Missing Tag Identification in COTS RFID Systems: Bridging the Gap between Theory and Practice

Jihong Yu, Wei Gong, *Member, IEEE*, Jiangchuan Liu, *Fellow, IEEE*, Lin Chen, *Member, IEEE*, Kehao Wang, and Rongrong Zhang

Abstract—With rapid development of radio frequency identification (RFID) technology, ever-increasing research effort has been dedicated to devising various RFID-enabled services. The missing tag identification, which is to identify all missing tags, is one of the most important services in many Internet-of-Things applications such as inventory management. Prior work on missing tag detection all rely on hash functions implemented at individual tags. However, in reality hash functions are not supported by commercial off-the-shelf (COTS) RFID tags. To bridge this gap between theory and practice, this paper is devoted to detecting missing tags with COTS Gen2 devices. We first introduce a point-to-multipoint protocol, named P2M that works in an analog frame slotted Aloha paradigm to interrogate tags and collect their electronic product codes (EPCs). A missing tag will be found if its EPC is not present in the collected ones. To reduce time cost of P2M resulted from tag response collisions, we further present a collision-free point-to-point protocol, named P2P that selectively specifies a tag to reply with its EPC in each slot. If the EPC is not received, this tag is regarded to be missing. We develop two bitmask selection methods to enable the selective query while reducing communication overhead. We implement P2M and P2P with COTS RFID devices and evaluate their performance under diverse settings.

Index Terms—RFID, IoT, missing tag identification, commercial Gen2 devices

1 Introduction

1.1 Background and Motivation

Recent years have witnessed an unprecedented development of the radio frequency identification (RFID) technology [1]. The distinct advantages of RFID, such as low manufacture cost of commercial tags (e.g., 5 cents per tag [2]), wireless non-line-of-sight communication and batched tag identification, make it widely deployed in various scenarios ranging from inventory control [3], [4], [5], supply chain management [6], [7], object localization [8], [9], to human-computer interaction [10].

To enable worldwide commercial implementation of RFID, the EPCglobal, an organization that was formed in 2003, developed the Gen2 air interface protocol [11] for ultra-high-frequency (UHF) RFID systems. This protocol has been adopted as the ISO 18000-6C standard and has become mainstream specification worldwide for commercial off-the-shelf (COTS) RFID devices like ImpinJ [12] and ThingMagic series [13]. A Gen2 RFID system comprises two types of devices: passive RFID tags and RFID reader. A passive tag is a light-weight battery-free device that can record information of a physical object and is able to capture the energy in the wireless signal of its nearby RFID reader and modulate this signal by adjusting the impedance match on its antenna so that a message of zeros and ones is backscattered to the reader.

Identifying missing tags, *which is to completely pinpoint the tags that should be in the coverage range of the reader but are absent*, is one of the most important RFID-enabled services. According to the statistics presented in [14], inventory shrinkage, a combination of shoplifting, internal theft, administrative and paperwork error, and vendor fraud, resulted in about 49 billion dollars in loss for retailers in 2016. In this context, RFID provides a promising technology to reduce the financial loss by deploying a reader to monitor passive tags attached on products in its coverage range and conducting missing tag identification regularly to find missing items in time.

- *J. Yu is with the School of Information and Electronics, Beijing Institute of Technology, Beijing 100081E, China. E-mail: jihong-yu@hotmail.com.*
- *W. Gong is with the School of Computer Science and Technology, University of Science and Technology, Anhui 230022, China, and the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada. E-mail: weigong@ustc.edu.cn.*
- *J. Liu is with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada. E-mail: jcliu@sfu.ca.*
- *L. Chen is with the Laboratoire de Recherche en Informatique (LRI), Universit Paris-Sud, Orsay 91400, France, and the Institut Universitaire de France (IUF), 75231 Paris, France. E-mail: chen@lri.fr.*
- *K. Wang is with the Key Laboratory of Fiber Optic Sensing Technology and Information Processing, School of Information Engineering, Wuhan University of Technology, Hubei 430070, China. E-mail: kehao.wang@whut.edu.cn.*
- *R. Zhang is with Information Engineering College, Capital Normal University, Beijing, Beijing 100048, China. E-mail: s100101182@hotmail.com.*

Manuscript received 15 May 2018; revised 17 Sept. 2018; accepted 28 Nov. 2018. Date of publication 21 Dec. 2018; date of current version 3 Dec. 2019. (Corresponding author: Jihong Yu.)
Digital Object Identifier no. 10.1109/TMC.2018.2889068

1.2 Limitations of Prior Arts

Ten-year gap of missing tag identification with COTS Gen2 devices. The study of missing tag identification was initiated in the research community about 10 years ago, and ever since then ten-year effort has been dedicated to reducing communication overhead, producing a large body of work. However, none of the previous work is compatible with the Gen2 standard so that they cannot be implemented in practice, which leaves billions of deployed COTS tags behind. The failure of the prior work mainly results from the two reasons:

(1) *Hashing-dependent slot selection*: Prior work on missing tag identification requires the functionality of hashing in tags so that each tag can select and respond in a random but predictable slot corresponding to the hash value of its electronic product code (EPC) and a random seed. While the hashing functionality has never been implemented in any COTS tags as high energy consumption and manufacture cost will be incurred otherwise (e.g., over 1,000 gate equivalents for hardware), which is contradictory to what is expected of RFID.

(2) *Complete visibility for slot states*: Prior work must definitely know the states of each slot, e.g., empty and busy, which depends on the number of one-bit responses from tags in this slot, and exploits the empty slots that should be busy to identify missing tags. While a COTS Gen2 reader only reports successful reads in a time interval, disabling the utility of empty slots. Hence, the previous work cannot implemented in COTS RFID devices.

Motivated by the observations above, we argue that a systematical study on missing tags identification with COTS Gen2 devices is called for to maximize the function of widely deployed Gen2 RFID systems and to reduce financial losses.

1.3 Proposed Approach

To address this issue, we develop two protocols that are able to completely pinpoint missing tags while being compatible with the Gen2 standard and the existing COTS devices. Specifically, we first develop a point-to-multipoint protocol (named P2M). P2M employs Q-command to query the tags which is the de facto random access protocol in the Gen2 standard, and can fulfill the task within the bounded worst-case time by carefully configuring the interrogation duration 2^Q. In order to improve the time efficiency of P2M, we then design a point-to-point protocol (named P2P) that can singularize the tags in each slot with a selective bitmask and ensures successful communication in all slots. To this end, we propose two bitmask selection approaches making a tradeoff between communication overhead and computational complexity.

We implement P2M and P2P in extensive scenarios using COTS Gen2 devices: one ImpinJ reader and 20 ImpinJ Monza tags. The results show that P2P achieves time efficiency gains of about 4x and 6x over P2M on average in the identification of all missing tags and the detection of the first missing tag. We also confirm the correctness of bitmask selection approaches of P2P in larger systems.

2 System Model and Problem Statement

A typical Gen2 RFID system is consisted of a reader and multiple passive tags. The reader can charge, synchronize and collect information from tags, while tags each having an EPC are usually attached on physical objects, producing one-one map between a tag and an object. To interact with battery-free tags, the reader initially transmits a continuous wave to the tags. The tags capture energy from the incoming wave to power themselves on one hand and use this wave as a carrier to backscatter their information bits with ON-OFF keying on the other hand. Specifically, the tags send a '1' bit by adjusting the impedance match on their antennas to reflect the reader's wave and a '0' bit by remaining silent [15].

The Missing Tag Identification Problem. Consider a Gen2 system containing a reader and n tags $\{x_1, x_2, \cdots, x_n\}$ and that the reader knows all tag EPCs, there exists an event that m out of the n tags are missing due to the damage of these tags or the disappearance of their corresponding objects. *The missing tag identification problem is to exactly find the m missing tags.* In this problem, execution time that is measured as the time spent achieving the task is the most important metric. The earlier missing products are found, the more significantly financial loss is reduced.

Limitations of Prior Work. A large body of works are proposed to accelerate the identification process on the top of the assumption that response slots of tags are predictable via hashing operations. Though the works are promising in improving time efficiency, the reality is that the widely deployed Gen2 tags cannot support hash function that is the prerequisite of these works. Moreover, no manufacturer declares that hash function will be packaged into commercial tags in near future.

