On Fast and Reliable Missing Event Detection Protocol for Multitagged RFID Systems

Hao Liu, Rongrong Zhang, Lin Chen, *Member, IEEE*, Jihong Yu, *Member, IEEE*, Jiangchuan Liu, *Fellow, IEEE*, and Jianping An, *Member, IEEE*

***Abstract*—With the rapid development of radio-frequency identification (RFID) technology, the ever-increasing research effort has been dedicated to devising various RFID-enabled services. The missing event detection, the functionality of detecting missing objects, is one of the most important services in many Internet-of-Things applications such as inventory management. Prior detection protocols only work in single-tagged RFID systems and would waste much time on repeated checks on one object in the emerging multitagged systems where each object is attached by multiple tags, leaving efficient detection in the new scenario unaddressed. To bridge the gap, this article is devoted to detecting missing multitagged objects. The key technicality is to build a filter from a subset of tags instead of whole in prior works to avoid repeated detections of one object and reduce detection time. Specifically, we first provide a basic solution based on the Bloom filter which can specify only tags in the chosen subset to participate in the final detection. To further improve time efficiency, we propose an advanced protocol that exploits tag ID knowledge and sparsity of slots mapped by only tags in the chosen subset to build a more compact compressive filter. Moreover, a composite vector is used to efficiently coordinate tags to report its presence. We conduct theoretical analysis on optimum protocol parameters and extensive simulations to verify the feasibility of the protocols. The results show that the advanced protocol achieves more than 2× performance gain in terms of time efficiency over the Bloom filter-based basic protocol.**

***Index Terms*—Missing event detection, multitagged object, radio-frequency identification (RFID).**

I. Introduction

RADIO-FREQUENCY identification (RFID) technology plays a crucial role in the deployment of Internet of Things in various applications, such as inventory control [1], [2], supply chain management [3]–[5], and objects tracking [6] and locating [7]. An RFID system is composed of one/multiple readers and a large number of tags. Readers can query tags wirelessly. Each tag has a unique ID and can capture energy in the RF signal of a reader for computation and send message via backscatter communications [8].

Fast and reliable missing event detection is of practical importance in many RFID-enabled applications. According to the statistics, inventory shrinkage, a combination of shoplifting, internal theft, and paperwork error, resulted in $44 billion in loss for U.S. retailers in 2014 [9] and is costing U.K. retailers almost $13.4 billion annually [10]. In this context, the RFID-based item-level monitoring can help retailers from economic losses due to missing objects.

This article focuses on a variation on missing event detection problem different from prior works, motivated by the emerging deployment of multitagged RFID systems where each object in the coverage is attached with multiple tags. Attaching multiple tags on an object has advantages of enhanced security [11], [12], and accurate object state sensing [13]–[15]. This, however, brings a new challenge of repeated detection of a multitagged object in an enlarged system to fast and reliable missing event detection.

The prior works [16]–[25] are not designed for multitagged RFID systems and suffer from low time efficiency. The core reason lies in the potential detection of all tags in the system, degrading time efficiency from two aspects. First, the existing approaches do not differentiate the known tags in the system even when one of the tags on an object has been checked, wasting much time on repeated confirmation of the presence of an object. Second, there are severe interferences from responses of the Big tags on a checked present object to tags on the other objects. An alternative approach that avoids multiple checks on the presence of an object is selectively polling one tag on each object. Yet, this approach has to query each tag with a tedious 96-b ID, which is time consuming for large-scale systems. Therefore, how to efficiently detect a missing event in multitagged RFID systems is still an open question.

In this article, we devote the first formulation and study on the missing event detection problem in multitagged RFID systems. As analyzed above, the key guideline on the protocol design is to query a subset of tags instead of whole in the prior works. Our idea is to divide a protocol into two phases: 1) marking phase and 2) detection phase. The reader first arbitrarily chooses one tag from each object and exploits

Manuscript received January 10, 2020; revised March 16, 2020; accepted April 1, 2020. Date of publication April 8, 2020; date of current version October 9, 2020. This work was supported in part by the NSF of China under Grant 61901035 and Grant 61801064, in part by the Beijing Institute of Technology Research Fund Program for Young Scholars, and in part by the Young Elite Scientist Sponsorship Program by CAST and Chongqing Key Laboratory of Mobile Communications Technology. The work of Rongrong Zhang was supported by the Science and Technology Project of Beijing Municipal Education Commission under Grant KM202010028005. *(Corresponding author: Jihong Yu.)*

Hao Liu, Jihong Yu, and Jianping An are with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081, China (e-mail: 3120150371@bit.edu.cn; jihong.yu@bit.edu.cn; an@bit.edu.cn).

Rongrong Zhang is with the Information Engineering College, Capital Normal University, Beijing 100089, China (e-mail: zhangrr@cnu.edu.cn).

Lin Chen is with the School of Computer Science and Technology, Sun Yat-sen University, Guangzhou 510275, China (e-mail: chen@lri.fr).

Jiangchuan Liu is with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: jcliu@sfu.ca).

Digital Object Identifier 10.1109/JIOT.2020.2986352

their mappings to design a filter. The filter is able to mark the chosen tags to ask them for further detection in the second phase while sifting out and suppressing the remaining tags. The reader then interrogates the marked tags and detects missing event from their responses. Following this idea, we propose two concrete two-phase detection protocols, namely, basic protocol and advanced protocol. The main contributions of this article are articulated as follows.

1) We provide an efficient solution to the missing event detection problem in multitagged RFID systems, named basic protocol. In the first phase, we leverage Bloom filter to represent the chosen tags so that they can pass the membership test while the others are sifted out. A virtual Bloom filter is constructed from responses of the tags in the second phase, enabling missing tag detection.
2) We design an advanced protocol that is more time efficient. Exploiting properties of full knowledge on tags' IDs and sparsity of slots mapped by the chosen tags compared with the others, we propose a compressive filter that only needs one hashing operation for a tag but can achieve better marking efficiency than the Bloom filter. A composite vector built from multiple mappings of the marked tags is then used for the detection.
3) We investigate the performance of the proposed protocols both theoretically and experimentally. We derive optimum parameters used in the protocols which minimize communication overhead under the constraint of required detection reliability. On the other hand, extensive simulation results verify the effectiveness of both protocols on missing event detection, and show that the advanced protocol achieves a time efficiency gain of at least $2\times$ over the Bloom filter-based basic one.

II. Related Work

Missing tag detection plays a crucial role in RFID-enabled applications since it could monitor state (normal or broken) of tags and fast detect illegal movement of objects in the work region, such as misplacement and theft. The works on missing tag detection could be separated into two categories: 1) probabilistic [16]–[22] or 2) deterministic protocols [17]–[19].

Probabilistic protocols detect a missing tag event with a predefined probability. Tan *et al.* [16] initiated the study of probabilistic detection and proposed a solution called trusted reader protocol (TRP). TRP detects a missing tag event by comparing the precomputed slots with those picked by the tags in the population. If an expected singleton slot turns out to be an empty slot, then the missing event is detected. Follow-up works [20], [21] employ multiple seeds to increase the probability of the singleton slot, which reduces the useless empty and collision slots and thus achieves a better performance. RUN [22] and BMTD [23] are proposed to address the influence of unknown tags. Yu *et al.* [24] designed a suit of detection protocols for multicategories and multiregion RFID systems and studied how to detect missing tags by using COTS RFID devices [25].

Deterministic protocols, on the other hand, are able to exactly identify which tags are absent. Li *et al.* [17] developed a series of deterministic protocols to reduce the radio collision by reconciling collision slots and finally iron out a bit-level tag identification method by iteratively deactivating the tags of which the presence has been verified. Subsequently, Zhang *et al.* [18] proposed identification protocols which store and compare the bitmap of tag responses in all rounds and observe the change among the corresponding bits among all bitmaps to determine the present and absent tags. But how to configure the protocol parameters is not theoretically analyzed. More recently, Liu *et al.* [19] enhanced the work by reconciling both 2-collision and 3-collision slots and filtering the empty and unreconcilable collision slots to improve time efficiency.

We would like to emphasize that none of the prior works is designed to detect a missing event in a multitagged RFID system. In this scenario, all existed missing tag detection protocols cannot work effectively, because they have to detect all tags whose IDs are recorded in the reader, wasting too much time. In contrast, this article chooses a subset of these tags for detection, avoiding repeated checks of one object and their interferences to the other tags. Moreover, this article exploits tag knowability and slot sparsity jointly to improve time efficiency, which completely differs this article from the existing ones.

