# A Multiple Hashing Approach to Complete Identification of Missing RFID Tags

Xiulong Liu*, Keqiu Li*, Geyong Min‡, Yanming Shen*, Alex X. Liu § and Wenyu Qu†
*School of Computer Science and Technology, Dalian University of Technology, China
‡School of Computing, Informatics and Media, University of Bradford, UK
§Department of Computer Science and Technology, Nanjing University, China
†School of Information Science and Technology, Dalian Maritime University, China
Corresponding Author: Keqiu Li {keqiu@dlut.edu.cn}

***Abstract*—Owing to its superior properties such as fast identification and relatively long interrogating range over barcode systems, Radio Frequency Identification (RFID) technology has promising application prospects in inventory management. This paper studies the problem of missing tag identification, which is important in practice. Time efficiency is the key performance metric of missing tag identification. However, the existing protocols are ineffective in terms of execution time and can hardly satisfy the requirements of real-time applications. In this paper, a Multi-hashing based Missing Tag Identification (MMTI) protocol is proposed, which achieves better time efficiency by improving the utilization of the time frame used for identification. Specifically, the reader recursively sends bitmaps that reflect the current slot occupation state to guide the slot selection of the next hashing process, thereby changing more empty or collision slots to the expected singleton slots. We investigate the optimal parameter settings to maximize the performance of the MMTI protocol. Efficient methods are also proposed to extend the MMTI protocol to the multi-reader scenarios. We discuss the case of channel error and propose the countermeasures to make the MMTI workable in the scenarios with non-perfect communication channel. Extensive simulation experiments are conducted to evaluate the performance of MMTI, and the results demonstrate that the proposed MMTI protocol significantly outperforms the state-of-the-art protocols in both single- and multi-reader scenarios.**

***Index Terms*—RFID, missing tag identification, multi-hashing.**

## I. Introduction

Radio Frequency Identification (RFID) devices are widely deployed in many application scenarios such as supply chain management [1] [2] and inventory control [3] [4] [5], where the missing tag problem is an important but challenging issue [6] [7] [8] [9]. This issue can be generally classified into three categories: (1) missing-tag event detection protocols focus on detecting whether any RFID tags are missing or not instead of exactly pinpointing which tags are missing [6] [7] [9]; (2) probabilistic missing-tag identification protocols can pinpoint which RFID tags are missing (i.e., find out the ID information of the missing RFID tags), but do not guarantee to report all missing ones (e.g., Protocol 1 in [8]); (3) complete missing-tag identification protocols focus on pinpointing the ID information of all missing RFID tags and guarantee 100% reporting (e.g., IIP in [10], Protocols 2 and 3 in [8]).

This paper focuses on the third type of sub-problem—complete missing-tag identification, which is of great importance and irreplaceable in some scenarios. For example, in a warehouse that suffers from burglary, it is essential to monitor the items. However, simply detecting the missing-tag event is not enough. We also need to obtain the detailed information (e.g., category, price, etc.) of the missing items so as to assess the seriousness of the loss and take different countermeasures. In this situation, complete missing tag identification protocol is preferred. Despite of its practical importance, the problem of complete identification of missing tags is still under-investigated and solicits new efficient solutions.

To the best of our knowledge, the existing advanced protocols for addressing the problem of missing tag identification include: (1) the Iterative ID-free Protocol (IIP) proposed in [10]; and (2) a group of protocols proposed in [8]. In what follows, we will present and analyze these schemes, respectively.

The IIP scheme is based on the classical *Framed Slotted Aloha* communication mechanism. By pseudo-randomly employing a hash function (shared by both the reader and tags), the reader can predict the singleton slots, in which only one tag is expected to respond; the collision slots, in which two or more tags are expected to respond; and the empty slots, in which no tag is expected to respond. Based on the observation of the actual state of each slot, the reader identifies the missing tags. In IIP, for time-efficiency, the tag response is 1-

bit. If an expected singleton slot turns out to be empty, the reader asserts that the tag corresponding to this slot is missing. Although many recent studies (e.g., [11], [12], and [13]) have been reported to make use of the collision slots, IIP does not leverage the expected collision slots. The reasons are exemplified as follows. For an expected collision slot, if all tags corresponding to this slot respond as expected, the reader *senses a busy slot*. On the other hand, if only some of them are missing, and thus at least one tag is present and responds, the reader *still senses a busy slot*. Clearly, in this case, the reader cannot identify the missing tags. The reader can identify the missing tags during an expected collision slot unless all the tags corresponding to this slot are missing and this slot turns out to be empty. But this probability is very small. Therefore, without loss of generality, IIP only leverages the expected singleton slots. Whereas, the expected empty slots and the expected collision slots that account for nearly 48% are not used and become wasted, which leads to the deficiency of IIP.

In [8], Zhang *et al.* proposed three protocols to identify the missing tags in the multi-reader scenarios. Protocol 2 and 3 in [8] can identify *all* the missing tags but Protocol 1 cannot. The authors in [8] showed that their protocols reduce the time for identifying all the missing tags by up to 75% when compared with IIP. Their superiority over IIP benefits from the cooperation among RFID readers. In the single reader scenarios (or in the scenarios where the number of readers is small), IIP still runs faster than the protocols in [8].

This paper investigates a "multi-hashing" approach to relieve the deficiency of IIP and proposes the Multi-hashing based Missing Tag Identification (MMTI) protocol. In this protocol, multiple hashing processes are repeated to increase the proportion of the expected singleton slots—improving the utilization of time frame. The challenge is how to guarantee the achieved singleton slots will not be selected in the next hashing process. Accordingly, we investigate a *bitmap* to guide the next hashing process. Specifically, since the slot occupation states of the frame is predictable to the reader, it could construct a bitmap, in which '1s' indicate the singleton slots that cannot be selected in the next hashing process; and '0s' indicate the collision or empty slots that can be selected in the next hashing process. The reader then broadcasts this bitmap to guide the next hashing process. To maximize the performance of the proposed MMTI, we investigate the optimization of the involved parameters including frame size and the hashing count. Sufficient analysis and experiments manifest that MMTI reduces 32% and about 90% of the required execution time, when compared to IIP and Protocol 3 (the best protocol in [8]), respectively, in the *single reader* scenarios. In reality, a single reader usually cannot cover all the tags due to the limitation of communication range. Hence, this paper also proposes a "Bloom Filter" based method to extend MMTI to the multi-reader scenarios. The simulation results demonstrate that the proposed MMTI outperforms IIP and Protocol 3 in the multi-reader scenarios by reducing about 90% and 50% of the execution time, respectively. The major contributions of this paper are summarized as follows:

1) A Multi-hashing based Missing Tag Identification (MMTI) protocol is proposed to reduce the proportion of the expected empty slots and expected collision slots that are not leveraged and trigger the deficiency of the existing IIP scheme.
2) The optimal parameter settings of the proposed MMTI protocol are thoroughly investigated in order to maximize its performance.
3) Bloom Filter technique is introduced to further accelerate the proposed MMTI scheme for use in the multi-reader scenarios.
4) Extensive simulation experiments are conducted to evaluate the performance of the proposed MMTI protocol and manifest its efficiency over the other related protocols.

The rest of this paper is organized as follows. The related work is reviewed in Section II. Section III describes the problem to be addressed in this paper and presents the system model. We propose the MMTI protocol and present the related proofs in Section IV. In Section V, extensive simulation experiments are conducted to evaluate the performance of the MMTI protocol. Finally, this paper is concluded in Section VI.

## II. Related Work

In the current literature, many studies have been conducted to address various important problems in the field of RFID. Most of the previous work concentrated on the problem of tag identification, which is to identify the IDs from a large number of tags as quickly as possible. The existing tag identification schemes can be generally classified into Aloha-based schemes [14] [15] [16] and tree-based schemes [17] [18] [19]. In recent years, a new technical problem of information collection has attracted much attention, which aims to collect the information (e.g., the environment temperature) generated by the sensor-augmented RFID tags [20], instead of just simple ID information. In [20], Chen *et al.* proposed a multi-hashing approach named Multi-hash Information Collection protocol (MIC) to increase the utilization of the time frame. In MIC, the reader assigns tags to slots by using $k$ hashing functions,

which is equivalent to "multi-hashing". Intuitively, a single hashing function can generate about $37\%$ singleton slots (when the number of tags is the same as the number of slots). Two hashing functions can generate $37\% + (1 - 37\%) \times 37\%$ singleton slots. The more hashing functions are employed, the more singleton slots can be achieved. Then the reader sends a hash-selection vector to inform the tags which hashing functions they should adopt. However, in MIC, the tags have to store the whole hash-selection vector when searching their proper slots, which poses challenges on the very limited storage capacity of RFID tags, especially for the passive tags.

The *missing tag* problem, which is of great practical importance, also attracted much attention. In [6], Tan *et al.* proposed the Trust Reader Protocol (TRP) to detect the missing-tag event with a given probability when the number of missing tags exceeds a threshold. In [7], Luo *et al.* investigated *birthday paradox* to detect the missing tag event and presented the corresponding energy-time tradeoff. To further accelerate the detection process, a "multi-hashing" method was proposed to increase the utilization of the time frame. The Multi-Seed Missing-tag Detection (MSMD) [9] uses multiple hashing seeds to increase the proportion of the expected singleton slots in the time frame. Specifically, the reader uses a hashing seed to map the tags to slots, which is logically equivalent to a hashing process (multiple seeds correspond to multiple hashing processes). A slot may be an expected singleton using a seed, but a collision using another seed. To maximize the proportion of the expected singleton slots, the reader selects the best hashing seed for each slot such that this slot can be singleton. After that, the reader constructs a seed-selection vector $V$, which contains $f$ selectors, one for each slot in the time frame. Then, the reader broadcasts the seed-selection vector $V$ to tell the tags to choose the seed thereby guiding their slot selection (i.e., the multi-hashing processes). MSMD also suffers from the storage limitation. The protocols reported in [6] [7] [9] are able to detect the missing-tag event only, but cannot exactly pinpoint which RFID tags are missing.