Why is the Hashing Functionality not Supported by COTS tags? The main reason lies in high energy consumption and manufacture cost introduced by hardware design of hash function.[1] In particular, thousands of gate equivalents are required for current common hash functions, such as SHA-1 and SHA-256 [16] require 8,120 and 10,868 gate equivalents with power consumption 10.68 μA and 15.87 μA, respectively. Even the most compact hash function that is presented in theory and is not available for COTS tags, e.g., PRESENT-80 [17], still requires 1,075 gate equivalents. Considering huge market of RFID (e.g., 1.82×10^{10} tags in 2017), enabling hash function in tags will incur extremely high cost.

The Proposed Solutions without Requirement of Hash Function. It is still an open question how to identify all missing tags without the hash function in the Gen2 system. To bridge this gap, we design two Gen2-compatible missing tag identification protocols by using commands specified in the Gen2 standard, such as Q-command and *Select* command. As our protocols can be implemented in COTS RFID devices, they can be used to identify missing items in RFID-deployed scenarios like Walmart [18] and River Island [19], to reduce or even avoid financial loss resulted from product missing event.

1. Gate equivalent is a key performance metric in evaluating efficiency and availability of a hardware design. The more gate equivalents are required, the higher the implementation overhead and cost are.

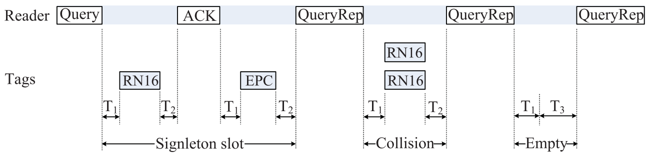


Fig. 1. Link timing of P2M communications. The Gen2 standard has strict requirement for each command format and link timing parameters T_1, T_2, and T_3 that stand for interval-command time, enabling the computation of overall interrogation time.

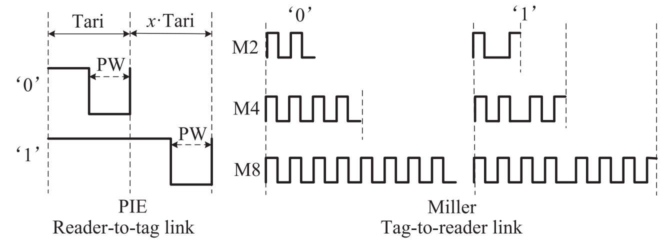


Fig. 2. Data encoding in the Gen2 standard.

In what follows, we describe P2M that behaves in a point-to-multipoint manner with the Q-command used to query all tags in the system. We then show the second work, namely P2P, which ensures point-to-point communication in each slot with an exclusive bitmask and avoids empty and collision slots.

3 P2M: Point-to-Multipoint Missing Tag Identification

In this section, we introduce the first Gen2-compatible protocol, the point-to-multipoint Q-query and its application to identify missing tags, and then show the parameter configuration and time cost computation.

3.1 Point-to-Multipoint Q-Query

The Gen2 standard specifies how the reader interrogates tags. First, the reader sends a *Query* command to initiate the interrogation. This command contains backscatter link frequency (BLF), tag-to-reader encoding method and a Q parameter used to specify the number of slots in this query round. With the parameter Q, each tag is able to determine its response slot by selecting a random value in $[0, 2^Q - 1)$ as its slot counter. If this counter is equal to 0, the tag replies immeditely with a 16-bit random number (RN16); otherwise it shall keep silent. Upon receiving an *RN16* from a tag, the reader transmits an *ACK* containing the decoded *RN16* to acknowledge this tag. If the tag confirms the correctness of the reader-to-tag *RN16*, the tag will backscatter its *EPC* to the reader. Subsequently, the reader issues a *QueryRep* to instruct tags to decrement their slot counters and the tags whose counters are equal to 0 reply with another *RN16*, indicating the start of a new slot.[2] Fig. 1 illustrates the Q-query process and shows that there is waitting time between two continous commands like T_1, T_2 and T_3.

Since the reader can collect all EPCs of the tags present in its coverage via the Q-query, it compares the collected EPCs with the ones recorded in the database. If some recorded EPCs are not present in the collected EPC set, these tags are missing and can thus be identified by the reader. This comparison is conducted at the end of the the Q-query. P2M is superior to the existing works since they need the knowledge of all slot states which cannot be obtained from COTS reader. The main question in P2M is when the Q-query should be terminated.

2. The counter of a tag in the Q-query measures the number of slots before it replies, thus setting a value to a tag's counter is equivalent to assigning a slot to this tag.

3.2 Encoding Methods

The quest for low cost, tiny size, and battery-free tags severely limits their computation and hardware capabilities. It is thus important and necessary to encode and decode data in an extremely simple and robust way. In practice, the reader-to-tag symbols are amplitude-modulated pulse interval encoding (PIE) symbols which an analogy comparator is adequate to decode. As shown in Fig. 2, symbol '0' in PIE comprises two intervals of the same length, namely power-on and power-off (PW: pulse width). Tari (Type A reference interval) is the duration of data-0, while the duration of data-1 is as long as $x \in [0.5, 1]$ times of data-0. The Tari values can be set as 6.25, 12.5, or 25 μs corresponding to the rates 160, 80, and 40 kbps. Different from the lightweight tags, the reader has the strong decoding capacity. The Gen2 standard specifies four encoding method for the tag-to-reader link, FM0, M2 (Miller2), M4 (Miller4), and M8 (Miller8). The data rate depends on the BLF and the used encoding method. For example, if BLF is 320 kHz, the data rates of FM0, M2, M4, and M8 are 320/1, 320/2 = 160, 320/4 = 80, 320/8 = 40 kbps, respectively.[3]

3.3 Configuration of the Parameter Q

From the description of the Q-query, we can observe that it is a random access process in nature, with tags randomly setting their individual counters in the beginning of the interrogation. The reader cannot predict the values picked by the tags. Consider an arbitrary slot i, there would be three states:

- If there is only one tag replying, i.e., this tag uniquely picks the value i, it is a singleton slot;
- if there are multiple tags replying, i.e., these tags pick the value i, it is a collision slot;
- if there is no tag replying, i.e., no tag selects the value i, it is an empty slot.

We make an illustration in Fig. 1 where one tag replies in the first slot and then two tags and no tag respond in the second and the third slots, respectively.

Among these states, only singleton slots are useful for EPC collection while collision and empty slots are useless, thus a natural optimisation criteria is to ensure with high probability that there exist n singleton slots in the interrogation, meaning that no collision occurs. Technically, the Q-query process can be formulated as the classic Ball-into-Bins problem [20]. Specifically, n tags are balls and 2^Q values (or slots) are bins. To avoid collisions with high probability, 2^Q needs to be set to $\Theta(n^2)$ [21]. Guided by this

3. The reader sets and packages the parameters, including encoding type and BLF, into a query command, and sends the command to tags.

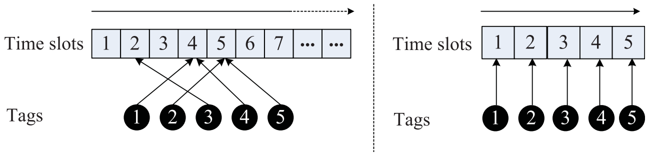


Fig. 3. Comparison of P2M (the left) and P2P (the right) for multiple tags. P2M would waste some slots that are collided (slots 4 and 5) or empty (slots 1, 3, 6, and 7). While P2P can selectively read tags and only needs five slots.

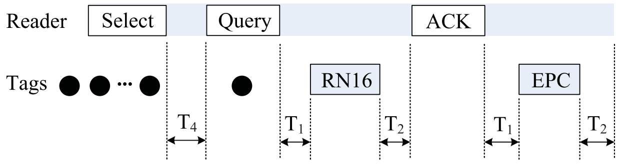


Fig. 4. Link timing of P2P communication where the black points represent tags. The Gen2 standard has strict requirement for each command format and link timing parameters T_1, T_2, and T_4 that stand for interval-command time, enabling the computation of overall interrogation time.

theoretical result, we set Q to $2\log n$ where $\log$ denotes the logarithm to the base 2. Under such configuration, the Q-query lasts n^2 slots.

By this setting, it is adequate for our point-to-multipoint protocol to know singleton slots, which fits well in today's COTS devices. In contrast, we note that existing works require the reader to report empty slots, which is unsupportable in the current COTS devices.

