = 0$ and p_{req}=0.95, i.e., there are no false positive tags in the searching result of STEP and E-STEP with a probability no smaller than 0.95.

E-STEP vs. STEP

Fig. 4.8(a) and Fig. 4.8(b) plot the execution time of STEP and E-STEP for different η when there are 200 and 2000 wanted tags, respectively. Two conclusions can be drawn. First, E-STEP significantly outperforms STEP when $|X| \ll |T|$. When X is small, most of the reader's local tags are nontarget tags, which can be quickly filtered out by the bit vector used in E-STEP. Compared with STEP, E-STEP reduces execution time by 81% when η=0.1 and 60% when η=0.9, respectively. Second, the improvement of E-STEP over STEP becomes less significant when either η or the

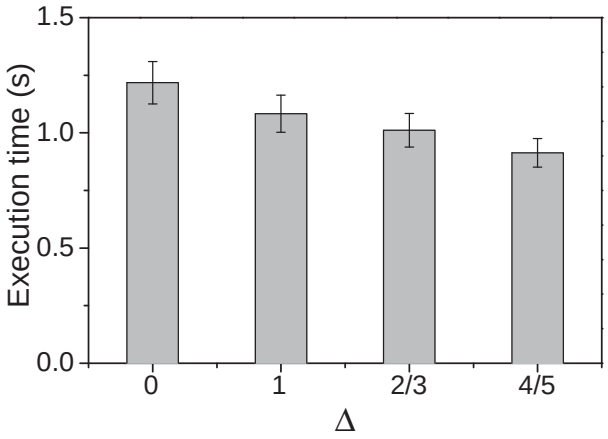


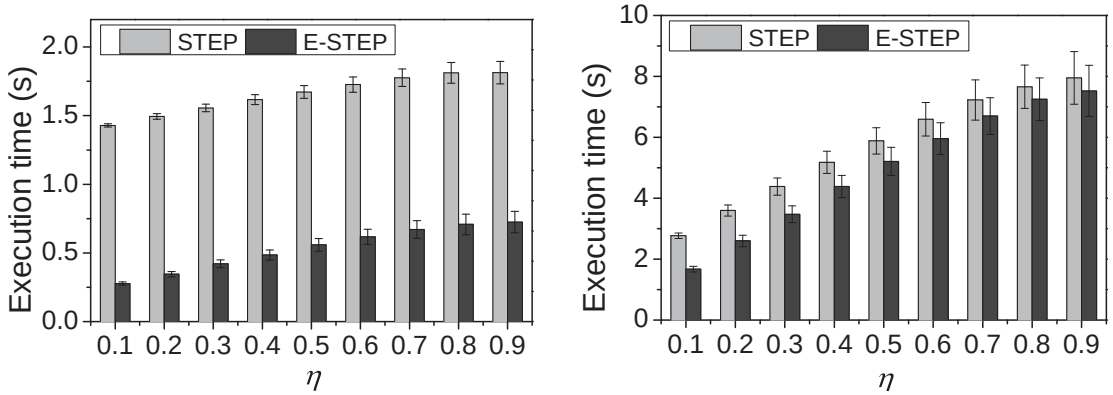
Fig. 4.7: Execution time of E-STEP when Δ changes.

wanted tag number ($|X|$) is large. When η or $|X|$ is large, most of the reader's local tags are target tags and cannot be filtered out by the bit vector. When $|X|$=2000, E-STEP reduces execution time by 40% and 6% when η=0.1 and η=0.9, respectively.

Comparison with CATS and ITSP

Fig. 4.9(a) plots the execution time of the three protocols in 500 runs when $|X| =$ 200, 2000, and 10000, respectively. For ITSP and CATS, we set the false positive tag ratio, defined as $p_f = \frac{|S-X \bigcap T|}{|X-T|}$, as 10^{-3}. All the three protocols perform well when $|X|$ is small, due to the high efficiency of the bit vector in filtering out nontarget tags. When $|X|$=200, the average execution time of E-STEP, ITSP, and CATS are 0.213s, 0.305s and 0.295s, respectively. Compared with CATS and ITSP, E-STEP reduces searching time by 30% and 28%, respectively. When $|X|$ is comparable to $|T|$, e.g., $|X| = 2000$, E-STEP performs much better than CATS and ITSP, using 47% and 46% less time, respectively.

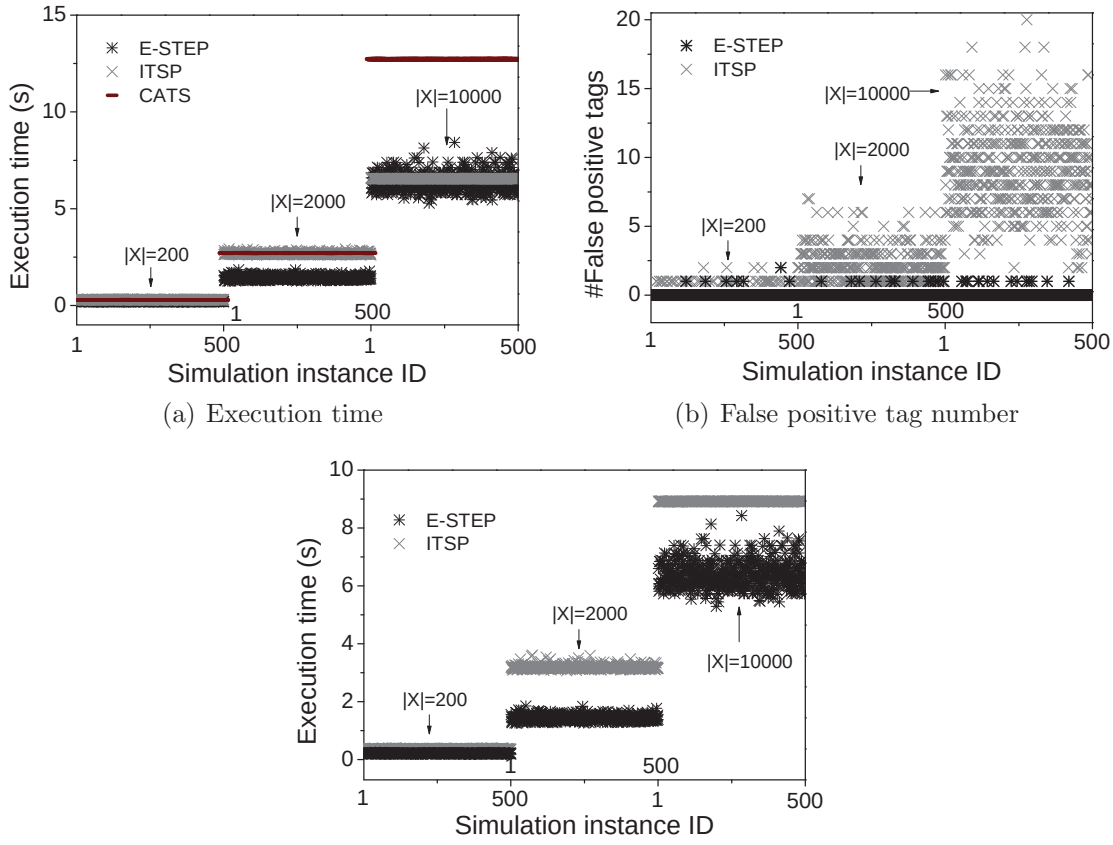
Both ITSP and E-STEP perform much better than CATS when $|X|$ is much larger



(a) $|X| = 200$ (b) $|X|$=2000

Fig. 4.8: Execution time of STEP and E-STEP when η changes ($|T|$=2000): (a) Small $|X|$; (b) large $|X|$.

than $|T|$. For example, when $|X| = 10000$, the average execution time of E-STEP, ITSP, and CATS are 0.639s, 0.656s, and 1.273s, respectively. E-STEP and ITSP use only about half of searching time of CATS. However, ITSP generates much more false positive tags than E-STEP does when $|X|$ is large. As shown in Fig. 4.9(b), there could be up to 20 false positive tags in ITSP. In contrast, the number of false positive tags of E-STEP is independent to $|X|$ and always smaller than 2. In fact, in most runs (1470 out of 1500) there are no false positive tags in E-STEP. Fig. 4.9(c) plots the execution time of E-STEP and ITSP when p_f is set at $= 10^{-4}$ in ITSP, in which case ITSP can achieve similar searching precision as E-STEP does. In this case, E-STEP obviously outperforms ITSP and reduces execution time by more than 28%.



(a) Execution time

(b) False positive tag number

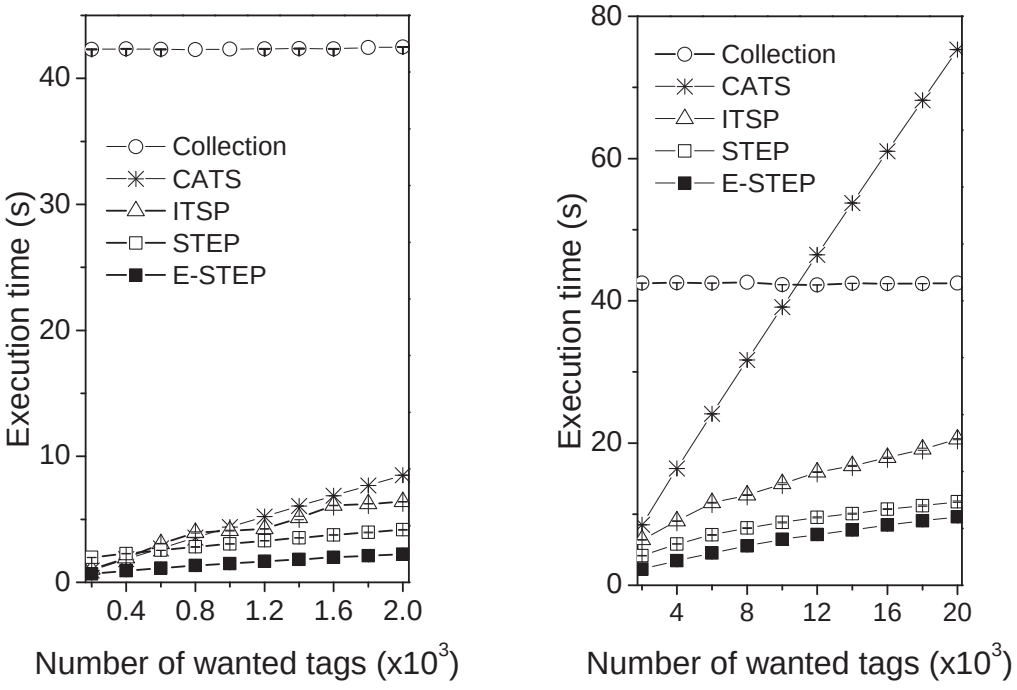
(c) Execution time

Fig. 4.9: Comparison between E-STEP, ITSP, and CATS: (a) Execution time, (b) false positive tag number, (c) execution when $p_f = 10^{-4}$ in ITSP.

