Fast Identification of the Missing Tags in a Large RFID System

Rui Zhang, Yunzhong Liu, Yanchao Zhang
School of Electrical, Computer, and Energy Engineering
Arizona State University, Tempe, AZ, USA
{ruizhang,liuyz,yczhang}@asu.edu

Jinyuan Sun
Department of Electrical Engineering and Computer Science
University of Tennessee, Knoxville, TN, USA
jysun@eecs.utk.edu

***Abstract*—RFID (radio-frequency identification) is an emerging technology with extensive applications such as transportation and logistics, object tracking, and inventory management. How to quickly identify the missing RFID tags and thus their associated objects is a practically important problem in many large-scale RFID systems. This paper presents three novel methods to quickly identify the missing tags in a large-scale RFID system of thousands of tags. Our protocols can reduce the time for identifying all the missing tags by up to 75% in comparison to the state of art.**

I. Introduction

RFID (radio-frequency identification) is an emerging technology with extensive applications such as transportation and logistics, asset tracking, inventory management, and healthcare. According to a market research report "Global RFID Market Analysis till 2010" by RNCOS, the global RFID market is expected to grow at a compound annual growth rate of around 17% in the period 2011–2013 to a value of approximately 9.7 billion US dollars, and the overall growth in RFID is expected to outpace other automatic identification technologies like barcode.

This paper considers a typical large-scale RFID system for automating inventory management and asset tracking over a large region, such as a super warehouse, a mega factory, a huge hospital complex, and a military base. As shown in Fig. 1, the system consists of a backend server, many fixed RFID readers, and a large number of RFID tags. Every RFID tag contains a unique numeric ID and is attached to a physical object (item or even human) to be monitored. The objects can arbitrarily move either proactively (e.g., vehicles or humans) or due to external forces (e.g., goods or tools carried by people), so do the associated RFID tags. The server records all the tag-object mappings and periodically instructs RFID readers to interrogate RFID tags. The transmission range of RFID tags ranges from several tens of feet for passive tags to hundreds of feet for semi-active or active tags. Therefore, sufficient RFID readers are deployed to ensure full coverage over the large region, with each in charge of a subregion called a *zone*. The readers communicate with the server via single-hop or multi-hop high-speed wireless links; e.g., the readers can form a wireless mesh network as shown in Fig. 1.

Quickly identifying the missing tags is critical in many large-scale RFID systems as outlined above [1]–[3]. In particular, since RFID tags are typically designed to be (nearly) non-removable from their associated physical objects, a missing tag highly indicates that the corresponding object is also missing from the monitored region. It is thus of paramount importance to quickly identify which tags are missing in order to take some countermeasures as soon as possible, especially when there are high-valued physical objects such as jewelries, weapons, and munitions. The most important criterion for missing-tag identification is to minimize the identification time for two main reasons. First, long identification time may make it impossible to retrieve the missing objects that may have been moved out of the monitored region, and a difference on the order of seconds may matter a lot.[1] Second, short identification time corresponds to lower overhead and can enable periodic missing-tag identification operations on a high frequency and thus nearly realtime asset tracking.

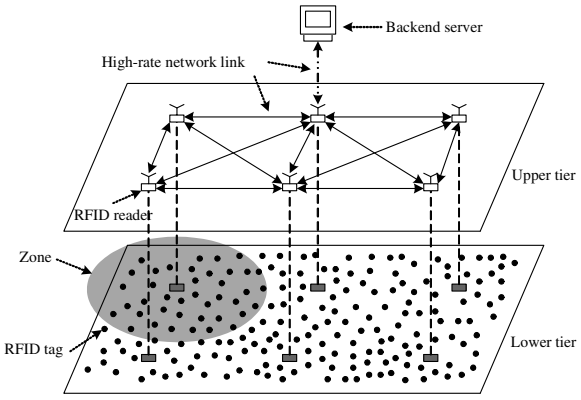


Fig. 1. A snapshot of a large-scale RFID system.

Despite significant research on RFID technologies, missing-tag identification in a large-scale RFID system remains challenging and under-explored. A naive solution is to collect the IDs of all present tags and then compare them with those recorded in the server to identify the missing ones. This method is, however, very time-consuming due to the extremely long time spent on resolving radio contention among a large number of tags competing for the same low-bandwidth channel to report their IDs [1]–[3]. Moreover, collecting all the tag IDs

[1]For example, stolen objects may be quickly loaded on a fleeting vehicle parking just outside the monitored region.

is vulnerable to dishonest RFID readers returning incorrect information to the server. For example, a dishonest employee can first collect all the tag IDs prior to the theft, and replay those stolen tags' IDs back to the server later [1], [2]. In [1], [2], Tan *et al.* proposed efficient methods without collecting tag IDs to detect whether there is any missing tag with probability α if the number of missing tags is larger than a predetermined threshold m, but their method cannot tell which tags are missing. Most recent work [3] presents a suite of novel protocols to identify which tags are missing with certainty, among which the most efficient one is the identification-free protocol (IIP). When being applied to a large-scale RFID system as in Fig. 1, the protocols proposed in [3] logically treat all the readers as one (see Section 2.1 of [3]). This is in effect equivalent to considering all the tags in a single *collision domain* in the sense that they all compete for the same low-bandwidth channel for responding to the virtual reader's interrogation.

We observe that the existence of multiple RFID readers is under-utilized in [3]. In particular, the transmissions from a particular tag only collide with those from other tags covered by the same reader. Then it is natural to let all the readers perform independent and parallel missing-tag identification in their respective coverage zone. Since a collision domain now only involves the tags within a reader's coverage zone instead of all the tags in the system, we expect the overall time on resolving tag collisions and thus the overall identification time to be much reduced in comparison to the IIP protocol in [3].

With the above insight, we propose three novel protocols for fast identification of the missing tags in a large-scale RFID system without explicitly collecting tag IDs. In our protocols, all RFID readers are instructed by the backend server to conduct multiple synchronized scans within their respective coverage zone and return the collected tag information to the server for further processing. In the first protocol, the server directly identifies tags that are absent in all zones and consider those missing in the system. This protocol is a probabilistic one that ensures all missing tags can be identified with a target probability α when the number of missing tags does not exceed a target threshold M. The second protocol eliminates the need to estimate M and lets the server identify the tags present in every zone through an iterative process and eventually identify all the missing ones as those not present in any zone. Our last protocol further improves the second protocol by suppressing the transmissions from the tags identified present in previous rounds. Since fewer tags answer the readers' interrogations as time goes by, the protocol execution time can be reduced. Extensive simulations show that our protocols can reduce the missing-tag identification time by up to 60.5%, 70.15%, and 75.26%, respectively, in comparison with the IIP protocol in [3].

