# Minimal Perfect Hashing-based Information Collection Protocol for RFID systems

Xin Xie Xiulong Liu Keqiu Li Bin Xiao Heng Qi

**Abstract**—For large-scale RFID systems, this paper studies the practically important problem of target tag information collection, which aims at collecting information from a specific set of target tags instead of all. However, the existing solutions are of low time-efficiency because of two reasons. First, the serious collisions among tags due to hashing randomness seriously reduce the frame utilization, whose upper bound is just 36.8%. Second, they cannot efficiently distinguish the target tags from the non-target tags and thus inevitably collect a lot of irrelevant information on non-target tags, which further deteriorates the effective utilization of the time frame. To overcome the above two drawbacks, this paper proposes the minimal Perfect hashing-based Information Collection (PIC) protocol, which first leverages lightweight indicator vectors to establish a one-to-one mapping between target tags and slots, thereby improving the frame utilization to nearly 100%; and then uses the novel data structure called Minimal Perfect Hashing based Filter (MPHF) to filter out the non-target tags, thereby preventing them from interfering with the process of collecting information from target tags. Sufficient theoretical analyses are also presented in this paper to minimize the execution time of the proposed PIC protocol. Extensive simulations are conducted to compare the proposed PIC protocol with prior works side-by-side. The simulation results demonstrate that PIC significantly outperforms the state-of-the-art protocols in terms of time-efficiency.

**Index Terms**—UHF RFID, TDMA protocol, perfect hashing, time-efficiency

## 1 Introduction

RADIO Frequency Identification (RFID) technology, with many attractive properties such as small size, low manufacturing cost and remote access, has been deployed in various applications such as modern supply chains [1], inventory monitoring [2], [3], [4], [5], [6], object tracking [7], [8], [9], [10], and theft surveillance , *etc.* A typical RFID system is composed of a controller, a reader and numerous tag-attached objects [11], [12]. The tag, a microchip being attached to the antenna [13], has a distinct serial number acting as the identity of a person or object [14]. It communicates with the reader through a wireless channel. The tag antenna draws in energy from the RF waves to power the chip, which generates a signal back to the reader. The reader can convert the radio waves reflected back from the tag into digital information, which is sent to the central controller who can make use of this information to support various applications.

The wide usage of RFID is envisioned in a scenario of "Internet-of-Things", a world in which billions of objects can report their location, identity and other characteristics (*e.g.*, price, place of origin and expiration date) over a single wireless channel. Such a huge volume of data will pose a challenge in terms of time-efficient information collection. In such a vision, an important problem is how to time-efficiently collect the information from a specific set of target tags. Note that, we do not concern with the information on the non-target tags due to various practical reasons. For example, the RFID devices have a low-rate communication channel, hence, given a limited time window, we obviously should let the reader collect information from the tags with high priorities. These high-priority tags are called target tags. Another example is in a multi-tenant warehouse, the tags of a tenant are called target tags from the view of this tenant, and the tags belonging to other tenant are called non-target tags. A tenant obviously concerns more with the information on the target tags.

To address the problem of target tag information collection, an immediate solution is to use the $Q$ protocol specified in the EPC Class-1 Gen-2 (C1G2) standard [15] in which all tags including target ones and non-target ones contend for a common slotted time frame. Each tag calculates a uniformly distributed hash function to randomly choose a time slot to report its stored information. A tag is able to successfully report its information if and only if it exclusively occupies a slot. On the contrary, if two or more tags simultaneously transmit information in the same slot, the reader cannot resolve any of them due to signal corruption. Because of the inherent randomness nature, the $Q$ protocol suffers from serious tag collisions, which result in that the frame utilization is up to 36.8%. More seriously, the $Q$ protocol cannot distinguish the target tags from the non-target tags, and just collect a lot of irrelevant information from non-target tags. This further wastes the limited channel resources. Another straightforward solution is to let the reader poll the $id$s of target tags one by one [16]. A tag listens to the channel and will report its information once it finds that its $id$ was just queried in the last time slot. Although simple, the polling method is also of low time-efficiency due to the heavy transmission of tag $id$s whose length is 96-bit long. To avoid the transmission of tag $id$s and improve the frame utilization, [17] proposed the Multi-hash Information Collection protocol (MIC) to obtain more singleton slots

*X. Xie, X. Liu, K. Li, and H. Qi are with the School of Computer Science and Technology, Dalian University of Technology, China. E-mail: {xiulongliudut,likeqiu}@gmail.com.*
*B. Xiao is with the department of Computing, the Hong Kong Polytechnic University, Hong Kong, China.*
*Corresponding authors: Xiulong Liu and Keqiu Li*

that contain successful transmission. Moreover, the reader is able to know the information received in the current slot is from which tag, because it knows the mapping between tags and slots by pre-computing. However, MIC is still of low time-efficiency because it does not consider the serious interferences from non-target tags. To resolve this issue, the state-of-the-art protocol called Enhance Tag Ordering Protocol (ETOP) [18] leverages a tag ordering vector to filter out the non-target tags for preventing them from interfering with the collection of target tags; and obtain a reporting order for target tags, thereby reducing the collisions among target tags. However, also due to hashing randomness, ETOP cannot completely avoid the collisions among target tags. It still requires the heavy polling method to query the information on the collided target tags, which renders its inefficiency.