III. System Model and Problem Formulation

A. System Model

We consider an RFID system of one reader[1] and a large number of tags where each physical object is attached by multiple tags [11], [15]. The reader is connected via high-speed channels with a back-end server of powerful computing capability. We regard the server and the reader as a single entity called *the reader* for simplicity [27], [28]. Generally, each tag has a unique ID and user-defined memory to achieve storage of the user-defined data while capable of performing certain computations such as hashing functions. Moreover, we assume that the reader has the IDs of all tags in the system.

The downlink (i.e., reader-to-tags) and uplink (i.e., tags-to-reader) communications experience different slot duration: 1) 96-b downlink slot duration from the reader to tags and 2) 1-b response slot from tags to the reader. We denote T_{id} and T_{short} as the length of a downlink slot and response slot, respectively. For an arbitrate response slot, there are three types of slot states: 1) if no tag relies on this slot, it is called an empty slot; 2) if a single tag replies, it is called a singleton slot; and 3) if multiple tags respond simultaneously, it is called a collision slot. The latter two states are referred to as a nonempty slot.

B. Problem Formulation

In this article, we are interested in detecting missing object events in a multitagged RFID system where n tags monitoring

[1]For multiple readers, we can treat them as a single virtual reader as in [24] and [26]. Specifically, the back-end server calculates all the parameters and constructs the filter vectors and sends them to all readers such that the readers broadcast the same parameters and filters to the tags. Consequently, the back-end server can synchronize the readers and we can logically consider them as a whole.

g objects and each object is tagged by multiple tags, i.e., $g < n$. Let m_a denote the number of missing objects, a missing event denotes the event that m_a exceeds a threshold M_a. Let P_d define the probability that the reader can find a missing event, we formulate the optimum missing event detection problem as follows. The missing multitagged object detection problem is to devise an algorithm of minimum execution time to find missing event with probability $P_d \geq \alpha$ when $m_a \geq M_a$, where α is the required detection reliability. Given the required probability, the key performance metric is communication overhead between the reader and tags spent in completing the detection task. In this article, the communication overhead means the execution time.

We would like to emphasize the main difference between the problem in this article and those in the prior works. In our problem, one missing object leads to multiple tags absent from the interrogation of the reader. Instead, an object and its attached tag are injective in the prior work. This difference makes the algorithm design in this article completely different, which can be interpreted as follows. If a tag is absent from the interrogation of the reader, the corresponding attached object can be regarded as missing in the prior work. This, however, does not hold for the multitagged system here. In the new scenario, the reader learns a missing object only when all its attached tags are absent. If we still use the prior algorithms to deal with the new problem, all tags on an object would respond to the interrogation, leading to severe interference and thus considerably degrading time efficiency.

Take an example to explain the difference. Consider 10 000 objects, there will be then 10 000 tags detected by the reader in an injective RFID system. Yet, the number will soar to 30 000 in a multitagged system where each object is attached by three tags if the existing algorithms are used, sharply increasing communication overhead. This urges us to investigate the following problem: can we design detection algorithms that can achieve the required detection reliability by interrogating only part of the tags in the system? We shall answer this question later in this article with a comprehensive investigation. Table I summaries main notations used in this article.

TABLE I
MAIN PARAMETER NOTATION

Symbols	Description
$\mathcal{G}_{\mathcal{A}}$	Set of representative tags
$\mathcal{G}_{\mathcal{B}}$	Set of pending tags
n	Number of tags in our system
g	Number of objects in our work region
P_d	Achieved detection probability
α	Requirement of detection probability
m_a	Number of actual missing representative tags
M_a	Least number of missing representative tags to satisfy detection requirement
f_1	Length of filter in marking via bloom filter
k_1	Number of mapping in marking via bloom filter
P_{fp_1}	Probability of false positives in marking
f_2	Length of filter in detection via bloom filter
k_2	Number of mapping in detection via bloom filter
P_{fp_2}	Probability of false positives in detection
λ	Marking Efficiency in the advanced protocol
f_d	The response frame length in the second phase of the advanced protocol

C. Design Rational

Recall the missing multitagged object detection problem, an object is missing if all of its attached tags are absent, but the absence of one tag indicates the potential missing object. Consequently, it is adequate to first probe one of the tags on an object instead of all for missing object event detection. If the probed tag is present, the tagged object must still locate in the coverage of the RFID system and we do not need to interrogate the other tags on this object, which reduces communication overhead. Otherwise, we would further poll the Big tags on the object, and a missing object can be found if all of them are absent. Since the percentage of missing objects is usually small, the idea above can improve time efficiency.

Following the guideline, we randomly choose a tag from each object, which is referred to as *representative tag*. These g tags constitute the representative tag set defined as $\mathcal{G}_{\mathcal{A}} = \{\text{tag}_1, \text{tag}_2, \ldots, \text{tag}_g\}$ where tag_i is a tag on the object i for $1 \leq i \leq g$. The set of the remaining tags named *pending tags* is denoted by $\mathcal{G}_{\mathcal{B}}$. We then are interested in interrogating the representative tags to detect the potential missing object event. Yet, the pending tags in $\mathcal{G}_{\mathcal{B}}$ would cause severe interference to representative tag detection. Therefore, an efficient scheme should be able to eliminate this negative impact.

In this article, we design two-phase protocols to address the problem.

1) *Marking Phase:* The task of phase 1 is to mark the representative tags for further detection while depressing the pending tags to abate their interference. The key to answering this question lies in designing a filter that is able to filter out the pending tags while ensuring all representative tags pass the test.
2) *Detecting Phase:* The reader then conduct missing object event detection in phase 2 by interrogating the remaining tags after the execution of phase 1. Therefore, we should ensure the efficiency of the two phases so that the overall time cost can be minimized. To this end, we propose two approaches. Note that a filter is an indicator vector with a certain number of elements each being either "0" or "1," and a position in an offline built filter corresponds to the slot in the same sequence of a frame during the online execution.

Basic Approach (Bloom Filter-Based Algorithm): The Bloom filter is a space-efficient probabilistic data structure for representing a set and supporting set membership queries. Its property can meet the design requirement analyzed above. Specifically, the reader first constructs a Bloom filter with the optimum parameters by encoding each tag in $\mathcal{G}_{\mathcal{A}}$, and transmits parameters and the filter to all tags. At the tag side, each tag uses the hash functions and the received parameters to map itself to several positions in the received filter. If all the value of these positions is "1," the tag knows it is a representative tag and will participate in the detection in phase 2. Otherwise, the tag is a pending tag and should turn to sleep and wait for the next activation command. This method is a direct application of a Bloom filter to achieve marking task. After the marking phase, the reader detects missing tags by constructing a virtual Bloom filter from the responses of the active tags. Since the reader can predict slot states, it can find a tag missing if there exist at least one of its mapped slots which are supposed to busy but turn out empty.

Advanced Approach (Compressive Filter-Based Algorithm): A Bloom filter can effectively complete the marking task, yet its performance is hindered by the tradeoff between filter length (i.e., frame size) and false positive ratio that tags in $\mathcal{G_B}$ are mistakenly marked with a certain probability. Specifically, reducing the false positive ratio is at the price of a longer filter. Especially, when $|\mathcal{G_B}|$ is considerably larger than $|\mathcal{G_A}|$, we should accordingly increase filter length to reduce the false positive ratio, and a higher false positive ratio leads to severe interference to the representative tags, otherwise.

To tackle the drawback of the basic approach, we develop a new filter that only needs one hash function rather than multiple ones in the Bloom filter but can achieve better performance. First, the reader employs one hash function to construct a filter where all positions are initialized to "0" and only those mapped by tag(s) from $\mathcal{G_A}$ are set to "1." Such a filter can mark tags in $\mathcal{G_A}$ and asks them to participate in the second phase. Second, to reduce the time cost spent on the filter transmission, we explore the sparsity of "1" in the filter to compress its size.

Specifically, the elements "0" in the filter are in the majority, and its proportion increases with the filter size and the difference of $\mathcal{G_B}$ and $\mathcal{G_A}$. Moreover, the filter performs as a binary test, it is thus adequate to inform the tags of the positions of "1" in the filter. Motivated by these observations, we design such a compressive algorithm that consecutive zeros between any two "1" in the filter are replaced by a binary bit sequence of fixed size. It is required that the denary value of the bit sequence is equal to the number of the consecutive zeros, which can be used to indicate the positions of "1" in the original filter. Through the optimum parameter configuration, the compressive filter can be significantly compacter than the original one.

In the second phase, since a missing tag will be found when it is mapped to a singleton slot, we aim to improve the communication efficiency by changing nonsingleton slots into singleton slots. At the reader side, it first offline maps each representative tag independently via different seeds and builds a composite vector by picking all singleton slots from the multiple mapping. It then broadcasts parameters, including the vector, its size, and the seeds. At the tag side, each tag maps to one position of the vector using one seed and should respond if founding the mapping slot is a singleton. From the responses of tags, the reader can check whether a representative tag is missing and decides whether to poll the remaining tags in the corresponding object to verify its existence.