II. Preliminaries

In this section, we will first present the problem formulation and the introduce a framed slotted ALOHA protocol underlying our protocols.

A. Problem Formulation

We assume a large-scale RFID system deployed over a large region as shown in Fig. 1, which consists of one backend server, L RFID readers, and N RFID tags. The large region is divided into L zones with each covered by one reader. Every reader can communicate with the tags within its coverage zone via a one-hop low-rate wireless link. In contrast, the readers and the server can communicate through a single-hop or multi-hop wireless or wired link. The tags can be passive tags which do not have a power source and only transmit a signal upon receiving RF energy emitted from a nearby reader, active or semi-active tags that are powered by a battery and have a longer transmission range, or a mixture of them. Since the objects and thus their associated RFID tags may move arbitrarily in the region, we make the practical assumption that the server and the readers do not know which tags are present in each zone.

To ensure full coverage of the monitored region, adjacent readers may have overlapping coverage zones. The optimal number and deployment of RFID readers depend on the size and shape of the monitored region, the number N of tags, the minimum transmission range among the N tags, the actual coverage requirement (say, k-coverage), and other factors. This problem can be solved by referring to the existing rich literature on sensor network coverage (see [4] and the references therein) and is orthogonal to our work in this paper.

Given the above system model, we aim to tackle the following problem. Suppose that the system administrator suspects that there are some tags and their associated objects missing from the monitored region, which might be stolen or moved out of the monitored region by mistake. He wants to actually identify all the missing tags with a confidence level $\alpha \leq 1$ as quickly as possible in order to take some necessary countermeasures. Note that he may also run missing-tag detection periodically.

B. Framed Slotted ALOHA Protocol

In our proposed protocols, every reader communicates with the tags in its coverage zone with the framed slotted ALOHA protocol, which is a popular anti-collision MAC protocol adopted by many RFID systems [1]–[3], [5]–[16] and works as follows. First, the reader broadcasts two parameters $\langle r, f \rangle$, where r is a random number and f is the number of time slots in one frame which specifies the scanning time window. We assume that the slots are numbered from 0 to $f-1$. Upon receiving $\langle r, f \rangle$, every tag replies in slot $h(ID \oplus r) \mod f$, where ID is its unique ID and $h(\cdot)$ denotes an efficient hash function which is available in RFID tags [1]–[3], [5]–[16].

There are three types of slots: an *empty* slot if no tag replies in that slot, a *singleton* slot if only one tag replies in that slot, and a *collision* slot if two or more slots reply in the same slot. The tags need send multi-bit long responses for the reader to distinguish empty, singleton, and collision slots. For example, according to the specification in the Philips I-Code system [17], 10-bit tag responses are required for this purpose and lead to a slot length of $t_s = 0.8$ ms [3].

III. Missing-Tag Identification Protocols

In this section, we present three novel protocols to quickly identify the missing tags in a large-scale RFID system. In all three protocols, the server instructs the L readers to perform multiple synchronized scans and aggregates the responses from the readers to identify the missing tags. To simplify the presentation, we first illustrate our protocols under the assumptions that there are no transmission errors and that tags are static during the execution of our protocols. Then we discuss the impact of tag mobility and imperfect wireless channels on our protocols in Section III-D.

A. Protocol 1: Using Empty Slots

Our first protocol exploits the empty slots observed in every scan. In particular, we observe that if a particular tag is in a particular zone, it should reply in a specific slot when that zone is scanned according to the framed slotted ALOHA protocol; otherwise, if that slot is observed empty, then that tag is certainly not in that zone. Based on this observation, the server can decide that a particular tag is missing if the tag is determined absent in all the L zones after multiple scans. Protocol 1 is also probabilistic in the sense that we aim to detect all the missing tags with probability no less than a predetermined confidence level α if there are at most M missing tags. In what follows, we first detail the basic operations of Protocol 1 and then discuss the impact of overlapping zones and the optimal selection of scanning parameters.

1) Detailed operations: Protocol 1 consists of multiple rounds. In each round, the server instructs the L readers to perform one synchronized scan and then processes the scan results returned by the readers.

Consider the nth round as an example, where $1 \leq n \leq \hat{n}_1$ and $\hat{n}_1$ denotes the total number of rounds to be discussed in Section III-A3. At the beginning of the nth round, the server broadcasts a fresh random number r and a frame length f to all the L readers, meaning that the response frame consists of f slots of length t_s. Each reader will locally broadcast the received $\langle r, f \rangle$ to the tags in its coverage zone. Let $\mathcal{T}$ denote all the tags that should be in the monitored region and T_i be the ID of the ith tag, where $1 \leq i \leq N$. According to the framed slotted ALOHA protocol, tag T_i should reply in slot $s_i = h(T_i \oplus r) \mod f$. At the end of the nth scan, every reader returns an *observation vector* of $2f$ bits to the server. Let $\mathcal{O}_{l,n}$ denote the observation vector generated by the lth ($1 \leq l \leq L$) reader in the nth round and $\mathcal{O}_{l,n}[j]$ denote bits j and $2j+1$ which indicate the status of slot j for $0 \leq j < f$. In particular, $\mathcal{O}_{l,n}[j]$ is equal to 00 for an empty slot, 01 for a singleton slot, and 11 for a collision slot.

After receiving $\{\mathcal{O}_{l,n}\}_{l=1}^{L}$, the server checks every tag against $\{\mathcal{O}_{l,n}\}_{l=1}^{L}$ to see whether it is absent in one or multiple zones. In particular, for every tag T_i, the server checks whether $\mathcal{O}_{l,n}[s_i] = 00$, for all $l \in [1, L]$. If so, T_i is certainly absent in zone l; otherwise, T_i may or may not be in zone l because there might be other tags that also replied to reader l in slot s_i. If T_i is found absent in all the L zones after $\hat{n}_1$ rounds, the server can claim it missing.

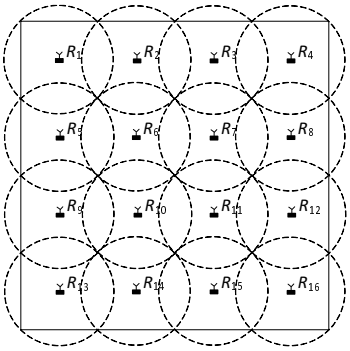


Fig. 2. An example of overlapping zones.

2) Overlapping scans: Adjacent readers need to have overlapping coverage zones to ensure full coverage over the monitored region. Now we discuss the impact of overlapping zones on Protocol 1 and necessary simple adaptations.