This paper proposes a time-efficient Perfect hashing-based Information Collection protocol (PIC) to completely filter out non-target tags and obtain a one-to-one mapping between target tags and slots. Then, the reader can collect the information from target tags without any collisions. Specifically, PIC consists of four phases: assignment phase, filtering phase, polling phase, collection phase. In the first phase, PIC uses multiple lightweight indicator vectors to recursively assign an exclusive slot to each target tags. In the second phase, PIC constructs a data structure called Minimal Perfect Hashing based Filter (MPHF), which is an array of $w$ elements, each element is the fingerprint of a target tag, where $w$ is the number of target tags. Then, the reader broadcasts the MPHF to filter out most of the non-target tags, thereby preventing them from interfering with the collection of target tags. In the third phase, a few non-target tags that are not filtered out by MPHF due to false positives will be deactivated by polling methods. In the fourth phase, the target tags arranging in a non-collision order will report their information one by one. Here, we face two technical challenges as follows. The first challenge is in setting an optimal length of the indicator vector. Although a long indicator vector can assign more target tags to exclusive slots, it also incurs more transmission overhead. The second challenge is in setting the false positive required in the filtering phase. If the false positive is set to a quite small value, less non-target tags will be left, but it requires a long fingerprint for each target tag and begets more transmission overhead. On the contrary, if the false positive is set to a large value, it requires less transmission for MPHF due to the shorter fingerprint for each target tag. However, more non-target target tags will be left to the heavy polling phase. Clearly, it is important to set a proper indicator length in the assignment phase and a proper false positive in the filtering phase. In this paper, we propose sufficient theoretical analyses to optimize these parameters thereby minimizing the total execution time of PIC. The contributions made in this paper can be summarized as follows.

- We propose a novel protocol called Perfect hashing-based Information Collection protocol (PIC) to collect the information from a specific set of target tags. Compared with prior work, our PIC can completely prevent the interferences from non-target tags and also avoid the collisions among target tags.
- We propose sufficient theoretical analyses to investigate the impact of the parameters involved in PIC, and optimize their values to minimize its execution time.
- We conduct extensive simulations to compare the proposed PIC protocol with prior works side-by-side. The simulation results demonstrate that our PIC significantly outperforms the state-of-the-art protocol by reducing 30%$\sim$50% of the execution time.

The rest of paper is organized as follows. Section 2 provides a brief review of prior works. Section 3 presents the system model and problem statement. The detailed design of PIC is described in Section 4. Section 5 presents how to optimize system parameter to minimize the execution time of PIC. Section 8 conducts extensive simulations to evaluate the performance of PIC and compares it with the related works. Finally, we conclude this paper in Section 9.

## 2 Related Work

One of the most fundamental tasks in RFID systems is to exactly identify the tag IDs. The key challenge is in resolving the serious tag collisions, which occur when two or more tags choose the same time slot to report their IDs. The solutions fall into three categories, frame-slotted ALOHA-based protocols [19], [20], tree-based protocols [21] and compressed sensing-based protocols [22], [23], [24]. ALOHA-based protocols divide the total time frame into multiple slots. Each tag randomly chooses one slot from the time frame to respond to the reader with its ID. Fundamentally, it is a Time Division Multiple Access (TDMA) communication mechanism. As for Tree-based protocols, the reader broadcasts a $0/1$ string to query the tags. A tag responds with its ID once it finds that the queried string is the prefix of its ID. A reader identifies a tag ID when only one tag responds. Finally, the most advanced compressed sensing-based protocols [22], [23], [24] treat tag collisions as a sparse rate-less code across the tags IDs, and they introduce a compressive sensing algorithm to identity tags from collisions signals, which are commonly regarded as useless in previous studies. The experiments results of the prototype implemented on the customized RFID devices reveal that the compressed sensing-based techniques can significantly improve the time-efficiency [24].