In what follows, we elaborate on the basic approach and the advanced one in subsequent.

IV. Basic Approach: Bloom Filter-Based Protocol

In the basic approach, the downlink and uplink Bloom filters are built in the two phase for missing event detection, respectively. In phase 1, the reader first constructs a Bloom filter to marking representative tags by encoding each tag in $\mathcal{G_A}$ according to the derived parameters and transmits the parameters and the constructed Bloom filter to tags. Tags conduct a membership test by checking the value of its mapping positions in the received filter. The detail of the method would be described as follows. In phase 2, the reader interrogates the remaining active tags with another suit of the derived parameters. Each tag should reply in its mapping slots, and a virtual Bloom filter can be constructed from the responses of all tags at the reader side for missing tag detection.

A. Protocol Description

The basic protocol consists of two phases: 1) marking phase and 2) detection phase, which is described as follows.

Marking Phase: In the beginning, the reader samples tags to participate in this process. To achieve sampling probability of p_1, the reader broadcasts parameters of length f_{sample}, seed s_{sample}, and threshold $\text{Th}_1 = \lceil p_1 f_{\text{sample}} \rceil$. Each tag hashes to $[0, f_{\text{sample}})$ with s_{sample}. If the result is smaller than Th_1, it will take part in this process, and keep sleep, otherwise.

The rest of the first phase can be executed in multiple rounds, which is decided by the parameter configuration to be discussed in Section IV-B. Recall that the objective of this phase is to filter out the pending tags in $\mathcal{G_B}$. We consider the ith round mark of $\mathcal{G_A}$, $1 \le i \le R_1$, where R_1 is the number of executed rounds. Let B_i be the number of the still active pending tags at the beginning of this round.

The reader offline constructs a f_1-b Bloom filter BV_i by mapping each tag ID in $\mathcal{G_A}$ to k_1 positions under seed s_i and set their value to "1." Then, the reader broadcasts the parameters and BV_i. Each unmarked tag uses the same parameters to map itself to k_1 positions as the reader dose. If the tag finds all the mapped k_1 bits in BV_i is ones, it passes the filter and waits for the detection in the second phase. Otherwise, it will keep sleep and cannot take part in the rest of the protocol. The Bloom filter has no false negative, i.e., tags in $\mathcal{G_A}$ must pass the test, but suffers from false positive. A tag in $\mathcal{G_B}$ may also pass the check. We denote by g_i the number of the tags filtered out in this round. After all R_1 rounds, there will be $B_{R_1} - g_{R_1}$ active tags in $\mathcal{G_B}$ which will access to the second phase.

Detection Phase: This phase aims to detect potential missing representative tags in $\mathcal{G_A}$ with the presence of $B_{R_1} - g_{R_1}$ active tags of $\mathcal{G_B}$. Similar to the first phase, the reader also first samples the remaining tags with sampling probability of p_2 and threshold $Th_2 = \lceil p_2 f_{sample} \rceil$. The rest of the second phase is executed in multiple rounds, which is decided by the parameter configuration to be discussed in Section IV-B.

Denote by R_2 the number of the rounds in this phase. Consider an arbitrate round i, different from the first phase, a Bloom filter will be built from the responses of the tags, which is used by the reader to check the existence of each tag. To this end, the reader broadcasts the parameters, including filter size f_2, the number of hush functions k_2, and seed s_2^*. Each tag then maps itself to k_2 slots and will reply in these slots. At the reader side, it can build a Bloom filter by setting positions corresponding to busy slots to "1." As the reader knows all IDs, it can predict every slot state and can thus detect a missing tag if there exists at least one "0" at its mapped k_2 positions.

Although there exists false positive and the interference of some pending tags, we could configure parameters used in the protocol so that the required detection reliability can meet within the minimum communication overhead. The analysis will be introduced in Section IV-B.

B. Parameter Optimization

The execution time of the basic protocol mainly consists of two parts: the communication cost in the marking phase and that spent on the detection.

1) We start with the analysis of the first part. The execution time of the marking phase could be expressed as

$$T_m = T_{m\_\text{ini}} + f_1 R_1 \frac{T_{\text{id}}}{96} \tag{1}$$

where T_{m_ini} is the constant time cost of the parameter transmission. The goal is thus to minimize $f_1 R_1 (T_{\text{id}}/96)$.

It is known that the false positives of the Bloom filter are

$$P_{fp_1} = \left[1 - \left(1 - \frac{1}{f_1}\right)^{k_1 A}\right]^{k_1} \approx \left(1 - e^{-\frac{k_1 A}{f_1}}\right)^{k_1} \tag{2}$$

where $A = A_{\text{orig}} p_1$ is the number of tags passing the sampling in $\mathcal{G}_{\mathcal{A}}$, k_1 is the number of hash functions, and f_1 is the length of the Bloom filter (i.e., frame size). Consider an arbitrary round, if the k_1 slots mapped by a tag in $\mathcal{G}_{\mathcal{B}}$ are same as those in $\mathcal{G}_{\mathcal{A}}$, then it cannot be filtered out in this round. The probability of this event is (2). Therefore, the probability that a tag in $\mathcal{G}_{\mathcal{B}}$ remains active after the marking phase can be written as

$$P_{fp_1}^{R_1} = \left(1 - e^{-\frac{k_1 A}{f_1}}\right)^{k_1 R_1} \tag{3}$$

where R_1 is the number of executing rounds.

We calculate the first order of differential function and obtain the minimum value of $P_{fp_1}^{R_1}$ is $(1/2)^{(f_1 R_1/A)\ln 2} \approx 0.6185^{(f_1 R_1/A)}$ when $k_1 = (f_1/A)\ln 2$. Therefore, the key is to minimizing $f_1 R_1$. Due to the fact that a smaller $f_1 R_1$ results in more active pending tags and more interferences to the detection phase, we thus jointly minimize the cost with the second phase.

2) We define the cost of execution time in the detection phase as T_d

$$T_d = T_{d\_ini} + f_2 R_2 T_{\text{short}}. \tag{4}$$

Similarly, we should minimize $f_2 R_2$ for time cost optimization with the constraint of the detection reliability. To this end, we first calculate the probability of false positives in the detection phase, which is expressed as follows:

$$P_{fp_2}^{R_2} = \left(1 - e^{-\frac{k_2 A'}{f_2}}\right)^{k_2 R_2} \tag{5}$$

where f_2 is the frame length, k_2 is the number of mappings (i.e., the number of hash functions) in a frame, and A' is the number of tags responding to the interrogation. Denote by A_r the number of the remaining tags after the first phase and m is the number of missing tag, then $A' = (A_r - m)p_2$. Similar to $P_{fp_1}^{R_1}$, we have the minimum $P_{fp_2}^{R_2}$

$$P_{fp_2}^{R_2} = 0.6185^{\frac{f_2 R_2}{A'}}. \tag{6}$$

We denote by P_d the probability that a missing event could be detected in $\mathcal{G}_{\mathcal{A}}$. As we should detect the missing event when $m_a \geq M_a$, P_d could be derived as

$$P_d = 1 - \left[1 - p_1 + p_1\left(1 - p_2 + p_2 P_{fp2}^{R_2}\right)\right]^{M_a} \tag{7}$$

where p_1 and p_2 are sampling ratios in the two phases, respectively. In order to meet the system requirement in detection, P_d should be greater than α, then we have

$$P_{fp_2}^{R_2} \leq \frac{\frac{(1-\alpha)^{\frac{1}{M_a}} + p_1 - 1}{p_1} + p_2 - 1}{p_2}. \tag{8}$$

It is required that

$$p_1 p_2 > 1 - (1 - \alpha)^{\frac{1}{M_a}}. \tag{9}$$

As it is adequate to set $P_d = \alpha$, we have

$$f_2 R_2 = \frac{A'}{-(\ln(2))^2} \times \left[\ln\left(\frac{(1-\alpha)^{\frac{1}{M_a}} + p_1 - 1}{p_1} + p_2 - 1\right) - \ln(p_2)\right]. \tag{10}$$

Recall that A' is the number of tags responding to the interrogation, including partial tags of $\mathcal{G}_{\mathcal{A}}$ and a few of $\mathcal{G}_{\mathcal{B}}$, as it is enough to find missing tags when $m_a \geq M_a$, we can rewrite A' for the parameter settings as

$$A' = \left(A + B P_{fp_1}^{R_1} - M_a\right) p_2 \tag{11}$$

where $B = B_1 = B_{orig} p_1$. Substituting (6) and (11) into (10), we have

$$f_2 R_2 = \frac{\left[\ln\left(\frac{(1-\alpha)^{\frac{1}{M_a}} + p_1 - 1}{p_1} + p_2 - 1\right) - \ln(p_2)\right]}{-(\ln(2))^2} \times \left(A + B 0.6185^{\frac{f_1 R_1}{A}} - M_a\right) p_2. \tag{12}$$