The Iterative ID-free Protocol (IIP) presented in [10] can completely identify the missing tags and guarantee 100% reporting. As aforementioned, the singleton slots in the IIP protocol are used to verify the presence of the tags. The expected empty slots and collision slots contribute nothing to missing tag identification and are wasted. The inefficiency due to the collision slots has been noticed in [10], and the authors investigated a method to turn some of the collision slots into singleton slots. However, the empty slots are not discussed and still wasted directly. According to the theoretical analysis in [10], we find that the expected empty slots and expected collision slots still account for nearly 48% even though they have tried to turn some collision slots into singleton slots. Clearly, IIP still has a large space for improvement. In reality, a single reader usually cannot cover the whole monitoring area. In [8], Zhang *et al.* proposed three protocols to identify missing tags in the multi-reader scenarios, where all the readers perform *synchronized and parallel* scans. Protocol 2 and 3 in [8] can identify *all* the missing ones while Protocol 1 cannot. The authors in [8] claimed that their protocols reduce the time for identifying all the missing tags by up to 75% in comparison to IIP. Their superiority over IIP benefits from the cooperation of the readers. In the single reader scenarios (or in the scenarios where the number of readers is small), IIP still runs faster than the protocols in [8].

Actually there is another type of "missing" tag problem [21], in which the "missing" tags represent the tags that are left unread due to errors in the communication link towards the reader, e.g., caused by the obstacles in the radio path. In other words, the study in [21] investigated the problem of tag identification (i.e., reading the tag IDs) in a scenario with non-perfect communication channels. The authors studied how to minimize the probability of missing (miss-reading) a tag, which is different from the missing tag identification problem addressed in this study.

## III. System and Problem

### A. Problem Description

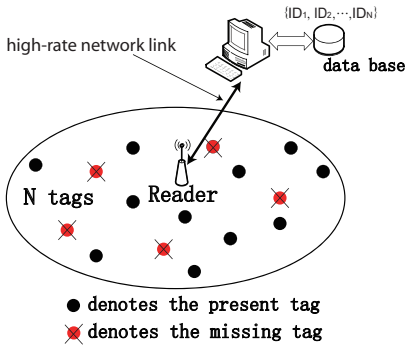


Fig. 1: Problem description.

Assume each item under monitoring in a warehouse is attached with an RFID tag. An RFID reader located in the center of this warehouse periodically scans the tags within its interrogation range. This scenario is illustrated

in Fig. 1 where there is one reader and $N$ tags and all the tags are within the range of this reader. It is worth noting that the proposed MMTI protocol can also work in the scenario with multiple readers deployed. For the purpose of clarity, we first consider the case of a single reader. After that, we present the method to extend the protocol to multi-reader scenarios. Let $S_{all}$ denote the tag set, where $S_{all} = \{Tag_1, \ldots, Tag_i, \ldots, Tag_N\}$. All tags are equipped with the same uniform hash function $H(\cdot)$, and each of them possesses a unique tag ID. The reader is able to access the tag IDs stored in a database [6] [7] [8] [9] [10]. Some tags may be missing due to theft or other reasons. The problem studied in this paper is to completely identify all the missing RFID tags in a fast way. Table I summarizes the notations used this paper.

TABLE I: Notations

| Symbols | Descriptions |
|---|---|
| $N$ | The number of tags in the system |
| $S_{all}$ | The set of all tags in the system |
| $S_{miss}$ | The set of missing tags |
| $ID_i$ | The ID of $Tag_i$ |
| $S$ | The set of tags whose presence has not been verified |
| $N^*$ | The number of tags participating in this slotted frame |
| $f$ | The frame size, i.e. the number of slots available in this frame |
| $slot_x$ | The $x^{th}$ slot in a time frame |
| $\rho$ | The load factor, $N^*/f$ |
| $R$ | The random number that is fresh in each round |
| $H(\cdot)$ | The hash generator with a uniform distribution |
| $m$ | The hashing count in a round of MMTI |

### B. Time Slots

The proposed MMTI protocol is based on the slotted Aloha communication mechanism which will be briefed below. The communication between the reader and the tags is in a time-slotted way. The reader synchronizes the slots by broadcasting the *end_slot* command. Each tag has a *slot_clock* which is initialized with a random slot number. A tag down-counts its *slot_clock* one each time when the reader indicates that the current slot has ended. A tag responds when its *slot_clock* reaches zero. According to Philips I-Code [22], we have the following two claims: (1) if each tag response is at least 10 bits, the reader can distinguish three types of slots: the *empty* slot in which no tag responds in the slot; the *singleton* slot in which exactly one tag responds; and the *collision* slot in which more than one tag responds. (2) if each tag response is less than 10 bits (e.g., 1 bit only), the reader can distinguish two types of slots only: the *idle* slot in which no tag responds; and the *busy* slot in which at least one tag responds. The above two claims have also been adopted in literature [5] [7] [9] [10].

Based on Philips I-Code [22], Li *et al.* [10] presented a method of classifying the time slots based on their length: *tag slots*, *long slots* and *short slots*. The length of a tag slot is denoted as $t_{id}$, which allows the transmission of a tag ID (96 bits), either from the reader to the tags or from a tag to the reader. The length of a long slot is denoted as $t_l$, which can afford transmitting a long response containing 10-bit information. The length of a short slot is denoted as $t_s$, which allows the transmission of a short response conveying only 1-bit information. This gives an approximate transmission rate of $96/(2.4 * 10^{-3}) = 40$Kb/s [8].

## IV. Multi-hashing based Missing Tag Identification Protocol

Recall the state-of-the-art IIP scheme, empty and collision slots accounting for about $48\%$ are wasted and thus trigger its deficiency. To overcome this problem, a method that can increase the proportion of singleton slots is desirable. This paper is inspired by [20] [9] and also leverages multi-hashing idea to relieve the inefficient of the IIP scheme. But it is worth noting that the detailed protocol design of our multi-hashing idea is different from that in [20] [9]. And the proposed MMTI protocol does not suffer the limitation of storage. In this section, we first present the intuitive advantage of multi-hashing that inspires the proposed MMTI protocol. After that, we give the detailed protocol design and investigate the involved parameter settings including the frame size $f$ as well as the hashing count $m$ to maximize its performance.

### A. Intuitive Motivation of Multi-Hashing

The MMTI protocol is proposed to reduce the expected empty slots and the expected collision slots. Fig. 2 illustrates the intuitive motivation of multi-hashing. Before exemplifying the basic multi-hashing idea, for the purpose of clarity, we define two types of tags: (1) singleton tag that picks an expected singleton slot in the multi-hashing process; (2) collision tag that picks an expected collision slot. The process that each tag pseudo-randomly picks a slot from a given slot set is referred to as a hashing. As illustrated in Fig. 2 (a), after the first hashing, 4 singleton slots (marked by *circle*) can be used to identify the presence of the corresponding 4 singleton tags (marked by *circle*). Whereas, 6 slots (empty or collision) are not used and wasted. Clearly, if the hashing is implemented for just once, the slotted frame is of low utilization.

A multi-hashing method can improve the efficiency. As illustrated in Fig. 2(b), keeping the singleton mapping (derived from the first hashing) unchanged, we further implement the second hashing between the 6 collision tags and the 6 non-singleton (empty or collision) slots to improve the utilization of the slotted frame. In other words, the singleton tags will not participate in the second round of hashing. Moreover, only the non-singleton slots can be picked in the second hashing. After that, we can obtain 2 more singleton slots (marked by *hexagon*).

Note that, the hashing process is just to get a virtual mapping between the slots and the tags. The slotted time frame has not been executed yet. As exemplified in Fig. 2, after two hashing processes, 6 tags are assigned to exclusive slots. That is, 6 slots are supposed to be singleton in the following actual time frame. If we iteratively implement more hashing processes, more slots will become singleton. So intuitively, the multi-hashing based protocol can better utilize the time slots thereby relieving the deficiency of the IIP scheme.

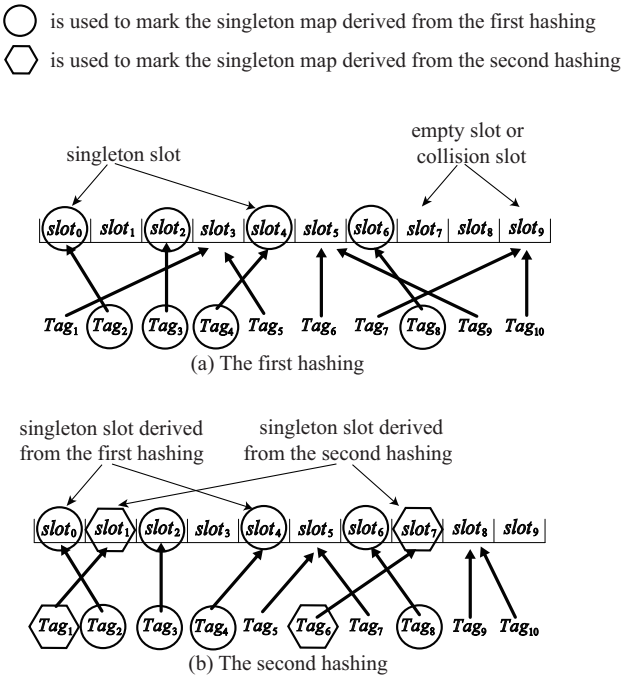


Fig. 2: Multi (two)-hashing processes.