The tags in the overlapping zones can receive from and transmit to multiple readers. To ensure that these tags can each receive at least one complete message $\langle r, f \rangle$ in each scan, the server perform simple scheduling to avoid message collisions. In particular, the server can partition the L readers into multiple sets according to their locations such that the readers from the same set do not have overlapping coverage zones. For example, if the readers are deployed in a grid fashion as in Fig. 2, the readers can be divided into four sets. Then when one set of readers are scheduled to broadcast $\langle r, f \rangle$, the readers in other sets keep silent.

Protocol 1 requires every tag to reply only once in each scan even if it may receive the same $\langle r, f \rangle$ from multiple readers. Since multiple readers can receive the same response from every tag in their overlapping zones, their observation vectors for the server will contain redundant information for the same tag. Fortunately, such redundant information does not affect the identification performance of Protocol 1 which solely depends on observed empty slots. In short, even if there are overlapping zones, the missing tags can still be identified because they will still be identified absent in all the zones.

3) Optimal scan-parameter selection: Now we discuss the optimal parameters $\langle r, f \rangle$ to minimize the total scan time. Protocol 1 uses the same frame length f and a unique r for each scan. Recall that $\hat{n}_1$ denotes the total number of rounds needed in Protocol 1. Our target is to minimize $\hat{n}_1 f t_s$ to identify all the missing tags with probability no less than α when there are no more than M missing tags.[2]

To make the derivation tractable, we assume that the N tags are uniformly distributed. Let ρ denote the tag density equal to N divided by the area of the monitored region. Assuming that the coverage zone of every reader is a circle with radius R, there would be about $\rho\pi R^2$ tags in every zone. In addition, we assume that there are far less than $\rho\pi R^2$ missing tags in

[2]As in [3], here we ignore the time for the readers to broadcast $\langle r, f \rangle$, which is negligible in contrast to $\hat{n}_1 f t_s$ [3].

every zone, which generally holds in practice (e.g., thieves will not steal too many objects from the same zone to avoid being easily detected). Then there are approximately $\rho\pi R^2$ tags replying to every reader in every scan.

Assume that tag T_i is actually missing. For the nth round in zone l, the probability that T_i can be found absent in this zone is equal to the probability that slot $s_i = h(T_i \oplus r) \mod f$ is empty, which can be approximated by $(1 - \frac{1}{f})^{\rho\pi R^2} \approx \exp(-\frac{\rho\pi R^2}{f})$. It follows that the server can find T_i absent in zone l after n scans with probability $1 - q^n$, where $q = 1 - \exp(-\frac{\rho\pi R^2}{f})$.

The server can claim T_i missing from the monitored region after n rounds if T_i is found absent in all the L zones, which occurs with probability $(1-q^n)^L$. Recall that the server expects at most $M \ll N$ tags to be missing. To ensure that all the M missing tags can be identified with probability no less than α after n rounds, the following constrain must hold.

$$(1 - q^n)^{LM} \geq \alpha \, . \tag{1}$$

Given $L, R, \rho, t_s, M,$ and α, we then have the following optimization problem.

$$\begin{aligned} &\min_{f,n\in\mathbb{Z}+} fnt_s \\ &\textbf{s.t.} \quad (1 - q^n)^{LM} \geq \alpha \, . \end{aligned} \tag{2}$$

Intuitively, the total scan time is minimized when the equality holds in the constraint, i.e., $\hat{n}_1 = \log_q(1 - \alpha^{\frac{1}{LM}})$. The minimization problem can then be transformed to

$$\min_{f\in\mathbb{Z}+} t_s f \log_q(1 - \alpha^{\frac{1}{LM}}), \tag{3}$$

which the server can solve by offline exhaustive search to find the optimal f. With the optimal f in place, the server can stop instructing the readers to do further scanning when the readers have finished $\hat{n}_1 = \log_q(1 - \alpha^{\frac{1}{LM}})$ scans.

B. Protocol 2: Using Empty, Singleton, and Collision Slots

Protocol 1 only uses empty slots and also requires estimating the maximum number M of possible missing tags. In addition, it may not be able to detect all the missing tags due to its probabilistic nature. We propose Protocol 2 to eliminate these drawbacks. The basic idea is to let the server fully exploit all observed empty, singleton, and collision slots to identify the tags in every zone and eventually deduce the missing tags as those not found in any zone.

Protocol 2 also consists of multiple rounds and uses an iterative process. Let $\mathcal{T}$ denote the tags that should be in the monitored region before the protocol execution. Initially, the server assumes that every tag in $\mathcal{T}$ could be in every zone and can gradually reduce such ambiguity. Specifically, in every round, the server instructs the readers to broadcast $\langle r, f\rangle$ to the tags in their respective coverage zones as in Protocol 1. The server can identify some tags in $\mathcal{T}$ that are present or absent in every zone by resolving some empty, singleton, and collision slots. Protocol 2 requires the server to buffer all the observation vectors collected in previous rounds. The information about newly identified tags present or absent in every zone in every round could also help resolve some unresolved singleton and collision slots in the observation vectors collected in previous rounds based on the following observations.

- **Observation 1**: *If a tag is absent in one zone in one round, it should also be absent in all the other rounds in the same zone.*
- **Observation 2**: *If a tag is present in one zone in one round, it should be present in all the other rounds in the same zone and absent in all the other non-overlapping zones in all the rounds.*

In other words, any tag in $\mathcal{T}$ which was newly identified present or absent in one zone could trigger a chain reaction and lead to more resolvable non-empty slots in all the observation vectors and thus more tags in $\mathcal{T}$ identified present or absent in every zone. The server starts a new round until there are no more tags that could be identified present or absent in every zone. Protocol 2 terminates until every tag in $\mathcal{T}$ has been identified present or absent in every zone. Finally, the server can claim the tags not in any zone to be missing.

1) Detailed operations: Before detailing Protocol 2, we first define three types of tag sets. Let $\mathcal{P}_{l,n}$ denote the set of tags which have been identified present in zone l after n rounds, $\mathcal{A}_{l,n}$ denote the set of tags which have been identified absent in zone l after n rounds, and $\mathcal{U}_{l,n}$ denote the set of tags in $\mathcal{T}$ which have not been identified present or absent in zone l after n rounds. The tags in $\mathcal{U}_{l,n}$ are referred to as *unidentified* tags. Before the execution of Protocol 2, the server sets $\mathcal{P}_{l,0} = \phi$, $\mathcal{A}_{l,0} = \phi$, and $\mathcal{U}_{l,0} = \mathcal{T}$ for $1 \leq l \leq L$. As the protocol proceeds, the number of unidentified tags will gradually reduce to zero, but we always have $\mathcal{T} = \mathcal{P}_{l,n} \bigcup \mathcal{A}_{l,n} \bigcup \mathcal{U}_{l,n}$ for $1 \leq l \leq L$, which means that any tag in $\mathcal{T}$ can only be present, absent, or unidentified in zone l after round n. Given these definitions, Protocol 2 terminates in round $\hat{n}_2$ when all the tags in $\mathcal{T}$ have been identified either present or absent in every zone, i.e., $\bigcup_{l=1}^{L} \mathcal{U}_{l,\hat{n}_2} = \phi$. Then the server can easily identify the missing tags as those in $\mathcal{T} - \bigcup_{l=1}^{L} \mathcal{P}_{l,\hat{n}_2}$.