With the development of sensor-augmented tags, RFID tags can provide not only the IDs but also various information such as temperature, humidity and light intensity. Due to the low communication rate of RFID channel, how to collect information from massive tags is a challenging problem. Chen *et al.* designed a protocol to improve the utilization of the time frame by using multiple hashing functions to resolve tag collisions in the time frame [17]. Yue *et al.* proposed a Bloom filter-based protocol [25] to collect tag information in the multi-reader scenarios, which can let each reader efficiently obtain the ID set of tags that are within its interrogation range. In some application scenarios, we may only desire to collect the information generated by a specific set of target tags. For the problem of target tag information collection, Yan *et al.* proposed a tag-ordering vector based protocol to filter out the non-target tags [18], and thus prevent them from interfering with the collection of target tags. Moreover, the tag-ordering vector can also avoid executing the collision slots and empty slots.

However, the target tags that are mapped to the collision bit will report their information only when their IDs are queried by the RFID reader. Due to the transmission of target tag IDs that are 96-bit long, it is still of low time-efficiency.

## 3 Preliminary and Problem Definition

### 3.1 System model

In our model, the RFID system is composed of three components: a central controller, an RFID reader, and a large number of tags. The central controller is connected to the RFID reader via a high speed network link, and has powerful computation and storage capability. It controls the reader to interrogate the RFID tags, and then tackles the data reported from the reader. The reader is equipped with multiple antennas to cover a large number of tags. The tags are powered up by the radio waves transmitted by the reader, and then communicate with the reader by backscattering the RF carrier according to the commands received from the reader. The passive tags are required to implement a set of mandatory commands defined by EPC global Class1 Gen2 (C1G2) standard [26]. In addition, it provides flexibility for manufacturers to implement customized commands in conformance with the standard (*e.g.*, random number generator and lightweight hash function). Each tag contains a 96-bit ID according to C1G2, which is a unique identifier of the RFID tag. The central controller stores all the IDs of the tags within the interrogation range and knows all mandatory and customized commands implemented on tags. Thus, it is able to predict the action of the tags.

### 3.2 Communication Model

The reader periodically broadcasts synchronization commands to create a slotted time frame. Upon receiving the command of starting frame, a tag will randomly choose a slot from the time frame and respond to the reader with their IDs or information in that slot. We assume that the communication channel between the reader and the tags is reliable, and the communication errors are handled by low level protocols, *i.e.*, tags can correctly receive commands from the readers and the reader can correctly detect the responses from the tags. The main notations used in this paper are summarized in Table 1.

### 3.3 Problem Definition

This paper addresses the problem of collecting information from target tags. Formally, the problem is defined as follows: given a set of registered tags $\mathbb{U}$ with size $u$ and a set of target tags $\mathbb{W}$ with size $w$. Here, $\mathbb{W}$ is a subset of $\mathbb{U}$, *i.e.*, $\mathbb{W} \subseteq \mathbb{U}$. We want to use the RFID reader to efficiently collect information from $\mathbb{W}$. This problem may arise in many applications. For example, a consumer may want to collect detailed product information on items in his/her shopping list.

## 4 Detailed Design of the PIC Protocol

In this section, we will present a novel protocol called Perfect hashing-based Information Collection (PIC) protocol. Before presenting the detailed protocol design, we discuss the limitations of the existing solutions, which motivates us to take further effort to study this problem of target tag information collection. Then, we give the overview of the proposed PIC protocol. PIC consists of four phases: *assignment phase* is to assign each target tag with an exclusive slot; *filtering phase* is to filter out the non-target tags, and thus preventing them from interfering with the collection of target tag; *polling phase* is to deactivate the non-target tags that are not filtered out due to false positives; *collection phase* is to collect the information from target tags without any interferences from non-target tags and also without any collisions among target tags.

TABLE 1
Main notations.

| Notations | Descriptions |
|---|---|
| $\mathbb{U}$ | set of tags in the interrogation zone, $u = \|\mathbb{U}\|$ |
| $\mathbb{W}$ | set of target tags, $w = \|\mathbb{W}\|$ |
| MPHF | minimal perfect hashing filter |
| $H(\cdot)$ | lightweight uniform hash function |
| $F(\cdot)$ | fingerprint generation function |
| $r$ | random seed |
| $a$ | num of assigned target tags |
| $V$ | indicator vector, $v = \|V\|$ |
| $d$ | length of fingerprint |
| $x$ | load factor of the indicator vector, $x = w/v$ |
| $c$ | average cost for assigning a target tags |
| $p$ | proportion of assigned target tags |
| $l$ | length of the queried information |
| $\rho$ | ratio of target tags to integrated tags, $\rho = w/u$ |
| $v_{\mathrm{i}}$ | length of indicator vector in $i$th round |
| $w_{\mathrm{i}}$ | num of unassigned target tags before $i$th round |
| $\Delta w_{\mathrm{i}}$ | num of target tags assigned in $i$th round |
| $p_f$ | false positive probability of the filter |
| $p_e$ | error probability of falsely identifying tags |
| $R_r/R_t$ | reader to tags/tags to reader rate |