### B. Protocol Design

The proposed MMTI protocol consists of multiple rounds, each of them consists of three phases: *Pre-identification* phase, *Identification* phase and *Acknowledgment* phase. In the *Pre-identification* phase, the hashing process is iteratively implemented for $m$ times. In an arbitrary hashing process, each non-singleton tag pseudo-randomly chooses a slot from the non-singleton slot set. For clarity of description, we first describe the MMTI protocol with $m = 2$ (i.e., in each round, the hashing process is iteratively implemented twice in the *Pre-identification* phase). After the first phase, each RFID tag determines a time slot within the following frame and responds in the picked slot. Since the tags have ability to send a 1-bit response to the reader [9] [10] [11] [23], for the purpose of saving time, the proposed MMTI lets each tag respond only 1 bit information to announce its presence in the picked slot. According to the actual state of an expected singleton slot, the reader identifies the presence of the corresponding tag. Specifically, if an expected singleton slot turns out to be an empty slot, the tag corresponding to this slot is missing. In the *Acknowledgment phase*, the reader transmits a $bitmapAck$ to inform the tags if they have declared their presence successfully in this round. According to the $bitmapAck$, each tag determines to participate in the next round or not. In an arbitrary round, the details of three phases are presented as follows.

*1) Pre-identification phase:* In the first phase, the proposed MMTI protocol aims to generate $(m-1)$ bitmaps on the reader side when $m$ hashings are made. Generally, between any two consecutive hashing processes, each bitmap represents the slot occupation states of the current round of hashing, and is used to guide the slot selection of the next hashing. After the bitmaps are generated, they are transmitted one by one. Then the tags go through the $(m-1)$ bitmaps, one after another, until they find their slots to respond and the announce their presence. The detailed procedures are presented as follows.

The reader first broadcasts a query $\langle R, f\rangle$, where $R$ is a random number (fresh in each round) and $f$ is the frame size. Each tag, say $Tag_i$, receives the query $\langle R, f\rangle$ and uses the hash generator $H(\cdot)$, $R$, $f$ and its $ID_i$ to pseudo-randomly pick $slot_x$, where $x = H(ID_i, R) \mod f$, whose result is within $[0, f-1]$. This is the first hashing process.

By employing the same $H(\cdot)$, $R$ and $f$, the reader can predict the locations of the expected empty slots, the expected singleton slots and the expected collision slots after the first hashing process. The reader constructs an $f$-bits $bitmap$, in which '1s' represent the expected singleton slots that cannot be selected in the next hashing process; '0s' represent the expected empty slots or expected collision slots that can be selected in the next hashing.

The reader counts the number of non-singleton (empty

or collision) slots denoted as $z$, and broadcasts it together with a new random number $R'$. Each tag receives $z$ and $R'$ and computes $j = H(ID_i, R') \mod z$, whose result is within $[0, z-1]$, where $j$ means that the $j^{th}$ non-singleton slot is the candidate slot that $Tag_i$ may pick in the second hashing process. The reader broadcasts the $bitmap$ constructed above. Each tag, say $Tag_i$, receives this $bitmap$ sequentially. $Tag_i$ checks the $x^{th}$ bit in this $bitmap$ and finds out the index (denoted as $y$) of the $j^{th}$ '0' in the $bitmap$ as well. If $Tag_i$ finds that the $x^{th}$ bit in the $bitmap$ is '1', it learns that the $slot_x$ derived from the first hashing is an expected singleton slot, then it picks the $slot_x$. Otherwise, $Tag_i$ picks the $slot_y$, which is equivalent to the second hashing.

Because the reader knows all the parameters, *tags-slots mapping results* of the above two-hashing processes are predictable to the reader.

*2) Identification phase:* In this phase, the MMTI protocol actually executes the slotted frame. Similar with IIP [10], the proposed MMTI leverages the observations of the expected singleton slots to identify the missing tags. Specifically, if the reader receives a response in an expected singleton slot, it can assert the presence of the corresponding tag. On the other hand, if an expected singleton slot turns out to be empty, the corresponding tag must be missing and is pinpointed by the reader. To check if an expected singleton slots is empty or non-empty, 1-bit tag response is adequate.

At the end of this phase, the reader constructs the $f$-bits $bitmapAck$. If $slot_k$ is expected to be a singleton slot, the reader sets $bitmapAck[k]$='1'. Otherwise, the reader set $bitmapAck[k]$='0'.

*3) Acknowledgment phase:* When all the slots in the frame have been counted, the reader broadcasts the $bitmapAck$ constructed above. Each tag receives the $bitmapAck$ sequentially and checks whether it has been verified successfully. Specifically, if a tag picked and responded in the $slot_k$, it checks the $k^{th}$ bit in the $bitmapAck$. If the $k^{th}$ bit is '1', the tag learns that its presence is noticed by the reader, then this tag keeps silent and will not participate in the following rounds. Otherwise, this tag continues to participate in the next round.

In this round, a fraction of the tags are verified and the other tags will participate in the next round. The MMTI protocol repeats the round described above until all the presence of all tags is verified.

In the Pre-identification phase and the Acknowledgement phase, a tag sequentially receives the bitmaps which may be very long. As there is no need for a tag to store the whole bitmap, the long bitmap is divided into segments of 96-bits for transmission in long slots. A segment of the bitmap becomes useless and can be erased after being checked by a tag. Hence, a tag needs to store only one segment (96-bits) at the same time. So the storage requirement is not an obstacle to the proposed MMTI protocol.

A random number $R$ is picked in the above protocol design to perform the hashing. It is worth noting that, because the reader knows all tag IDs, instead of randomly picking $R$, it is able to select an ideal $R$ that can achieve a better mapping between the tags and the slots. Then, the performance of the proposed MMTI can be further improved. But for a fair comparison with the other two main benchmark protocols proposed in [8], [10], where $R$ is also picked as a random number, this paper still randomly picks $R$ in each process of hashing.

Instead of broadcasting a bitmap for each hashing round, the reader may persistently pre-compute the hashing processes until each tag selects a singleton slot. Then, the reader broadcasts the map to tell the tags which slots they should pick. As a result, a single frame is adequate to complete the identification of all missing tags. This persistent hashing scheme has a good potential in terms of time-efficiency, but requires each tag to store the whole map, which poses challenges on the very limited storage capacity of the RFID tags. The MMTI protocol proposed in this paper is specifically for the tags with low storage space.

### C. Choosing an Optimal Frame Size $f$

TABLE II: Notations used in the following proving process

| Symbols | Descriptions |
|---|---|
| $s_1$ | The expected number of singleton slots derived from the first hashing |
| $s_1'$ | The expected number of non-singleton slots derived from the first hashing |
| $n_1$ | The expected number of singleton tags derived from the first hashing |
| $n_1'$ | The expected number of collision tags derived from the first hashing |
| $s_2$ | The expected number of new singleton slots derived from the second hashing |
| $n_2$ | The expected number of new singleton tags derived from the second hashing |
| $N_{total}$ | The total number of expected singleton slots derived from two hashing processes, i.e., $N_{total} = s_1 + s_2$ |
| $T$ | The whole execution time of this round |

The MMTI protocol repeats multiple rounds to identify the presence of all tags. In an arbitrary round, let $N^*$ denote the number of the tags that should participate in

this round. Clearly, $N^* = N$ in the first round. In what follows, we will present the detailed analysis of how to choose the optimal frame size $f$ in order to achieve the best time-efficiency in each round. Table II summaries the notations used below.

*Theorem 1:* For the special case of $m = 2$ (i.e., two hashing processes are conducted in each round), MMTI achieves the best time-efficiency if we set $f = N^*$ in each round.

*Proof:* First, let us consider how many tags are expected to be singleton in this round. In the first hashing process, given the frame size *f*, each tag has the probability $\frac{1}{f}$ to select a specific slot during the first hashing process. For $N^*$ tags in total, the probability $p_1$ that a slot becomes expected singleton slot is given as follows:

$$\begin{aligned} p_1 &= \binom{N^*}{1} \times \frac{1}{f} \times (1-\frac{1}{f})^{N^*-1} \\ &\approx \frac{N^*}{f} \times e^{-\frac{N^*}{f}} \\ &= \rho \times e^{-\rho} \end{aligned} \tag{1}$$

Since $f$ is normally large, in Eq. (1), $p_1$ can be simplified to $\frac{N^*}{f} \times e^{-\frac{N^*}{f}}$. For the clarity of presentation, we denote $\frac{N^*}{f}$ as $\rho$, and $\rho$ is referred to as the load factor meaning the number of tags "loaded" in the current frame.