Given the above definitions, the server knows that the observation vector returned by reader l in round $n+1$ must contain the responses from the tags in $\mathcal{P}_{l,n}$ and could also contain the responses from some tags in $\mathcal{U}_{l,n}$. To utilize this observation, the server constructs two matrixes $U_n := [u_{l,s}]_{L\times f}$ and $P_n := [p_{l,s}]_{L\times f}$ at the end of round n, where $u_{l,s}$ and $p_{l,s}$ denote the number of tags in $\mathcal{U}_{l,n}$ and $\mathcal{P}_{l,n}$ that are expected to reply in slot s in round $n+1$ according to the framed slotted ALOHA protocol, respectively.

Without loss of generality, we take round $n+1$ as an example to illustrate the immediate operations to update the three tag sets for every zone. After receiving the observation vectors $\{\mathcal{O}_{l,n+1}\}_{l=1}^{L}$, the server checks the status of every slot s in $\mathcal{O}_{l,n+1}$ to identify more tags present in every zone $l \in [1, L]$ according to the following cases.

- **Case 1**: If slot s is a singleton slot, $u_{l,s} = 1$, and $p_{l,s} = 0$, i.e., only one unidentified tag in $\mathcal{U}_{l,n}$ is expected to reply in a singleton slot s, tag T_i is certainly present in zone l.

- **Case 2**: If slot s is a collision slot, and $u_{l,s} + p_{l,s} = 2$, i.e., only two tags in $\mathcal{P}_{l,n} \bigcup \mathcal{U}_{l,n}$ are supposed to reply in a collision slot s, both tags are present in zone l.

Let Υ_l denote the set of tags newly identified present in zone l according to either case above. The server then constructs $\mathcal{P}_{l,n+1} = \mathcal{P}_{l,n} \bigcup \Upsilon_l$ and $\mathcal{U}_{l,n+1} = \mathcal{U}_{l,n} - \Upsilon_l$.

The server also checks the status of every slot s in $\mathcal{O}_{l,n+1}$ to identify more tags absent in every zone $l \in [1, L]$ according to the following cases.

- **Case 3**: If slot s is an empty slot, and $u_{l,s} > 0$, i.e., some unidentified tags in $\mathcal{U}_{l,n}$ are expected to reply in an empty slot s, all these tags are absent in zone l.
- **Case 4**: If slot s is a singleton slot, $u_{l,s} > 0$, and $p_{l,s} = 1$, i.e., a present tag and some unidentified tags are all expected to reply in a singleton slot s, all the unidentified tags are absent in zone l.

Let Ψ_l denote the set of tags newly identified absent in zone l according to either case above. The server then constructs $\mathcal{A}_{l,n+1} = \mathcal{A}_{l,n} \bigcup \Psi_l$ and $\mathcal{U}_{l,n+1} = \mathcal{U}_{l,n} - \Psi_l$.

The iterative process of Protocol 2 works on the combination of the above two observations and four cases. In particular, assume that the server has constructed $\mathcal{P}_{l,n+1}$, $\mathcal{A}_{l,n+1}$, and $\mathcal{U}_{l,n+1}$ according to Cases 1 to 4 after receiving $\{\mathcal{O}_{l,n+1}\}_{l=1}^{L}$ for every zone $l \in [1, L]$. According to Observation 2, the tags newly identified present in zone l, i.e., Υ_l, should be absent in every zone j that does not overlap with zone l. Therefore, for every zone $j \in [1, L]$ that does not overlap with zone l, the server updates $\mathcal{A}_{j,n+1} = \mathcal{A}_{j,n+1} \bigcup \Upsilon_l$ and $\mathcal{U}_{j,n+1} = \mathcal{U}_{j,n+1} - \Upsilon_l$. In addition, according to Observation 1, all the tags in $\mathcal{A}_{l,n+1}$ should also be absent in zone l in all previous rounds, while according to Observation 2, all the tags in $\mathcal{P}_{l,n+1}$ should also be present in zone l in all previous rounds. The server thus further sets $\mathcal{A}_{l,k} = \mathcal{A}_{l,n+1}$, $\mathcal{P}_{l,k} = \mathcal{P}_{l,n+1}$, and $\mathcal{U}_{l,k} = \mathcal{U}_{l,n+1}$ for every round $k \in [1, n]$. Next, the server updates the matrixes U_k and P_k for every round $k \in [1, n+1]$ and tries to identify more tags present or absent in every zone by reexamining all the other Ln observation vectors according to the above four cases and two observations.

In short, every tag newly identified present or absent in every zone will trigger the updates of the present-tag, absent-tag, and unidentified-tag sets in all the zones and rounds and lead to more tags identified present or absent in every zone. This chain reaction stops until there is no more change in any present-tag, absent-tag, and unidentified-tag sets. If $\bigcup_{l=1}^{L} \mathcal{U}_{l,n+1} = \phi$, the server can determine the missing tags to be those in $\mathcal{T} - \bigcup_{l=1}^{L} \mathcal{P}_{l,n+1} = \bigcap_{l=1}^{L} \mathcal{A}_{l,n+1}$, i.e., the tags in none of the L zones. Otherwise, the server initiates another round and repeats the above iterative identification process. Also note that $\bigcap_{l=1}^{L} \mathcal{A}_{l,n+1}$ always contain the missing tags identified until round n.

Unlike in Protocol 1, we are unable to derive the optimal frame length f for Protocol 2. We, however, will show that Protocol 2 still outperforms Protocol 1 even using the same frame length as in Protocol 1.

C. Protocol 3: Suppressing Identified Tags

We further propose Protocol 3 to improve the performance of Protocol 2. The basic idea is to keep the tags identified present in previous rounds from replying to the readers in subsequent rounds. In this way, the number of singleton and collision slots is expected to be much reduced, which would make it much quicker to classify the remaining unidentified tags as either present or absent in any zone. The overall missing-tag identification time can thus be further reduced.

Protocol 3 is based on Protocol 2 and selectively adds one suppression process between two consecutive rounds. For this purpose, the server just needs to record whether a tag in any present-tag set $\mathcal{P}_{l,n}$ has been suppressed. After the iterative identification process stops in every round n according to Protocol 2, the server computes the ratio of the number of unsuppressed present tags in $\mathcal{P}_{l,n}$ to the number of unidentified tags, i.e., $|\mathcal{U}_{l,n}|$, in every zone l. If the average ratio across all the L zones is larger than a predetermined suppression threshold δ, the server instructs all the readers to broadcast a suppression command which asks some identified present tags to keep silent in subsequent rounds.