### 4.1 Motivation

To collect the information stored in RFID tags, an immediate method is to let the reader query the tag IDs one by one, and a tag will respond with the stored information once it finds its ID is queried. Although this method is simple, it is low-efficient due to the heavy transmission of tag IDs. Alternatively, to avoid the redundant transmission of tag IDs, [27] proposed a hashing-mapping method. Specifically, each tag uses its ID to calculate the randomly distributed hash function $H(id, r) \mod f$ to choose a slot to reply its information, where $r$ is a publicly known seed received from the reader, and $f$ is the number of slots in the forthcoming frame. We can predict which slot an arbitrary tag will choose to reply its information (*i.e.*, the mapping between tags and slots), because we know all the tag IDs as well as all the hashing parameters. Then, when the reader successfully receives a tag response, we could know this information is from which tag, although this tag did not reply its ID. The hashing-based method is much more efficient than the polling-based method, because it does not require the transmission of tag IDs. The reader can collect a tag information when only one tag replies in a slot, which is called

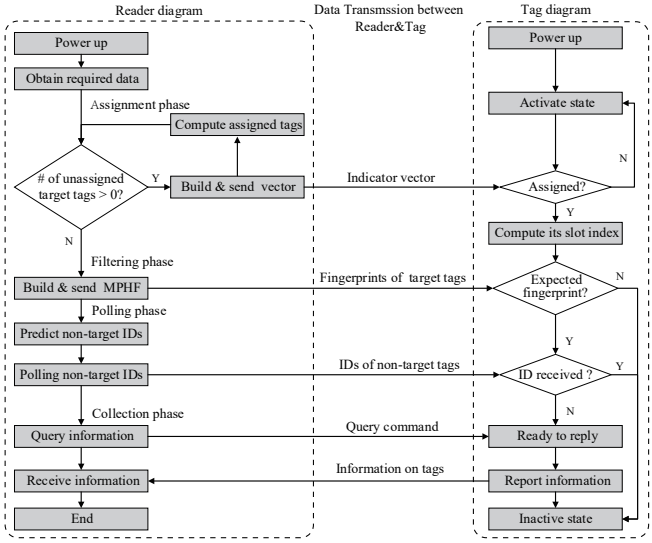



Fig. 1. Overview of the minimal perfect hashing-based protocol (PIC)

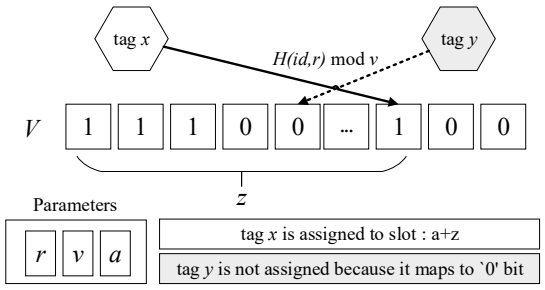



Fig. 2. $V$ is the indicator vector broadcast by the reader, and each bit whose value is one in $V$ represents an assigned tag. $a$ is the total number of assigned tags in previous rounds, and $z$ is the number of ones prior to the representative bit of tag $x$.

singleton slot. However, because of hashing collisions, two tags may choose the same slot to reply their information. Then, the reader cannot successfully receive any of them due to signal corruption. Such type of slots are called collision slots. Moreover, the empty slots in which no tag replies are also wasted. The throughput *i.e.*, the ratio of singleton slots, is up to 36.8% [27], which becomes the new bottleneck. With the same motivation of improving frame utilization in many high-level literatures [18], [25], [27], [28], [29], [30], this paper proposes the minimal perfect hashing technique to establish a one-to-one mapping between target tags and slots in time frame. Based on this bijective mapping between target tags and slots, we can build a better Bloom filter (*i.e.*, MPHF) to efficient filter out non-target tags, and thus preventing them from interfering with the collection of target tags. Besides, each target tag can also transmit its information to the reader based on this mapping to completely avoid the collisions among target tags.

### 4.2 Protocol Overview

The basic idea of PIC is to completely resolve target tag collisions and completely deactivate non-target tags during the time frame. PIC consists of four major phases: assignment, filtering, polling and collection as shown in Fig 1. The assignment phase is to resolve collisions among target tags by assigning each each target tag to an exclusive slot in the time frame whose size equals the number of target tags. It consists of multiple assignment rounds, during an arbitrary round, the reader first predicts the hashing result of each target tag to construct an indicator vector. Then, it broadcasts this indicator vector to assign a slot index to tags that hash to the bits containing one target tag, which ensures each target tag is assigned to an exclusive slot. Since the non-target tags also receive the indicator vectors during the assignment phase, they are also mapped to slots in the time frame. To completely deactivate non-target tags, in the filtering phase, we first propose the Minimal Perfect Hashing Filter (MPHF) technique to filter out most of non-target tags. However, due to the probabilistic nature of bloom filtering, a small ratio of non-target tags will be mistakenly recognized as target tags. Since we know all the ids and the used hash parameters, we could predict which non-target tags are mistakenly recognized as target tags. To completely eliminate these tags, in the polling phase, we let the reader poll the IDs of them one by one, and the tags who find their IDs are polled will turn into inactive state. Finally, in the collection phase, the reader starts a time frame to collect information from target tags. Since all non-target tags have been deactivated in filtering and polling phases and all target tags have been assigned to distinct slots in the assignment phase, each target tag can send the data stored on it without collisions, and the reader can exactly know the sender of each received data package. In the following, we will present the detailed processes of the above four phases.