Each of the $f$ slots in the current frame has the probability $p_1$ to be a singleton slot. Let $s_1$ be the expected number of the singleton slots derived from the first hashing. We have:

$$s_1 = f \times p_1 = f \times \rho \times e^{-\rho} = N^* \times e^{-\rho} \tag{2}$$

Let $s_1'$ denote the expected number of non-singleton slots derived from the first hashing process. Clearly, $s_1'$ can be written as:

$$s_1' = f - s_1 = f - N^* \times e^{-\rho} \tag{3}$$

Let $n_1$ and $n_1'$ denote the expected number of the singleton tags and the expected number of the collision tags, respectively. $n_1$ and $n_1'$ can be given by:

$$n_1 = s_1 = N^* \times e^{-\rho} \tag{4}$$

$$n_1' = N^* - n_1 = N^* - N^* \times e^{-\rho} \tag{5}$$

In the *second* hashing process, the $n_1'$ collision tags re-select slots from the set of $s_1'$ non-singleton slots. The probability $p_2$ that a certain non-singleton slot becomes a singleton slot can be given as follows:

$$\begin{aligned} p_2 &= \binom{n_1'}{1} \times (\frac{1}{s_1'}) \times (1-\frac{1}{s_1'})^{n_1'-1} \\ &= n_1' \times \frac{1}{s_1'} \times (1-\frac{1}{s_1'})^{n_1'-1} \end{aligned} \tag{6}$$

Each of the $s_1'$ non-singleton slots has the probability $p_2$ to be a singleton slot in the second hashing process. Let $s_2$ denote the expected number of new singleton slots derived from the second hashing process. According to Eq. (6), $s_2$ is given by:

$$s_2 = s_1' \times p_2 = n_1' \times (1-\frac{1}{s_1'})^{n_1'-1} \approx n_1' \times e^{-\frac{n_1'}{s_1'}} \tag{7}$$

According to Eqs. (3) and (5), by replacing $s_1'$ and $n_1'$ in Eq. (7), we have:

$$s_2 \approx (N^* - N^* \times e^{-\rho}) \times e^{-\frac{N^* - N^* \times e^{-\rho}}{f - N^* \times e^{-\rho}}} \tag{8}$$

The expected number, $N_{total}$, of singleton slots derived from two hashing processes is given as follows:

$$\begin{aligned} N_{total} &= s_1 + s_2 \\ &\approx N^* \times e^{-\rho} + (N^* - N^* \times e^{-\rho}) \times e^{-\frac{N^* - N^* \times e^{-\rho}}{f - N^* \times e^{-\rho}}} \end{aligned} \tag{9}$$

Then let us consider how long it takes in this round. The size of the *bitmap* transmitted in the *Pre-identification phase* and the *bitmapAck* transmitted in the *Acknowledgment phase* may be very long. The reader divides them into segments of 96-bits (equivalent to the length of the tag ID) and transmits each segment in a tag slot with length of $t_{id}$, i.e. 2.4ms. The length of each short slot in the *Identification phase* is $t_s$, i.e., 0.4ms. The time for transmitting the parameters $f$, $R$ and $z$ is negligible and can be ignored. Hence, the execution time of this round is:

$$\begin{aligned} T &= \left\lceil \frac{f}{96} \right\rceil \times t_{id} + f \times t_s + \left\lceil \frac{f}{96} \right\rceil \times t_{id} \\ &= 2 \times \left\lceil \frac{f}{96} \right\rceil \times t_{id} + f \times t_s \end{aligned} \tag{10}$$

According to Eqs. (9) and (10), in this round, the average time for identifying the presence of a tag is:

$$\begin{aligned} \frac{T}{N_{total}} &= \frac{2 \times \left\lceil \frac{f}{96} \right\rceil \times t_{id} + f \times t_s}{N^* \times e^{-\rho} + (N^* - N^* \times e^{-\rho}) \times e^{-\frac{N^* - N^* \times e^{-\rho}}{f - N^* \times e^{-\rho}}}} \\ &\approx \frac{2 \times \frac{f}{96} \times t_{id} + f \times t_s}{N^* \times e^{-\rho} + (N^* - N^* \times e^{-\rho}) \times e^{-\frac{N^* - N^* \times e^{-\rho}}{f - N^* \times e^{-\rho}}}} \\ &\approx \frac{0.45}{\rho \times e^{-\rho} + \rho \times (1 - e^{-\rho}) \times e^{-\frac{\rho - \rho \times e^{-\rho}}{1 - \rho \times e^{-\rho}}}} \end{aligned} \tag{11}$$

Following [10], we find the optimal from the average values, without considering the variance. Clearly, the average time for identifying a tag, $\frac{T}{N_{total}}$, is a function of $\rho$, where $\rho > 0$. To obtain the value of $\rho$ that minimizes the $\frac{T}{N_{total}}$, we differentiate $\frac{T}{N_{total}}$ and set the result to *zero* as follows:

$$(\frac{T}{N_{total}})' = \frac{0.45(\rho - 1) \times A(\rho)}{(\rho \times e^{-\rho} + \rho \times (1 - e^{-\rho}) \times e^{-\frac{\rho - \rho \times e^{-\rho}}{1 - \rho \times e^{-\rho}}})^2} = 0, \tag{12}$$

where

$$A(\rho) = e^{-\rho} + \frac{1-(1+\rho^2)e^{-\rho}+(1+\rho)\rho e^{-2\rho}-\rho^2 e^{-3\rho}}{(1-\rho e^{-\rho})^2} e^{-\frac{\rho-\rho e^{-\rho}}{1-\rho e^{-\rho}}} \quad (13)$$

In the Appendix-A, we prove that $A(\rho)$ in Eq. (12) *is always larger than 0*. Hence, we have: $(\frac{T}{N_{total}})' > 0$ when $\rho > 1$; $(\frac{T}{N_{total}})' = 0$ when $\rho = 1$; $(\frac{T}{N_{total}})' < 0$ when $\rho < 1$. Then it is proven that $\frac{T}{N_{total}}$ is *minimized* when $\rho = 1$ (i.e., $\frac{N^*}{f} = 1$). In other words, MMTI ($m = 2$, i.e., two hashing processes in each round) achieves the best time-efficiency when $f = N^*$.

■

### *D. Choosing an Optimal Hashing Count $m$*

In the first phase, the proposed MMTI protocol employs a $bitmap$ to guide the next $hashing$ process. We have described the MMTI protocol with the special case of $m = 2$. Intuitively, we can implement the hashing process for more times in each round to further improve the efficiency. To achieve the MMTI protocol with $m = \tau$, where $\tau > 2$, we just need to iteratively execute the hashing processes in the *Pre-identification* phase. Can we repeat the hashing process until each tag chooses an expected singleton slot in the frame? In this case, the utilization of the frame is 100%. However, the transmission of a bitmap for guiding the next hashing is not cost-free and generates overhead, and excessive hashing may beget inefficiency instead. In what follows, we will present the analysis to optimize the hashing count $m$ in *Pre-identification* phase of each round.

In the previous subsection, we have proved that, under the condition of $m = 2$, the MMTI protocol achieves the best efficiency when $f = N^*$ in each round. Hence, we still set $f = N^*$ in each round of the MMTI protocol with $m > 2$.

*Theorem 2:* Under the condition of $f = N^*$ in each round. If the hashing process is iteratively implemented for 5 times in the *Pre-identification* phase of each round, the MMTI protocol achieves the best efficiency.

*Proof:* We first consider how many tags are expected to be verified in this round. According to Eq. (2), we have:

$$s_1 = N^* \times e^{-\rho} = N^* \times e^{-1} \quad (14)$$

After the first hashing, we expect to obtain $N^* \times \frac{1}{e}$ singleton slots. Then there are $N^* \times (1-\frac{1}{e})$ non-singleton (empty or collision) slots remaining.

According to Eq. (8), and due to $\rho$=1, we have:

$$\begin{aligned} s_2 &\approx (N^* - N^* \times e^{-\rho}) \times e^{-\frac{N^* - N^* \times e^{-\rho}}{f - N^* \times e^{-\rho}}} \\ &\approx N^* \times (1-e^{-1}) \times e^{-1} \end{aligned} \quad (15)$$

That is, after the second hashing process, we expect to obtain $N^* \times (1-e^{-1}) \times e^{-1}$ more singleton slots. There are $N^* \times (1-e^{-1})^2$ non-singleton slots remaining.

We iteratively implement the hashing process for more times to achieve more singleton slots. After each hashing, $\frac{1}{e}$ of the remaining non-singleton slots are expected to become singleton slots. Let $s_i$ denote the number of the new singleton slots derived from the $i^{th}$ hashing process. $s_i$ is given by:

$$s_i = N^* \times (1-e^{-1})^{i-1} \times e^{-1} \quad (16)$$

In other words, we expect to obtain $N^* \times (1-e^{-1})^{i-1} \times e^{-1}$ more singleton slots from the $i^{th}$ hashing process. Let $N_m$ denote the total number of all singleton slots we expect to obtain after implementing hashing for $m$ times in this round. As the singleton slots are used to identify the presence of the corresponding tags, the presence of $N_m$ tags can be verified in this round. We have:

$$N_m = \sum_{i=1}^{m} s_i = \sum_{i=1}^{m} N^* \times (1-e^{-1})^{i-1} \times e^{-1} \quad (17)$$

Then, let us consider the execution time of this round. In the *Pre-identification phase*, in order not to 'disturb' the achieved singleton slots in the next hashing, the reader sends a bitmap, which reflects the current slot occupation states in time frame ($f$ slots), to guide the next hashing. Hence, the size of the bitmap reflecting the states of the whole time frame should always be $f$ bits. For $m$ times hashing, the reader needs to broadcast $bitmap$ ($f$ bits) for $m-1$ times. Each $bitmap$ is divided into 96-bits segments, and each segment is transmitted in a long slot with length of $t_{id}$, i.e., 2.4ms. There are $f$ short slots in the *Identification phase* and the length of a short slot is $t_s$, i.e. 0.4ms. In the *Acknowledgment phase*, the reader transmits the $bitmapAck$, which is also divided into 96-bits segments and the transmission of each segment occupies a long slot $t_{id}$. Thus, the total time of this round, $T_m$, is denoted as:

$$\begin{aligned} T_m &= (m-1) \times \left\lceil \frac{f}{96} \right\rceil \times t_{id} + f \times t_s + \left\lceil \frac{f}{96} \right\rceil \times t_{id} \\ &= m \times \left\lceil \frac{f}{96} \right\rceil \times t_{id} + f \times t_s \end{aligned} \quad (18)$$