4.5.3 Multiple Reader Scenario

Impact of Wanted Tag Number

Fig. 4.10 plots the execution time of different protocols when the wanted tag number increases. We have two observations. First, in all the considered protocols (except Collection), the execution time increases when there are more wanted tags. CATS's execution time increases much faster than the other three protocols. It performs even



(a) Small $|X|$ (b) Large $|X|$

Fig. 4.10: Execution time of different protocols when the number of wanted tags changes: (a) Small $|X|$; (b) large $|X|$. 64 readers, $|T|$=50000, η=0.2.

better than STEP and ITSP when $|X| \leq 400$, but performs even worse than Collection when $|X| \geq 10000$. Compared with CATS, STEP and E-STEP reduce execution time by up to 84% and 87%, respectively. Second, there is a cross point between STEP's and ITSP's execution time: ITSP uses less time than STEP when $|X| \leq 400$, but uses more time than STEP when $|X|$ is large. E-STEP always outperforms STEP and ITSP. Compared with ITSP and STEP, E-STEP reduces execution time by 60% and 52% in average, respectively.

Fig. 4.11 plots the number of false positive tags (N_δ) in E-STEP and ITSP [3]

[3]STEP and CATS have the similar false positive tags as E-STEP and ITSP, respectively.

when $|X|$ increases from 2000 to 20000. In ITSP the false positive tags increase proportionally to the wanted tags, while in E-STEP the number of false positive tags is not affected by the wanted tags and is always small. Recall that E-STEP can guarantee $P\{N_\delta < \Delta\} \geq p_{req}$. In our simulation setting, the theoretical expected number of false positive tags in E-STEP is approximately $\Delta * (1 - p_{req}) * M = 1 * (1 - 0.95) * 64$=3.2, which well coincides with the simulation results. In contrast, N_δ increases from 10 to 60 in ITSP. Define the searching precision as the ratio of N_δ to $|X \bigcap T|$. E-STEP promotes the searching precision by nearly an order of magnitude with only half searching time.

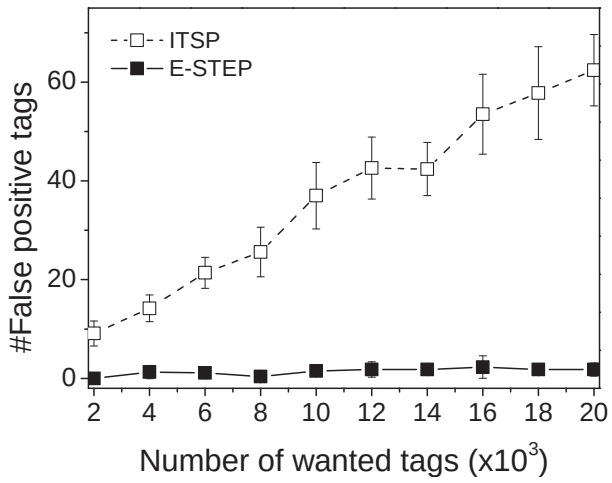
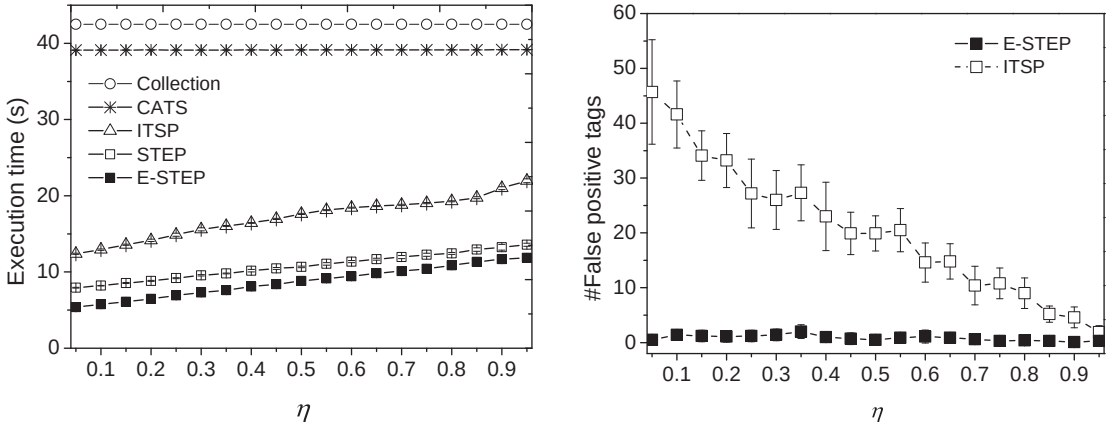


Fig. 4.11: Number of false positive tags in E-STEP and ITSP when $|X|$ increases from 2000 to 20000. 64 readers, $|T|$=50000, η=0.2.

Impact of Target Tag Ratio

Fig. 4.12(a) plots the execution time of different protocols when η varies from 0.05 to 0.95. While the execution time of STEP, E-STEP, ITSP all increases when η becomes large, the execution time of CATS is not affected by η. This is because



(a) Execution time (b) N_δ

Fig. 4.12: Execution time and N_δ in different protocols when η changes: (a) Execution time; (b) false positive tag number. 64 readers, $|T|$=50000, $|X|$=10000.

in CATS the size of the Bloom filter is set according to $|X|$, which is independent to η. In contrast, when η increases, there would be more target tags in each reader's region, in which case our protocols and ITSP need to run more rounds to meet the precision requirement. However, STEP and E-STEP always outperform CATS and ITSP. Compared with CATS and ITSP, E-STEP reduces execution time by up to 86% and 57%, respectively.

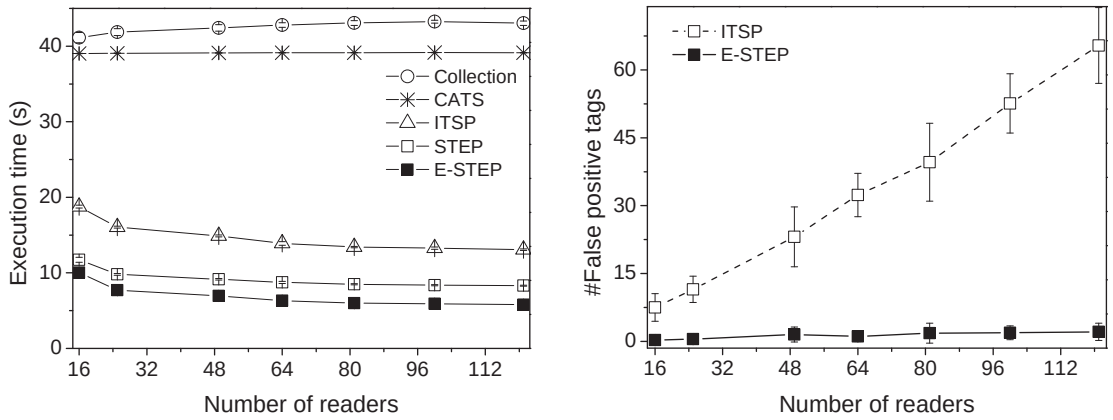
Fig. 4.12(b) plots the number of false positive tags in ITSP and E-STEP when η increases. Along with the increase of η, the number of nontarget tags (i.e., $X - T$) becomes smaller, and thus the number of false positive tags in ITSP decreases, as validated by Fig. 4.12(b). In contrast, the false positive tags in E-STEP are not affected by η and are always less than that in ITSP.

Impact of Reader Number

We change the number of readers in the system by varying the size of the deployment region to investigate how the system scale impacts different protocols' performance. We increase the number of readers ($|M|$) from 16 to 121 and keeps the tag density and distance between adjacent readers unchanged. This means that $|T|$ increases from 12500 (when M=16) to 100000 (when M=121).

Fig. 4.13(a) and Fig. 4.13(b) plot the execution time and the number of false positive tags in different protocols, respectively. Contrary to the intuition that the execution time will increase when the system scales up, it is shown in Fig. 4.13(a) that the execution times in both our protocols and ITSP decrease. The reason is as follows. Because the number of target tags is fixed, the target tags in each reader's interrogation region become less when M increases. This is equivalent to decreasing η, in which case nontarget tags will be filtered out more efficiently. In contrast, the execution time of CATS is not affected by η and remains unchanged. Moreover, we should point out that although the number of readers increases, the total rounds to schedule all the readers to work does not change. Actually, in all the cases the readers are scheduled in 4 rounds. In average, compared with CATS and ITSP, E-STEP reduces execution time by 82 percent and 53 percent, respectively.

The numbers of false positive tags in E-STEP and ITSP when M increases are plotted in Fig. 4.13(b). In both our protocol and ITSP, N_δ increases when M becomes larger. However, our protocol achieves much higher searching precision than ITSP does. The false positive tag number in the searching result of E-STEP is smaller than 5 even when there are more than 100 readers in the system, while the false positive tag number of ITSP is larger than 50 in the same case.



(a) Execution time (b) N_δ

Fig. 4.13: Execution time and N_δ in different protocols when M increases: (a) Execution time, (b) false positive tag number. $|T|$ increases from 12500 (M=16) to 100000 (M=121), $|X|$=10,000, η=0.2.

4.6 Discussions on Implementation Issues

The STEP protocol could be implemented on current EPC C1G2 EPCglobal [2008] tags and readers after some slight modifications are made on them. First, the reader needs to know the hash function that tags use to select transmission slots in order to calculate the testing slots. To achieve this goal, we can let the readers and tags use the same hash function to select transmission slots. Second, STEP requires tags to transmit short responses rather than their IDs when receiving the searching command. This can be accomplished by adding a new state in the tag-side software implementation and triggering the short-response mode in this state. Third, the reader needs to detect the status of each slot and send different acknowledgement (ACK or NAK) to tags accordingly. All these modifications are in the software level and could be accomplished by adding some new states and corresponding transition

conditions in the software implementation.

E-STEP uses the indicator vector technique that could be implemented on EPC G1G2 compliant tags by using the method given in Chen et al. [2011]. To broadcast an indicator vector, the reader first divides the vector into segments of 96 bits and encapsulates each segment into a tag ID. The segments are broadcasted one after another. A tag buffers the segment in which its corresponding bit resides in and acts correspondingly. To further enhance the robustness of the indicator vector, cyclic-redundancy check (CRC) code could be added into each segment to help tags check whether the segment is correctly received.