### 4.3 Assignment Phase

To remove target tag collisions, the reader starts the assignment phase to assign target tags in set $\mathbb{W}$ to $w$ consecutive indexes range from 0 to $w-1$. The assignment phase consists of multiple rounds, during an arbitrary round, says $i$, the reader broadcasts an indicator vector, says $V_{\rm i}$, to assign a part of target tags to their slot indexes. Let $\mathbb{W}_{\rm i}$ be the number of unassigned target tags at the beginning of $i$th round. To construct $V_{\rm i}$, the reader hashes each tag in set $\mathbb{W}_{\rm i}$ to a representative bit in $V_{\rm i}$ with index of $H(id,r) \mod v_{\rm i}$, where $id$ is the target tag ID, $r$ is a random seed, and $v_{\rm i}$ is the size of $V_{\rm i}$. In Section 5.1, we have presented how to set $v_{\rm i}$ according to the number of unassigned tags $w_i$. As to the construction of the indicator vector, a bit in the $V_{\rm i}$ is set to '1' if only one target tag is mapped to this bit; a bit in the $V_{\rm i}$ is set to '0' if no or more than 2 target tags are mapped to this bit. The reader then broadcasts the indicator vector $V_i$, the vector size $v_i$, the random seed $r$ and the total number of tags that have been assigned in previous rounds, $a$.

With the vector size $v_{\rm i}$ and random seed $r$, a tag calculates $H(id,r) \mod v_{\rm i}$ to find out the location of its representative bit in $V_i$. If its representative bit is '0', the tag will not be assigned (*e.g.*, tag $y$ in Fig. 2); in contrary, if the representative bit is '1', the tag will be assigned (*e.g.*, tag $x$ in Fig. 2). This scheme guarantees that each target tag is assigned to an exclusive slot. For an arbitrary tag whose representative bit is '1', it calculates the assigned slot index by summing the number of '1s' before its representative bit

in the vector and the number of target tags that have been successfully assigned in previous rounds. The assigned tags ignore the the following indicator vectors and are handled in the subsequent filtering phase for determining whether they are target tags. Because the reader knows all the IDs in the system and all the IDs of target tags, it can pre-compute the set of tags that can be assigned by indicator vector $V_\mathrm{i}$, thus knowing the updated set of unassigned tags after broadcasting $V_\mathrm{i}$, namely $\mathbb{W}_{\mathrm{i}+1}$. With $\mathbb{W}_{\mathrm{i}+1}$, the reader starts the next assignment round, says $i+1$, for further tag assignment. Once the reader finds out there are no unassigned target tags, the assignment phase is terminated.

Let $w_\mathrm{i}$ denote the number of target-tags that participate in the $i$-th assignment round. The number of target tags that can be successfully assigned in this round is expected to be $\binom{w_\mathrm{i}}{1}(\frac{1}{v_\mathrm{i}})(1-\frac{1}{v_\mathrm{i}})^{w_\mathrm{i}-1} \times v_\mathrm{i} \approx w_\mathrm{i} e^{-\frac{w_\mathrm{i}}{v_\mathrm{i}}}$, where $e$ is the natural constant. For example, if we set the length of the indicator vector to the number of target tags that have not yet been assigned, $w_\mathrm{i} \times \frac{1}{e}$ target tags will be assigned, *i.e.*, $\frac{1}{e} \approx 36.8\%$ of the non-assigned target tags will be assigned in each round. Now, let's consider the transmission overhead of this case. The indicator vector is $w$-bit for the first round; it is $w(1-\frac{1}{e})$-bit for the second round, because $w(1-\frac{1}{e})$ target will participate in this round on expectation; $w(1-\frac{1}{e})^2$-bit for the third round; $w(1-\frac{1}{e})^{i-1}$-bit for the $i$-th round. The total transmission overhead is calculated as $\sum_{i=1}^{+\infty} w(1-\frac{1}{e})^{i-1} = we[1-(1-\frac{1}{e})^{+\infty}]$, by relaxing, whose upper bound is $we$. That is, the transmission complexity of the slot assignment phase is just $O(w)$.