As a result, the average time for identifying the presence of a tag in this round is:

$$\begin{aligned} \frac{T_m}{N_m} &= \frac{m \times \lceil \frac{f}{96} \rceil \times t_{id} + f \times t_s}{\sum_{i=0}^{m-1} N^* \times (1-e^{-1})^i \times e^{-1}} \\ &= \frac{m \times \lceil \frac{f}{96} \rceil \times 2.4 + f \times 0.4}{\sum_{i=0}^{m-1} N^* \times (1-e^{-1})^i \times e^{-1}} \\ &= \frac{m \times \lceil \frac{f}{96} \rceil \times 2.4 + f \times 0.4}{N^*[1-(1-e^{-1})^m]} \end{aligned} \quad (19)$$

Since we set $f$ to $N^*$ in this round, hence, we get

$$\frac{T_m}{N_m} \approx \frac{\frac{m}{96} \times 2.4 + 0.4}{1-(1-e^{-1})^m} \quad (20)$$

According to Eq. (20), the average time $\frac{T_m}{N_m}$ is a *discrete* function of $m$, where $m = 1, 2, 3, \cdots$. We need to find the optimal hashing count $m$ that minimizes $\frac{T_m}{N_m}$. We first assume $\frac{T_m}{N_m}$ is a *continuous* function of $m$, where $m > 0$. The derivative of $\frac{T_m}{N_m}$ is denoted as $(\frac{T_m}{N_m})'$ and given as follows:

$$(\frac{T_m}{N_m})' = \frac{\frac{2.4}{96}[1-(1-\frac{1}{e})^m] + (\frac{2.4}{96}m + 0.4) \times (1-\frac{1}{e})^m \times ln(1-\frac{1}{e})}{[1-(1-\frac{1}{e})^m]^2} \quad (21)$$

By solving $(\frac{T_m}{N_m})' = 0$ in Eq. (21), we get $m = 5.1696$. Moreover, in Appendix-B, we prove that $(\frac{T_m}{N_m})''$ is always larger than 0, where $(\frac{T_m}{N_m})''$ denotes the derivative of $(\frac{T_m}{N_m})'$. Therefore, we have $(\frac{T_m}{N_m})' > 0$ when $m > 5.1696$; $(\frac{T_m}{N_m})' = 0$ when $m = 5.1696$; $(\frac{T_m}{N_m})' < 0$ when $m < 5.1696$. That is, $m = 5.1696$ is the minimum point of $\frac{T_m}{N_m}$. Actually, $m$ is a discrete variable and cannot be set to 5.1696. Hence, the adjacent two points $m = 5$ and $m = 6$ are the only two optimal candidates. $\frac{T_m}{N_m}$ is 0.5839 ms when $m = 5$, and 0.5875 ms when $m = 6$. Clearly, $m = 5$ is the optimal setting, i.e., the hashing count $m$ is optimized to 5 in each round. ■

In Theorem 2, it has been proven that 5 times hashing processes are optimal in each round. Intuitively, fewer than 5 hashings decrease the utilization of the time frame (i.e., the proportion of the singleton slots is low). On the other hand, more than 5 hashings increase the slot utilization but cause too much overhead for transmitting the bitmaps, and the added overhead outweighs the gain of slot utilization. The simulation results shown in Table III also validate the above analysis.

### *E. Bloom-Filter based MMTI Protocol in Multi-reader Scenarios*

For clarity of description, we have described and analyzed the proposed MMTI protocol for the case of single reader. In some large-scale application scenarios, due to the limitation of communication range, one reader usually cannot cover all the tags. In what follows, we present how to extent the MMTI protocol to the multi-reader scenarios. We assume there are $\ell$ readers in this multi-reader system, and they are numbered from 1 to $\ell$. All the tags are uniformly placed and covered by at least one reader. Let $\aleph_i$ denote the tag set covered by reader $r_i$, and obviously $\aleph_1 \bigcup \aleph_2 \bigcup \cdots \bigcup \aleph_\ell = S_{all} - S_{miss}$. Note that, if the adjacent readers simultaneously interrogate the overlapped tags, reader-collision will occur. In [24], Yang *et al.* proposed a protocol stack called Season to solve this reader-collision problem. This problem is beyond the scope of this paper, we do not discuss this issue in detail due to the space limitation.

*1) A Bloom-Filter based Method:* Bloom Filter is a binary array data structure proposed by B. H. BLOOM at 1970s in [25], which was used to check if an element belongs to a given set, with allowance of *false positive*—an element that does not belong to a set but is detected as an element of this set. On the contrary, it involves no *false negative*—an element belonging to a set but is not detected as its member. *This technique can be introduced to make a reader know which tags may locate within its interrogation range.* Using Bloom filter technique, an arbitrary reader, say $r_i$, is able to learn that $|\aleph_i| + p \times (N - |\aleph_i|)$ (on average) tags are within its coverage, where $p$ denotes the probability of false positive. We denote this tag set as $\hat{\aleph}_i$, and it is known to the reader $r_i$. The reader $r_i$ executes the MMTI protocol using $\hat{\aleph}_i$ as the input. We denote the missing tag set returned by $r_i$ as $S_i$. Then, $\hat{\aleph}_i - S_i$ should be the actual set of tags that are confined within the interrogating range of reader $r_i$. We can get the set of all present tags by aggregating all the subsets $\hat{\aleph}_i - S_i$, where $i \in [1, \ell]$. That is, $\bigcup_{i=1}^{\ell}(\hat{\aleph}_i - S_i)$ is the set of all present tags in the system. Thus, $S_{all} - \bigcup_{i=1}^{\ell}(\hat{\aleph}_i - S_i)$ is the final set of missing tags.

*2) Analyzing the Performance of Paralleled MMTI:* Although "Bloom Filter" technique has been used in the RFID related papers (e.g., [26] and [23]), this paper presents a different theoretical analysis on the bloom-filter parameters for combination with the proposed MMTI. For an arbitrary reader $r_i$, its execution time of Bloom-Filter is $-\frac{|\aleph_i| \times lnp}{(ln2)^2} \times t_{bit}$ ms, where $t_{bit}$ is the time (in milliseconds) for transmitting a bit, according to [26]. Since $|\aleph_i| + p \times (N - |\aleph_i|)$ (on average) tags will be the input of MMTI executed by reader $r_i$, then, the time cost of MMTI on $r_i$ is about $0.58 \times (|\aleph_i| + p \times (N - |\aleph_i|))$ ms. Combining the above two parts of time, the total execution time for $r_i$ is $T_{r_i} = -\frac{|\aleph_i| \times lnp}{(ln2)^2} \times t_{bit} + 0.58(|\aleph_i| + p \times (N - |\aleph_i|))$ ms. We should configure the false positive $p$ so as to minimize this execution time. By setting its derivative $(T_{r_i})' = 0$, we have $p = \frac{|\aleph_i| \times t_{bit}}{0.58 \times (ln2)^2 (N - |\aleph_i|)}$. Furthermore, we observe that $(T_{r_i})' > 0$ when $p > \frac{|\aleph_i| \times t_{bit}}{0.58 \times (ln2)^2 (N - |\aleph_i|)}$; $(T_{r_i})' < 0$ when $p < \frac{|\aleph_i| \times t_{bit}}{0.58 \times (ln2)^2 (N - |\aleph_i|)}$. Therefore, the execution time $T_{r_i}$ is minimized when $p$ is set to $\frac{|\aleph_i| \times t_{bit}}{0.58 \times (ln2)^2 (N - |\aleph_i|)}$. And the global execution time is the largest execution time among all readers. Note that, the actual set $\aleph_i$ of tags that are covered by reader $r_i$ is not known in prior. Fortunately, we do not need to know exactly which tags are within $r_i$; we only need its cardinality $|\aleph_i|$, which can be roughly approximated by $Coverage\ Area\ of\ r_i \times Tag\ Density$.

### F. Impact on Channel Errors

The paper first assumes that there is a perfect communication channel between the reader and tags. However, in the real environment, the communication channel is error-prone [10] [12] [20] [27] [28]. The white noise may corrupt the data exchanged between the reader and tags, e.g., '0' becomes '1' or '1' becomes '0' [10]. The signals are even not detected at all due to path loss. In most literature [12] [20] [28] that consider the channel error, CRC (Cyclic Redundancy Code) is used to verify the correctness of the exchanged data between the reader and tags. Since most previous literature assumes the reader has adequate power to interrogate all tags in its vicinity [20] [29], this paper assumes that the signals on the reader-to-tags link will not be lost. In contrary, the signals on the tags-to-reader link may be lost because of the weak capability of tags. In what follows, we discuss the various channel errors in detail.

*1) Channel Errors during Transmission of the Parameters $R$, $f$ and $z$:* The correct transmission of the parameters $R$, $f$ and $z$ is crucial to the proposed MMTI protocol. To ensure the correct transmission of those parameters, the reader can generate a 10-bit checksum [28] of these parameters and appends the checksum to the back of the transmitted parameters. The tags check the correctness of the checksum after receiving the $< parameters, checksum >$. If the received parameters and the received checksum do not match, the tag will reply an NACK signal so as to force the reader to retransmit the $< parameters, checksum >$. If the reader senses a strong pulse (may be a single NACK or a mix of multiple NACKs), it will retransmit this binary group until it is transmitted correctly.