4.7 Summary

Searching a particular set of tags in a multi-reader RFID system is very important to many RFID applications but has not been thoroughly solved yet. In this chapter, we propose two highly time-efficient tag searching protocols, STEP and E-STEP, which are applicable to a broad application scenarios. Our protocols iteratively eliminate nontarget tags round by round to obtain the searching result, and minimize the average time to eliminate nontarget tags in every round to achieve high time-efficiency. Simulation results exhibit the superior performance of our protocols. Compared with the best existing solutions, our best protocol reduces searching time by at most 76%. Furthermore, our solution can guarantee that the number of false positive tags is bounded by a constant threshold.

We observe that there is no solution that always outperforms other solutions in all scenarios. In the future, we will investigate how to adaptively combine different solutions according to differen scenarios to further reduce the searching time.

Chapter 5

Efficient Tag Stocktaking in Highly Dynamic RFID Systems

An RFID system can greatly improve the efficiency of tagged object inventory setup and update. It is necessary to periodically take stock of tags and update the inventory accordingly (i.e., deleting absent tags and adding new tags) in dynamic scenarios such as warehouses and shopping malls. Fast tag stocktaking is critical for the dynamic RFID system management. Previous work can take stock of tags by either collecting IDs of all tags in the system, which is known to be inefficient, or broadcasting a long indicator vector to save tag identification time, which is not compatible with current commercial-off-the-shelf (COTS) tags. Although unknown tag identification protocols can quickly find out the unknown tags in the system, they will fail if any missing tags in the system as well. In this chapter, we propose HARN, a protocol that can quickly take stock of tags in dynamic RFID systems but use only one more hash in the standard EPC C1G2 protocol. HARN leverages a new hash to generate a 16-bit field to replace the original 16-bit random number. Such replacement enables the transmitted 16-bit string to contain more information to identify present tags, which can save the tedious ID transmission from known tags to

readers and greatly improve the inventory process. HARN is compatible with COTS RFID tags and can be easily applied in a real RFID system. To further improve the performance of HARN, we also devise an analog network coding (ANC) based approach to extract useful information from the collided signal when multiple tags transmit simultaneously. Simulation results demonstrate up to 3.8x (when ANC is not used) and 18x (when ANC is used) boosts in stocktaking throughput compared to the state-of-the-art solutions in dynamic RFID systems.

5.1 Overview

Warehouses or shopping malls have to periodically take stock of the products and accordingly update the inventories stored in the databases (delete the absent products and add the new products). The RFID technology enables the system to quickly update the inventory by taking stock of all tags that are attached to products. Obviously, fast tag stocktaking is practically important to efficient warehouse management.

A common approach for RFID system management is to take stock of tags by collecting all tag IDs (i.e. tag identification). Tag identification refers to collecting IDs of all the tags in the reader's interrogation region, which is the most fundamental issue in RFID systems. Existing protocols mainly focus on identifying tag IDs with static methods [Kang et al., 2012, Namboodiri and Gao, 2010, Shahzad and Liu, 2013, Zhang et al., 2010, Zhen et al., 2005], namely, they always directly collect all tag IDs even when a large fraction of tags have been previously identified. These protocols fall into two different categories: Tree-based protocols [Shahzad and Liu, 2013] and ALOHA-based protocols [Namboodiri and Gao, 2010, Zhen et al., 2005]. Previous

researches either focus on tuning parameters in ALOHA-based protocols to enhance the time/energy efficiency, or designing smart tree traversal schemes to reduce time wasted in visiting collision slots that could not be used to collect tag IDs.

However, tag identification protocols are highly inefficient solutions for tag stocktaking, because they have to unnecessarily re-collect a large number (maybe, unfortunately, the most) of tags that have already identified before. For example, assume that there are 1000 tags in the reader's interrogation region and only 200 of them are *unknown tags*, i.e., tags that entered the system after the last identification operation and have not been identified yet. Only these 200 unknown tags need to be identified. The other 800 tags whose IDs have already been collected, referred to as *known tags* hereinafter, need not to be identified again. Identification protocols designed for static RFID systems cannot recognize which tags are unknown and which are known, and thus have to identify all the 1000 tags from scratch, resulting in low time efficiency and consequently low stocktaking throughput with respect to unknown tags. It seems that the reader could quickly identify all the unknown tags with unknown tag identification protocols. However, these protocols cannot tolerate any missing tags in the system, which will cause the failure of finding unknown tags. Furthermore, these protocols cannot identify missing tags, which is also important when taking stock of tags. Hence, an efficient tag stocktaking protocol should quickly identify both unknown tags and missing tags in the system.

An important issue has to be considered is that the designed protocol should be *compatible with current commercial-off-the-shelf (COTS) tags*,e.g., EPC Class-1 Generation-2 tags. There are already a few solutions on (unknown) tag identification in dynamic RFID systems [Liu et al., 2014a,b, 2012b, 2014c, Sheng et al., 2010].

They are not compatible with current COTS tag standards. They commonly require the reader to broadcast a long *indicator vector* [Chen et al., 2011] to tags to suppress responses from known tags. This will inevitably cause three problems. First, the indicator vector technique is not supported by current COTS tags. Second, it requires high computation ability at the tag side. Third, such indicator vector-based approaches usually suffer from the *hidden tag* problem, i.e., a unknown tag is hidden by some known tags and cannot be successfully recognized and identified.

In this chapter, we present the design and evaluation of a novel COTS-tag-compatible stocktaking protocol HARN that achieves superior performance in dynamic RFID systems. In HARN, by making the most of a designed *RN16* (i.e., a bit-string used only to resolve contention between tags in the standard EPC C1G2 protocol), the reader could quickly distinguish unknown tags from known tags, which can save the tedious ID transmission from known tags to readers and greatly improve the stocktaking process. In summary, we make the following major contributions:

- We propose HARN, a novel tag stocktaking protocol with *HA*sh-based *RN*16 generation that exploits previously collected tag information to distinguish known tags from unknown tags. HARN is compatible with the standard EPC C1G2 protocol and can greatly boost identification throughput of unknown tags in dynamic RFID systems.

- We develop an analog network coding (ANC) based approach to extract useful information from the collided signal when multiple tags transmit simultaneously, and integrate it with HARN to further promote stocktaking throughput of HARN.

- We conduct extensive simulations to evaluate the performance of HARN. Experimental results demonstrate up to 3.8x (when ANC is not used) and 18x (when ANC is used) boosts in stocktaking throughput with respect to unknown tags when compared with state-of-the-art solutions.

The rest of the chapter is organized as follows. Section 5.2 introduces background knowledge of EPC C1G2 protocol. Section 5.3 presents the design and analysis of HARN. The ANC based enhancement and its integration with HARN is described in Section 5.4. Evaluation results based on simulation experiments are reported in Section 5.5. Finally, Section 5.6 close this chapter with summary remarks.

5.2 Background

5.2.1 Revisiting EPC C1G2 Protocol

To achieve better time efficiency, we slightly modify the EPC C1G2 protocol and make the most of RN16 for fast tag stocktaking. In the standard EPC protocol (Fig. 5.1), the only purpose of RN16 is to contend for channel access. The RN16 field is blindly generated and it is independent of the tag's ID. Even when the reader knows IDs of known tags, it has no way to judge whether a received RN16 is from a known tag or a unknown tag without collecting its ID. In HARN, we design a hash-based approach to RN16 generation that bridges a tag's ID and its RN16 field, which helps distinguish whether a received RN16 is sent by a unknown tag or a known tag, and then save the tedious ID transmission from known tags to readers and greatly improve the inventory process.

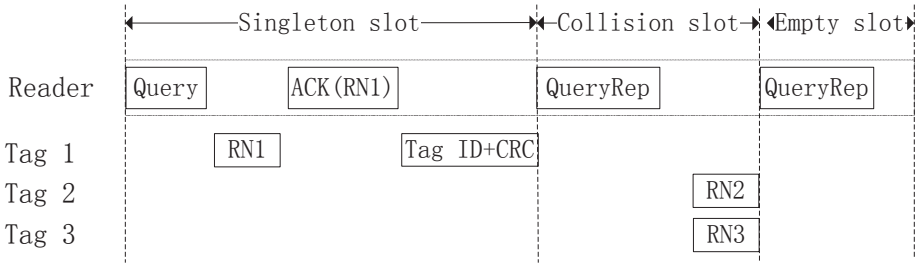


Fig. 5.1: Illustration of the standard EPC C1G2 identification protocol.

5.2.2 Extracting Useful Information From Collided Signals

In the standard EPC C1G2 protocol, only singleton slots can be used to collect tag IDs, which limits its identification throughput. In the past several years, some signal processing techniques are developed to extract useful information from collided signals, e.g., analog network coding (ANC) [Gollakota and Katabi, 2008, Katti et al., 2007, Zhang et al., 2010] and cross correlation-based approaches [Liu et al., 2012a]. Assume that the receiver receives a collided signal mixed by k signals from different sources. If the receiver knows arbitrary $k-1$ out of the k signals contributing to the collided signal, it could decode the remaining signal by using ANC. In contrast, if the reader knows that the received signal must be from a signal pool containing a set of candidate signals, it could use cross correlation technique to detect which candidate signals are contained in the collided signal, and which are not. We will leverage these techniques to enhance the performance of our protocol.

5.2.3 Statement of The Problem

We consider a RFID system that consists of N known tags $KT = \{t_1, t_2, \ldots, t_N\}$ and M unknown tags $FT = \{u_1, u_2, \ldots, u_M\}$. The reader knows IDs of known tags and has to collect IDs of unknown tags. Because the system is dynamic, KT might

change frequently in different identification operations. It is also possible that some known tags recorded in KT have already left the system after the last identification operation. These tags are usually termed as *missing tags* [Li et al., 2013, Luo et al., 2012, Zheng and Li, 2013a,a]. Different from previous unknown tag identification protocols that do not allow missing tags [Liu et al., 2014b, 2012b, 2014c, Sheng et al., 2010], our approach can *tolerate the existence of missing tags*. That is, our protocol does not require all the tags recorded in KT are present in the system during the identification operation.

The problem in the paper is how to quickly take stock of all tags in the system, with IDs of known tags as a priori. Our aim is to design an efficient tag stocktaking protocol that is compatible with tags following the EPC C1G2 specification, i.e., it relies on only mandatory abilities that must be implemented by current COTS tags such as *hashing* and *random number generating*. Our protocol does not rely on techniques that are not supported by current COTS tags, e.g., the indicator vector technique that is the basis of many previous solutions [Liu et al., 2014b, 2012b, 2014c, Sheng et al., 2010].

The worst and the best solutions An intuitive solution to this problem is to identify all the tags in the system. We call this solution as the *Baseline* solution, and it will have the worst performance because all known tags are unnecessarily re-identified. In contrast, the ideal solution is to collect the IDs from only the unknown tags (i.e. tags in FT), and meanwhile figures out which tags recorded in the database are missing. We call this solution as the *Ideal* solution.