### 4.4 Filtering Phase

Although we have assigned each target tag to an exclusive slot to avoid collisions among them, each non-target tag in the interrogation zone can also be assigned to these slots by the indicator vectors. Since the number of non-target tags are usually much larger than that of target tags, the non-target tags cause serious collisions and prevent the reader from efficiently collecting information from target tags. Hence, before starting the time frame to collect information from target tags, we need to deactivate all the non-target tags at first.

In the filtering phase, the reader first broadcasts a data structure called Minimal Perfect Hashing Filter (MPHF) to efficiently filter out most of the non-target tags, thereby preventing them from interfering with the information collection of target tags. Fundamentally, MPHF is a filter that consists of $w$ elements, and each of them is a $d$-bit fingerprint of the correspond target tag. The reader sequentially sends each fingerprint in MPHF in a slot of the forthcoming time frame. If a tag who plans to reply in the $x$-th slot finds that the $x$-th fingerprint in MPHF does not match with its own fingerprint, it will recognize that it is a non-target tag and will keep silent in that slot. Thus, most of non-target tags can be filtered out, and won't interfere with the information collection from target tags. Each fingerprint is computed based on the ID of the correspond target tag. We will introduce several suitable fingerprint generators $F(\cdot)$ in Section 7.1, including uniform hash, id slicing and Cyclic Redundancy Checksum. The broadcasting order is based on the slot indexes assigned to target tags in the previous phase.

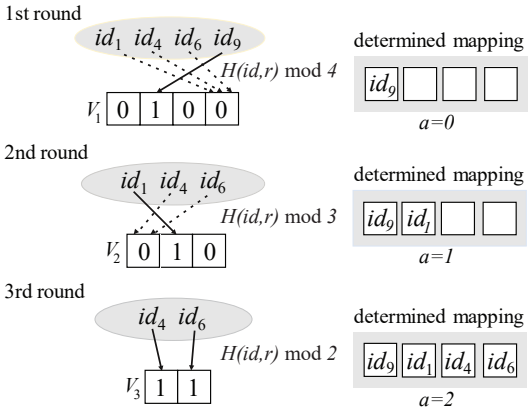



Fig. 3. Exemplifying the process of the assignment phase

### 4.5 Polling Phase

When the fingerprint is relatively long, a non-target tag can be successfully recognized and filtered out with a high probability. However, due to the probabilistic nature of Bloom filtering techniques, a small ratio of non-target tags cannot be deactivated. As we know all the tag IDs and the hashing parameters used in filtering phase, we can predict which non-target tags pass the filtering phase and still keep active. To completely deactivate these non-target tags, in phase three, the reader broadcasts the IDs of such left non-target tags. If a tag finds that its ID is broadcasted by the reader, it knows that it is a non-target tag and will keep silent in the final reporting phase.

### 4.6 Collection Phase

Finally, the reader applies the frame-slotted protocol to collect information from target tags. At the beginning of this phase, the reader issues a `Query` command to start a time frame of $w$ slots. When receiving the `Query` command, each tag loads the slot index, which is obtained in phase one, in its slot counter. The tag whose slot counter $sc$ equals zero immediately transmits its information to the reader. We can confirm that only one target tag responds to the reader during each slot of the time frame because we have assigned tags in set $\mathbb{W}$ to distinct slots in phase one and deactivated all tags in set $\mathbb{U}-\mathbb{W}$ in phase two. As a result, information sent by an arbitrary target tag can be successfully received by the reader. At the end of each slot, the reader broadcasts a `QueryRep` command to start the next slot. After receiving `QueryRep` command, each tag decreases its slot counter by one, *i.e.*, $sc = sc-1$ and responds to the reader immediately if its slot counter equals zero.

In summary, after the assignment phase, each target tag is assigned to an exclusive slot, *i.e.*, there is a one-to-one mapping between target tags and the same number of slots. After the filtering phase, most of the non-target tags are filtered out by MPHF broadcasted from the reader. However, a small number of non-target tags still remain active due to false positives of MPHF. In the polling phase, the reader deactivates these left non-target tags that pass the

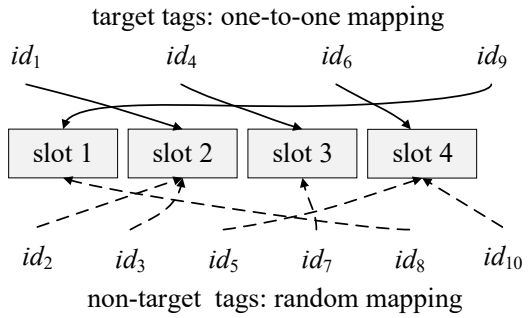



Fig. 4. The mapping results of tags in set $\mathbb{U}$ and slots

check of MPHF. In the collection phase, the reader listens to the channel, and receives the information from each target tag in a non-collision order. Since our protocol deactivates all non-target tags with high-efficient MPHF and removes tag collisions in frame slotted Aloha, the throughput of PIC is improved to be 100%, which significantly improves the efficiency of target tag information collection.