3) From (12), we can observe that T_d increases with the decrease of $f_1 R_1$ that is determined by the first phase. Define the overall time cost of the basic protocol as T_{whole}, we have

$$T_{\text{whole}} = T_{m\_\text{ini}} + T_{d\_\text{ini}} + f_1 R_1 \frac{T_{\text{id}}}{96} + f_2 R_2 T_{\text{short}}. \tag{13}$$

Since T_{g_ini} and T_{d_ini} are constants and too small compared with the other parts. Hence, we ignore them in the subsequent optimization. The overall cost is simplified as

$$\begin{aligned}\hat{T} &= f_1 R_1 \frac{T_{\text{id}}}{96} + f_2 R_2 T_{\text{short}} \\ &= \frac{\left[\ln\left(\frac{(1-\alpha)^{\frac{1}{M_a}} + p_1 - 1}{p_1} + p_2 - 1\right) - \ln(p_2)\right]}{-(\ln(2))^2} T_{\text{short}} \\ &\quad \times \left(A + B 0.6185^{\frac{f_1 R_1}{A}} - M_a\right) p_2 + \frac{T_{id}}{96} f_1 R_1.\end{aligned} \tag{14}$$

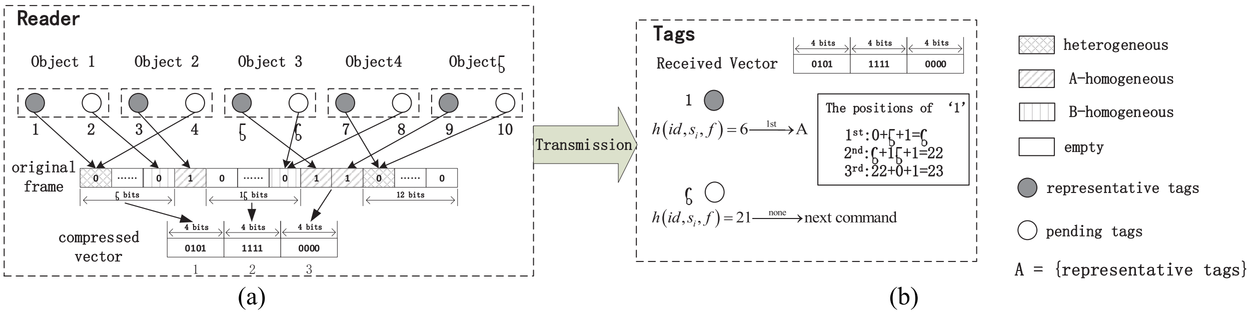


Fig. 1. Illustration of a compressive filter and the checking process at tag side.

Denote $u = f_1 R_1$, we derive the differential of $\hat{T}$ with u

$$\frac{\partial \hat{T}}{\partial u} = \left[\ln\left(\frac{(1-\alpha)^{\frac{1}{M_a}}+p_1-1}{p_1}+p_2-1\right)-\ln(p_2)\right]T_{short} \times \frac{Bp_2 0.6185^{\frac{u}{A}}}{A}+\frac{T_{\mathrm{id}}}{96}. \tag{15}$$

Let $(\partial \hat{T}/\partial u) = 0$, we could get the minimum overall when

$$u = -\frac{A}{(\ln 2)^2} \times \ln\left(\frac{-\frac{T_{\mathrm{id}}}{96}A}{T_{\mathrm{short}}Bp_2\left(\ln\left(\frac{(1-\alpha)^{\frac{1}{M_a}}+p_1-1}{p_1}+p_2-1\right)-\ln p_2\right)}\right). \tag{16}$$

Parameter Configuration: Given the sampling ratios p_1 and p_2 meeting (9), the value of f_1 and R_1 can be chosen so that (16) holds. Once they are fixed, we can get f_2 and R_2 following (12). Finally, the optimal parameters can be configured for the basic protocol.

V. Advanced Approach: Compressive Filter-Based Protocol

The Bloom filter can effectively complete the marking task, yet its performance is hindered by the tradeoff between filter length (i.e., frame size) and the false positive ratio that tags in $\mathcal{G_B}$ are mistakenly marked with a certain probability. Specifically, reducing the false positive ratio is at the price of a longer filter. Especially, when $|\mathcal{G_B}|$ is considerably larger than $|\mathcal{G_A}|$, we should accordingly increase filter length to reduce the false positive ratio, and a higher false positive ratio leads to severe interference to the representative tags, otherwise.

To tackle the drawback of the basic protocol, we develop an advanced protocol containing a new filter for the marking phase that only needs one hash function rather than multiple in Bloom filter but can achieve better marking performance, and a composite filter picking all singleton slots from multiple mappings. The improvement in the first phase results from two aspects: 1) the knowledge on IDs of all tags and 2) the sparsity of the original vector. The first one enables the reader to encode the mappings of both representative and pending tags instead of only the former in the basic protocol, making the filter more informative. The second one makes compression of the filter possible reducing communication cost.

A. Protocol Description

The advanced protocol also consists of two phases: 1) marking phase and 2) detection phase. In the first phase, we use one hash function to encode mappings of all tags and exploit the sparsity of "1" to build a compressive filter to mark representative tags. In the second phase, we pick singleton slots from multiple random mappings of a tag to build a composite filter informing a remaining active tag after the first phase of its response slot and conduct the detection. Note that the position of a filter and a slot of a frame is injective.

1) Marking Phase: In the marking phase, the reader first samples the tags with a ratio of p_1. Then, the marking phase works in multiple rounds. Consider an arbitrary round i, the reader offline employs one hash function to encode all unmarked tags to an f_i-b vector where all positions are initialized to "0." Since the reader knows IDs of all tags, it can predict A-homogeneous positions that are mapped only by tag(s) of $\mathcal{G_A}$, B-homogeneous positions that are mapped only by tag(s) of $\mathcal{G_B}$, heterogeneous positions that are mapped by tags of $\mathcal{G_A}$ and $\mathcal{G_B}$, and empty positions. Consequently, the reader only sets the A-homogeneous positions of the vector to "1" instead of both homogeneous and heterogeneous positions in the basic protocol. Note that a position in an offline built vector corresponds to the slot in the same sequence of a frame during the online execution.

Let us take Fig. 1(a) as a toy example. The first position is heterogeneous because it is mapped by tag 1 in set $\mathcal{G_A}$ and tag 4 in set $\mathcal{G_B}$. The 21st position is also set to 0 since it is a B-homogeneous position mapped by tag 6 and tag 8 of set $\mathcal{G_B}$. On the contrary, the 6th and 22nd positions are A-homogeneous since they are mapped by tags in the group $\mathcal{G_A}$. Following the rule, we can build the original vector as "0000_0100_0000_0000_0000_0110_0000_0000_000."

After the original vector is built, we start to compress it, which is motivated by the sparsity of "1" as shown in Fig. 1(a). We exploit the distance between two "1" to indicate the positions of "1" in the vector. Because the distance is usually short, the vector length can be significantly reduced. Specifically, the reader replaces each segment of consecutive zeros between "1" by the number of consecutive zeros in this segment. To this end, the reader first finds the longest segment of consecutive zeros in the original vector and records the length of zeros as $L_i^{\max}$. Second, each segment of consecutive zeros is converted to a binary sequence of $l_i = \lceil \log_2(L_i^{\max}+1) \rceil$ bits whose decimal value is equal to the number of consecutive zeros, and the compressive filter is finally constructed. If the compressive filter is longer than 96 b, the reader can divide it into parts and transmit each part in T_{id}.

For instance in Fig. 1(a), the longest segment of 15 zeros is converted to the number 15, which is compressed from 15 b to 4 b, and the other segments are also represented as 4-b sequences. Consequently, the reader can get a 12-b compressive filter compressed from the 35-b original vector. The compression ratio is $12/35 \approx 0.34$.

The reader then broadcasts parameters, including original vector size f_i, the segment size l_i and seed s_i. We will analyze how to set the parameters in Section V-B. The reader also sends the compressive filter to tags. At the tag side, after receiving the filter, it calculates the decimal value of each l_i-b segment starting from the head of the filter and outputs the same number of consecutive zeros. Repeat this for all segments, a tag can learn all positions of value "1" among $[1, f_j]$. It then can directly check from the compressed filter whether it is a representative tag. Specifically, the tag hashes itself to a slot among $[1, f_j]$. It then subtracts the sum shown in Fig. 1(b) from its hash value until the result is nonpositive. It can be marked as a representative tag if the result is zero. Otherwise, it waits for the following marking round. Note that it means two consecutive "1" that the decimal value of a compressed segment is 0. Moreover, the length of the reconstructed vector may be smaller than f_i because the consecutive zeros at the end of the original vector are omitted for saving time cost. The tag just needs to fill with several zeros at the end of the reconstructed vector to reach f_i. After multiround execution, all sampled representative tags can be marked and accessed to the detection phase, while the others keep silent.