5.3 Design of HARN

In this section, we first present the detailed design of the HARN protocol, then analyze its performance in identifying unknown tags, and finally discuss its ability to handle the hidden tag problem.

5.3.1 A Hash-based Approach to Generating RN

In the original EPC C1G2 protocol, tags uses their RNs[1] to contend for channel access. A tag's RN field is independent of its ID. Thus, the reader cannot differentiate whether a received RN is from a known tag or from a unknown tag, even in the case that it knows the IDs of known tags. If the received RN can be used to distinguish unknown tags from known tags, the reader can identify only unknown tags and avoid wasteful re-identification of known tags.

Our solution is to generate the RN field for a tag by hashing its ID to a value in the range $[0, 2^{16} - 1]$. Tag t's RN is generated as $RN_t = H(ID_t) \mod 2^{16}$, where ID_t is t's tag ID and H is a uniform hash function known to both the reader and the tag. With this method, the reader can predict what RN it will receive if a known tag transmits. Furthermore, as the value of t's slot counter is calculated according to its RN filed [EPCglobal, 2008], *the reader can predict in which slot a known tag will backscatter its RN*. Thus, when the reader receives a RN that is different from corresponding known tag's RN, it knows that the RN must be from a unknown tag and can collect its ID immediately.

Besides the RN used to contend for channel access, we introduce a second RN to help quickly suppress responses from known tags. For tag t, its second RN is

[1]For clarity in presentation, in the rest of the paper we use RN and RN16 interchangeably.

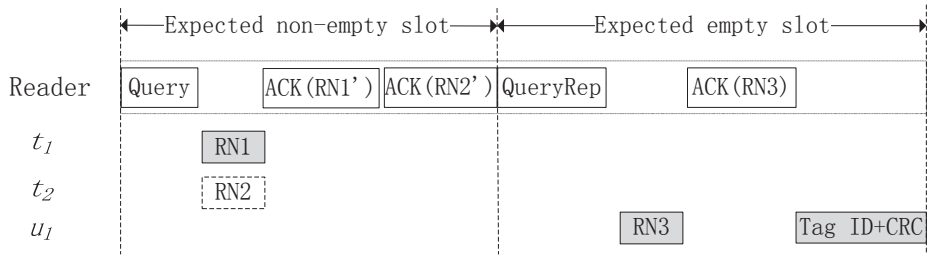


Fig. 5.2: Illustration of the HARN protocol.

generated as $RN'_t = H(ID^r_t) \mod 2^{16}$, where ID^r_t is the reverse of ID_t. As to be explained in Section 5.3.2, it should be assured that $RN'_t \neq RN_t$ to comply with the standard protocol. If the two generated RNs are equal to each other, we let $RN'_t = RN_t + 1$. The reader uses RN' to suppress responses from known tags.

5.3.2 Description of The HARN Protocol

HARN consists of multiple frames. The first frame is special: Besides identifying unknown tags, it suppresses all the known tags to make them enter the acknowledged state and, meanwhile, identifies missing tags if there are any. After all the known tags are suppressed, the remaining unknown tags are identified in the following frames.

Before starting the first frame, the reader calculates the expected status of every slot in the frame according to the IDs of known tags. Recall that the reader knows which slot a known tag t_i will select because it knows t_i's ID. It then predicts what RNs it will receive from known tags in each slot. As shown in Fig. 5.2, the reader acts as below in each slot:

- In an *expected empty slot*, i.e., no known tag will transmit in this slot, if the reader successfully receives an RN, which must be from a unknown tag, it broadcasts an ACK command containing the received RN. Otherwise, if the reader

receives no signal or detects a collision, it broadcasts a QueryRep command to move to the next slot.

- If at least one known tag should transmit in this slot, in which case the slot is named as an *expected non-empty slot*, the reader sends a series of ACK commands to suppress known tags mapped to this slot. Assume that k known tags $\{t_1, \ldots, t_k\}$ select this slot. For each tag t_i $(1 \leq i \leq k)$, the reader broadcasts an ACK command containing t_i*'s second random number (*RN'*)* to suppress its participation in the current identification operation. Note that a tag will transmit its ID to the reader when it receives an ACK command containing its RN. That's why we require $RN \neq RN'$ for tags, because otherwise the known tags will send their IDs to the reader.

Upon receiving the commands from the reader, tags act as below accordingly:

- When tag t receives a Query/QueryRep command and its slot counter equals zero in response to the received command, it backscatters its RN to the reader.
- If the tag receives an ACK command containing the same RN as its RN, it transmits its tag ID along with the CRC to the reader and then enters the *acknowledged* state.
- If the tag receives an ACK command containing a RN that is equal to its RN', it enters the *acknowledged* state.

Compatibility Analyses In HARN, the tag may take three different actions in response to the received commands. The first two actions are exactly the same as in the standard EPC protocol. For the third case, the tag will keep in the *ready*

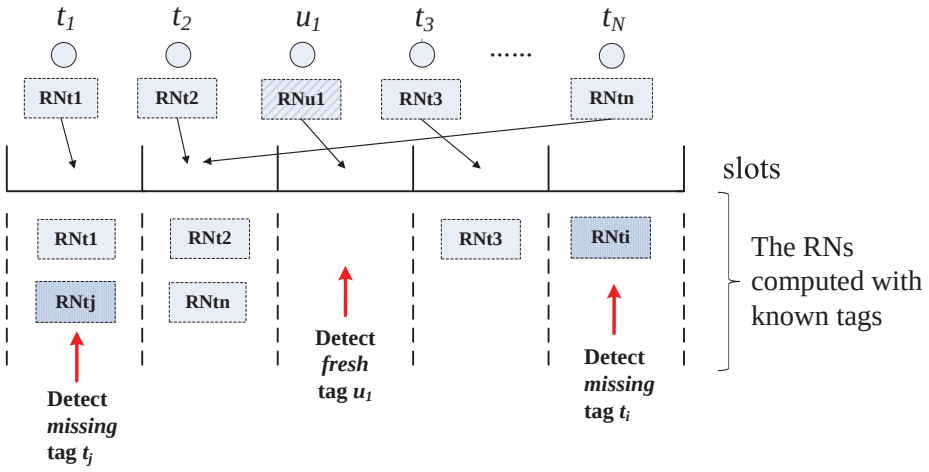


Fig. 5.3: Identify unknown tags and detect missing tags in HARN.

state and participate in the following frames in the original EPC protocol. HARN inherits this operation, but adds only an *additional branch* that makes the tag enter acknowledged state when the received "suppression" RN matches its RN'. This does not change any other logic of the original protocol and complies with the standard protocol.

Missing Tag Detection In expected non-empty slots, the reader also detects if there are any missing tags. For each known tag t_i selecting this slot, the reader detects whether t_i' RN signal appears in the received signal by performing a cross correlation [Liu et al., 2012a] between the received signal and t_i's RN signal, which could be recorded when identifying t_i or be locally generated by using t_i's RN value. Existence of a peak value means that t_i's signal is in the received signal. Otherwise, t_i must be a missing tag and its ID should be removed from the known tag set KT.

Fig. 5.3 illustrates how HARN identifies unknown tags and detects missing tags in the first frame.

5.3.3 Minimizing Per Tag Identification Time

We now discuss how to set the frame length to minimize the average identification time of unknown tags.

The First Frame

Besides identifying unknown tags, the main purpose of the first frame is to suppress responses from all the known tags. Assume that the frame size is f. We use N_e to denote the number of expected empty slots in the frame, and use N_s to denote the number of slots in which unknown tags are successfully identified. The total duration of the frame is

$$T = T_{id} * N_s + T_e * (N_e - N_s) + N * T_{ack} + (f - N_e) * T_{qrn}, \tag{5.1}$$

where T_{id} and T_e indicate the duration of a single slot and an empty slot, T_{ack} indicates the time to send an ACK command containing a RN, and T_{qrn} represents the time needed to send a Query plus a RN, respectively. The third term in equation (5.1) indicates the time spent in suppressing known tags by sending ACK commands, and the fourth term indicates the time spent in receiving RNs transmitted by tags.

As there are N known tags and M unknown tags, N_s and N_e can be calculated as

$$N_e = f * (1 - \frac{1}{f})^N \approx f * e^{-N/f} \tag{5.2}$$

and

$$N_s = f * \frac{M}{f}(1 - \frac{1}{f})^{M+N-1} \approx M * e^{-(M+N)/f}. \tag{5.3}$$

Then the average time to identify a unknown tag is

$$T_f = \frac{T}{N_s} = T_{id} + T_e * (\frac{N_e}{N_s} - 1) + T_{ack} * \frac{N}{N_s} + T_{qrn} * \frac{f - N_e}{N_s}. \tag{5.4}$$

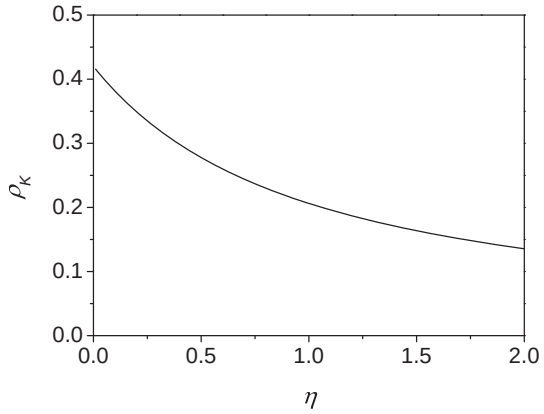


Fig. 5.4: Optimal ρ_K in the first frame of HARN for different η.