### 4.7 Illustrative Example

For clarity, we give an example to illustrate the operations of the proposed PIC protocol. In this example, there are ten tags $\mathbb{U} = \{id_1, id_2, \cdots, id_{10}\}$, and four among them are target tags $\mathbb{W} = \{id_1, id_4, id_6, id_9\}$. As illustrated in Fig. 3, the slot assignment phase consists 3 rounds in this example. In each round, the reader generates an indicator vector by hashing each non-assigned target tag to a bit. As exemplified in Fig. 3, $id_{\text{i}}$ is mapped to the bit with index of $H(id_{\text{i}}, r) \mod 4$, and this bit is called the representative bit of $id_{\text{i}}$. In the indicator vector, '1' indicates that only one tag is mapped to this bit; '0' indicates that no or more than two tags are mapped to this bit. After generating such an indicator vector $V_1$, the reader broadcasts it to the tags. Each tag receives this vector and checks its representative bit. If the representative bit is 0, the tag is not assigned because it shares this bit with other tag(s); If the representative bit is 1, the tag will be assigned to the slot with index of $(a + x)$, where $a$ indicates the number of target tags that are successfully assigned in previous rounds, and $x$ is the number of '1s' preceding this bit.

For example, in the first round, $id_9$ is successfully assigned to the slot with index of $(a+x) = (0+0) = 0$, where $a$ is obviously 0 because no target tag has been assigned yet, and $x = 0$ because no bit '1' appears preceding its representative bit. On the contrary, $id_1$, $id_4$, and $id_6$ are not assigned, because they are mapped to the same bit due to hashing collision (their representative bit is 0).

The non-assigned target tags (*i.e.*, $id_1$, $id_4$, and $id_6$) need to participate in the second round. $id_1$ is successfully assigned to the slot with index of $(a + x) = (1 + 0) = 1$, where $a = 1$ because one target tag (*i.e.*, $id_9$) has been assigned in the previous round, and $x = 0$ because no bit '1' appears preceding its representative bit.

The remaining non-assigned target tags (*i.e.*, $id_4$ and $id_6$) need to participate in the third round. $id_4$ is successfully assigned to the slot with index of $(a + x) = (2 + 0) = 2$, where $a = 2$ because two target tag (*i.e.*, $id_9$ and $id_1$) have been assigned in previous rounds, and $x = 0$ because no bit '1' appear preceding its representative bit. $id_6$ is successfully assigned to the slot with index of $(a + x) = (2 + 1) = 3$, where $a = 2$ because two target tag (*i.e.*, $id_9$ and $id_1$) have been assigned in previous rounds, and $x = 1$ because one bit '1' (*i.e.*, the bit to which $id_4$ is mapped) appears preceding its representative bit. So far, each target tag is successfully assigned to an exclusive slot in the time frame. It should be noted that each non-target tag also receives $V_1$, $V_2$ and $V_3$ and is assigned to a slot accordingly. We do not describe how the non-target tags do in the example for the purpose of clarity. But note that, non-target tags also participate in this phase and eventually choose a slot in random manner. Fig. 4 shows the final mapping between tags and slots, in which each target tag occupies a slot that contains no other target tags but may contain one or more non-target tags.

TABLE 2
Fingerprints of tags

| tag | fingerprint | tag | fingerprint |
|---|---|---|---|
| $id_1$ | 1111 | $id_6$ | 1101 |
| $id_2$ | 0001 | $id_7$ | 1111 |
| $id_3$ | 0010 | $id_8$ | 1110 |
| $id_4$ | 1100 | $id_9$ | 0101 |
| $id_5$ | 0001 | $id_{10}$ | 1011 |

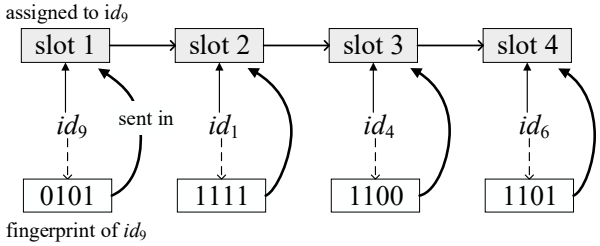



Fig. 5. The MPHF structure consists of $w = 4$ elements, each of them is a $d = 4$-bit fingerprint of a target tag