Let us take Fig. 1(b) as an example to illustrate the decompression process at the tag side. From the received compressive filter, tag 3 can learn that there are five zeros until the first "1," matching with its mapping, it can thus be marked. In contrast, tag 6 mapped to the 21st slot finds the value at the 21st position of the original vector is "0," which can be inferred from 15 zeros between the first and second "1." It thus knows that it should keep silent in the rest of the protocol.

2) Detection Phase: In this phase, the reader first samples the tags marked in the first phase with a ratio of p_2. The reader then constructs a composite vector from multiple mappings of the sampled tags. Define the composite vector length as f_d and seed sequence $\{s_1, s_2, \ldots, s_l\}$. We will analyze how to set the parameters in Section V-B. The reader maps a tag to $H(\text{id}, s_j, f_d)$th position of the jth vector in the jth mapping where $1 \le j \le l$. After l mappings of all tags, the reader can obtain l vectors and uses them to composites a vector storing indexes of seeds that contribute to singleton slots. Specifically, the f_d-b composite vector is initialized to null. For each of its positions i, the reader picks a seed that makes one of the ith positions in the obtained l vectors singleton, for example, s_j, and sets the ith position in the composite vector to j. Repeating these operations for all f_d positions, the reader can obtain the expected composite vector.

After the offline construction of the composite vector, the reader broadcasts the vector length f_d, seed sequence $\{s_1, s_2, \ldots, s_l\}$, and the composite vector. The reader then sends another interrogation command to ask the qualified tag to respond, subsequently. At the tag side, for each slot, each tag uses a seed to map itself to a position of the vector and checks whether the sequence of the position in the vector is equal to the slot in the frame and whether the seed index in this position of the vector is equal to the seed used in this mapping. If both of them hold, the tag will respond in this slot. Otherwise, it uses another seed and repeats the above operations. At the reader side, the reader can compare the observed slot states with the predicted ones. It can detect a missing tag if a predicted singleton slot turns out to be empty.

B. Parameter Setting

We here introduce how to set parameters so that the detection reliability can meet and the communication cost can be minimized. To make the analysis feasible, we separately analyze the communication cost of the two phases.

1) Optimum Parameters for the Marking Phase: In an arbitrary round i of this phase, the objective is to maximize the marking efficiency λ_i. The ratio of the number ϕ_i of sampled representative tags in $\mathcal{G_A}$ marked in this round to the execution time t_i of this round. It implies that more tags can be marked in a unit time when λ increases. Let f_i^c define the compressive filter length in this round, we have

$$\lambda_i = \frac{\phi_i}{t_i} = \frac{\phi_i}{\frac{f_i^c}{96} T_{\text{id}}}. \tag{17}$$

As ϕ_i and f_i^c depend on f_i, the key is to find the optimum f_i.

Let n_i be the number of sampled representative tags unmarked at the beginning of the round, and when all sampled representative tags are marked after I rounds, n_I is equal to the number of sampled pending tags in $\mathcal{G_B}$ in this phase. Denote by ϕ_i' the number of sampled representative tags unmarked at the beginning of the round, we have

$$\begin{aligned} n_{i+1} &= n_i - \phi_i \\ \phi_{i+1}' &= \phi_i' - \phi_i. \end{aligned} \tag{18}$$

Since the protocol is probabilistic, we derive the expected value of ϕ_i, and the result is stated in the following lemma.

Lemma 1: Given the original vector size f_i at the ith round, the expected number of sampled representative tags marked in this round should be

$$\phi_i = \phi_i' \left(1 - \frac{1}{f_i}\right)^{n_i - \phi_i'}. \tag{19}$$

Proof: We first study the event that j sampled representative tags are mapped to a same slot. Its probability, defined as P_A^i, consists of three parts: 1) the probability of a arbitrary slot mapped by j tags which is $(1/f_i)^j(1-[1/f_i])^{n_i-j}$; 2) $\binom{n_i}{j}$ kinds of combination of j tags; and 3) the probability of j tags being representative tags which is equal to $\mathcal{G_A}$ is $\binom{\phi_i'}{j}/\binom{n_i}{j}$. It thus holds that

$$P_A^i = \binom{\phi_i'}{j}\left(\frac{1}{f_i}\right)^j\left(1 - \frac{1}{f_i}\right)^{n_i - j}. \tag{20}$$

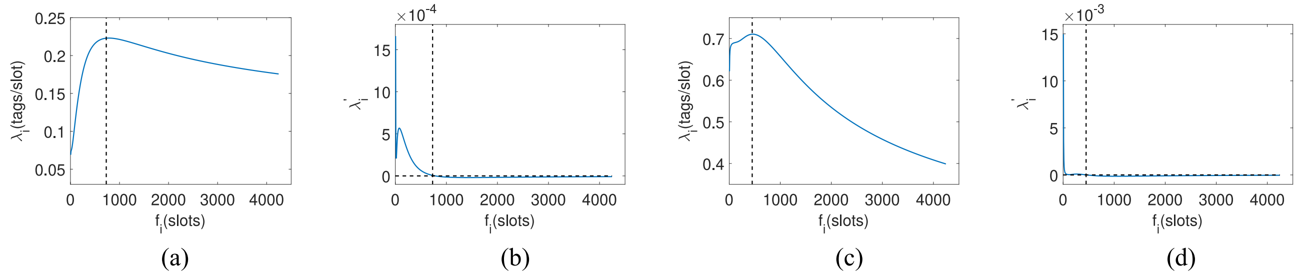


Fig. 2. Illustration of properties of marking phase λ_i versus original vector size f_i. (a) Marking efficiency when $n_i = 1000$ and $\phi_i' = 100$. (b) Derivation of marking efficiency when $n_i = 1000$ and $\phi_i' = 100$. (c) Marking efficiency when $n_i = 2000$ and $\phi_i' = 1000$. (d) Derivation of marking efficiency when $n_i = 2000$ and $\phi_i' = 1000$.

Hence, the expected number of sampled tags in group $\mathcal{G}_\mathcal{A}$ mapped to a slot is $\sum_{j=0}^{\phi_i'} j\binom{\phi_i'}{j}([1/f_i])^j(1-[1/f_i])^{n_i-j}$, and the number of the sampled tags in group $\mathcal{G}_\mathcal{A}$ marked by the compressive vector could be written as

$$\phi_i = f_i\sum_{j=0}^{\phi_i'} j\binom{\phi_i'}{j}\left(\frac{1}{f_i}\right)^j\left(1-\frac{1}{f_i}\right)^{n_i-j}.$$

After algebraic operations, the lemma can be proven. ■

From the construction of the compressive filter, we can find the following relation between f_i^c and f_i:

$$f_i^c = f_i\left(1-\frac{1}{f_i}\right)^{n_i-\phi_i'}\left(1-\left(1-\frac{1}{f_i}\right)^{\phi_i'}\right) \times \log_2\left(\frac{1}{\left(1-\frac{1}{f_i}\right)^{n_i-\phi_i'}\left(1-\left(1-\frac{1}{f_i}\right)^{\phi_i'}\right)}+1\right) \quad (21)$$

where the multiplicators at the two sides of the multiplication sign are the expected number of A-homogeneous positions and the average length of consecutive zeros in original vector, i.e., l_i, respectively. The relation among n_i, ϕ_i, and ϕ_i' also satisfies (18). Substituting (21) into (17), we can approximately have

$$\lambda_i = \frac{96}{T_{id}}\frac{\phi_i'}{f_i\left(1-\left(1-\frac{1}{f_i}\right)^{\phi_i'}\right)} \times \frac{1}{\log_2\left(\frac{1}{\left(1-\frac{1}{f_i}\right)^{n_i-\phi_i'}\left(1-\left(1-\frac{1}{f_i}\right)^{\phi_i'}\right)}+1\right)}. \quad (22)$$

To accelerate the mark phase, we should select an optimum f_i that maximizes the marking efficiency λ_i. To this end, we conduct theoretical analysis and provide a upper bound for the optimum f_i, which is stated in the following theorem.

Theorem 1: Given n_i and ϕ_i' that are known at the beginning of round i, the optimum f_i falls in $[1, ([n_i^2]/[n_i - 0.5\phi_i'])]$.