We define the ratio of unknown tags to known tags as $\eta = M/N$, and define the load factor with respect to known tags as $\rho_K = N/f$. Then we have

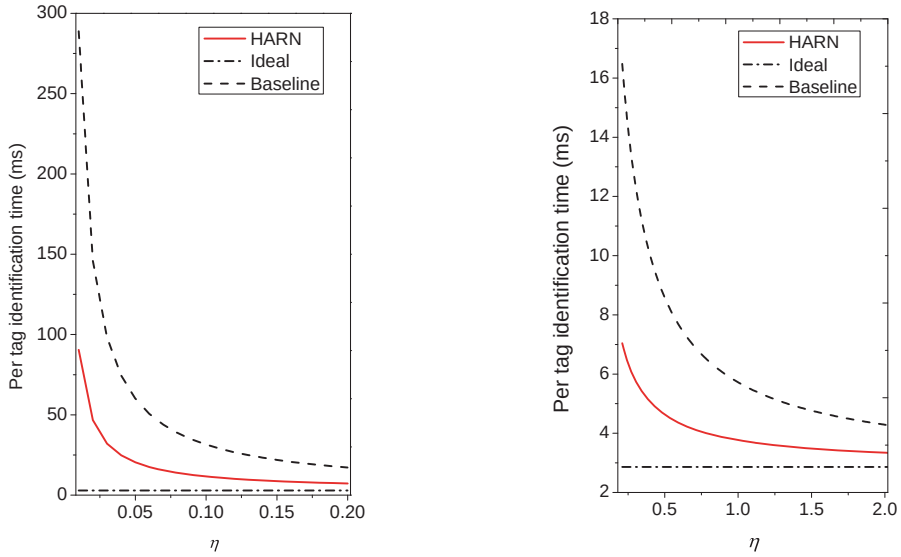
$$T_f = T_{id} + T_e(\frac{e^{\eta\rho_K}}{\eta\rho_K} - 1) + \frac{T_{ack}}{\eta}e^{(1+\eta)\rho_K} + T_{qrn}\frac{e^{(1+\eta)\rho_K} - e^{\eta\rho_K}}{\eta\rho_K} \tag{5.5}$$

To minimize T_f, we let $\frac{\partial T_f}{\partial \rho_K} = 0$, and know that the minimum value of T_f is attained when

$$\begin{aligned} T_e\frac{e^{\eta\rho_K}(\eta\rho_K - 1)}{\eta\rho_K} + T_{ack}\frac{e^{(1+\eta)\rho_K}(1+\eta)}{\eta} \\ +T_{qrn}\frac{\rho_K e^{(1+\eta)\rho_K} + (\eta-1)(e^{(1+\eta)\rho_K} - e^{\eta\rho_K})}{\eta\rho_K^2} = 0. \end{aligned} \tag{5.6}$$

Fig. 5.4 plots the optimal ρ_K to minimize T_f for different η according to the time specification in the EPC C1G2 standard [EPCglobal, 2008]. It can be observed that ρ_K decreases when η increases, which means that a longer frame should be used when there are more unknown tags in the system.

Fig. 5.5 plots T_f in the first frame of HARN and compares it with the Baseline and the Ideal solution. It can be observed that T_f decreases when η increases. When η increases, a longer frame is used as shown in Fig. 5.4, and more unknown tags



(a) Small η (b) Large η

Fig. 5.5: Average time to identify unknown tags in the first frame of HARN.

could be identified in the first frame. This means that the overhead in suppressing known tags are amortized by more unknown tags, and thus the per unknown tag identification time is reduced. Compared with the Baseline solution, HARN reduces per tag identification time by up to 69 percent. We should point out only a small fraction of unknown tags ($e^{-(1+\eta)\rho_K}$) are identified in the first frame, and the other unknown tags are identified without interference from known tags in the following frames. Thus the per tag identification time averaged over all the unknown tags is low.

Other Frames

Different from previous schemes that try to maximize the ratio of singleton slots in the frame, we try to minimize the average time to identify a tag because different slots have different duration time. We use T_c to denote the duration of a collision

slot. Let N_e, N_s, and N_c be the number of empty, singleton, and collision slots in the frame, respectively, and let M_a be the number of remaining active unknown tags. Then the total duration of the frame is

$$T = N_e * T_e + N_s * T_{id} + N_c * T_c, \tag{5.7}$$

where

$$N_e \approx f * e^{-M_a/f}, N_s \approx M_a * e^{-M_a/f}, N_c = f - N_e - N_s. \tag{5.8}$$

Let $\rho = M_a/f$. The average time to identify unknown a tag is

$$\frac{T}{N_s} = T_{id} + T_e * \frac{1}{\rho} + T_c * \frac{e^{\rho} - \rho - 1}{\rho}, \tag{5.9}$$

whose minimum values is attained when

$$e^{\rho}(\rho - 1) + 1 - \frac{T_e}{T_c} = 0. \tag{5.10}$$

According to the EPC C1G2 specification, $T_e/T_c = 0.184/0.44 = 0.4182$. Thus the optimal ρ is 0.7155, in which case the average time to identify a unknown tag is approximately 2.86 ms. That is, in the Ideal solution, the average time to identify a unknown tag is 2.86 ms.

5.3.4 Identification Probability of Unknown Tags

It is possible that some unknown tags might be incorrectly suppressed in the first frame and thus cannot be identified successfully. We call such tags as *hidden tags* as they seem to be "hidden" by the protocol. The hidden tag problem is a common problem that torments all previous protocols targeting unknown tag identification [Liu et al., 2014a,b,c, Sheng et al., 2010]. We now calculate the probability that a unknown tag can be successfully identified in HARN.

In HARN, unknown tags can be hidden in only the first frame. A unknown tag t is incorrectly suppressed when it meets two conditions: 1) It selects an expected non-empty slot in the frame, and 2) Among the known tags selecting the same slot of t, at least one known tag's RN' equals t's RN'. We use P_h to denote the probability that t is incorrectly suppressed, and use $P_{h|k}$ to denote the conditional probability that t is suppressed when there are exactly k known tags selecting the same slot of t. The we have

$$P_h = \sum_{k=1}^{N} P_k * P_{h|k}, \tag{5.11}$$

where P_K denotes the probability that exactly k known tags select the same slot as t, which can be calculated as

$$P_k = \binom{N}{k} (\frac{1}{f})^k (1 - \frac{1}{f})^{N-k}. \tag{5.12}$$

To calculate $P_{h|k}$, we consider the expected number of distinct RN's when k known tags choose their RN's independently in the range $[0, 2^{16} - 1]$, which is denoted as E_k. Actually, E_k equals the expected number of non-empty slots in a frame containing $Q = 2^{16}$ slots when k tags choose their slots independently. It is easy to calculate E_k as

$$E_k = Q(1 - (1 - \frac{1}{Q})^k) \approx Q * (1 - e^{-k/Q}). \tag{5.13}$$

In practice, k is far less than Q, in which case $E_k \approx k$. Thus,

$$P_{h|k} = \frac{E_k}{Q} \approx \frac{k}{Q}. \tag{5.14}$$

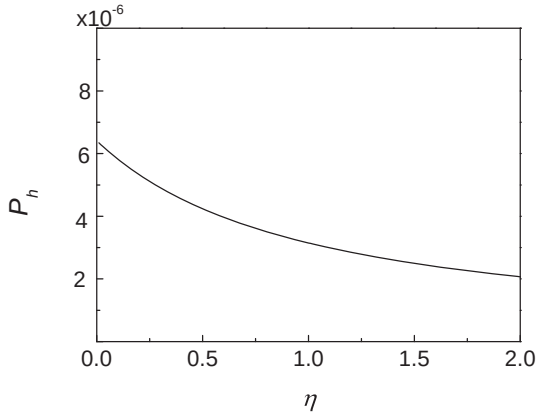


Fig. 5.6: Probability of hiding unknown tags in HARN for different η.

Substituting equation (5.12) and equation (5.14) into equation (5.11), we get

$$\begin{aligned} P_h &= \sum_{k=1}^{N} P_k * P_{h|k} \\ &= \sum_{k=1}^{N} \frac{N!}{k!(N-k)!}(\frac{1}{f})^k(1-\frac{1}{f})^{N-k}\frac{k}{Q} \\ &= \frac{N}{Q*f}\sum_{k=1}^{N}\frac{(N-1)!}{(k-1)!(N-k)!}(\frac{1}{f})^{k-1}(1-\frac{1}{f})^{N-k} \\ &\leq \frac{N}{Q*f} = \rho_K * \frac{1}{Q}. \end{aligned} \tag{5.15}$$

As ρ_K is determined by η, P_h is also determined by η. Fig. 5.6 plots the probability that unknown tags are hidden in HARN for different η. It shows that P_h decreases when η increases, and it is always smaller than 10^{-5}. This means that HARN can well handle the hidden tag problem in real RFID system, even when there are thousands of unknown tags in the reader's interrogation region.

5.4 HARN Enhancement

In the first frame of HARN, unknown tags could be identified in only expected empty slots. In this section, we propose an approach to identifying unknown tags in

expected non-empty slots based on analog network coding, which further improves the identification throughput of unknown tags.

5.4.1 Extracting RN in Expected Non-empty Slots

Non-empty slots are not used to identify unknown tags in the first frame of HARN, which limits its performance. Consider a slot in which a known tag t_1 and a unknown tag u_1 backscatter their RNs simultaneously. The reader will receive a collided signal. However, as the reader knows the signal of t_1's RN[2], it can extract u_1's RN by subtracting t_1's RN signal from the received collided signal, which could be done by using analog network coding [Katti et al., 2007, Zhang et al., 2010].

We briefly introduce how to separate the unknown tag's RN field from the collided signal. More details can be found in [Katti et al., 2007, Zhang et al., 2010]. The signal of t_1's RN value can be represented as

$$s_k[n] = A_k[n]e^{i\theta[n]}, \tag{5.16}$$

where A_k is the amplitude of the RN signal and $\theta_k[n]$ is the phase of the n-th sample of the signal. Similarly, the transmitted signal of u_1's RN can be represented as

$$s_u[n] = A_u[n]e^{i\phi[n]}, \tag{5.17}$$

where A_u and $\phi[n]$ is the amplitude and the phase of the n-th sample of u_1's RN signal, respectively.

Due to channel response and phase shift, the received signal of the RNs of t_1 and u_1 would be distorted. The received signal can be represented as

$$y_k[n] = h_k A_k e^{i(\theta[n]+\gamma_k)}, y_u[n] = h_u A_u e^{i(\phi[n]+\gamma_u)}, \tag{5.18}$$

[2]This signal can be recorded in a server when identifying t_1, or can be locally regenerated by the reader because it knows the value of t_1' RN.

where $h_k(h_u)$ is the channel response between $t_1(u_1)$ and the reader, and $\gamma_k(\gamma_u)$ is the phase shift that depends on the distance between the reader and $t_1(u_1)$.

When t_1 and u_1 transmit their RNs simultaneously, the received signal is the mixture of the two signals

$$y[n] = y_k[n] + y_u[n]. \tag{5.19}$$

Because y_k is known, the reader can subtract y_k from the received signal y to obtain y_u. It can then decode y_u to obtain the RN transmitted by the unknown tag. After the reader obtains the unknown tag's RN, it broadcasts an ACK command containing the decoded RN to collect the unknown tag's ID. The unknown tag's RN could also be successfully extracted when more than one known tags transmit their RNs, providing that the reader knows RN signals of these known tags and the signal to noise ratio (SNR) of the mixed signal is high enough [Katti et al., 2007, Zhang et al., 2010].

However, as the reader has no way to know whether there are unknown tags backscattering their RNs, it just blindly separates a RN value and assumes that it is transmitted by a unknown tag. This may result in some useless RNs. For example, when more than two unknown tags transmit their RNs in the same slot, the extracted RN would be incorrect and cannot be used to collect unknown tag's ID. Another case is that some known tags are missing and the extracted RN is incorrect even when there is only one unknown tag transmitting. However, this will not break the correctness of our protocol, as to be discussed in Section 5.7.