3.4 Calculation of the Interrogation Duration

As shown in Fig. 1, the three types of slots differ in their slot duration. Thus the first step in the interrogation duration computation is to figure out the number of slots in each type. Recall that we set $Q = 2\log n$ to ensure no collision and that there are m missing tags, there would be $n - m$ singleton slots and $n^2 - n + m$ empty slots in the interrogation. As a result, the key is to compute the sizes of singleton and empty slots. To do so, we further zoom in on each slot in Fig. 1, and obtain the following observations:

- Singleton slot size: A singleton slot is composed of an *RN16*, an *ACK*, an *EPC*, and the inter-command time T_1 and T_2. Thus we can calculate a singleton slot size as $ACK \cdot \text{Tari} + \frac{RN16+EPC}{\text{BLF}/j} + 2(T_1 + T_2)$ where $j \in \{1,2,3,4\}$ indicates different tag-to-reader encoding methods.[4]
- Empty slot size: An empty slot comprises two intervals of commands T_1 and T_3, thus its length is equal to $T_1 + T_3$.
- Inter-slot time: There is a *Query* command in the beginning of the interrogation and a *QueryRep* between any two continuous slots, so the overall inter-slot time in the whole interrogation should be $(Query + (n^2 - 1) \cdot QueryRep) \cdot \text{Tari}$.

Following these observations, now we are able to formulate the overall interrogation time of P2M is $(n-m) \cdot (ACK \cdot \text{Tari} + \frac{RN16+EPC}{\text{BLF}/j} + 2(T_1 + T_2)) + \big(Query + (n^2 - 1) \cdot QueryRep\big) \cdot \text{Tari} + (n^2 - n + m)(T_1 + T_3)$.

4 P2P: Point-to-Point Missing Tag Identification

Our first proposition presented previously follows the point-to-multipoint paradigm. Due to its random nature, multiple tags may reply with *RN16* simultaneously, leading to decoding failure at the reader. To deal with tag collisions, P2M sets Q to $2\log n$, which results in considerable empty slots and wastes time. To avoid collision events while improving time efficiency, we propose P2P that performs as a point-to-point paradigm, which is able to singularize tags in every slot. As shown in Fig. 3, the reader cannot control the response slots of tags in P2M such that it suffers from collisions. In contrast, P2P can assign the reply order and avoids collisions, such as tags 1-5 respond in slots 1 to 5 in sequence. In what follows, we first elaborate the missing tag identification process, then demonstrate how to build effective and efficient bitmasks.

4.1 Point-to-Point Selective Query

The Gen2 standard provides a command *Select* that allows the reader to selectively read a subset of tags based on user-defined criteria. As shown in Fig. 4, the selective query includes two phases: tags filtering and tag query. First, the reader issues a *Select* that specifies a bitmask and an action that will be performed by the tags. On receiving *Select*, each tag checks whether it matches the reader-to-tag bitmask. If yes, it will assert its flag variable `SL`; otherwise it will deassert the `SL`. By carefully designing the bitmask, we can ensure only one tag can pass the bitmask comparison, which will be presented shortly. Then the reader further sends a *Query* that specifies the tags with asserted `SL` to reply. Since only one tag meet the requirement in P2P, this tag is the only one replying to the *Query* with its *RN16*. Subsequently, the reader transmits *ACK* with the decoded *RN16* and prepares to receive the *EPC* of this tag. When this query finishes, the reader will repeat the above process to read tags one by one.

The desired property of P2P is its capacity to specify an individual tag to reply. If there is no response from this tag, the reader will know its absence. As a result, P2P can identify all m missing tags after n selective queries. Moreover, P2P can also detect a missing tag in at most $n - m$ selective queries.

4.2 Calculation of the Overall P2P Execution Time

As shown in Figs. 1 and 4, the length of a P2P selective query on a present tag contains a *Select*, T_4, a *Query*, and a singleton slot whose length is equal to that in P2M. If a missing tag is queried, the components of this query duration are almost same as the prior except that slot duration becomes to empty slot size instead of singleton slot size. Thus, recall Section 3.4, we know that it takes P2P time of $(Select + Query + ACK) \cdot \text{Tari} + \frac{RN16+EPC}{\text{BLF}/j} + T_4 + 2(T_1 + T_2)$ to achieve a selective query on a present tag, where $j \in \{1,2,3,4\}$ indicates different tag-to-reader encoding methods.[4] As a consequence, the overall time cost of P2P is

4. Either a preamble or a frame-sync will be prepended to every command, such as *RN16*, *EPC*, *ACK*, *Query*, *QueryRep*, and *Select*. In addition, tags reply PC (protocol control) and CRC along with their *EPC*s. We use these commands to represent their individual length plus the extra length (bits).

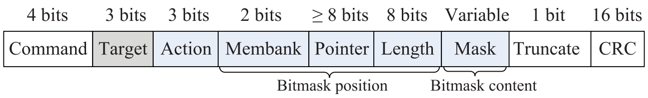


Fig. 5. *Select* command: `MemBank`, `Pointer`, and `Length` specify the bitmask position that the tag needs to search in its memory; `Mask` records the bitmask content that the tag will compare with.

TABLE 1
Tag Response to Action

Action code	Tag matching	Tag not-matching
000_2	assert `SL`	deassert `SL`
001_2	assert `SL`	do nothing
010_2	do nothing	deassert `SL`
011_2	negate `SL`	do nothing
100_2	deassert `SL`	assert `SL`
101_2	deassert `SL`	do nothing
110_2	do nothing	assert `SL`
111_2	do nothing	negate `SL`

$(n-m)\big((Select + Query + ACK)\text{Tari} + \frac{RN16+EPC}{\text{BLF}/j} + T_4 + 2(T_1+T_2)\big) + m\cdot((Select + Query)\cdot \text{Tari} + T_4 + T_1 + T_3)$.

Having described the process of P2P, we next explain how *Select*, the key function in P2P, is designed in our missing tag detection protocol.

4.3 *Select* Function

There are six mandatory fields in the *Select* command as shown in Fig. 5, we introduce five fields relevant to our design.

(1) `Action` specifies eight types of tag behaviour which are listed in Table 1. In our scenario, we use the first type, i.e., `Action` $= 000_2$, to specify tag action. Specifically, tags that match the received bitmask, called matching tags, will assert `SL`, while the other tags, called not-matching tags, will deassert `SL`.

(2) `MemBank` indicates which tag memory model a tag will search to compare with the received bitmask. The `MemBank` $= 00_2$ is reserved memory storing passwords associated with the tag. If `MemBank` $= 01_2, 10_2, 11_2$ then the tag searches for the bitmask in the EPC memory bank that stores the tag EPC, TID memory bank that specifies the permalocked tag and manufacture specific information, and User memory bank that can be written with user-defined data. We employ the EPC memory bank in this paper, i.e., `MemBank` $= 01_2$.

(3) `Pointer` records a starting bit position in the chosen `MemBank` for the bitmask comparison.

(4) `Length` specifies the bitmask length. If `MemBank` $= M$, `Pointer` $= p$ and `Length` $= l$ then the tag compares the bitmask with the bits starting from the p-th bit to the $(p+l-1)$-th bit in its memory model M.

(5) `Mask` records the bitmask content that is a bit string. The tag compares it with the specified bit string in its memory.

From the description above, we observe that the combination of `MemBank`, `Pointer` and `Length` specifies the position of the bit string that the tag needs to search for in its memory while `Mask` records the bitmask content that the tag will compare with the bit string. Thus, we use BM to represent a bitmask, that is to say, $BM = (M, p, l, Mask)$.

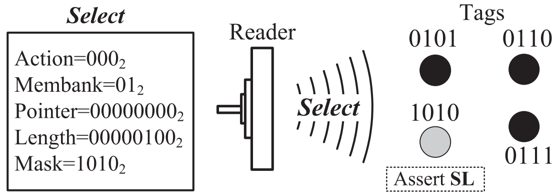


Fig. 6. Illustration of a selective query in P2P. There are four tags with EPCs: 0101, 0110, 1010, 0111, respectively. With the configuration: `Action` = 000_2, `Membank` = 01_2, `Pointer` = 00000000_2, `Length` = 00000100_2, `Mask` = 1010_2, the reader asks the tags to compare the bit string from the 1st to the 4th bit in their individual EPCs with the content in `Mask` of the received *Select* [5].

```
TagFilter t1 = settings.getFilters().getTagFilter1();
        t1.setBitCount(4);
        t1.setBitPointer(0x20);
        t1.setMemoryBank(MemoryBank.Epc);
        t1.setFilterOp(TagFilterOp.Match);
        t1.setTagMask("A");
        settings.getFilters().setMode(TagFilterMode.OnlyFilter1);
```

Fig. 7. Implementation of *Select* command in Fig. 6.