The suppression command for reader $l \in [1, L]$ comprises an f-bit vector $S_{l,n}$ indicating the tags in zone l that should keep silent from now on. Based on the same $\langle r, f \rangle$ used in this round, $S_{l,n}$ is constructed to suppress as many tags in $\mathcal{P}_{l,n}$ as possible while not suppressing any unidentified tag in $\mathcal{U}_{l,n}$. In particular, the server sets the jth bit of $S_{l,n}$ to 0 if there is at least one tag $T_i \in \mathcal{U}_{l,n}$ that satisfies $h(T_i \oplus r) \mod f = j$; otherwise, it sets the jth bit of $S_{l,n}$ to 1. After receiving $r, f, S_{l,n}$ from reader l, every tag T_x computes $s_x = h(T_x \oplus r) \mod f$ and checks whether the s_xth bit of $S_{l,n}$ is 1. If so, it knows that it has been identified present in zone l and will keep silent in subsequent rounds.

Special care need be taken for the tags in overlapping zones. In particular, the server uses the same scheduling method for transmitting the scan message $\langle r, f \rangle$ to avoid the collisions among the suppression commands. A tag, say T_x, in an overlapping zone will receive multiple suppression commands and need to keep silent as long as it needs to do so according to at least one suppression command. In addition, the server marks T_x suppressed in every present-tag set $\mathcal{P}_{l,n}$ (for all $l \in [1, L]$) where it resides. It is obvious that this measure will not cause any ambiguity in subsequent rounds.

The iterative identification process in Protocol 2 need be slightly modified to deal with suppressed tags. Specifically, in round $n+1$, the server only expects replies from the unsuppressed tags in $\mathcal{P}_{l,n}$ and the unidentified tags in $\mathcal{U}_{l,n}$ for any $l \in [1, L]$. In addition, after the execution of Protocol 3, the server need broadcast a special command via the readers to reactivate all the suppressed tags.

D. Impact of Tag Mobility

Our previous protocol illustrations have implicitly assumed that the tags do not cross zone boundaries during the protocol execution. In practice, there might be a few tags moving into another zone during the very short execution time of all our

protocols. Fortunately, all our protocols can easily cope with tag mobility after the following simple adaptation.

For all three protocols, we introduce a short polling phase before broadcasting the scan parameters $\langle r, f \rangle$ in every round to deal with tag mobility. At the beginning of every polling phase, every reader broadcasts a polling request containing its ID. Consider tag T_i as an example. We require T_i to record the set of reader IDs it heard during the previous polling phase n, which is denoted by $\mathcal{R}_{i,n}$ and requires $|\mathcal{R}_{i,n}|\lceil \log_2(L) \rceil$ bits. Since tag T_i is in one zone or the overlapping zone of a few readers, i.e, $|\mathcal{R}_{i,n}|$ is equal to one or a very small integer, this memory cost is negligible. Assume that tag T_i hears $\mathcal{R}_{i,n+1}$ in the polling phase $n+1$. If $\mathcal{R}_{i,n+1} = \mathcal{R}_{i,n}$, then tag T_i did not cross any zone boundary; otherwise, it has left zones $\mathcal{R}_{i,n} - \mathcal{R}_{i,n+1}$ and entered zones $\mathcal{R}_{i,n+1} - \mathcal{R}_{i,n}$ in round n. Then T_i returns $\mathcal{R}_{i,n}$, $\mathcal{R}_{i,n+1}$, and $h(T_i \oplus r)$. The server can easily figure out which tag is crossing the boundary based on $h(T_i \oplus r)$, as all the random numbers and the corresponding hash computations can be done offline before the protocol execution. In rare cases, there might be few other tags near T_i which also crossed zone boundaries in round n. The possible collisions of these few transmissions can be easily resolved, e.g., using random backoff. For simplicity, our following discussion assumes one moving tag T_i. The readers in $\mathcal{R}_{i,n+1}$ then each forward T_i's response to the server for further processing which differs in the three protocols.

For Protocol 1, the server simply records T_i as a present one, and all the other operations remain. Note that T_i may be misidentified missing in the original Protocol 1 because Protocol 1 only considers empty slots. In particular, assume that tag T_i moved from zone $\mathcal{R}_{i,n} = \{l_1\}$ to $\mathcal{R}_{i,n+1} = \{l_2\}$ and that it has been identified absent in zone l_2 in previous rounds. According to Protocol 1, T_i is always considered absent in zone l_2 and will also be identified absent in all the other zones because T_i is indeed not there. As a result, T_i will be eventually misidentified missing. With our adaptation in place, the server can eliminate such false positives.

For Protocols 2 and 3, the server considers T_i present in zones $\mathcal{R}_{i,n}$ in the previous n rounds and suppressed afterwards. In addition, suppose that T_i should have replied in slot s_i in round n. Since T_i may cross the zone boundaries before or after slot s_i, it is difficult to tell which readers have received T_i response in round n. To deal with this situation, the server marks the s_ith slot in the nth observation vector from every reader in $\mathcal{R}_{i,n+1} \bigcup \mathcal{R}_{i,n}$ as *unusable* and will not use these slots in subsequent iterative identification. For this to work, we also let the server postpone the processing of the observation vectors collected in round n to the end of the polling phase $n+1$.

It is worth noting that since all our protocols can finish in very short time, the crossing-boundary events are rare if the monitored objects and thus their associate tags do not move too fast, which is generally true in practice. Even if a tag crosses the zone boundaries during the protocol execution, there will be only a few readers or zones involved which have very small impact on our protocols.

Another related issue is that some tags may be missing during the protocol execution and may not be identified by our protocols and also the protocols in [3]. Thanks to the very short execution time of all our protocols, these newly missing tags can be quickly identified in the next protocol execution.

E. Impact of Imperfect Wireless Channels

If the wireless channel is not error-free, there might be false positives and negatives in our protocols and also the protocols in [3] (see Section 4.6 of [3]). Since the readers can broadcast messages with high power and also use effective error-correction codes (ECCs) to ensure reliable transmission of their messages to the tags, we shall only discuss the impact of unreliable transmissions from the tags to the readers on our protocols and the necessary adaptations.

1) Impact on Protocol 1: In Protocol 1, the server exploits empty slots to detect missing tags. There are two cases related to transmission errors. First, high channel noise may cause some empty slots to be detected as non-empty ones. If this occurs, the corresponding tags cannot be immediately detected to be absent in the corresponding zones. The total identification time thus would be slightly increased, but all the missing tags can still be detected with overwhelming probability after a sufficient number of rounds. Second, the response from a tag may not be detected by the corresponding reader, in which case the tag will be misclassified absent in that zone. A simple countermeasure is to increase the threshold for identifying absent tags. In particular, a tag is considered absent in a zone if the reader observes multiple empty slots in which the tag should have replied if residing in that zone.