Proof: As it is unfeasible to directly derive optimum f_i from (22), we derive an upper bound of f_i and prove that λ_i is a decreasing function with respect to f_i when f_i exceeds this upper bound. As a result, the optimum f_i maximizing λ_i can be found between 1 and this upper bound.

Let $b = 1-(1-[1/f_i])^{\phi_i'}$. We can write

$$\frac{1}{\lambda_i} = \frac{T_{id}}{96\phi_i'}f_i b\log\left(1+\frac{1}{(1-b)^{\frac{n_i}{\phi_i'}-1}b}\right).$$

We can check that $(1/[(1-b)^{(n_i/\phi_i')-1}b])$ is decreasing in b for $0 \le b \le [(\phi_i')/n_i]$. Hence, $\log(1+(1/[(1-b)^{(n_i/\phi_i')-1}b]))$ is decreasing in b. Note that it easy to check that b also decreases with f_i, $\log(1+(1/[(1-b)^{(n_i/\phi_i')-1}b]))$ is thus increasing in f_i. On the other hand, regard $y = f_i b$ as a function of f_i, we can derive that

$$y' = 1-\left(\frac{f-1}{f}\right)^{\phi_i'}\left(1+\frac{\phi_i'}{f_i-1}\right) > 0. \quad (23)$$

Therefore, $(1/\lambda_i)$ is increasing in f_i when $0 \le b \le (\phi_i'/n_i)$. To establish the inequalities, f_i should satisfy that

$$f_i \ge \frac{1}{1-\left(1-\frac{\phi_i'}{n_i}\right)^{\frac{1}{\phi_i'}}}. \quad (24)$$

By applying the Taylor series $1-zx < (1-x)^z < 1-zx+0.5zx^2$, we have

$$\frac{n_i-0.5\phi_i'}{n_i^2} < 1-\left(1-\frac{\phi_i'}{n_i}\right)^{\frac{1}{\phi_i'}} < \frac{1}{n_i}.$$

Hence, it is adequate to guarantee that $(1/\lambda_i)$ is increasing in f_i for $f_i \ge [n_i^2/(n_i-0.5\phi_i')]$. Consequently, λ_i is decreasing when $f_i \ge [(n_i^2)/(n_i-0.5\phi_i')]$. It thus suffices to search f_i to find its optimum value until λ_i starts to decrease. The theorem follows from here. ■

To understand the properties of λ_i, we depict its numerical results with $(96/T_{\text{id}})$ omitted in Fig. 2 under diverse n_i and ϕ_i'. It can be observed that there exists an optimum f_i maximizing λ_i, which matches with the analysis stated in Theorem 1.

2) Optimum Parameters for the Detection Phase: The execution time in this phase is mainly spent on the composite vector transmission and the tags' responses. It is written as

$$T_d = \frac{f_d\lceil\log_2(l+1)\rceil}{96}T_{\text{id}} + f_d T_{\text{short}}. \quad (25)$$

Our goal is to minimize f with the constraint of the detection reliability requirement. We first derive the detection probability of our approach. Let n_A define the number of the representative tags marked in the first phase, and p_2 be the sampling

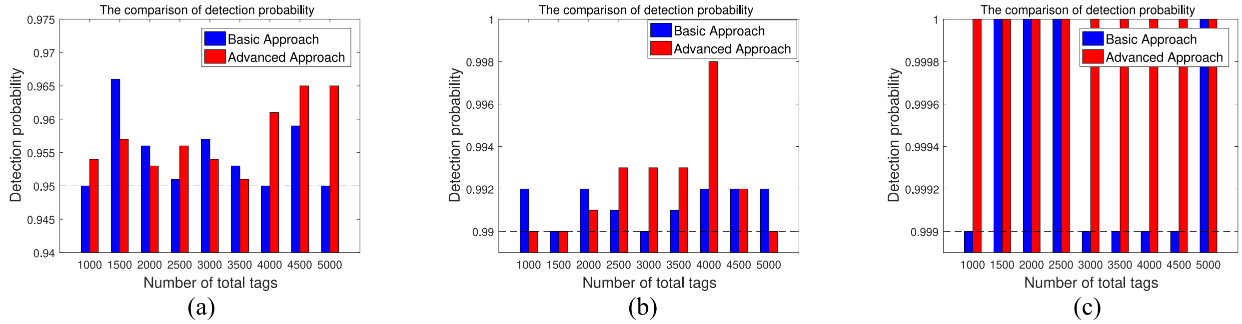


Fig. 3. Achieved detection probability with the number of total tags varied from 1000 to 5000 when the threshold of missing objects is $M_a = 2$ and the required detection probability is (a) $\alpha = 95\%$, (b) $\alpha = 99\%$, and (c) $\alpha = 99.9\%$.

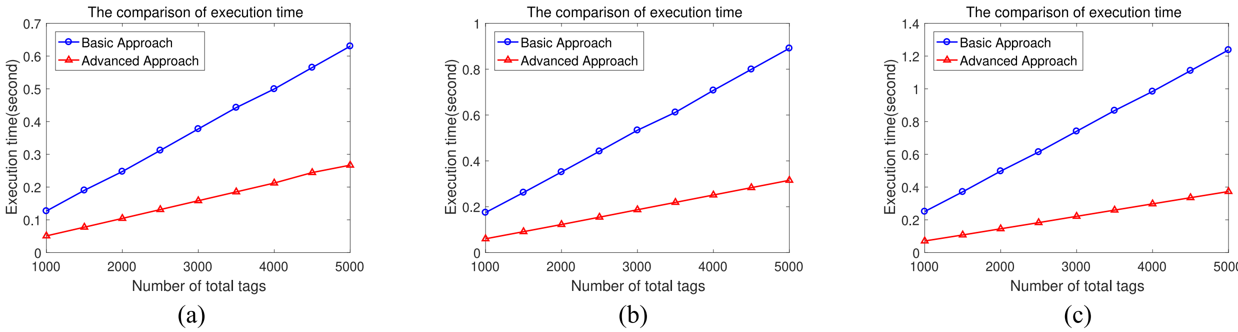


Fig. 4. Execution time with the number of total tags varied from 1000 to 5000 when the threshold of missing objects is $M_a = 2$ and the required detection reliability is (a) $\alpha = 95\%$, (b) $\alpha = 99\%$, and (c) $\alpha = 99.9\%$.

ratio in the second phase. Then, the probability $P_j(p_2)$ that j marked representative tags are sampled in the detection could be expressed as

$$P_j(p_2) = \binom{n_A}{j} p_2^j (1-p_2)^{n_A - j}. \tag{26}$$

We then recursively derive the probability that an arbitrary slot is singleton after l mappings given a j

$$\begin{aligned} Ps_l &= Ps_{l-1} \\ &\quad + (1 - Ps_{l-1}) \binom{j - r_{l-1}}{1} \left(\frac{1}{f_d}\right) \left(1 - \frac{1}{f_d}\right)^{j - r_{l-1} - 1} \\ r_l &= \lfloor f_d Ps_l \rfloor. \end{aligned} \tag{27}$$

Thus, the probability that an arbitrary slot is singleton in our protocol after l mapping is

$$P_l = \sum_{j=0}^{n_A} P_j(p_2) Ps_l. \tag{28}$$

Since an arbitrary tag is mapped to a singleton slot with the probability of $(f_d P_l / n_A)$, the missing event detection probability in the advanced protocol can be approximately derived as

$$\begin{aligned} P_d &= 1 - \left(1 - p_1 + p_1 \left(1 - \frac{f_d P_l(p_2)}{n_A}\right)\right)^{M_a} \\ &= 1 - \left(1 - \frac{f_d P_l(p_2)}{|\mathcal{G}_{\mathcal{A}}|}\right)^{M_a}. \end{aligned} \tag{29}$$

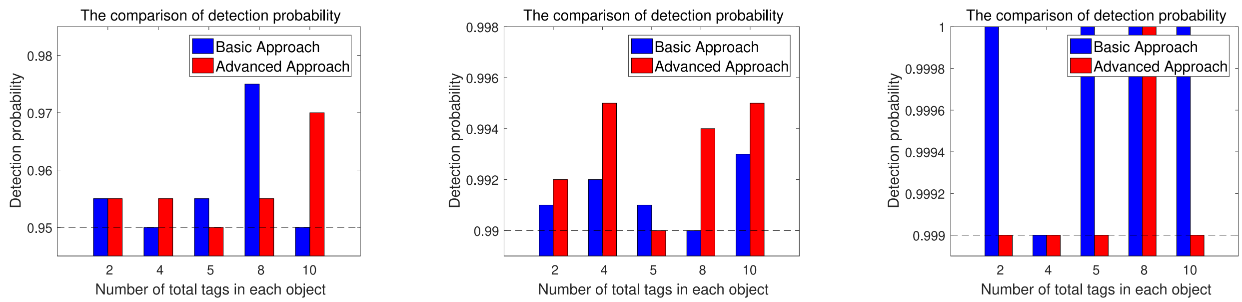
Note that M_a is a given threshold. Consequently, we should pick f_d and p_2 so that $P_d \geq \alpha$.