In P2P, we build the bitmask from a tag EPC by setting `MemBank` $= 01_2$. The EPC is unique and has been stored in tags, thus P2P does not need to write new data to tags. We take an example to further illustrate its application. As shown in Fig. 6, the reader sends a *Select* specifying the EPC 1010 as the bitmask.[5] Upon receiving this command, each tag checks the bit string from the first to the fourth bit in its EPC and compares it with the received one in the `Mask`. Since only the grey tag meets the criterion, it will assert its `SL` and wait for the incoming *Query*, while the others will keep silent. We present an implementation of this example in Java in Fig. 7. As tag EPC starts from the 32nd bit in the memory, the pointer in the implementation is set to 0x20.

So far we have introduced the framework of P2P and the *Select* function, the final question left is how to effectively and efficiently configure the bitmask, i.e., `Mask`. We attack the configuration of bitmask in the next subsection.

4.4 Bitmask Selection

Recall that in P2P, the reader seeks to distinguish a tag from the others in every slot. To do so, a direct way is setting `Mask` to the tag EPC, as the toy example in Fig. 6. Although such configuration is effective, it suffers from low efficiency. Recall Fig. 5, a *Select* command is 45-bit long excluding the `Mask`.[6] If we use 96-bit EPC in `Mask` which is over two times of the other fields and over the two thirds of the whole *Select*. If we can use a shorter `Mask`, the efficiency will be improved. For example, reconfiguring *Select* in Fig. 6 to `Pointer` = 00000000_2, `Length` = 00000001_2, `Mask` = 1_2 when the tags compare the first bit of their EPCs with the `Mask`, we can make the gray tag the only one to meet the requirement with 1-bit mask instead of previous 4 bits.

5. Usually a Gen2 tag has a 96-bit EPC. In this example, we assume the EPC length is four for simplicity.

6. The format of `Pointer` is an extensible bit vector that contains one or multiple 8-bit blocks. With one block, it can represent numeric values between 0 and 2^7. For the value over 2^7, it must add another block. Since the EPC length used in this paper is 96 bits, it is enough to use one block, that is to say, field `Pointer` is 8-bit long.

Inspirited by the example above, we exploit the potential of building a bitmask with a portion of a tag EPC instead of the whole. Although 96∼496-bit EPC can be supported by tags like ImpinJ Monza tags, we use 96-bit EPC in this paper, but our work can be directly used in the scenarios where EPC length is over 96 bits. We know that 96-bit strings can uniquely identify $2^{96} = 7.9 \times 10^{28}$ tags at most. Since the number of the tags in a system is usually much smaller than this quantity, the present EPCs in a Gen2 system are sparse compared with overall 2^{96} EPCs. We can exploit this sparsity to design more efficient bitmask selection methods. Note that their efficiency is more significant for tags with longer EPC, e.g., 496-bit EPC.

Algorithm 1. Deterministic Bitmask Selection

```
   Input: Tag set {x_1, x_2, ..., x_n}
1  Initialisation: l ← 1, N ← ∅, j ← 0, S* ← ∅
2  while j ≤ n do
3    while l ≤ a log n do
4      p ← 0
5      while p ≤ a log n − l do
6        N ← {x_1, x_2, ..., x_n}; S ← ∅
7        Indicator=1
8        Choose an arbitrary tag x from N − S − S*
9        for each j ∈ N/x do
10         if x(p, l) == j(p, l) then
11           S ← S ∪ x; Indicator=0
12           p ← p + 1; Jump to Line 6
13         end
14       end
15       if Indicator==1 then
16         Record x, p, and l; S* ← S* ∪ x
17         j ← j + 1; Jump to Line 2
18       else
19         p ← p + 1
20       end
21     end
22     l ← l + 1
23   end
24   j ← j + 1
25 end
26 Return a collection of x(p, l)
```

4.4.1 A Deterministic Bitmask Selection Algorithm

We first design a deterministic algorithm, whose core idea is to use only a portion of a tag EPC as bitmask so that only one tag matches. The fields `Length` and `Pointer` specify the length and the starting position of the bit string in tag memory which will be compared with the received bitmask, we denote them by l and p, respectively. Since we select l consecutive bits from an $a \log n$-bit EPC, l could be equal to a value between 1 to $a \log n$, and there are $a \log n - l + 1$ segments in all in an EPC corresponding to $p = 0 : a \log n - l$. For instance, if $l = 2$ in Fig. 6, we have three segments for the grey tag from left to right, namely 10, 01, 10. As a result, we can find an optimal bitmask in each slot, i.e., the shortest bitmask that can make a tag singular in a slot, through the following three-dimensional search (Algorithm 1). In the algorithm, $x(p, l)$ denotes a string from the p-th bit to $(p + l - 1)$-th bit in the EPC of tag x; $a = \frac{EPC}{\log n}$. The Algorithm, whose core steps are explained below, outputs the shortest bitmask specifying `Pointer`, `Length` and `Mask`.

- First, let $l = 1$, and we arbitrarily pick one out of n tags.
- Second, given l and this tag EPC, we compare its first l-bit segment, i.e., the leftmost, with those of the other $n - 1$ tags EPCs. If we find the segment unique, it can be used as a bitmask and `Pointer` $= 00000000_2$, then the searching process will be terminated; otherwise, this tag is regarded useless temporally, and we choose another one from the $n - 1$ tags to compare its first l-bit segment with those in the other $n - 1$ tags EPCs. This step runs until either a unique l-bit segment is found or any two tags has compared with each other.
- Third, if we fail to find a unique l-bit segment in the second step, we repeat the operations in the second step with the second l-bit segment. If it succeeds this time, this segment is assigned to `Mask` and `Pointer` is equal to 00000001_2; otherwise we set $l = l + 1$. The third step stops if a bitmask is found or $l = a \log n$. If a bitmask is found, that is to say, we can selectively query a tag matched this bitmask, then the algorithm keeps running to look for a bitmask for another tag.

From the description above, we can interpret the three dimensions in our algorithm as follows:

- Comparing between any two tags;
- Sliding `Pointer` p from 0 to $a \log n - l$;
- Incrementing l from 1 to $a \log n$.

Our algorithm can deterministically find an optimal bitmask. We now analyze its computational complexity. As we explained previously, the complexity of our algorithm can be decomposed into three parts: 1) $\mathcal{O}(n^2)$ operations for each (p, l); 2) $\mathcal{O}(\log n^2)$ combinations of (p, l); 3) the algorithm needs to find a bitmask for all n tags. The overall computational complexity sums up to $\mathcal{O}(n^3(\log n)^2)$.

4.4.2 A Probabilistic Bitmask Selection Algorithm

We next devise a probabilistic Bitmask selection algorithm that ensures a unique bitmask with a required success probability. Compared with the deterministic algorithm, the probabilistic algorithm has three advantages:

- Reduced complexity. The probabilistic scheme reduces the complexity from $O(n^3)$ to $O(n^2)$ in the worst case. In practice the gain can be more significant. Low complexity is desired especially for handhold mobile reader which has limited computational capacity.
- Tunable accuracy. As a desired property, the accuracy of the probabilistic algorithm can be tuned to strike a balance between the accuracy and computation and communication overhead.
- Better applicability. The probabilistic algorithm can be used to identify missing tags even when there are new tags that are not recorded in the database, but the deterministic one cannot conduct this task. This will be discussed at the end of this section.

In the probabilistic algorithm, we divide a tag EPC into $\lfloor\frac{|EPC|}{l}\rfloor$ non-overlapping segments, i.e., $\lfloor\frac{a \log n}{l}\rfloor$ segments. For example, if $l = 2$ in Fig. 6 where an EPC is assumed to be four bits long, we have two such segments for the black tag 0111 from left to right, namely 01 and 11. This method is formally stated in Algorithm 2 that operates as follows:

- First, we set l and another parameter z that stands for the execution rounds of this algorithm. How to configure the parameters will be introduced shortly.
- Second, we arbitrarily choose a tag and select the first (leftmost) segment of its EPC. Then, we compare this segment with those of the other $n-1$ tags. If this segment is unique, we use it as a bitmask and set Pointer= 00000000_2, then the algorithm stops; otherwise, we select the second segment, and repeat the operations above. The algorithm terminates when a unique segment is found or the number of the executed rounds exceeds z.