2) Impact on Protocols 2 and 3: Protocols 2 and 3 both require the readers to precisely distinguish empty, singleton, and collision slots. As in [3], we can take a few simple countermeasures to alleviate the impact of transmission errors. We propose an adaptive countermeasure to alleviate the impact of transmission errors. In particular, assume that the readers can perform online estimation of the channel condition, which is a routine operation available in many wireless devices. If the channel is considered unreliable, the reader instructs the tags in its coverage zone to enter an *error-avoidance* phase. During this phase, the tags need send longer responses protected by an efficient error-correction code (ECC), and a slot is considered singleton if the reader can receive a correct response passing the ECC check in that slot. In addition, the reader performs multiple scans with the same parameter $\langle r, f \rangle$ in the same round and determines the status of every slot using the majority rule. Once the channel condition becomes better, the reader instructs the tags to return to the regular phase. More illustrations about this adaptive countermeasure are omitted here due to the space limitation.

IV. Performance Evaluation

In this section, we evaluate the performance of our three protocols. In particular, we first analyze their computational complexity at the server side and execution time (i.e., missing-tag identification time). We then use simulations to compare our protocols with IIP, the best protocol in [3].

A. Analysis of Computational Complexity

In this section, we briefly analyze the computational complexity of our three protocols at the server side.

All three protocols require the server to decide the frame length f and also perform at most N hash computations for every round to decide the slot in which every tag should reply in that round. In all our protocols, f is chosen according to the process in Section III-A3 before the protocol execution. In addition, the server can choose the random numbers and perform all the hash computations before the protocol execution. Since such offline computations do not affect the overall missing-tag identification time, we shall ignore them hereafter. Also note that such offline computations of the hash values and frame length are also implicitly assumed in [3].

We first analyze the online computational cost of Protocol 1. Assume that Protocol 1 is completed in $\hat{n}_1$ rounds. In every round, for every tag T_i, the server need check whether slot $s_i = h(T_i \oplus r)$ is empty in every of the L zones, i.e., checking whether bit s_i is zero in every observation vector. This involves at most $\hat{n}_1 NL$ checking operations in total.

Now we analyze the online computational costs of Protocols 2 and 3. Assume that Protocol 2 is completed in $\hat{n}_2$ rounds. In every round, the server need check the status of every slot in every of the L newly received observation vectors, based on which to immediately identify more tags present or absent in every zone according to Cases 1 to 4. If we define the checking and resulting tag identification of every slot as a basic operation, this process involves totally $\hat{n}_2 fL$ basic operations because an observation vector corresponds to f slots. In addition, every tag newly identified present or absent in every zone will trigger the reprocessing of the observation vectors in at most $\hat{n}_2 - 1$ other rounds due to the iterative identification process. Since every tag can be identified either present or absent only once in every zone, it can trigger at most $L(\hat{n}_2 - 1)$ basic operations. Therefore, we can upper-bound the online computation cost of Protocol 2 by $\hat{n}_2 fL + NL(\hat{n}_2 - 1)$ basic operations. Since Protocol 3 is based on Protocol 2, its online computational cost can be similarly upper-bounded by $\hat{n}_3 fL + NL(\hat{n}_3 - 1)$ basic operations, where $\hat{n}_3$ denotes the total number of rounds needed in Protocol 3.

B. Analysis of Protocol Execution Time

For fair comparison with IIP, we adopt the same simulation parameters in [3] to directly use their results. In particular, it takes $t_s = 0.8$ ms and $t_{tag} = 2.4$ ms for a tag to transmit a multi-bit response and its 96-bit ID, respectively [3], based on the specification of the Philips I-Code system [17]. This gives an approximate transmission rate of $96/(2.4 * 10^{-3}) = 40$ kb/s. As in [3], we assume that the reader transmits to the tags at the same rate.

For both our protocols and IIP, the protocol execution time comprises the time incurred by reader-server communications, the time caused by reader-tag communications (called *total scan time*), and the total computation time at the server. Since the server communicates with the readers over high-speed links, the first part is normally negligible in comparison with the other two parts and will be ignored hereafter for both our protocols and IIP. Since the respective total computation time of all our protocols has been analyzed in Section IV-A, we only discuss their respective total scan time in what follows.

TABLE I
SCAN-TIME ANALYSIS

Protocols	**Total Scan Time** (in ms)
Protocol 1	$(0.8f + 2.72(0.25\kappa_1 + 2)\kappa_2)\hat{n}_1$
Protocol 2	$(0.8f + 2.72(0.25\kappa_1 + 2)\kappa_2)\hat{n}_2$
Protocol 3	$(0.8f + 2.72(0.25\kappa_1 + 2)\kappa_2)\hat{n}_3 + 0.025fn_s$
IIP	$0.86N$

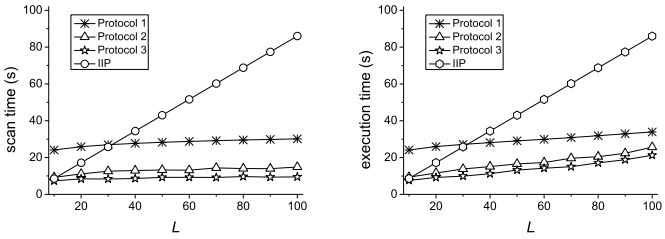
In all our protocols, the scan time in every round includes the time for every reader to broadcast $\langle r, f \rangle$, a short polling duration to deal with tag mobility, and f slots (i.e., $0.8f$ ms) to collect tag responses. The first part is negligible compared to the other parts. For example, if r and f are of 64 and 16 bits, respectively, it only takes 2 ms to transmit $\langle r, f \rangle$. If we assume a grid deployment of RFID readers as in Fig. 2 and use the scheduling method mentioned in Section III-A2 to avoid collisions, it takes about 8 ms to broadcast $\langle r, f \rangle$. Since the first part is the same for all our protocols and also IIP, we shall ignore it as in [3].

To derive the length of the polling phase, we assume that a reader ID is of 10 bits, which can accommodate up to 1024 readers. It then takes about 0.25 ms to broadcast the reader ID in the polling phase, which is considered negligible. Assuming that every tag can receive the polling request from at most κ_1 readers simultaneously, a slot number is of 16 bits, and a hash value is of 64 bits, a polling response (including the tag's slot number and one hash value in the first scanning round and reader IDs) is thus of $10\kappa_1 + 80$ bits and takes $0.25\kappa_1 + 2$ ms to transmit. Also assuming that there are at most κ_2 tags competing to report in the same polling phase, it takes $2.72(0.25\kappa_1 + 2)\kappa_2$ ms to collect their responses using ALOHA-based protocols such as EDFSA [8]. Note that the polling phase is a synchronized process in all L zones and thus incurs about $2.72(0.25\kappa_1+2)\kappa_2$ ms. We expect κ_1 and κ_2 to be very small in practice. Finally, every round in Protocol 3 may involve broadcasting an f-bit suppression command, which corresponds to $0.025f$ ms.