5.4.2 Protocol Description

We name the enhanced HARN protocol as HARN-ANC. In HARN-ANC, tags act as same as in HARN, while the reader broadcasts an additional ACK command

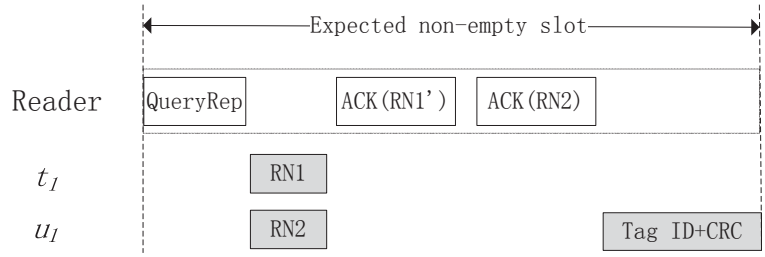


Fig. 5.7: Illustration of HARN-ANC.

containing the extracted RN at the end of each expected non-empty slot, after suppressing all the known tags in this slot. For example, as shown in Fig. 5.7, after the reader receives the signal mixed by RN1 and RN2, it first separates RN2 from the mixed signal, then suppresses responses from the known tag t_1 by broadcasting an ACK command containing RN1. However, different from in the HARN protocol, in HARN-ANC the reader broadcasts another ACK command containing the separated RN2, expecting to identify a unknown tag in this slot. In this example, the reader correctly separates RN2 and successfully identifies u_1.

In order to separate RN2 from the mixed signal, the reader needs to know the signal of RN2. To do this, the reader records the corresponding RN when it successfully receives a tag ID in the slot. In the example given in Fig. 5.7, if the reader successfully receives u_1's ID, it will record the separated signal into a database and mark it as u_1's RN signal. In contrast, if the reader does not receive any valid ID in the slot, which means that the extracted RN is useless, it ignores the responses and will not record the extracted RN signal into the database. With this method, the reader can correctly record the received RN signals for known tags.

The useless RN does not affect the correctness of the protocol. First, if u_1 receives a RN different from its own RN value, it will simply ignore this command and

join the next frame for identification. Second, if there are more than one unknown tags transmitting their RNs in this slot simultaneously, the extracted RN would be different from the RNs of the two unknown tags' RNs[3], and they all will ignore the following ACK command. Thus the correctness of this protocol holds even when a useless RN is extracted.

5.4.3 Optimal Frame Size Setting

We now discuss how to set the size of the first frame in HARN-ANC to minimize the average time to identify unknown tags. The second and the following frames in HARN-ANC use the same frame size setting as in HARN. In the following analyses, we assume that the RN signal transmitted by a unknown tag could be always successfully separated from the mixed signal. The impact of failing in separating the signal on the performance of HARN-ANC is evaluated by simulation in Section 5.5.

Let N_s^f be the number of singleton slots with respect to unknown tags in the frame, and let $r_s^f = N_s^f/f$ be the ratio of such singleton slots in the frame. By using the notations defined in Section 5.3.3, the total time of the frame can be calculated as

$$T = T_{id} * N_s^f + N * T_{ack} + N_e * (1 - r_s^f) * T_e + (f - N_e) * (1 - r_s^f) * T_{ack}, \tag{5.20}$$

where the last term indicates the time wasted in broadcasting useless RNs in expected non-empty slots. r_s^f and N_s^f can be calculated as

$$\begin{aligned} r_s^f &= \frac{M}{f}(1 - \frac{1}{f})^{M-1} \approx \frac{M}{f}e^{-M/f}, \\ N_s^f &= f * r_s^f \approx M * e^{-M/f}. \end{aligned}$$

[3]The extreme scenario is that all the unknown tags selecting the same slot have the same RN, in which case they all transmit their IDs and a collision would be detected. However, the probability of this case is smaller than $\frac{1}{f*2^{16}}$ where f is the frame size, which could be ignored in practice.

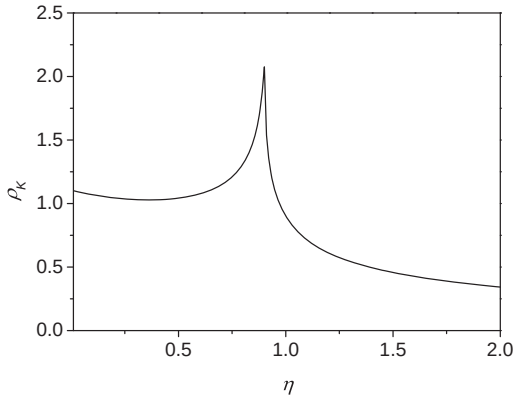


Fig. 5.8: Optimal ρ_K for the first frame in HARN-ANC for different η.

The average time to identify a unknown tag is

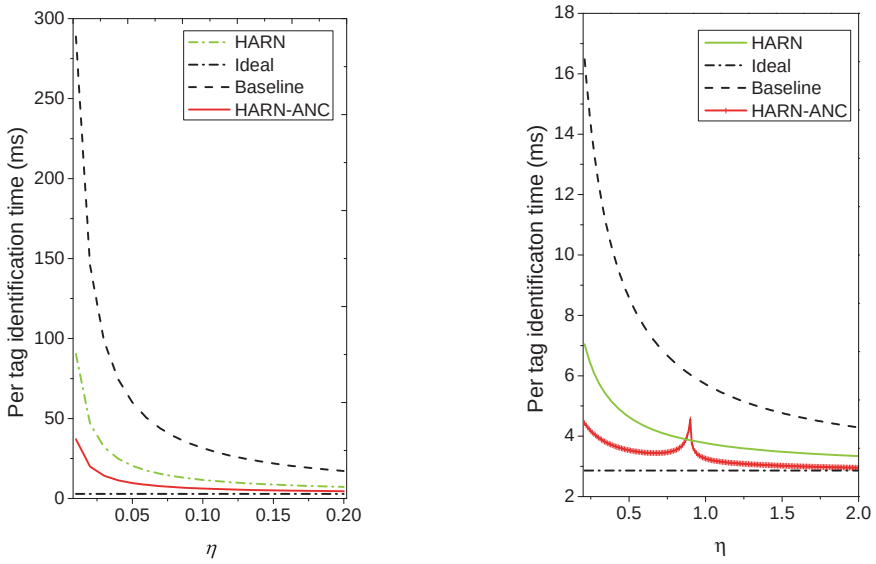
$$T_f = T_{id} + T_{ack}\frac{e^{\eta\rho_K}}{\eta} + T_e * \frac{e^{(\eta-1)\rho_K}(1-\eta\rho_K e^{-\eta\rho_K})}{\eta\rho_K} + T_{ack} * \frac{(e^{\eta\rho_K} - e^{(\eta-1)\rho_K})(1-\eta\rho_K e^{-\eta\rho_K})}{\eta\rho_K}. \quad (5.21)$$

To minimize T_f, we let $\frac{\partial T_f}{\partial \rho_K} = 0$, and finds that the minimum value of T_f is attained when

$$T_{ack} * e^{\eta\rho_n}(1-\frac{1}{\eta^2}) + \frac{T_e}{\eta\rho_n^2}[e^{(\eta-1)\rho_n}((\eta-1)\rho_n - 1) + \eta\rho_n^2 e^{-\rho_n}] + \frac{T_{ack}}{\eta\rho_n^2}[\rho_n(\eta e^{\eta\rho_n} - (\eta-1)e^{(\eta-1)\rho_n}) - (e^{\eta\rho_n} - e^{(\eta-1)\rho_n}) - \eta\rho_n^2 e^{-\rho_n}] = 0. \quad (5.22)$$

Fig. 5.8 plots the optimal ρ_K for the first frame in HARN-ANC for different η. It shows that ρ_K first increases when η increases and reaches the largest value when η is about 1, then it decreases again. ρ_K is always smaller than 2, guaranteeing that the probability that unknown tags are hidden is smaller than 10^{-5} in most cases.

Accordingly, Fig. 5.9 plots the average time used to identify a unknown tag in the first frame of HARN-ANC for different η. We can see that by exploiting expected



(a) Small η (b) Large η

Fig. 5.9: Average identification time in the first frame for different η.

non-empty slots to identify unknown tags, HARN-ANC effectively reduces per tag identification time in most cases. However, when η is about 1, there is a sudden increase in per tag identification time. The reason is that in such cases, only very few unknown tags can be identified in the first frame as ρ_K is larger than 1, and thus each identified unknown tag needs to share more overhead in suppressing the known tags. However, the per tag identification time in HARN-ANC, when averaged over all the unknown tags, is still significantly lower than that in HARN, as to be shown in Section 5.5. As for the first frame, HARN-ANC significantly outperforms HARN in most cases. Compared with Baseline and HARN, HARN-ANC reduces per tag identification time by up to 87% and 58%, respectively. Furthermore, when η is larger than 1, the performance of HARN-ANC is nearly the same as the Ideal solution, showing its superior performance in identifying unknown tags.

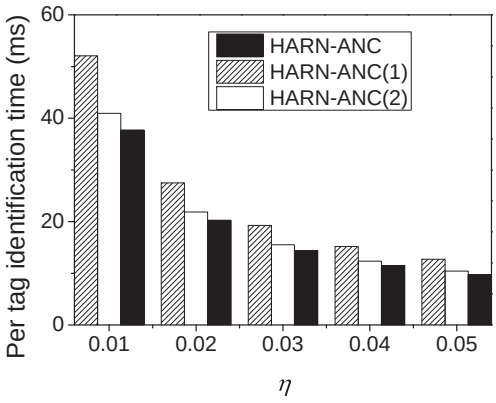


Fig. 5.10: Tag identification time in HARN-ANC with different RN extraction ability. HARN-ANC(1) represents that a unknown tag's RN can be successfully extracted when at most one known tag replies, while HARN-ANC(2) represents that the RN can be extracted when at most two known tags reply.

5.5 Performance Evaluation

We evaluated the performance of HARN and HARN-ANC by using a simulator developed with Matlab. We compare the performance of our protocols with Baseline and Ideal, as well as the IFUTI protocol [Liu et al., 2014b] that represents state-of-the-art solutions based on indicator vector. The execution time of different protocols are calculated according to the EPC C1G2 UHF tag specification [EPCglobal, 2008] when the data rate between the reader and tags is set at 62.5 Kbps.