It is obvious that the complexity of the probabilistic method is $\mathcal{O}(n \cdot z)$ where $z \leq \lfloor \frac{a \log n}{l} \rfloor$. To find bitmasks for all tags in P2P, this method needs to run n times, so the overall complexity is $\mathcal{O}(n^2 \log n)$.

Algorithm 2. Probabilistic Bitmask Selection

Input: Tag set $\{x_1, x_2, \ldots, x_n\}$, l, z
Initialisation: $N \leftarrow \{x_1, x_2, \ldots, x_n\}$, $k \leftarrow 1$, $p \leftarrow 0$; choose an arbitrary tag x from N
while $k \leq z$ **do**
  Indicator=1
  for *each* $j \in N/x$ **do**
    if $x(p,l) == j(p,l)$ **then**
      Indicator=0
    end
  end
  if *Indicator==1* **then**
    Stop
  else
    $p \leftarrow p + l$; $k \leftarrow k + 1$
  end
end
Return $x(p,l)$

Next, we move to the analysis of parameter configuration. Since each bit in EPC is generated randomly in practice, the strings of $\lfloor \frac{a \log n}{l} \rfloor$ non-overlapping segments are mutually independent. The algorithm would run k rounds if the first $k-1$ rounds fail where $k \leq z$, thus the probability distribution of the number of executed rounds, defined as K, can be formulated as a geometric distribution.

Consider an arbitrary round, finding unique bitmasks for all n tags is equal to the event that the selected l-bit segments are different from each other. The probability of this event is $e^{-\frac{n^2}{2^{l+1}}}$ [20]. As a result, we have

$$\Pr(K = k) = (1 - e^{-\frac{n^2}{2^{l+1}}})^{k-1} \cdot e^{-\frac{n^2}{2^{l+1}}}.$$

Hence, the success probability after z rounds, defined as P_s can be calculated as

$$P_s = \sum_{k=1}^{z} (1 - e^{-\frac{n^2}{2^{l+1}}})^{k-1} \cdot e^{-\frac{n^2}{2^{l+1}}} = 1 - (1 - e^{-\frac{n^2}{2^{l+1}}})^{z}.$$

Denote by α the required success probability of finding bitmasks for n tags, we can get the relationship of l and z:

$$P_s = \alpha \Longrightarrow \log(1 - \alpha) = z \log(1 - e^{-\frac{n^2}{2^{l+1}}}). \quad (1)$$

To solve this equation, we can first specify value for either l or z, and derive the other. P_s monotonously increases with l and z, thus the selection of l and z indicates the tradeoff between computational complexity and communication overhead. For example, *let* $z = 1$, *the complexity will be reduced to* $\mathcal{O}(1)$ while l reaches its maximum value $\log \frac{n^2}{-\ln \alpha} - 1$ from (1). If the required α is equal to 99 percent and $n = 2^{10}$, then $\log \frac{n^2}{-\ln \alpha} - 1 \approx 26$. In contrast, if let $z = \lfloor \frac{96}{l} \rfloor$ under the same requirement, we have $l = 20$ while $z = 4$. Note that the value of l cannot exceed the length of a tag EPC.

4.5 Missing Tag Identification with New Tags

In this part, we discuss whether P2P can be used to identify missing tags in the scenario with the arrival of new tags that are not recorded in the database. To do so, we study in two cases: P2P with the deterministic algorithm and P2P with the probabilistic algorithm.

In the first case, P2P cannot be used in such a coexistence scenario as the deterministic algorithm must search for all EPCs of the tags in the database to find a unique bitmask while those of the new tags are not recorded. As a result, some new tags may also match the selected bitmask, colliding with the response of the known tag, which makes P2P fail.

In the second case, P2P can be adapted for the coexistence scenario if the number of the new tags can be estimated or the reader knows the range of the new tag population. Assume the upper bound of the new tag populations is $\overline{u}$, given the required α, we can calculate the needed bitmask length l and the number of the execution rounds from the following equation such that the identification probability is at least α,

$$\log(1 - \alpha) = z \log(1 - e^{-\frac{(n+\overline{u})^2}{2^{l+1}}}).$$

Note that when the tag EPC is 96-bit long, P2P can deterministically identify all missing tags if $l = 96$.

5 Implementation

5.1 Implementation Setup

COTS Gen2 Devices. We use one ImpinJ R420 reader and 20 ImpinJ Monza-4 UHF tags in our implementation. These devices are completely compile with the Gen2 standard. The missing identification programs are written in Java on the top of ImpinJ SDK v.1.28.0.1. In particular, the ImpinJ R420 reader supports Q-query and selective query. The ImpinJ Monza-4 tags have 96-bit EPCs.

Parameters. The transmission power of the reader is set to 30 dbm, and its reception sensitivity is -70 dbm. We implement three tag-to-reader encoding methods: M2, M4, M8. As the ImpinJ reader can support three combinations, we vary the tag-to-read link rate from 320 kbps with M2, to 68.5 kbps with M4, to 21.33 kbps with M8. In PMP, we set $Q = 2 \log n$ where n is the number of the tags in the Gen2 system, which will be set to 5, 10, 20, respectively. We will investigate the correctness of the

TABLE 2
Tag EPCs in the Implementation

i	x_i	x_{i+5}	x_{i+10}	x_{i+15}
1	2E4E6693572D3A8D185E0988	110B1D467E616FCA07E03A31	6402201E11FA2CB336243D3A	29B66F4D3EBD748A42352298
2	06DD7F27437B193326BA3F35	70A575FE134C343C67F778CA	37A721130D0879BC3BAA253E	3636306E7A131BFF738758C6
3	415859552FF64559679B4EFE	300833B2DDD9140000000000	4EB922210CEF339B2B3C0F4B	2FE666A910E74FB543FE5D83
4	76317A5F05056B4072D21075	49D87D2252B13F24278A24CF	75643B7A0D806EA8286E08BD	22A03BE81F5F28F552EF2011
5	7BD8536F240C0F0C19C2534A	2E8B6D541CCD447E0B7C684D	57EA364D50A277C53EB21B13	1B48018C6AB05C2274F13B9F

deterministic bitmask selection method and the probabilistic method, but use the former in the implementation of P2P while the later will be used in the subsequent experiments where the system scales.

5.2 Implementation Results

We evaluate the performance of the proposed missing tag identification protocols, namely P2M and P2P. We would like to note that this paper focuses on performance comparison in the same settings rather than maximizing the throughput.

Protocol Investigation. Before the formal comparison, we first present how the deterministic bitmask selection method works. We start with $n = 5$ tags whose EPCs are listed in the first column of Table 2, i.e., tags x_1—x_5. Running Algorithm 1, we can first set the bitmask $BM = (000_2, 6, 1, 0_2)$ to query tag x_3, then use $BM = (000_2, 9, 1, 0_2)$, $BM = (000_2, 11, 1, 0_2)$, $BM = (000_2, 37, 1, 0_2)$, $BM = (000_2, 39, 1, 0_2)$ in sequence to query tags x_4, x_1, x_2, x_5, respectively. That is to say, it is sufficient for P2P to use one-bit bitmask in this case. For illustration, we take a toy example where only the first two words of tag EPCs are searched, as shown in Fig. 8. Comparing this example with the prior, we can observe that searching more positions in EPC will yield shorter bitmasks.

We further execute Algorithm 1 to build the bitmasks for the cases of $n = 10$ and $n = 20$ corresponding to the first two columns and all tags in Table 2, respectively. The results for $n = 5, 10, 20$ are shown in Tables 3, 4, and 5, respectively. Note that we employ $\texttt{MemBank} = 000_2$ in P2P, and we just list $(p, l, Mask)$ for each tag for illustrative clarity.

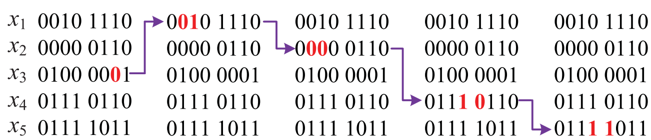


Fig. 8. The bitmasks used in P2P. There are five tags and we present the first two words of EPCs in binary. we can first set the bitmask $BM = (000_2, 6, 1, 0_2)$ to query tag x_3, then use $BM = (000_2, 1, 2, 01_2)$, $BM = (000_2, 1, 2, 00_2)$, $BM = (000_2, 3, 2, 10_2)$, $BM = (000_2, 3, 2, 11_2)$ in sequence to query tags x_1, x_2, x_4, x_5, respectively.