Table I summarizes the total scan time of all three protocols, where Protocols 1 to 3 are assumed to finish in $\hat{n}_1$, $\hat{n}_2$, and $\hat{n}_3$ rounds, respectively, and n_s denotes the total number of suppression operations in Protocol 3. In addition, the total scan time $0.86N$ of IIP is directly obtained from [3].

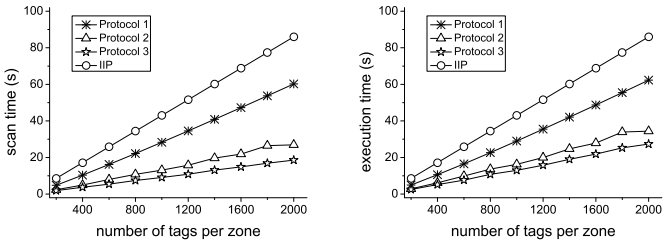
C. Simulation Results

We simulated a region consisting of square zones as shown in Fig. 2, where a reader is in the center of every zone and covers the entire zone. The simulations were done on a Dell XPS Studio 9100 desktop with an Intel i7 920 CPU and 9G RAM. Since the computation cost of IIP is not given or evaluated in [3], we shall only consider its total identification time $0.86N$ in all comparisons, which is clearly in favor of IIP. As in [3], we did not simulate transmission errors or tag



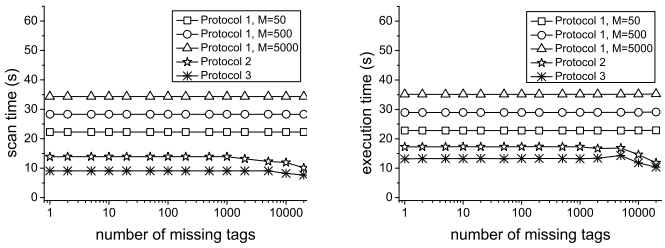
(a) Total scan time (b) Protocol execution time

Fig. 3. Impact of L, the number of zones.



(a) Total scan time (b) Protocol execution time

Fig. 4. Impact of tag density.



(a) Total scan time (b) Protocol execution time

Fig. 5. Impact of the number of missing tags.

mobility for simplicity and lack of space (we set $\kappa_2 = 0$ in Table I).

Unless specified otherwise, the following simulation settings were used by default. We simulated 50,000 tags uniformly distributed in a region of 5×10 square zones, among which 50 are missing. In addition, we set $\alpha = 0.99$ and $M = 500$ (the expected maximum number of missing tags) for Protocol 1 and the suppression hold $\delta = 0.5$ for Protocol 3. We also fixed the tag density to 1000 tags/zone and the missing-tag ratio to 0.001.

1) The impact of L: Fig. 3 shows the impact of L on Protocols 1∼3 and IIP. Since the tag density is fixed, increasing L is equivalent to increasing the total number of tags in the system. We can see from Fig. 3(a) that the total scan time of IIP is shorter than that of Protocols 1∼3 when L is small and increases linearly as L increases. In contrast, the total scan time of Protocols 1∼3 is much shorter than that of IIP when L is relatively large, say $L > 20$ (i.e., more than 20,000 tags) and is insensitive to L. This result is not surprising, as IIP treats the whole region as a single collision domain and has a scan time proportional to the total number of tags, while all our protocols treat every zone as a collision domain and perform synchronized and parallel scans in the L zones. In other words, our protocols depend on the number of tags in every zone instead of the total number of tags. In addition, Fig. 3(b) shows the protocol execution time (i.e., total missing-tag identification time) varying with L, which includes the total scan time and also the computation time for all our protocols. We can see that, even if the computation time of IIP was not considered in its favor, all our protocols could still detect the missing tags in much shorter time. For example, when $L = 100$, Protocols 1∼3 can reduce the missing-tag identification time by 60.5%, 70.15%, and 75.26%, respectively, in comparison with IIP.

TABLE II
MISSING-TAG DETECTION RATE OF PROTOCOL 1

	Identification Rate		
# of missing tags	$M = 50$	$M = 500$	$M = 5000$
1	100%	100%	100%
10	100%	100%	100%
100	99.4%	99.9%	100%
1000	99.43%	99.95%	100%
10000	99.93%	99.99%	99.99%

2) The impact of tag density: Fig. 4 shows the impact of tag density on Protocols 1∼3 and IIP. Since the tag density determines the size of the collision domain (either every zone in Protocols 1∼3 or the whole network in IIP), it is of no surprise to see that the total scan and execution times of all protocols increase almost linearly with the tag density. In all simulated cases, however, all our protocols still significantly outperform IIP, which is as expected.

3) The impact of number of missing tags: Fig. 5 compares our three protocols when the number of missing tags changes. Since all our protocols still significantly outperformed IIP in the simulated cases, the results for IIP are not shown in Fig. 5 for simplicity. We can see that the total scan time of Protocols 2 and 3 are relatively stable when the number of missing tags increases from 1 to 1000 and decreases as the number of missing tags further increases. This is as expected because Protocols 2 and 3 both detect present tags in every zone: the more missing tags, the fewer present tags per zone, and thus the shorter the total scan and execution times for both protocols. In contrast, Protocols 1's scan and execution times are not affected by the actual number of missing tags, but by the estimated maximum number of missing tags, i.e., M. Intuitively, the larger M, the longer the total scan and execution times, and vice versa, as observed in Fig. 5. One may think about conservatively estimating M to reduce the execution time of Protocol 1. Protocol 1, however, cannot detect all the missing tags when there is a significant underestimation of M, as shown in Table II. Therefore, there is a tradeoff between the missing-tag identification rate and the execution time for Protocol 1. In contrast, Protocols 2 and 3 both can always detect all the missing tags.

4) The progress of missing-tag detection: Fig. 6 sheds more light on the behavior of Protocols 1∼3. Under the default

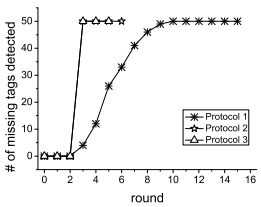


Fig. 6. Missing-tag detection progress.