*2) Channel Errors during Transmission of the Bitmaps:* As the bitmaps are used to guide the hashing processes, the correct transmission of the bitmaps is very crucial. To tackle the channel errors during transmission of the bitmaps, we can also use the method of appending a checksum. Specifically, we divide a bitmap into segments of 75-bit, each of which is given a segment number (11 bits), e.g., the segment number of the $10^{th}$ segment is "00000001010" (equal to the decimal value '10'). Later on, we will explain the function of the segment number. The reader generates a 10-bit checksum of $< segment\ number, segment >$ and appends the checksum to the back so as to generate a triple, i.e., $< segment\ number, segment, checksum >$. The reader transmits the triples to the tags one by one. Similarly, the tags check the correctness of the received triple by verifying the checksum. For a certain triple, some tags can correctly receive this triple, whereas, some other tags cannot correctly receive it and reply NACK signals to force the reader to retransmit the same triple. Note that, the retransmitted triples do not confuse the tags that have already correctly received the same triple. If a tag successively receives two or more triples with the same $segment\ number$, they are treated as the same one. A triple is successfully transmitted only when no NACK signal is replied. Then the next triple is transmitted.

*3) Channel Errors when the Tags Reply Responses:* Clearly, based on the above countermeasures, we can guarantee that the final virtual mapping between tags and slots is correct and is predictable by the reader. We do not care the channel errors in the expected empty slots and the errors in the expected collision slots, because the proposed MMTI protocol only leverages the expected singleton slots to identify the missing tags. Hence, we only discuss the impact of channel errors occurring in the expected singleton slots. If an expected singleton slot corresponds to a present tag, but due to the channel errors (e.g., path loss), its response is not sensed by the reader, then it is wrongly considered as a missing tag, namely the *false positive*. On the other hand, if an expected singleton slot corresponds to an actually missing tag, due to the channel errors (e.g., white noise), this missing tag is wrongly verified as a present one, namely the *false negative*. An efficient method to tackle the *false positive* (i.e., a present tag is considered as a missing one) is presented below. The reader requests the reported missing tags by their IDs one by one, then the fake 'missing' tags can be filtered out. As for the *false negative* (i.e., a missing tag is verified as a present one), even if a missing tag is not detected in the current execution, it will be detected with a high probability in the next round of execution, because the missing tag identification is usually periodically executed in reality.

## V. Performance Evaluation

### A. Simulation Setting

In this section, we evaluate the performance of the proposed MMTI protocol. The execution time for identifying all the missing tags is used as the performance criterion. Since IIP [10] and Protocol 3 [8] are the main benchmarks in this paper, we adopt the same parameter settings in [10] [8]. Specifically, transmission of a segment of 96 bits takes a tag slot (i.e., $t_{id} = 2.4ms$); transmission of 1 bit response to the reader takes a short slot (i.e., $t_s = 0.4ms$). We run each simulation for 1000 times and collect the average results.

### B. Validating the Optimal Hashing Count $m$

The first simulation set in Table III intends to investigate the impact of different hashing count $m$ on the

performance of the proposed MMTI protocol, where we simulate a single reader. We vary the hashing count $m$ from 2 to 6 in the first simulation set. As shown in Table III, the proposed MMTI protocol achieves the best time-efficiency when the hashing count $m = 5$, which coincides with the proof presented in Section IV-D.

TABLE III: The execution time of MMTI with different hashing count $m$ (Single Reader).

| N | Execution time (s) | | | | |
|---|---|---|---|---|---|
| | m=2 | m=3 | m=4 | m=5 | m=6 |
| 5000 | 3.78 | 3.21 | 3.01 | 2.95 | 2.98 |
| 10000 | 7.52 | 6.38 | 5.97 | 5.87 | 5.91 |
| 20000 | 15.02 | 12.74 | 11.93 | 11.71 | 11.78 |
| 30000 | 22.51 | 19.10 | 17.90 | 17.55 | 17.67 |
| 40000 | 30.01 | 25.46 | 23.83 | 23.39 | 23.53 |
| 50000 | 37.50 | 31.81 | 29.79 | 29.24 | 29.41 |

### C. Execution Time

In this subsection, we mainly compare the MMTI protocol with the state-of-the-art protocols, Iterative ID-free Protocol (IIP) in [10] and the protocols in [8]. As Protocol 3 is the best one in [8], we only select Protocol 3 as one of the benchmarks in this paper. The solutions to *tag identification* problem can also solve the problem of identifying the missing tags, by comparing the collected ID information with the original ID information stored in database. Hence, we also compare the MMTI protocol with the most outstanding tag identification protocols, the Enhanced Dynamic Framed Slotted ALOHA (EDFSA) [16] and the Binary Tree Protocol (BTP) [17]. As aforementioned, in the single reader scenarios, IIP outperforms Protocol 3; whereas, Protocol 3 shows superiority over IIP in the multi-reader scenarios where the number of readers is large. Therefore, in order to fully evaluate the proposed protocol, we conduct simulations for the single reader case and for the multi-reader case, respectively.

*1) Single Reader Case:* As shown in Table IV, extensive simulations are conducted to evaluate the efficiency of the MMTI protocol, where a single reader is simulated. In [10], the authors presented a lower bound for missing tag identification protocol for the single reader case. Specifically, $N$ tags respond 1-bit announcement to the reader one by one. If we assume no coordination information is transmitted, the execution time is $N \times t_s$, which is treated as the lower bound for any missing tag identification protocols. In this paper, we also compare the proposed protocol with this lower bound. The tag number $N$ varies from 5,000 to 50,000. The simulation results in Table IV demonstrate that the proposed MMTI protocol performs much better than the BTP protocol and the EDFSA protocol. For example, when $N$ is 50,000, the execution time of BTP and EDFSA is 404.46 seconds and 387.80 seconds, respectively. And the MMTI protocol is 29.24 seconds, which outperforms the BTP and EDFSA by 92.78% and 92.47%, respectively. Moreover, the MMTI protocol cuts the execution time by about 32% when compared to the IIP protocol. For example, when $N$ is 50,000, the execution time of the proposed MMTI protocol is 32.3% less than that of IIP. Because Protocol 3 does not possess superiority *in the single reader scenarios*, it is not surprising that Protocol 3 performs the worst in this simulation set.

*2) Multiple Reader Case:* As one of the main benchmarks, Protocol 3 possesses great superiority over IIP in the multi-reader scenarios. In this simulation set, we mainly compare the proposed MMTI with IIP and Protocol 3, which are the state-of-the-art missing tag identification protocols. For fair comparison with the protocols in [8], we adopt the same simulation settings in [8], and thus simulated a region consisting of square zones, where $L$ denotes the number of the readers. And we assume the tags are uniformly distributed in the monitoring region, where $D$ is used to denote tag density. In what follows, we conduct two simulation sets to investigate the impact of $D$ and $L$ on the performance of MMTI, respectively.

The simulation results in Table V show the impact of the tag density $D$ on the performance of MMTI, IIP and Protocol 3, where the reader number $L$ is fixed to 50 and the tag density $D$ varies from 400/zone to 2000/zone. The simulation results in Table V demonstrate that the proposed MMTI protocol still performs much better than both IIP and Protocol 3. The reason is that using the Bloom Filter method parallelizes the execution of all readers thereby accelerating the identification process. For example, when $D = 2000$, $L = 50$ (meaning $2000 \times 50$=$100,000$ tags), the execution time of IIP and Protocol 3 is 86.07s and 18.69s, respectively. And the execution time of MMTI is just 7.68s, representing 91.1% and 58.9% reduction compared with IIP and Protocol 3.

The simulation results in Table VI show the impact of the reader number $L$ on the performance of MMTI, IIP and Protocol 3, where the tag density $D$ is fixed to 1000/zone and the reader number (or zone number) $L$ varies from 20 to 100. As shown in Table VI, the proposed MMTI protocol still considerably outperforms both IIP and Protocol 3. For example, when $L = 100$,

TABLE IV: The Execution Time With Respect to Tag Number $N$ (Single Reader Case).

| N | Execution time (s) | | | | | |
|---|---|---|---|---|---|---|
| | BTP | EDFSA | Protocol 3 in [8] | IIP | MMTI (m=5) | Lower Bound |
| 5000 | 40.02 | 38.68 | 47.36 | 4.37 | 2.94 | 2 |
| 10000 | 80.87 | 77.15 | 94.38 | 8.68 | 5.86 | 4 |
| 20000 | 162.04 | 157.23 | 188.43 | 17.29 | 11.72 | 8 |
| 30000 | 242.93 | 231.94 | 282.68 | 25.97 | 17.55 | 12 |
| 40000 | 324.21 | 311.41 | 376.52 | 34.48 | 23.38 | 16 |
| 50000 | 404.46 | 387.80 | 470.57 | 43.22 | 29.25 | 20 |

TABLE V: The Execution Time with Respect to Tag Density $D$, when the Reader Number $L = 50$.

| D | Execution time (s) | | |
|---|---|---|---|
| | IIP | Protocol 3 in [8] | MMTI (m=5) |
| 400 | 17.24 | 4.05 | 1.54 |
| 800 | 34.42 | 7.62 | 3.07 |
| 1200 | 51.66 | 11.43 | 4.61 |
| 1600 | 68.86 | 14.82 | 6.14 |
| 2000 | 86.07 | 18.69 | 7.68 |

$D = 1000$ (meaning $100 \times 1000$=100,000 tags), the execution time of IIP and Protocol 3 is 86.05s and 9.76s, respectively. And the execution time of MMTI is just 3.96s, which outperforms IIP and Protocol 3 by reducing 95.4% and 59.4%, respectively. Moreover, simulation results in Table VI show that the total execution time of MMTI keeps steady when the reader number $L$ increases, which demonstrates the good scalability of the proposed MMTI protocol.