Protocol Comparison. From this part, we begin to compare the performance of P2M with P2P using the deterministic bitmask selection method in terms of execution time spent in identifying all missing tags and detecting the first missing tag under three different tag-to-reader encoding methods supported by an ImpinJ reader, namely M2, M4, and M8.

First, we investigate the impact of overall tag population n on the performance of P2M and P2P. To this end, we fix the number of missing tags $m = 0$ while increasing n from 5, to 10, to 20. As shown in Fig. 9, P2P can query all tags within significantly less time than P2M, and the performance gain soars with the increment in the number of the tags in the system. Meanwhile, the execution time of P2M experiences more sharp increase than P2P does. For example in Fig. 9a, when the tag population is 5, P2P is $1.5\times$ better than P2M. While this number increases to $5\times$ when there are 20 tags. The performance gain of P2P comes from the point-to-point design as it is able to successfully read a tag in every slot, but it takes $\mathcal{O}(n)$ slots for P2M to access a tag on average.

Second, we move to study how P2M and P2P perform under different missing tag population in the system. To do so, we fix the number of overall tags $n = 20$ while changing the number of missing tags m as $m = 4, 6, 8, 12$. The experimental results are depicted in Fig. 10. From these results, we can observe the following phenomenons:

- Overall performance: P2P remarkably outperforms P2M. Specifically, the identification cost of PMP, as shown in Fig. 10a, falls into the range between 0.56 s and 1.49 s, which is $2.8\times$ to $5.4\times$ more than that of

TABLE 3
Bitmasks for x_1-x_5

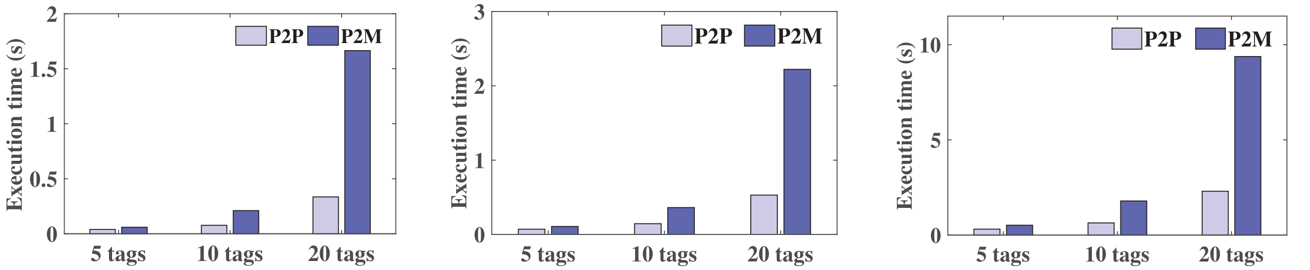
i	x_i
1	$(11, 1, 0_2)$
2	$(37, 1, 0_2)$
3	$(6, 1, 0_2)$
4	$(9, 1, 0_2)$
5	$(39, 1, 0_2)$

TABLE 4
Bitmasks for x_1-x_{10}

i	x_i	x_{i+5}
1	$(13, 2, 11_2)$	$(2, 2, 01_2)$
2	$(45, 2, 01_2)$	$(8, 2, 10_2)$
3	$(21, 2, 00_2)$	$(32, 1, 1_2)$
4	$(10, 2, 11_2)$	$(40, 2, 10_2)$
5	$(3, 2, 11_2)$	$(19, 2, 01_2)$

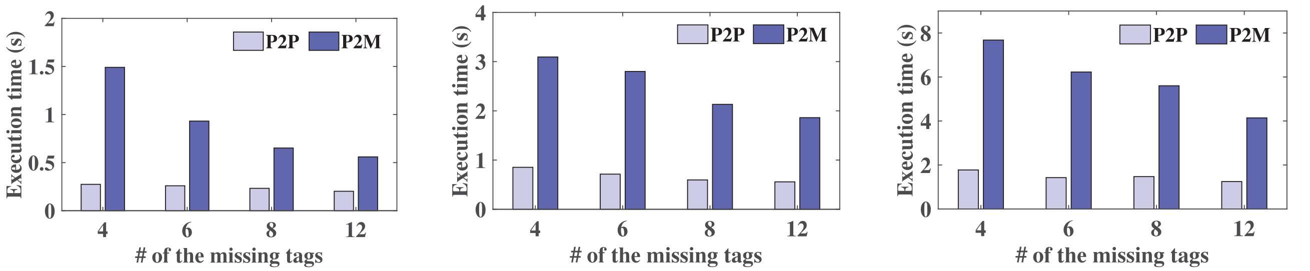
TABLE 5
Bitmasks for x_1-x_{20}

i	x_i	x_{i+5}	x_{i+10}	x_{i+15}
1	$(11, 3, 011_2)$	$(7, 3, 100_2)$	$(1, 3, 110_2)$	$(33, 3, 110_2)$
2	$(1, 3, 000_2)$	$(6, 3, 001_2)$	$(13, 3, 111_2)$	$(35, 3, 01_2)$
3	$(20, 3, 100_2)$	$(4, 4, 1111_2)$	$(52, 2, 00_2)$	$(2, 2, 01_2)$
4	$(11, 3, 100_2)$	$(74, 3, 001_2)$	$(5, 3, 101_2)$	$(4, 3, 001_2)$
5	$(70, 2, 01_2)$	$(55, 3, 001_2)$	$(68, 2, 11_2)$	$(18, 2, 00_2)$



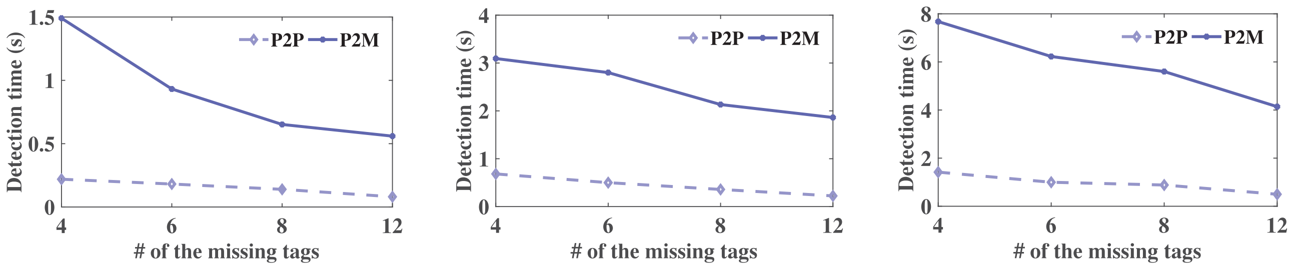
(a) Tag-to-reader encoding method: M2 (b) Tag-to-reader encoding method: M4 (c) Tag-to-reader encoding method: M8

Fig. 9. Performance comparison with different numbers of overall tags under three tag-to-reader rates: M2 (320 kbps) > M4 (68.5 kbps) > M8 (21.33 kbps).



(a) Tag-to-reader encoding method: M2 (b) Tag-to-reader encoding method: M4 (c) Tag-to-reader encoding method: M8

Fig. 10. Performance comparison with different missing tag population under three tag-to-reader rates: M2 (320 kbps) > M4 (68.5 kbps) > M8 (21.33 kbps).



(a) Tag-to-reader encoding method: M2 (b) Tag-to-reader encoding method: M4 (c) Tag-to-reader encoding method: M8

Fig. 11. Performance comparison in terms of detection time indicating the time of finding the first missing tag.

P2P. In the other tag-to-reader rates, P2P achieves at least $3.3\times$ performance gain over P2M. This is primarily due to the point-to-point query paradigm that reads tags in sequence while P2M needs more time to tackle collisions.

- Impact of missing tags: As the number of missing tags increases, the execution time of P2M decreases more significantly than P2P. For instance in Fig. 10a, the reduction of P2M is 62.2 percent, which is 2.4 times that of P2P. This can be interpreted as follows: the increase of missing tags reduces tag collisions in P2M but has a lower impact on P2P as it employs point-to-point query.