5.5.1 HARN-ANC with Different RN Extraction Ability

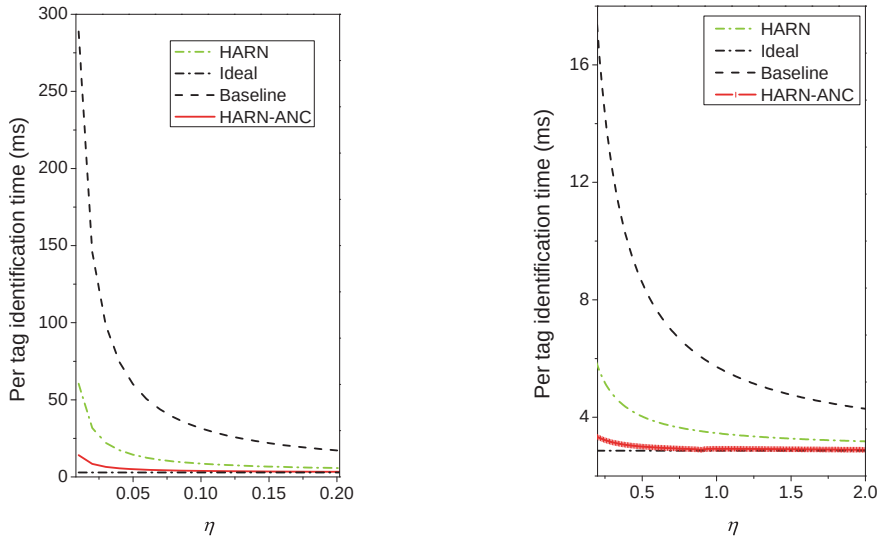
In the analyses of HARN-ANC in Section 5.4.3, we assume that the unknown tag's RN could be always successfully extracted, no matter how many known tags transmit their RNs in the same slot. In practice, the RN might not be successfully extracted when there are too many known tags transmitting simultaneously. Fig. 5.10 plots the performance of HARN-ANC with different ability in extracting the unknown tag's,

where HARN-ANC(k) means that the unknown tag's RN can be extracted when at most k known tags transmit in the same slot. It shows that HARN-ANC(2) performs nearly the same as HARN-ANC, due to the fact that with our frame size setting most collisions are caused by only two known tags. According to the results reported in [Zhang et al., 2010] and [Katti et al., 2007], the RN could be separated even when there are 3 or 4 signals transmitting simultaneously. Thus in practice HARN-ANC could achieve the performance given by the analyses in Section 5.4.3.

5.5.2 Impact of The Ratio of Unknown Tags

We now investigate how the ratio of unknown tags to known tags, namely η, affects the performance of HARN and HARN-ANC. Note that the analyses in Section 5.3.3 and Section 5.4.3 only show the per tag identification time *in the first frame* of HARN and HARN-ANC, respectively. Only a small fraction of unknown tags are identified in the first frame; the rest unknown tags are actually identified with the same performance as in the Ideal solution because all the known tags have been suppressed in the first frame.

Fig. 5.11 plots the per tag identification time *averaged over all the unknown tags* in different protocols. The advantages of HARN and HARN-ANC are significant when η is small. Compared with Baseline, HARN and HARN-ANC reduces per tag identification time by at most 79 percent and at most 90 percent, respectively. They respectively reflect about 3.8x and 11x increases in identification throughput. HARN-ANC outperforms HARN due to its ability to identify unknown tags in expected non-empty slots of the first frame. Furthermore, HARN-ANC performs nearly the same as the Ideal solution when $\eta \geq 0.7$, showing its superior performance in highly dynamic RFID systems.

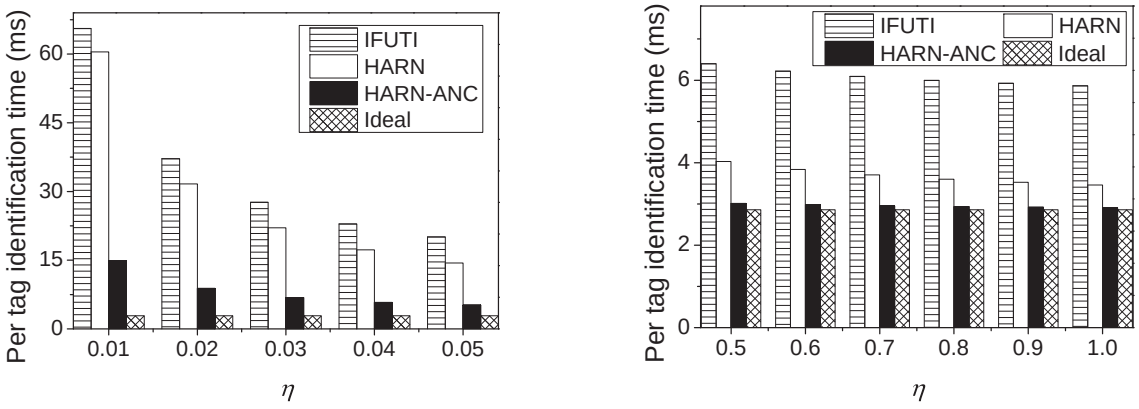


(a) Small η (b) Large η

Fig. 5.11: Average identification time for different η.

5.5.3 Comparison with State-of-the-art Solutions

We also compare HARN and HARN-ANC with IFUTI [Liu et al., 2014b], the state-of-the-art unknown tag identification protocol for dynamic RFID systems that builds on top of the indicator vector technique. IFUTI is a probabilistic approach to identifying unknown tags that leverages a filter vector broadcasted by the reader to suppress responses from known tags. Fig. 5.12(a) plots per tag identification time in different protocols when η is small, with the identification probability of IFUTI setting at 0.99. HARN performs only slightly better than IFUTI when η is small, but significantly outperforms IFUTI when η is large. Compared with IFUTI, HARN reduces per tag identification time by at least 8% and at most 41%, which are equivalent to approximately 9% and 70% increases in identification throughput, respectively. HARN-ANC outperforms IFUTI significantly in all cases. Compared with IFUTI, HARN-ANC improves identification throughput by up to 3.1x and 1.7x



(a) Small η (b) Large η

Fig. 5.12: Comparison with IFUTI [Liu et al., 2014b].

in average. Furthermore, we can observe that while HARN-ANC performs nearly the same as the Ideal solution, IFUTI still results in significantly larger per tag identification time than the Ideal solution even when η is large.

5.6 Summary

This chapter presents the HARN protocol that can boost identification throughput in dynamic RFID systems. HARN requires only very slight modification of the standard protocol, making it easy to be implemented with COTS tags. We also exploit the analog network coding technique to further improve the performance of HARN. Simulation results demonstrate up to 3.8x increase in identification throughput when compared with the standard protocol. When the ANC is integrated into HARN, the throughput increase can be up to 11x. Compared with the state-of-the-art solutions for unknown tag identification, HARN increases identification throughput by factors between 9% and 70%, while HARN-ANC improves identification throughput by up

to 3.1x and 1.7x in average, respectively. In the future, we plan to evaluate the performance of HARN and HARN-ANC by conducting testbed experiments with COTS tags.

Chapter 6

Conclusion and Future Work

6.1 Conclusions

In the thesis, we have proposed a series of efficient tag scanning protocols for three important applications in large-scale RFID systems, namely unknown tag identification in Chapter 3, tag searching in Chapter 4 and tag stocktaking in Chapter 5. We try to avoid the unnecessarily ID transmission, to obtain the require information, during the scanning operation. Obviously, the shorter signal length to be transmitted, the more efficiency it promises to large RFID systems.

We first study how to fast and complete unknown tag identification for large RFID systems. The proposed protocol recognizes known tags without ID transmission and deactivates them to prohibit their further replies. By employing two novel techniques slot pairing and multiple reselections to resolve the known tag collision, the enhanced protocols greatly shorten the time to deactivate the known tags and then quickly identify unknown tags.

We then propose two highly time-efficient tag searching protocols, STEP and E-STEP. STEP employs the novel technique testing slot to iteratively eliminate non-target tags round by round to obtain the searching result. Analysis shows that the

efficiency of STEP degrades when the ratio of the nontarget tags to the non-wanted tags in the readers region decreases. To further reduce the searching time, E-STEP uses a novel approach to dynamically adjust the ratio of the two types of tags. Furthermore, our solution can guarantee that the number of false positive tags is bounded by a constant threshold.

At last we present the HARN protocol that can boost tag stocktaking throughput in dynamic RFID systems. HARN requires only very slight modification of the standard protocol, making it easy to be implemented with COTS tags. We also exploit the analog network coding technique to further improve the performance of HARN.

6.2 Future Work

Future work lies in the following two directions. First, with the widespread of active tags, the time efficiency is not the only concern for the research of RFID system. Since the active tags are battery-powered, it is necessary to pay more attention to the energy efficient protocols of tag scanning. Second, currently evaluating research on large-scale RFID systems depends primarily on simulation, whose results are usually blamed for not convincing enough. We plan to evaluate and refine the proposed protocols in real RFID systems with COTS tags.

Bibliography

Daniel Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4):251–256, 1979.

K. Bu, B. Xiao, Q. Xiao, and S. Chen. Efficient Pinpointing of Misplaced Tags in Large RFID Systems. In *Proc of IEEE SECON*, pages 260–268, 2011.

Kai Bu, Bin Xiao, Qingjun Xiao, and Shigang Chen. Efficient misplaced-tag pinpointing in large RFID systems. *IEEE Transactions on Parallel and Distributed Systens*, 23(11):2094–2106, 2012.

J.R. Cha and J.H. Kim. Novel anti-collision algorithms for fast object identification in RFID system. In *Proc. of IEEE ICPADS*, volume 2, pages 63–67, 2005.

Min Chen, Wen Luo, Zhen Mo, Shigang Chen, and Yuguang Fang. An efficient tag search protocol in large-scale RFID systems. In *Proc. of Infocom*, pages 899–907, 2013.

Shigang Chen, Ming Zhang, and Bin Xiao. Efficient information collection protocols for sensor-augmented RFID networks. In *Proc. of Infocom*, pages 3101–3109, 2011.

R. Das and P. Harrop. RFID Forecasts, Players and Opportunities 2014-2024, 2013. http://www.centrenational-rfid.com/docs/users/file/RFID_Forecasts_2014_2024.pdf.

EPCglobal. EPC Radio-Frequency Identity Protocols Class-1 Gen-2 UHF RFID Protocol for Communications at 860 MHz-960 MHz, 2008.

K. Finkenzeller. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards and Identification*. John Wiley, 2003.

Shyamnath Gollakota and Dina Katabi. Zigzag decoding: combating hidden terminals in wireless networks. In *Proceedings of the ACM SIGCOMM*, pages 159–170, 2008.

Wei Gong, Kebin Liu, Xin Miao, and Haoxiang Liu. Arbitrarily Accurate Approximation Scheme for Large-Scale RFID Cardinality Estimation. In *Proceedings of the 33rd IEEE International Conference on Computer Communications (Infocom)*, pages 477–485, 2014.

Hao Han, Bo Sheng, Chiu Chiang Tan, Qun Li, Weizhen Mao, and Sanglu Lu. Counting RFID tags efficiently and anonymously. In *Proc. of Infocom*, pages 1028–1036, 2010.