TABLE VI: The Execution Time with Respect to Reader Number $L$, when the Tag Density $D = 1000\ tags/zone$.

| L | Execution time (s) | | |
|---|---|---|---|
| | IIP | Protocol 3 in [8] | MMTI (m=5) |
| 20 | 17.23 | 8.45 | 3.64 |
| 40 | 34.44 | 8.69 | 3.80 |
| 60 | 51.62 | 9.17 | 3.88 |
| 80 | 68.87 | 9.64 | 3.92 |
| 100 | 86.05 | 9.76 | 3.96 |

### D. The Impact of the Tag Number

According to the simulation results shown in Table IV, we can find that the average time taken by the proposed

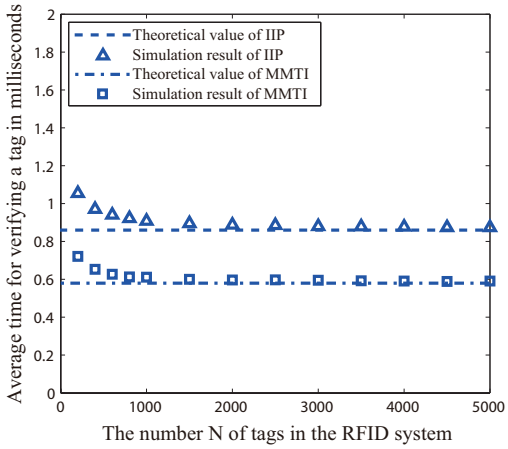


Fig. 3: Comparison with the IIP protocol in a small scale RFID system.

MMTI protocol to verify a tag is approximately the same as the theoretical value obtained in Section IV-D, 0.58 $ms$ per tag in a large scale system (i.e. with tens of thousands of tags). However, when the system is in a small scale, i.e., containing just hundreds of tags, the average time for verifying a tag in simulations is slightly higher than the theoretical value, as illustrated in Fig. 3. Why does this phenomenon occur? The reason is briefly exemplified as follows. Assume there are just 50 tags in the small scale MMTI system. According to the above description of the MMTI protocol, the $bitmap$ will be 50 bits (shorter than 96 bits) in the first round, but still occupies a tag slot, 2.4 $ms$. This overhead becomes notable, when $N$ (i.e. the number of tags in the system) is small. Hence, the average time taken for verifying a tag becomes slightly higher than the theoretical value. On the other hand, in a large scale system, this overhead still exists. But, due to the large number of tags, this small overhead is shared by a large number, $N$, of tags, then this overhead can be ignored.

The simulation results approach the theoretical value as the system scales up. As illustrated in Fig. 3, the MMTI protocol still outperforms the IIP protocol even in a small-scale RFID system.

### *E. The Impact of Channel Error*

Since the communication channel is not always perfect in reality, the channel errors (e.g., white noise, path loss, etc.) often degrade the performance of MMTI or even give rise to the false of the identification results. Hence, it is necessary to evaluate the performance of the proposed MMTI protocol when the channel is not error-free. Let $P_{error}$ denote the probability that each bit of the transmitted parameters or the bitmaps becomes wrong. If the channel error occurs, the retransmission is required.

*1) The Impact on the Total Execution Time:* In this set of simulation experiments, we investigate the impact of the channel errors on the total execution time of the proposed MMTI protocol under different channel conditions, where $P_{error}$ varies from $0.1\%$ to $1\%$. The simulation results depicted in Fig. 4 reveal that the MMTI protocol consumes more time in the scenarios with channel errors than in the perfect scenarios. Given a perfect communication channel, the MMTI with CRC consumes more time than the pure MMTI because the transmission of CRC generates extra overhead.

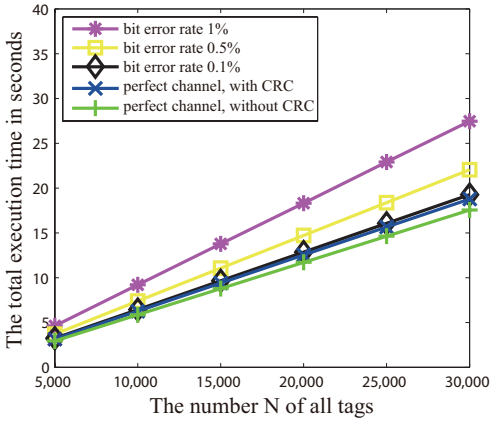


Fig. 4: Evaluating the performance of the MMTI protocol under different channel conditions.

*2) The Impact on the Identification Accuracy:* If the communication channel is error-free, the proposed MMTI can completely identify all the missing tags without any false. However, as aforementioned, the channel errors lead to false negative (i.e., missing tags are verified as present ones) and false positive (i.e., present tags are reported as missing ones) of identification. We use the ratio of false negative and the ratio of false positive to evaluate the identification accuracy of the MMTI protocol. The simulation results shown in Figs. 5 (a) and (b) demonstrate that both the ratio of false negative and the ratio of false positive fluctuate around the bit error probability $P_{error}$, which is normally quite small. As mentioned in Section IV-F3, the relatively small number of false positive tags can be reverified by a sample polling method. Moreover, the small number of false negative tags will be discovered with a high probability in the next execution [10].

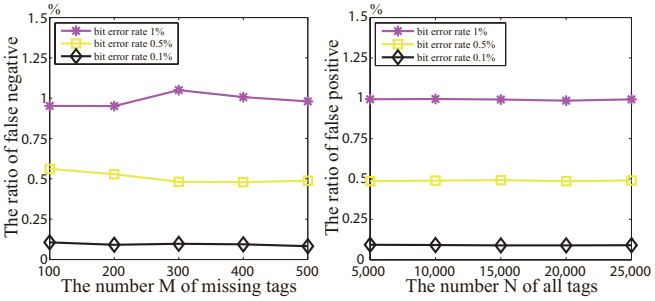


Fig. 5: Evaluating the identification accuracy of the MMTI protocol under different channel conditions. (a) The ratio of false negative; (b) the ratio of false positive.

## VI. Conclusions

This paper has studied an important problem of how to completely identify the missing RFID tags in an efficient way. The solutions to this problem are soliciting in many practical scenarios such as warehouse management, supply chain management and prison monitoring. For relieving the deficiency of the state-of-the-art protocols, this paper has proposed a Multi-hashing based Missing Tag Identification (MMTI) protocol. We have further investigated the optimal setting of the parameters $f$ and $m$ to maximize the performance of the proposed MMTI protocol. Extensive simulation experiments have been conducted to evaluate the efficiency of the proposed MMTI protocol. The results manifest that this protocol considerably outperforms the state-of-the-art protocols in both single reader scenarios and multi-reader scenarios.

## Acknowledgment

The authors would also like to sincerely thank the editors and anonymous reviewers for their thoughtful suggestions and constructive comments, which have greatly helped and inspired us to improve the quality of

this paper. This work was supported by NSFC (Grant No.s 60973117, 61173160, 61173162, 60903154, and 61321491), New Century Excellent Talents in University (NCET) of Ministry of Education of China, and the National Science Foundation for Distinguished Young Scholars of China (Grant No. 61225010).