Under the same settings as the above, we further compare P2M and P2P in terms of missing tag detection time that is the time spent in finding the first missing tag. It can be observed from Fig. 11 that P2P is able to detect the first missing tag within quite less time than P2M. In particular, When there are 12 missing tags, it takes P2M with M2 nearly $7\times$ time as much as P2P to find the first missing tag. The performance gap between them reaches over $8\times$ when M4 and M8 are used. Look at Figs. 10 and 11, we can also find that the detection time of P2P significantly reduces especially in the presence of more missing tags while that of P2M does not change. This difference is resulted from the nature of P2P and P2M that the former can learn existence or absence of a tag in each slot but the latter cannot know tag states until the execution of the whole frame. That said, P2P can find a missing tag after probing $n-m$ tags in the worst case while P2M is expected to query all n tags.

Correctness of the Probabilistic Bitmask Selection Method. Having implemented P2M and P2P with 20 ImpinJ tags, we move to confirm the correctness of the probabilistic bitmask selection method in this part. Revisiting Table 2 where the 20 tag EPCs are listed, we first check whether Algorithm 2 works in 5-tag scenario. To assess the reliability of the probabilistic method, we set $\alpha = 0.99$ and 0.999, and run Algorithm 2 for $\frac{1}{1-\alpha}$ times. Each time we randomly select 5 out of 20 tag EPCs. If bitmask collisions among tags arise more than one times, we claim the failure of Algorithm 2. We record in Table 6 the combinations of l and z that fulfill the required probability. The results show that with α increased Algorithm 2 needs to use longer bitmask or run more

TABLE 6
Bitmask Length l and Execution Rounds z When $n = 5$

α \ l	3	4	5	6	7	8–10	11	12	13	14
0.99	20	8	5	3	2	2	1	-	-	-
0.999	30	12	7	4	3	3	3	3	3	1

TABLE 7
Bitmask Length l and Execution Rounds z: (l, z)

α \ n	50	100	150	200	250	300
0.99	(11, 6)	(13, 6)	(15, 4)	(15, 6)	(16, 5)	(17, 4)
0.999	(12, 6)	(14, 7)	(15, 6)	(16, 6)	(16, 5)	(18, 5)

rounds, which is in correspondence with the analytical results. Moreover, given an α, the increase of either l or z can yield a smaller value of the other, confirming the trade-off between communication overhead and computational complexity.

To evaluate the impact of system scale, we increase the number of tags from 50 to 300 with step length of 50, and generate tag EPCs at random. From the results, we observe that Algorithm 2 can achieve the required probability α with the tag population increased. Since the maximum bitmask length can be directly computed from (1) with $z = 1$, we only list the combinations of the minimum bitmask length and execution rounds that make Algorithm 2 successful in Table 7. We can find that either bitmask length or execution rounds increase when the system scales or the required success probability becomes higher, which corresponds to the analytical result.

Performance Evaluation under Larger Systems. We further show how the time efficiency of the proposed protocols changes as system scales up. To this end, we set parameters following the Gen2 standard and specification of ImpinJ reader as follows: Tari $= 12.5\mu s$, BLF $= 640$ kHz. We use FM0 and M4 as encoding methods for the tag-to-reader link, respectively. Accordingly, the data rate defined by r is $1/BLF$ and $4/BLF$. The time durations are $T_1 = T_3 = \max(2.75Tari, 10r)$, $T_2 = 3r$, and $T_4 = 5.4r$. We vary the number of the overall tags from 500 to 10,000 and set $\alpha = 0.999$ when the required bitmask length l is 27, 29, 31, 33, 36 and the execution round of the probabilistic algorithm z equals to 1. In addition, define γ as the ratio of the number of the missing tags to that of the overall tags, we set it to 0.3 and 0.6. We listed the results in Tables 8 and 9.

We can observe that the increment in the execution time of P2M follows a square pattern of that in the number of the overall tags. The pattern becomes linear in P2P. Consequently P2P is considerably more time-efficient than P2M. We can also find that the ratio γ of the missing tag population has more impact on P2P than P2M. This is because the increase of γ leads to less success slots and more empty slots in P2P. And empty slot is shorter than success slot. Yet due to the change of empty slots resulted from the increase of γ in P2M, which is in the order of magnitude $\mathcal{O}(n)$, is significantly smaller than the original number of empty slots, i.e., $\mathcal{O}(n^2)$.

TABLE 8
Execution Time of P2M and P2P with FM0

Protocol	γ	500	1,000	2,000	4,000	10,000
P2M	0.3	29.86	119.10	475.70	1901.40	1,1878.48
	0.6	29.78	118.94	475.37	1,900.76	1,1876.87
P2P	0.3	0.79	1.60	3.26	6.62	16.92
	0.6	0.72	1.46	3.00	6.06	15.52

TABLE 9
Execution Time of P2M and P2P with M4

Protocol	γ	500	1,000	2,000	4,000	10,000
P2M	0.3	44.17	175.84	701.67	2,803.31	1,7508.01
	0.6	44.00	175.48	701.00	2,801.93	1,7505.00
P2P	0.3	1.085	2.19	4.44	9.0	22.82
	0.6	0.91	1.84	3.72	7.55	19.24

6 Related Work

In this section, we briefly summarize the existing missing tag monitoring protocols that can be classified into two categories: probabilistic detection and deterministic identification.

Probabilistic Missing tag Detection. This type of protocol detects a missing tag event with a predefined probability. Tan et al. initiate the study of missing tag detection and propose a solution called TRP in [22]. To detects a missing tag event, TRP first builds a virtual bitmap by using a hash function to predict response slots of tags and compares it with actual slot states measured from the response of the tags in the population. If an expected busy (singleton or collision) slot turns out to be empty, then the tag(s) corresponding to this slot is regarded to be absent. Because the probability of a collision slot to have only missing tags is very low when missing tag size is small, collision slots are less useful than the singleton ones. Given the importance of singleton slots, follow-up works [23], [24] employ multiple seeds to tune empty and collision slots to singleton slots, which increases the detection probability and thus improves time efficiency. Subsequently, the existence of unknown tags that would make an empty slot a missing tag mapped to become busy and will interfere with the detection. To deal with the interference, the work [25] and Yu et al. [26] expand the frame size in the detection with unknown tag size and design Bloom filter from the known tags to depress the unknown ones, respectively. Consider a different kind of application scenario, Yu et al. [27] design several Bloom-filter based approaches to detect missing tags in RFID systems where multi-category tags are distributed in multiple regions. More recently, Yang et al. [28] develop an on-tag hashing function that needs to write offline calculated hash values to all tags, and illustrate how to use this function to probabilistically detect missing tags.

Deterministic Missing tag Identification. Deterministic protocols are to exactly identify which tags are absent. Li et al. develop a series of identification protocols in [29] to reduce the time cost step by step by reconciling 2-collision slots and iteratively deactivating the tags of which the presence has been verified, respectively. Zhang et al. propose identification protocols in [30] which store and compare the bitmaps of tag responses in all rounds and look for changes at the corresponding bits among all bitmaps to determine the present and absent tags. Liu et al. [31] essentially combine the multi-seed method in [23] with the deactive-based method in [29] to improve the identification performance. Subsequently, Liu et al. [32] further enhance the prior work by reconciling both 2-collision and 3-collision slots and filtering the empty and unreconcilable collision slots to improve time efficiency. Recently, physical-layer information is exploited to accelerate missing tag identification. Zheng et al. [33] measure changes of signal strength in each slot and model missing tag identification using Compressing Sensing, which reduces time cost towards the same order of magnitude as missing tag population. In contrast, Chen et al. [34] use changes of signal strength in each slot to construct a Bloom filter, which can achieve the similar time efficiency while handling arbitrary number of missing tags.

Compared with the previous work, the novelty of this paper lies in that we design bitmask selection methods and conduct deterministic missing tag identification using COTS RFID devices without requirement for hash functions at tags and for writing hash values to tags.

7 Conclusion

In this paper, we have proposed two protocols enabling the missing tag identification service with COTS RFID reader and tags. Specifically, we first used Q-query to develop a point-to-multipoint protocol that operates in an analog frame slotted Aloha paradigm to collect tag EPCs. A missing tag can be found out if the collected EPC set does not contain its EPC. We then devised a point-to-point protocol that employs a bitmask to specify one tag to reply in each slot so that tag response collisions are avoided and time efficiency is improved. Moreover, we presented two bitmask selection methods to build compact bitmasks. The proposed protocols were implemented in ImpinJ reader and tags, and the extensive results showed that they are able to achieve missing tag identification task.