To this end, we could fix the value of p_2 and $P_d(p_2, f_d)$ is degraded into a function of f_d. Our goal is then turned to minimize f_d with $P_d(p_2, f_d) \geq \alpha$. After getting the optimum f_d for a given p_2, we start to introduce how to select p_2. When the sampling probability is too small to satisfy $P_d(p_2, f_d) \geq \alpha$, we cannot find suitable f_d. Hence, we could set a upper bound for f_d. If $P_d(p_2, f_d) < \alpha$ when f_d is greater than the upper bound, we should increase sampling probability p_2 to do another searching. Finally, we could find the minimum sampling probability $p_{\min}$ that just satisfies the requirement. Then, we will search the minimum f_d in $p_{\min} \leq p_2 \leq 1$.

Now, we will discuss the influence of the value in multiple mapping. Fixing f_d while increasing l, we observe that the improvement shrinks rapidly from $l = 7$ to 15, since a bigger l would increase execution time according to (25). Therefore, we can search for the optimal value of l.

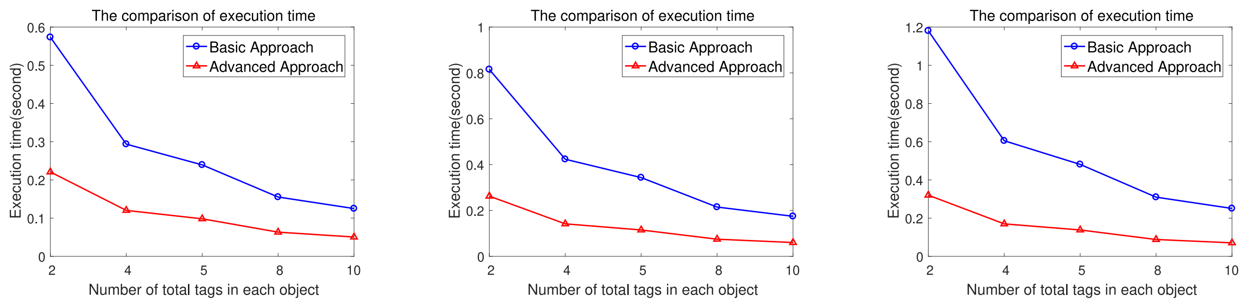
VI. Performance Evaluation

In this section, we evaluate the performance of the proposed basic and advanced protocols in terms of detection probability and execution time in multitagged RFID systems. The timing parameters in the simulation follow the EPC-global Gen-2 standard. Specifically, any two consecutive communications between the reader and tags are separated by a blank interval lasting for 266.4 μs. The transmission rate is 40.97 kb/s when a response slot T_{short} is 290.81 μs and a 96-b slot T_{id} is 2609.76 μs, which include a blank interval. The parameters, such as the filter and vector size, are set according to the theoretical analysis. In the simulation, we verify the effectiveness of the two protocols in addressing the missing event detection



(a) (b) (c)

Fig. 5. Detection probability with the number of tags on each object varied from 2 to 10 when the number of total tags is set to 1000, the number of missing objects is $M_a = 2$, and the required detection probability is (a) $\alpha = 95\%$, (b) $\alpha = 99\%$, and (c) $\alpha = 99.9\%$.



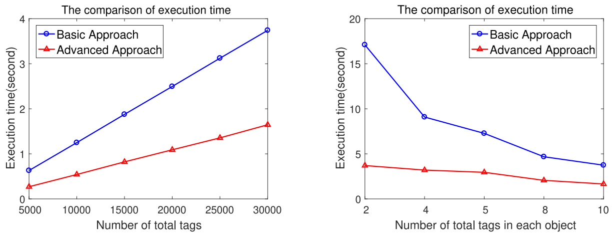
(a) (b) (c)

Fig. 6. Execution time with the number of tags on each object varied from 2 to 10 when the number of total tags is set to 1000, the number of missing objects is $M_a = 2$, and the required detection probability is (a) $\alpha = 95\%$, (b) $\alpha = 99\%$, and (c) $\alpha = 99.9\%$.

problem, where the results are obtained from 1000 independent runs. We also investigate the impacts of system scale and the number of tags on one object on their performance.

Performance Verification: We here verify the effectiveness and the efficiency of the proposed protocols under three scenarios. In the simulation, the threshold of missing objects is set to $M_a = 2$, and the required detection reliability varies from $\alpha = 95\%$ to $\alpha = 99\%$ and to $\alpha = 99.9\%$ in the first two scenarios and is fixed to $\alpha = 95\%$ in the third scenario.

1) In the first scenario, there exist ten tags on each object and the number of overall tags varies from 1000 to 5000. The simulation results of detection probability and execution time are depicted in Figs. 3 and 4. The results show that both the Bloom filter-based basic protocol and the compressive filter-based advanced one can meet the detection reliability requirement and they spend more time to detect a missing event as the number of overall tags increases. This can be interpreted as follows. As the number of objects increases, there are more representative tags that need to be marked and detected, leading to longer execution time. We can also observe that the advanced protocol needs significantly less time to detect missing event than the basic one under the same required detection reliability. As shown in Fig. 4(c), when the number of total tags is 5000, the execution time of the basic protocol is 1.24 s while the advanced protocol spends 0.38 s which is $3\times$ faster than the basic protocol.
2) In the second scenario, we study how the number of tags on one object influences the detection probability and execution time. To this end, we set the total number of tags in the system to 1000 and vary the number of tags in each object A from 2 to 10. From the results recorded in Figs. 5 and 6, we can draw similar conclusions with those in the first scenario that both protocols can complete the detection task with the required reliability satisfied, and the advanced protocol is more time efficient. In addition, the performance gain in terms of the execution time of the advanced protocol is at least $2\times$, and reaches $4\times$ when the required detection reliability is 99.9% and there are two tags on each object, as shown in Fig. 6(c).
3) In the third scenario, we focus on the time efficiency of the two protocols in large-scale systems, which is one of the most important metrics in RFID-enabled applications. The experiment consists of two cases. The number of tags on each object is fixed to ten and the number of total tags varies from 5000 to 30 000 in the first case; in contrast, we set the number of total tags to 30 000 but change the number of tags on each object from 2 to 10 in the second case. Fig. 7(a) illustrates the impact of the system scale on the execution time in the first case. We can observe that the two protocols experience longer execution time as the system scales up. But the advanced protocol performs better. Fig. 7(b) records the simulation results in the second case, which also confirms the superiority of the advanced protocol to the basic one. Moreover, it can be observed from the two figures that the advanced protocol achieves at least $2\times$ performance gain.



(a) (b)

Fig. 7. (a) Execution time with the number of tags varied from 5000 to 30000 when each object is attached by ten tags and the required detection probability is $\alpha = 95\%$. (b) Execution time with the number of tags on each object varied from 2 to 10 when the number of total tags is 30 000, required detection probability is $\alpha = 95\%$. (a) Case 1. (b) Case 2.

VII. Conclusion

This article has addressed a variation on the missing event detection problem arising from multitagged RFID systems where each object is tagged by multiple tags. Application of prior works to the new problem suffers low time efficiency due to repeated checks of one object. To overcome this drawback, we have provided two solutions, namely, the basic protocol and the advanced protocol. The former uses the Bloom filter to ask a subset of tags in the system to report their presence. The latter exploits knowability of each tag mapping and sparsity of slots mapped only by tag(s) of the chosen subset to build a compact compressive filter and a composite vector from multiple mappings of each tag. We have also derived the optimum parameters used in the protocols and conduct extensive simulations. The results confirm the effectiveness of the protocols and the superiority of the advanced protocol in terms of time efficiency under required detection reliability.

References

[1] H. Guo, C. He, N. Wang, and M. Bolic, "PSR: A novel high-efficiency and easy-to-implement parallel algorithm for anticollision in RFID systems," *IEEE Trans. Ind. Informat.*, vol. 12, no. 3, pp. 1134–1145, Jun. 2016.

[2] X. Li *et al.*, "Efficient range queries for large-scale sensor-augmented RFID systems," *IEEE/ACM Trans. Netw.*, vol. 27, no. 5, pp. 1873–1886, Oct. 2019.

[3] L. Zhang, W. Xiang, X. Tang, Q. Li, and Q. Yan, "A time-and energy-aware collision tree protocol for efficient large-scale RFID tag identification," *IEEE Trans. Ind. Informat.*, vol. 14, no. 6, pp. 2406–2417, Jun. 2018.