## REFERENCES

[1] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: Indoor Location Sensing Using Active RFID," *Wireless networks*, vol. 10, no. 6, pp. 701–710, 2004.
[2] L. Xie, B. Sheng, C. C. Tan, H. Han, Q. Li, and D. Chen, "Efficient Tag Identification in Mobile RFID Systems," *Proc. of IEEE INFOCOM*, 2010.
[3] M. Kodialam, T. Nandagopal, and W. C. Lau, "Anonymous Tracking using RFID tags," *Proc. of IEEE INFOCOM*, 2007.
[4] B. Sheng, Q. Li, and W. Mao, "Efficient Continuous Scanning in RFID systems," *Proc. of IEEE INFOCOM*, 2010.
[5] K. Bu, B. Xiao, Q. Xiao, and S. Chen, "Efficient Pinpointing of Misplaced Tags in large RFID Systems," *Proc. of IEEE SECON*, 2011.
[6] C. C. Tan, B. Sheng, and Q. Li, "How to Monitor for Missing RFID tags," *Proc. of IEEE ICDCS*, 2008.
[7] W. Luo, S. Chen, T. Li, and S. Chen, "Efficient Missing Tag Detection in RFID Systems," *Proc. of IEEE INFOCOM*, 2011.
[8] R. Zhang, Y. Liu, Y. Zhang, and J. Sun, "Fast Identification of the Missing Tags in a Large RFID System," *Proc. of IEEE SECON*, 2011.
[9] W. Luo, S. Chen, T. Li, and Y. Qiao, "Probabilistic Missing-tag Detection and Energy-Time Tradeoff in Large-scale RFID Systems," *Proc. of ACM MobiHoc*, 2012.
[10] T. Li, S. Chen, and Y. Ling, "Identifying the Missing Tags in a Large RFID System," *Proc. of ACM MobiHoc*, 2010.
[11] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk, "Efficient and Reliable Low-Power Backscatter Networks," *Proc. of ACM SIGCOMM*, 2012.
[12] M. Zhang, T. Li, S. Chen, and B. Li, "Using Analog Network Coding to Improve the RFID Reading Throughput," *Proc. of IEEE ICDCS*, 2010.
[13] L. Kang, K. Wu, J. Zhang, and H. Tan, "Decoding the Collisions in RFID Systems," *Proc. of IEEE INFOCOM*, 2011.
[14] J. I. Capetenakis, "Tree Algorithms for Packet Broadcast Channels," *IEEE Transactions on Information Theory*, vol. IT-25, no. 5, pp. 505–515, 1979.
[15] L. G. Roberts, "Aloha Packet System with and without Slots and capture," *ACM SIGCOMM Computer Communication Review*, vol. 5, no. 2, pp. 28–42, 1975.
[16] S. Lee, S. Joo, and C. Lee, "An Enhanced Dynamic Framed Slotted ALOHA Algorithm for RFID Tag Identification," *Proc. of IEEE MobiQuitous*, 2005.
[17] J. Myung and W. Lee, "Adaptive Splitting Protocols for RFID Tag Collision Arbitration," *Proc. of ACM MobiHoc*, 2006.
[18] N. Bhandari, A. Sahoo, and S. Iyer, "Intelligent Query Tree (IQT) Protocol to Improve RFID Tag Read Efficiency," *Proc. of IEEE ICIT*, 2006.
[19] V. Namboodiri and L. Gao, "Energy-Aware Tag Anti-Collision Protocols for RFID Systems," *Proc. of IEEE PerCom*, 2007.
[20] S. Chen, M. Zhang, and B. Xiao, "Efficient Information Collection Protocols for Sensor-augmented RFID Networks," *Proc. of IEEE INFOCOM*, 2011.
[21] R. Jacobsen, K. F. Nielsen, P. Popovski, and T. Larsen, "Reliable Identification of RFID Tags Using Multiple Independent Reader Sessions," *Proc. of IEEE RFID*, 2009.
[22] P. Semiconductors, "I-CODE Smart Label RFID Tags," *http://www.nxp.com/acrobat_download/other/identification/SL092030.pdf*, Jan 2004.
[23] Y. Zheng and M. Li, "Fast Tag Searching Protocol for Large-Scale RFID Systems," *Proc. of IEEE ICNP*, 2011.
[24] L. Yang, J. Han, C. Wang, T. Gu, and Y. Liu, "Season: Shelving Interference and Joint Identification in Large-scale RFID Systems," *Proc. of IEEE INFOCOM*, 2011.
[25] B. H. BLOOM, "Space/Time Tradeoffs in Hash Coding with Allowable Errors," *Communications of the ACM*, 1970.
[26] H. Yue, C. Zhang, M. Pan, Y. Fang, and S. Chen, "A Time-efficient Information-Collection Protocols for Large-scale RFID Systems," *Proc. of IEEE INFOCOM*, 2012.
[27] Y.-C. Lai and C.-C. Lin, "Two blocking algorithms on adaptive binary splitting: single and pair resolutions for rfid tag identification," *IEEE/ACM Transactions on Networking (TON)*, vol. 17, no. 3, pp. 962–975, 2009.
[28] Y. Qiao, S. Chen, T. Li, and S. Chen, "Energy-efficient Polling Protocols in RFID Systems," *Proc. of ACM MobiHoc*, 2011.
[29] M. Shahzad and A. X. Liu, "Every Bit Counts - Fast and Scalable RFID Estimation," *Proc. of ACM MobiCom*, 2012.

## APPENDIX

### A. Proving $A(\rho) > 0$:

*Proof Objective:* $A(\rho)$ in Eq. (12) is always larger than 0, where $A(\rho)$ is given as follows:

$$e^{-\rho} + \frac{1-(1+\rho^2)e^{-\rho}+(1+\rho)\rho e^{-2\rho}-\rho^2 e^{-3\rho}}{(1-\rho e^{-\rho})^2} e^{-\frac{\rho-\rho e^{-\rho}}{1-\rho e^{-\rho}}} \tag{22}$$

*Proof:* For clarity, we denote $1-(1+\rho^2)e^{-\rho}+(1+\rho)\rho e^{-2\rho}-\rho^2 e^{-3\rho}$ as $B(\rho)$; and denote $1-\rho e^{-\rho}$ as $C(\rho)$, then we have:

$$A(\rho) = e^{-\rho} + \frac{B(\rho)}{C(\rho)^2} e^{-\frac{\rho-\rho e^{-\rho}}{C(\rho)}} \tag{23}$$

Clearly, if we prove that $B(\rho) > 0$ and $C(\rho) \neq 0$, then $A(\rho) > 0$ is proved.

*Step 1: Proving $B(\rho) > 0$:*

$$\begin{aligned} B(\rho) &= 1-(1+\rho^2)e^{-\rho}+(1+\rho)\rho e^{-2\rho}-\rho^2 e^{-3\rho} \\ &> 1-(1+\rho^2)e^{-\rho}+\rho^2 e^{-2\rho}-\rho^2 e^{-3\rho} \\ &= 1-(1+\rho^2)e^{-\rho}+\rho^2(e^{-2\rho}-e^{-3\rho}) \\ &> \underbrace{1-(1+\rho^2)e^{-\rho}}_{denoted\ as\ D(\rho)} \end{aligned} \tag{24}$$

Let $1-(1+\rho^2)e^{-\rho}$ be denoted as $D(\rho)$, and we get its derivative as follows:

$$D(\rho)' = (\rho-1)^2 e^{-\rho} \geq 0 \tag{25}$$

As shown in Eq. (25), the derivative of $D(\rho)$ is always larger than (or equal to) 0, thence, $D(\rho)$ is a increasing function with respect to $\rho$. Therefore, we have $D(\rho) > D(0)$, where $D(0) = 0$, i.e., $D(\rho) > 0$. According to Eq. (24), we have $B(\rho) > D(\rho) > 0$.

*Step 2: Proving $C(\rho) \neq 0$:*

We get the derivative of $C(\rho) = 1-\rho e^{-\rho}$ as follows:

$$C(\rho)' = (\rho-1)e^{-\rho}, \tag{26}$$

where we have $C(\rho)' > 0$ when $\rho > 1$; $C(\rho)' = 0$ when $\rho = 1$; $C(\rho)' < 0$ when $\rho < 1$. Hence, $C(\rho)$ is minimized to $C(1)$, where $C(1) = 1 - e^{-1}$. Clearly, $C(\rho) \neq 0$.

Since we have proved $B(\rho) > 0$ and $C(\rho) \neq 0$, it is easy to know that $A(\rho)$ in Eq. (23) is larger than 0. ■

### *B. Proving* $(\frac{T_m}{N_m})'' > 0$*:*

*Proof Objective:* The derivative $(\frac{T_m}{N_m})''$ of $(\frac{T_m}{N_m})'$ is always larger than 0, where $(\frac{T_m}{N_m})'$ is given as follows:

$$(\frac{T_m}{N_m})' = \frac{\frac{2.4}{96}[1-(1-\frac{1}{e})^m] + (\frac{2.4}{96}m + 0.4) \times (1-\frac{1}{e})^m \times ln(1-\frac{1}{e})}{[1-(1-\frac{1}{e})^m]^2} \tag{27}$$

*Proof:* Also for clarity, we denote the $\frac{2.4}{96}[1-(1-\frac{1}{e})^m] + (\frac{2.4}{96}m + 0.4) \times (1-\frac{1}{e})^m \times ln(1-\frac{1}{e})$ as $F(\rho)$; and denote $[1-(1-\frac{1}{e})^m]^2$ as $G(\rho)$, then we have:

$$(\frac{T_m}{N_m})' = \frac{F(\rho)}{G(\rho)} \tag{28}$$

The derivative of $(\frac{T_m}{N_m})'$ is denoted as $(\frac{T_m}{N_m})''$ and given as follows:

$$(\frac{T_m}{N_m})'' = \frac{F(\rho)}{G(\rho)} = \frac{F(\rho)'G(\rho) - F(\rho)G(\rho)'}{G(\rho)^2} \tag{29}$$

Obviously, $G(\rho)$ is not equal to 0, hence, if we prove that $F(\rho)'G(\rho) - F(\rho)G(\rho)'$ is larger than 0, the final proving objective $(\frac{T_m}{N_m})'' > 0$ is proved. In what follows, we prove that $F(\rho)'G(\rho) - F(\rho)G(\rho)' > 0$.

$$\begin{aligned}
&F(\rho)'G(\rho) - F(\rho)G(\rho)' \\
=&(\frac{2.4}{96}m + 0.4) \times ln^2(1-\frac{1}{e}) \times (1-\frac{1}{e})^m \times \\
&\{[1-(1-\frac{1}{e})^m]^2 + 2(1-\frac{1}{e})^m \times [1-(1-\frac{1}{e})^m]\} + \\
&2 \times \frac{2.4}{96} \times ln(1-\frac{1}{e}) \times (1-\frac{1}{e})^m \times [1-(1-\frac{1}{e})^m]^2 \\
>&(\frac{2.4}{96}m + 0.4) \times ln^2(1-\frac{1}{e}) \times (1-\frac{1}{e})^m [1-(1-\frac{1}{e})^m]^2 \\
&- \frac{2.4}{96}(1-\frac{1}{e})^m \times [1-(1-\frac{1}{e})^m]^2 \\
>&(1-\frac{1}{e})^m \times [1-(1-\frac{1}{e})^m]^2 \times [\frac{2.4}{96}ln^2(1-\frac{1}{e})m + 0.0592] \\
>&0
\end{aligned} \tag{30}$$

According to Eq. 29 and Eq. 30, we prove that $(\frac{T_m}{N_m})''$ is larger than 0. ■