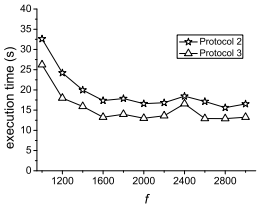


Fig. 7. Impact of frame length f.

settings with 50 missing tags, we can see that the number of missing tags detected by Protocol 1 gradually increases. In addition, although Protocol 1 has detected all 50 missing tags after 10 rounds, it still need run 6 more rounds to reach a confidence level of α. In contrast, Protocols 2 and 3 both detect all 50 missing tags after 3 rounds and stop after 6 and 5 rounds, respectively, until identifying all the present tags.

5) The impact of frame length f on Protocols 2 and 3: In our previous results, the frame length f used for Protocols 2 and 3 is the same as the optimal one derived for Protocol 1. Fig. 7 shows the impact of different f on Protocols 2 and 3 under the default settings, based on which the optimal frame length for Protocol 1 is 2211. As we can see, the execution time of Protocols 2 and 3 is insensitive to f as long as f is sufficiently large. The reason is that, the larger f, the fewer scan rounds needed, and vice versa, while the scan time is determined by the product of f and the number of scan rounds. Therefore, although it is difficult (if not impossible) to derive the optimal frame length for Protocols 2 and 3, it is safe to just use the optimal frame length derived for Protocol 1.

To sum up, all our protocols can quickly identify all the missing RFID tags in a large-scale RFID system and outperform the existing solutions by a significant margin. Among our protocols, Protocol 3 is the best choice.

V. Other Related Work

Besides [1]–[3], there is some other work loosely related to ours in this paper. Most previous work on RFID systems can be classified into two categories based on how they resolve the collisions among multiple RFID tags transmitting to the same RFID reader. The first category [18]–[21] involves a tree-based anti-collision protocol, in which one binary tree is formed by assigning every tag ID to a unique leaf node and labeling every non-leaf node by the common prefix of its two children. The second category [1]–[3], [5]–[16] employs a framed slotted ALOHA protocol which will be introduced in detail in Section II-B. Our work in this paper belongs to the second category. In [10], [11], [13], a few probabilistic models were presented to fast estimate the number of tags in large-scale RFID systems. Some probabilistic algorithms were proposed in [12] to efficiently find popular categories in RFID systems. In [14], multiple readers are scheduled for scanning to bound the number of unread tags. In [15], a probabilistic model was proposed to identify moving tags. These schemes are not directly applicable to missing-tag identification.

VI. Conclusion

In this paper, we presented three novel protocols to quickly identify the missing tags in a large-scale RFID system. Simulation results confirmed the significant improvement of all our protocols over the state of art [3].

Acknowledgement

This work was supported in part by the US National Science Foundation under grants CNS-1122697 and CNS-0844972 (CAREER). We would also like to thank anonymous reviewers for their constructive comments and helpful advice.

References

[1] C. Tan, B. Sheng, and Q. Li, "How to monitor for missing RFID tags," in *ICDCS'08*, Beijing, China, Jun. 2008, pp. 295–302.

[2] ——, "Efficient techniques for monitoring missing RFID tags," *IEEE Trans. Wireless Commun.*, vol. 9, no. 6, pp. 1882–1889, June 2010.

[3] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large RFID system," in *ACM Mobihoc'10*, Chicago, IL, Sep. 2010.

[4] C. Zhang, Y. Zhang, and Y. Fang, "A coverage inference protocol for wireless sensor networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 6, pp. 850–864, June 2010.

[5] H. Vogt, "Efficient object identification with passive RFID tags," in *IEEE Pervasive'02*, Zurich, Switzerland, Aug. 2002, pp. 98–113.

[6] B. Zhen, M. Kobayashi, and M. Shimizu, "Framed aloha for multiple RFID objects identification," *IEICE Transactions on Communications*, vol. E88-B, no. 3, pp. 991–999, Mar. 2005.

[7] J. Cha and J. Kim, "Novel anti-collision algorithms for fast object identification in RFID system," in *ICPADS'05*, Fukuoka, Japan, Jul. 2005, pp. 63–67.

[8] S. Lee, S. Joo, and C. Lee, "An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification," in *IEEE MobiQuitous'05*, San Diego, CA, Jul. 2005, pp. 166–174.

[9] J. Zhai and G. Wang, "An anti-collision algorithm using two-functioned estimation for RFID tags," in *ICCSA'05*, Singapore, May 2005.

[10] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems," in *ACM MOBICOM'06*, Los Angeles, CA, Sep. 2006, pp. 322–333.

[11] C. Qian, H. Ngan, and Y. Liu, "Cardinality estimation for large-scale RFID systems," in *IEEE PerCom'08*, Hong Kong, China, Mar. 2008.

[12] B. Sheng, C. Tan, Q. Li, and W. Mao, "Finding popular categories for RFID tags," in *ACM Mobihoc'08*, Hong Kong, China, May 2008, pp. 159–168.

[13] H. Han, B. Sheng, C. Tan, Q. Li, W. Mao, and S. Lu, "Counting RFID tags efficiently and anonymously," in *INFOCOM'10*, San Diego, CA, Mar. 2010.

[14] S. Tang, J. Yuan, X. Li, G. Chen, Y. Liu, and J. Zhao, "Raspberry: A stable reader activation scheduling protocol in multi-reader RFID systems," in *IEEE ICNP'09*, Princeton, NJ, Oct. 2009, pp. 304–313.

[15] L. Xie, B. Sheng, C. Tan, H. Han, Q. Li, and D. Chen, "Efficient tag identification in mobile RFID systems," in *INFOCOM'10*, San Diego, CA, Mar. 2010.

[16] L. Yang, J. Han, Y. Qi, and Y. Liu, "Identification-free batch authentication for RFID tags," in *ICNP'10*, Kyoto, Japan, Oct. 2010.

[17] P. Semiconductors, "I-CODE UID smart label IC functional specification," available at http://www.nxp.com/documents/data_sheet/SL092030.pdf, Jan. 2004.

[18] D. Hush and C. Wood, "Analysis of tree algorithm for RFID arbitration," in *IEEE ISIT'98*, Cambridge, MA, Aug. 1998, pp. 107–107.

[19] C. Law, K. Lee, and K. Siu, "Efficient memoryless protocol for tag identification," in *DIAL-M'00*, Boston, MA, Aug. 2000, pp. 75–84.

[20] F. Zhou, C. Chen, D. Jin, C. Huang, and H. Min, "Evaluating and optimizing power consumption of anti-collision protocols for applications in RFID systems," in *ISLPED'04*, Newport, CA, Aug. 2004, pp. 357–362.

[21] J. Myung and W. Lee, "Adaptive splitting protocols for RFID tag collision arbitration," in *ACM Mobihoc'06*, Florence, Italy, May 2006, pp. 202–213.