Acknowledgments

This work is supported in part by the Beijing Institute of Technology Research Fund Program for Young Scholars and was supported in part by a Canada NSERC Discovery Grant, and an NSERC E.W.R. Steacie Memorial Fellowship. Part of the work of K. Wang and L. Chen and R. Zhang is supported by NSF of China (no. 61672395), the CNRS PEPS project MIRFID, and NSF of China (no. 61801064).

References

[1] F. Klaus, *RFID Handbook: Radio-frecuency Identification: Fundamentals and Applications*. Hoboken, NJ, USA: Wiley, 1999.

[2] RFID Journal, "A 5-cent breakthrough," 2009. [Online]. Available: http://www.rfidjournal.com/articles/view?2295

[3] 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.

[4] Barcoding Inc., "How RFID works for inventory control in the warehouse," 2016. [Online]. Available: http://www.barcoding.com/rfid/inventory-control.shtml

[5] X. Liu, et al., "Fast identification of blocked rfid tags," *IEEE Trans. Mobile Comput.*, vol. 17, no. 9, pp. 2041–2054, Sep. 2018.

[6] C.-H. Lee and C.-W. Chung, "Efficient storage scheme and query processing for supply chain management using RFID," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 291–302.

[7] J. Liu, et al., "Efficient information sampling in multi-category rfid systems," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 159–172, Feb. 2019.

[8] J. Han, et al., "Twins: Device-free object tracking using passive tags," *IEEE/ACM Trans. Netw.*, vol. 24, no. 3, pp. 1605–1617, Jun. 2016.

[9] J. Liu, et al., "Tag-compass: Determining the spatial direction of an object with small dimensions," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.

[10] J. Wang, D. Vasisht, and D. Katabi, "RF-IDraw: Virtual touch screen in the air using RF signals," in *Proc. ACM SIGCOMM*, 2014, pp. 235–246.

[11] EPCglobal Inc., "Class-1 generation-2 UHF RFID protocol for communications at 860 mhz-960 mhz," 2005. [Online]. Available: http://www.gs1.org

[12] ImpingJ Inc., "Impingj connectivity devices." (2010). [Online]. Available: https://www.impinj.com/

[13] Thingmagic Inc., "Thingmagic products." (2011). [Online]. Available: http://www.thingmagic.com/index.php

[14] Chain Store Age, "Retailers losing billions to inventory shrink," 2017. [Online]. Available: https://nrf.com

[15] K. Finkenzeller, *RFID Handbook*. Hoboken, NJ, USA: Wiley, 2010.

[16] M. Feldhofer and C. Rechberger, "A case against currently used hash functions in rfid protocols," in *Proc. Int. Conf. Move Meaningful Internet Syst.*, 2006, pp. 372–381.

[17] C. Rolfes, A. Poschmann, G. Leander, and C. Paar, "Ultra-lightweight implementations for smart devices–security for 1000 gate equivalents," in *Proc. Int. Conf. Smart Card Res. Adv. Appl.*, 2008, pp. 89–103.

[18] Systemid, "Immediate inventory management: Everyone wins with RFID technology at Walmart," 2012. [Online]. Available: http://www.systemid.com/learn/

[19] RFID Journal, "RFID technology is boosting sales and customer engagement for retailers," 2017. [Online]. Available: https://www.raconteur.net/business/

[20] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[21] K. Beyer, et al., "On synopses for distinct-value estimation under multiset operations," in *Proc. 2007 ACM SIGMOD Int. Conf. Manage. Data*, 2007, pp. 199–210.

[22] C. C. Tan, B. Sheng, and Q. Li, "How to monitor for missing RFID tags," in *Proc. 28th Int. Conf. Distrib. Comput. Syst.*, 2008, pp. 295–302.

[23] 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 Proc. 13th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2012, pp. 95–104.

[24] 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.

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

[26] J. Yu, et al., "Finding needles in a haystack: Missing tag detection in large RFID systems," *IEEE Trans. Commun.*, vol. 65, no. 5, pp. 2036–2047, May 2017.

[27] 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.

[28] L. Yang, Q. Lin, C. Duan, and Z. An, "Analog on-tag hashing: Towards selective reading as hash primitives in gen2 rfid systems," in *Proc. ACM MobiCom*, 2017, pp. 301–314.

[29] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large RFID system," in *Proc. ACM MobiHoc*, 2010, pp. 1–10.

[30] R. Zhang, Y. Liu, Y. Zhang, and J. Sun, "Fast identification of the missing tags in a large RFID system," in *Proc. IEEE SECON*, 2011, pp. 278–286.

[31] X. Liu, et al., "A multiple hashing approach to complete identification of missing RFID tags," *IEEE Trans. Commun.*, vol. 62, no. 3, pp. 1046–1057, Mar. 2014.

[32] 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.

[33] Y. Zheng and M. Li, "P-mti: Physical-layer missing tag identification via compressive sensing," *IEEE/ACM Trans. Netw.*, vol. 23, no. 4, pp. 1356–1366, Aug. 2015.

[34] M. Chen, J. Liu, S. Chen, Y. Qiao, and Y. Zheng, "DBF: A general framework for anomaly detection in RFID systems," in *Proc. IEEE Conf. Comput. Commun.*, 2017, pp. 1–9.

Jihong Yu received the BE degree in communication engineering and the ME degree in communication and information systems from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2010 and 2013, respectively, and the PhD degree in computer science with the University of Paris-Sud, Orsay, France, in 2016. He was a research fellow with the School of Computing Science, Simon Fraser University, Canada. He currently is an associate professor with the School of Information and Electronics, Beijing Institute of Technology. His research interests include RFID technologies, wireless communications, and Internet of things.

Wei Gong (M'14) received the BS degree from the Department of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China, in 2003 and the MS and PhD degrees from the School of Software and Department of Computer Science and Technology, Tsinghua University, Beijing, China, in 2007 and 2012, respectively. He was a research fellow with the School of Computing Science, Simon Fraser University, Canada, and is a professor with the School of Computer Science and Technology, University of Science and Technology. His research interests include backscatter networks, mobile computing, and Internet of things. He is a member of the IEEE.

Jiangchuan Liu (S'01-M'03-SM'08-F'17) received the BEng (cum laude) degree from Tsinghua University, Beijing, China, in 1999, and the PhD degree from The Hong Kong University of Science and Technology, in 2003. He is currently a full professor (with University Professorship) with the School of Computing Science, Simon Fraser University, British Columbia, Canada. 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 a co-recipient of the Test of Time Paper Award of IEEE INFOCOM (2015), ACM TOMCCAP Nicolas D. Georganas Best Paper Award (2013), and ACM Multimedia Best Paper Award (2012). He is a fellow of the IEEE and an NSERC E.W.R. Steacie Memorial Fellow.

Lin Chen (S'07-M'10) received the BE degree in radio engineering from Southeast University, China, in 2002, the engineer diploma from Telecom ParisTech, Paris, in 2005, and the MS degree in networking from the University of Paris 6. He currently works as an associate professor with the Department of Computer Science, University of Paris-Sud. He serves as chair of the IEEE Special Interest Group on Green and Sustainable Networking and Computing with Cognition and Cooperation, IEEE Technical Committee on Green Communications and Computing. His main research interests include modeling and control for wireless networks, distributed algorithm design, and game theory. He is a member of the IEEE.

Kehao Wang received the BS degree in electrical engineering, the MS degree in communication and information system from the Wuhan University of Technology, Wuhan, China, in 2003 and 2006, respectively, and the PhD degree from the Department of Computer Science, University of Paris-Sud XI, Orsay, France, in 2012. He currently works as an associate professor with the Department of Information Engineering, Wuhan University of Technology. His research interests include cognitive radio networks and wireless network resource management.

Rongrong Zhang eceived the BE and ME degrees in communication and information systems from the Chongqing University of Posts and Telecommunications, Chongqing, China, in 2010 and 2013, respectively, and PhD degree in computer science at the University of Paris Descartes, France, in 2017. She was a research fellow in the School of Electrical Engineering and Computer Science, University of Ottawa, Ontario, Canada. She currently is an assistant professor in the Information Engineering College at Capital Normal University, Beijing, China. Her research interests focus on wireless communications and wireless area body networks for e-health applications.