D.R. Husn and C. Wood. Analysis of tree algorithm for RFID arbitrtion. In *Proc. of the IEEE International Symposium on Information Theory*, pages 107–107, 1998.

Lei Kang, Kaishun Wu, Jin Zhang, Haoyu Tan, and Lionel M Ni. DDC: A novel scheme to directly decode the collisions in UHF RFID systems. *IEEE Transactions on Parallel and Distributed Systems*, 23(2):263–270, 2012.

Sachin Katti, Shyamnath Gollakota, and Dina Katabi. Embracing wireless interference: analog network coding. In *Proceedings of the ACM SIGCOMM*, pages 397–408, 2007.

Dheerag K. Klair, Kwan-Wu Chian, and Raad Raad. A survey and tutorial of RFID anti-collision protocols. *IEEE Communications Survery and Tutorials*, 2(3):400–421, 2010.

D.K. Klair, K.W. Chin, and R. Raad. An investigation into thie energy eficiency of pure and slotted aloha based RFID anti-collision protocols. In *Proc. of IEEE WoWMoM*, pages 1–4, 2007.

M. Kodialam and T. Nandagopal. Fast and reliable estimation schemes in RFID systems. In *Proc. of ACM Mobicom*, pages 322–333, 2006.

Linghe Kong, Liang He, Yu Gu, Min-You Wu, and Tian He. A Parallel Identification Protocol for RFID Systems. In *Proceedings of the 33rd IEEE International Conference on Computer Communications (Infocom)*, pages 154–162, 2014.

Su-Ryun Lee, Sung-Don Joo, and Chae-Woo Lee. An enhanced dynamic framed slotted aloha algorithm for RFID tag identification. In *Proc. of MobiQuitous*, pages 166–174, 2005.

T. Li, S. Wu, S. Chen, and M. Yang. Energy efficient algorithms for the RFID estimation problem. In *Proc. of IEEE Infocom*, pages 1–9, 2010a.

Tao Li, Shigang Chen, and Yibei Ling. Identifying the missing tags in a large RFID system. In *Proc. of MobiHoc*, pages 1–10, 2010b.

Tao Li, Shigang Chen, and Yibei Ling. Efficient Protocols for Identifying the Missing Tags in a Large RFID System. *IEEE/ACM Transactions on Networking*, 21(6): 1974–1987, 2013.

Haoxiang Liu, Wei Gong, Xin Miao, Kebin Liu, and Wenbo He. Towards Adaptive Continuous Scanning in Large-Scale RFID Systems. In *Proceedings of the 33rd IEEE International Conference on Computer Communications (Infocom)*, pages 486–494, 2014a.

Hongbo Liu, Yu Gan, Jie Yang, Simon Sidhom, Yan Wang, Yingying Chen, and Fan Ye. Push the limit of WiFi based localization for smartphones. In *Proceedings of the 18th Annual International Conference on Mobile Computing and Networking (Mobicom)*, pages 305–316, 2012a.

Xiulong Liu, Keqiu Li, Geyong Min, Yanming Shen, Alex X. Liu, and Wenyu Qu. Completely pinpointing the missing RFID Tags in a time-efficient way. *To appear in IEEE Transactions on Computers*, 2013.

Xiulong Liu, Keqiu Li, Geyong Min, Kai Lin, Bin Xiao, Yanming Shen, and Wenyu Qu. Efficient Unknown Tag Identification Protocols in Large-Scale RFID Systems. *IEEE Transactions on Parallel and Distributed Systems*, pp(99):1–12, 2014b.

Xuan Liu, Shigeng Zhang, Kai Bu, and Bin Xiao. Complete and fast unknown tag identification in large RFID systems. In *Proceedings of the 9th IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS)*, pages 47–55, 2012b.

Xuan Liu, Bin Xiao, Shigeng Zhang, and Kai Bu. Unknown Tag Identification in Large RFID Systems: An Efficient and Complete Solution. *IEEE Transactions on Parallel and Distributed Systems*, pp(99):1–14, 2014c.

Wen Luo, Shigang Chen, Tao Li, and Yan Qiao. Probabilistic missing-tag detection and energy-time tradeoff in large-scale RFID systems. In *Proceedings of the 13t-n ACM International Symposium on Mobile Ad Hoc Networking and Computing (Mobihoc)*, pages 95–104, 2012.

J. Maneesilp, C. Wang, Wu. H., and N.F. Tzeng. RFID support for accurate 3D localization. *IEEE Transactions on Computers*, 62(7):1447–1459, 2013.

Jihoon Myung, Wonjun Lee, Jaideep Srivastava, and Timothy K. Shih. Tag-splitting: Adaptive collision arbitration protocols for RFID tag identification. *IEEE Transactions on Parallel and Distributed Systens*, 18(6):763–775, 2007.

Vinod Namboodiri and Lixin Gao. Energy-aware tag anticollision protocols for RFID systems. *IEEE Transactions on Mobile Computing*, 9(1):44–59, 2010.

Thomas F. La Porta, Gaia Maselli, and Chiara Petrioli. Anticollision protocols for single-reader RFID systems: Temporal analysis and optimization. *IEEE Transactions on Mobile Computing*, 10(2):267–279, 2011.

C. Qian, H. Ngan, and Y. Liu. Cardinality estimation for large-scale RFID systems. In *Proc. of IEEE Percom*, pages 30–39, 2008.

Chen Qian, Yunhuai Liu, Hoilun Ngan, and Lionel M. Ni. ASAP: Scalable identification and counting for contactless RFID systems. In *Proc. of ICDCS*, pages 52–61, 2010.

Chen Qian, Hoilun Ngan, Yunhao Liu, and Lionel M. Ni. Cardinality estimation for large-scale RFID systems. *IEEE Transactions on Parallel and Distributed Systens*, 22(9):1441–1454, 2011.

Yan Qiao, Shigang Chen, Tao Li, and Shiping Chen. Energy-efficient polling protocols in RFID systems. In *Proc. of MobiHoc*, page 25, 2011.

Philips Semiconductors. I-CODE smart label RFID tags, Jan. 2004. [online] Available: http://*www.nxp.com/acrobat_download/other/identification/SL*092030.*pdf*.

M. Shahzad and Alex X. Liu. Every bit counts: fast and scalable RFID estimation. In *Proc. ofACM Mobicom*, pages 365–376. ACM, 2012.

Muhammad Shahzad and Alex X. Liu. Probabilistic optimal tree hopping for RFID identification. In *Proceedings of ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, pages 293–304, 2013.

Bo Sheng, Qun Li, and Weizhen Mao. Efficient continuous scanning in RFID systems. In *Proc. of Infocom*, pages 1010–1018, 2010.

D.H. Shih, P.L. Sun, D.C. Yen, and S.M. Huang. Taxonomy and survey of RFID anti-collision protocols. *Computer communications*, 29(11):2150–2166, 2006.

Chiu Chiang Tan, Bo Sheng, and Qun Li. Efficient techniques for monitoring missing RFID tags. *IEEE Transactions on Wireless Communications*, 9(6):1882–1889, 2010.

Shaojie Tang, Jingyuan, Xiang-Yang Li, and Guihai Chen. Raspberry: A stable reader activation scheduling protocol in multi-reader RFID systems. In *Proc. of ICNP*, pages 304–313, 2009.

Shaojie Tang, Cheng Wang, Xiang-Yang Li, and Changjun Jiang. Reader activation scheduling in multi-reader RFID systems: A study of general case. In *Proc. of IPDPS*, pages 1147–1155, 2011.

Harald Vogt. Efficient object identification with passive RFID tags. In *Proc. of Pervasive*, pages 98–113, 2002.

J. Wang, H. Hassanieh, D. Katabi, and P. Indyk. Efficient and reliable low-power backscatter networks. *ACM SIGCOMM Computer Communication Review*, 42(4): 61–72, 2012.

Lei Xie, Bo Sheng, Chiu Chiang Tan, Hao Han, Qun Li, and Daoxu Chen. Efficient tag identification in mobile RFID systems. In *Proc. of Infocom*, pages 1001–1009, 2010.

Lei Xie, Qun Li, Xi Chen, Sanglu Lu, and Daoxu Chen. Continuous scanning with mobile reader in RFID systems: An experimental study. In *Proc. of Mobihoc*, pages 1–10, 2013.

Lei Yang, Jinsong Han, Yong Qi, Cheng Wang, Tao Gu, and Yunhao Liu. Season: Shelving interference and joint identification in large-scale RFID systems. In *Proc. of Infocom*, pages 3092–3100, 2011.

Lei Yang, Yong Qi, Jinsong Han, Cheng Wang, and Yunhao Liu. Shelving Interference and Joint Identification in Large-Scale RFID Systems. *IEEE Transactions on Parallel and Distributed Systems*, pp(99):1–11, 2014.

Hao Yue, Chi Zhang, Miao Pan, Yuguang Fang, and Shigang Chen. A time-efficient information collection protocol for large-scale RFID systems. In *Proc. of Infocom*, pages 2158–2166. IEEE, 2012.

Ming Zhang, Tao Li, Shigang Chen, and Bo Li. Using Analog Network Coding to Improve the RFID Reading Throughput. In *Proceedings of 2010 International Conference on Distributed Computing Systems (ICDCS)*, pages 547–556, 2010.

Rui Zhang, Yunzhong Liu, Yanchao Zhang, and Jinyuan Sun. Fast identification of the missing tags in a large RFID system. In *Proc. of SECON*, pages 278–286, 2011.

Bin Zhen, Mamoru Kobayashi, and Masashi Shimizu. Framed aloha for multiple RFID objects identification. *IEICE Transactions*, 88-B(3):991–999, 2005.

Yuanqing Zheng and Mo Li. PET: Probabilistic estimating tree for large-scale RFID estimation. *IEEE Transactions on Mobile Computing*, 11(11):1763–1774, 2012.

Yuanqing Zheng and Mo Li. P-MTI: Physical-layer Missing Tag Identification via compressive sensing. In *Proceedings of the 32nd IEEE International Conference on Computer Communications (Infocom)*, pages 917–925, 2013a.

Yuanqing Zheng and Mo Li. Fast Tag Searching Protocol for Large-Scale RFID Systems. *IEEE/ACM Transactions on Networking*, 21(3):924–934, 2013b.

Z. Zhou, H. Gupta, S.R. Das, and X. Zhu. Slotted scheduled tag access in multi-reader RFID systems. In *Proc. of IEEE ICNP*, pages 61–70, 2007.

Yanmin Zhu, Wenchao Jiang, Qian Zhang, and Haibing Guan. Energy-efficient identification in large-scale RFID systems with handheld reader. *To appear in IEEE Transactions on Parallel and Distributed Systens*, 2013.