[4] L. Zhang, W. Xiang, I. Atkinson, and X. Tang, "A time-efficient pairwise collision-resolving protocol for missing tag identification," *IEEE Trans. Commun.*, vol. 65, no. 12, pp. 5348–5361, Dec. 2017.

[5] L. Zhang, W. Xiang, and X. Tang, "An efficient bit-detecting protocol for continuous tag recognition in mobile RFID systems," *IEEE Trans. Mobile Comput.*, vol. 17, no. 3, pp. 503–516, Mar. 2018.

[6] W. Cheng, X. Cheng, M. Song, B. Chen, and W. W. Zhao, "On the design and deployment of RFID assisted navigation systems for VANETs," *IEEE Trans. Parallel Distrib. Syst.*, vol. 23, no. 7, pp. 1267–1274, Jul. 2012.

[7] X. Liu *et al.*, "Accurate localization of tagged objects using mobile RFID-augmented robots," *IEEE Trans. Mobile Comput.*, early access, Dec. 24, 2019. doi: 10.1109/TMC.2019.2962129.

[8] X. Gao, P. Wang, D. Niyato, K. Yang, and J. An, "Auction-based time scheduling for backscatter-aided RF-powered cognitive radio networks," *IEEE Trans. Wireless Commun.*, vol. 18, no. 3, pp. 1684–1697, Mar. 2019.

[9] National Retail Federation. (2015). *National Retail Security Survey*. [Online]. Available: https://nrf.com

[10] Crime and Tech. (2019). *Retail Security in Europe. Going Beyond Shrinkage*. [Online]. Available: https://checkpointsystems.com/uk/detail/369/uk-retailers-suffer-vmost-from-shrinkage

[11] L. Bolotnyy and G. Robins, "Multi-tag RFID systems," *Int. J. Internet Protocol Technol.*, vol. 2, no. 3, pp. 218–231, 2007.

[12] S. Dhal and I. Sengupta, "Protocol to authenticate the objects attached with multiple RFID tags," in *Emerging Trends in Computing and Communication*. New Delhi, India: Springer, 2014, pp. 149–156.

[13] L. Shangguan, Z. Yang, A. X. Liu, Z. Zhou, and Y. Liu, "Relative localization of {RFID} tags using spatial–temporal phase profiling," in *Proc. NSDI*, 2015, pp. 251–263.

[14] J. Liu, H. Dai, Y. Yan, X. Zhang, X. Chen, and L. Chen, "Is this side up? Detecting upside-down exception with passive RFID," in *Proc. IEEE SMARTCOMP*, 2017, pp. 1–2.

[15] D. Hochhalter, D. Bigelow, N. J. Witchey, and C. Milam, "RFID-based rack inventory management systems," U.S. Patent Appl. 15725 638, 2018.

[16] C. C. Tan, B. Sheng, and Q. Li, "How to monitor for missing RFID tags," in *Proc. IEEE ICDCS*, 2008, pp. 295–302.

[17] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large RFID system," in *Proc. 11th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2010, pp. 1–10.

[18] R. Zhang, Y. Liu, Y. Zhang, and J. Sun, "Fast identification of the missing tags in a large RFID system," in *Proc. 8th Annu. IEEE Commun. Soc. Conf. Sensor Mesh Ad Hoc Commun. Netw.*, 2011, pp. 278–286.

[19] X. Liu, K. Li, G. Min, Y. Shen, A. X. Liu, and W. Qu, "Completely pinpointing the missing RFID tags in a time-efficient way," *IEEE Trans. Comput.*, vol. 64, no. 1, pp. 87–96, Jan. 2015.

[20] W. Luo, S. Chen, T. Li, and Y. Qiao, "Probabilistic missing-tag detection and energy-time tradeoff in large-scale RFID systems," in *Proc. ACM MobiHoc*, 2012, pp. 95–104.

[21] W. Luo, S. Chen, Y. Qiao, and T. Li, "Missing-tag detection and energy-time tradeoff in large-scale RFID systems with unreliable channels," *IEEE/ACM Trans. Netw.*, vol. 22, no. 4, pp. 1079–1091, Aug. 2014.

[22] M. Shahzad and A. X. Liu, "Expecting the unexpected: Fast and reliable detection of missing RFID tags in the wild," in *Proc. IEEE INFOCOM*, 2015, pp. 1939–1947.

[23] J. Yu, L. Chen, R. Zhang, and K. Wang, "Finding needles in a haystack: Missing tag detection in large RFID systems," *IEEE Trans. Comput.*, vol. 65, no. 5, pp. 2036–2047, May 2017.

[24] J. Yu, L. Chen, R. Zhang, and K. Wang, "On missing tag detection in multiple-group multiple-region RFID systems," *IEEE Trans. Mobile Comput.*, vol. 16, no. 5, pp. 1371–1381, May 2017.

[25] J. Yu, W. Gong, J. Liu, L. Chen, K. Wang, and R. Zhang, "Missing tag identification in COTS RFID systems: Bridging the gap between theory and practice," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 130–141, Jan. 2020.

[26] J. Yu, W. Gong, J. Liu, L. Chen, and K. Wang, "On efficient tree-based tag search in large-scale RFID systems," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 42–55, Feb. 2019.

[27] J. Yu, J. Liu, R. Zhang, L. Chen, W. Gong, and S. Zhang, "Multi-seed group labeling in RFID systems," *IEEE Trans. Mobile Comput.*, early access, Aug. 14, 2019, doi: 10.1109/TMC.2019.2934445.

[28] W. Gong, J. Liu, and Z. Yang, "Fast and reliable unknown tag detection in large-scale RFID systems," in *Proc. ACM MobiHoc*, 2016, pp. 141–150.

Hao Liu received the B.E. degree in information engineering from Beijing Institute of Technology, Beijing, China, in 2014, where he is currently pursuing the Ph.D. degree.

His research interest includes RFID technology.

Rongrong Zhang received the B.E. and M.E. degrees in communication and information systems from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2010 and 2013, respectively, and the Ph.D. degree in computer science from the University of Paris Descartes, Paris, France, in 2017.

She was a Research Fellow with the School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, ON, Canada, and is an Associate Professor with Capital Normal University, Beijing, China. Her research interests focus on wireless body area networks and Internet of Things.

Lin Chen (Member, IEEE) received the B.E. degree in radio engineering from Southeast University, Nanjing, China, in 2002, the Engineering Diploma degree from Telecom ParisTech, Paris, France, in 2005, and the M.S. degree in networking from the University of Paris 6, Paris, in 2005.

He is currently a Professor with the School of Computer Science and Technology, Sun Yat-sen University, Guangzhou, China, and used to be an Associate Professor with the Department of Computer Science, University of Paris-Sud, Orsay, France. His main research interests include modeling and control for wireless networks, distributed algorithm design, and game theory.

Prof. Chen serves as the Chair of the IEEE Special Interest Group on Green and Sustainable Networking and Computing with Cognition and Cooperation and the IEEE Technical Committee on Green Communications and Computing.

Jihong Yu (Member, IEEE) received the B.E. degree in communication engineering and the M.E. degree in communication and information systems from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2010 and 2013, respectively, and the Ph.D. degree in computer science from the University of Paris-Sud, Orsay, France, in 2016.

He was a Postdoctoral Fellow with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. He is currently a Professor with the School of Information and Electronics, Beijing Institute of Technology, Beijing, China. His research interests include RFID, backscatter networking, and Internet of Things.

Jiangchuan Liu (Fellow, IEEE) received the B.Eng. degree *(cum laude)* from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from Hong Kong University of Science and Technology, Hong Kong, in 2003.

He is currently a Full Professor with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada.

Prof. Liu was a co-recipient of the Test of Time Paper Award of IEEE INFOCOM in 2015, the ACM TOMCCAP Nicolas D. Georganas Best Paper Award in 2013, and the ACM Multimedia Best Paper Award in 2012. He is a Steering Committee Member of the IEEE TRANSACTIONS ON MOBILE COMPUTING, and an Associate Editor of the IEEE/ACM TRANSACTIONS ON NETWORKING, the IEEE TRANSACTIONS ON BIG DATA, and the IEEE TRANSACTIONS ON MULTIMEDIA. He is an NSERC E.W.R. Steacie Memorial Fellow and a Fellow of the Canadian Academy of Engineering.

Jianping An (Member, IEEE) received the Ph.D. degree from Beijing Institute of Technology, Beijing, China, in 1996.

In 1995, he joined the School of Information and Electronics, Beijing Institute of Technology, where he is currently a Full Professor. He is currently the Dean of the School of Information and Electronics, Beijing Institute of Technology. His research interests are in the field of digital signal processing, wireless networks, and high-dynamic broadband wireless transmission technology.