Next, the reader starts the filtering phase and constructs a Minimal Perfect Hashing Filter (MPHF). Suppose the fingerprint of each tag is shown in Table 2. The constructed MPHF is shown in Fig. 5, which consists of 4 fingerprints with each is 4-bit. Then, the reader broadcasts the fingerprints of the target tags to filter out non-target tags. The broadcasting order is: $id_9$'fingerprint $\rightarrow$ $id_1$'fingerprint $\rightarrow$ $id_4$'fingerprint $\rightarrow$ $id_6$'fingerprint. Specifically, the reader first broadcasts '1011', and the non-target tag $id_8$ can be deactivated because it has a different fingerprint '1110'. Second, the reader broadcasts '1111', which deactivates non-target tags $id_2$ and $id_3$ because they have different fingerprints '0001' and '1100'. Third, the reader broadcasts '1100', and the non-target tag $id_7$ is deactivated because it has a different fingerprint '1111'. Finally, the reader broadcasts '1101' to deactivate non-target tags $id_5$ and $id_{10}$. Because we choose a large enough fingerprint length, all non-target tags have been deactivated in the filtering phase. Thus, we skip the polling phase and start a time frame containing 4 slots to

collect information from 4 target tags, and these target tags will report the information stored on them in the following order: $id_9 \rightarrow id_1 \rightarrow id_4 \rightarrow id_6$.

## 5 Parameter Optimization

In this section, we study how to optimize the parameters involved in our PIC protocol to minimize the transmission overheads of its assignment, filtering and polling phases, which mainly dominate its time-efficiency. We find that the length of bit vector $v$ and the length of fingerprint $d$ are two key parameters that affect the transmission overhead during the assignment phase and deactivating phase (*i.e.*, filtering + polling), respectively. Later in this section, we will present how to set proper values for $v$ and $d$ to minimize the transmission overhead.

### 5.1 Transmissions in Assignment Phase

To minimize the transmission overhead in the assignment phase, the key is to choose a suitable $v_{\rm i}$ during each assignment round. If we use a large $v_{\rm i}$, obviously, a large number of target tags can be assigned by this indicator vector. However, the transmitting overhead of this round will increase accordingly. On the contrary, if we use a small $v_{\rm i}$, the assignment efficiency will be low, the probabilities of assignment collisions (*i.e.*, more than one target tags share a common representative bit) will become serious. Hence, in this section, we propose theoretical analysis to optimize the $v_{\rm i}$ to minimize the transmission overhead $c$ per tag, which is the ratio of transmission overhead to the number $\Delta w$ of target tags that are successfully assigned to exclusive slots.

In an arbitrary round, says $i$, we know the total transmission overhead includes the indicator vector with size of $v_{\rm i}$ and the used parameters $\langle v_{\rm i}, r, a\rangle$. Since each parameter is 16-bit, the total transmission overhead in this round is $v_{\rm i}+64$ bits. Let $\Delta w_{\rm i}$ denote the number of target tags that are successfully assigned by this indicator vector, the per tag transmission overhead can be calculated as follows:

$$c = \frac{v_i+64}{\Delta w_{\rm i}}. \tag{1}$$

The number of assigned tags is expected to be:

$$\Delta w_{\rm i} = w_{\rm i} \times \left(1-\frac{1}{v_{\rm i}}\right)^{w_{\rm i}-1}. \tag{2}$$

By substituting Eq. 2 into Eq. 1, we can obtain the following equation:

$$c = \frac{v_{\rm i}+64}{\Delta w_{\rm i}} = \frac{v_{\rm i}+64}{w_{\rm i}\times\left(1-\frac{1}{v_{\rm i}}\right)^{w_{\rm i}-1}} \approx \frac{v_{\rm i}+64}{w_{\rm i}\times e^{-\frac{w_{\rm i}}{v_{\rm i}}}} \tag{3}$$

Since $w_{\rm i}$ is known to us, $c$ is a function of $v_{\rm i}$. To simplify this function, we substitute the load factor $x_{\rm i}$, which is the ratio of $w_{\rm i}$ to $v_{\rm i}$ in the above equation; then, we have:

$$c = \frac{1+\frac{64x_{\rm i}}{w_{\rm i}}}{x_{\rm i}\times e^{-x_{\rm i}}} \tag{4}$$

To compute the minimum value of $c$, we take the first-order derivative of Eq. 4 and set the right side to 0. Then, we have:

$$\begin{aligned} \frac{dc}{dx_{\rm i}} = & \quad \frac{x_{\rm i}e^{-x_{\rm i}}\times\frac{64x_{\rm i}}{w_{\rm i}} - \left(1+\frac{64x_{\rm i}}{w_{\rm i}}\right)(1-x_{\rm i})\,e^{-x_{\rm i}}}{x_{\rm i}^2 e^{-2x_{\rm i}}} = 0 \\ \Rightarrow & \quad 64x_{\rm i}^2 + w_{\rm i}x_{\rm i} - w_{\rm i} = 0 \end{aligned} \tag{5}$$

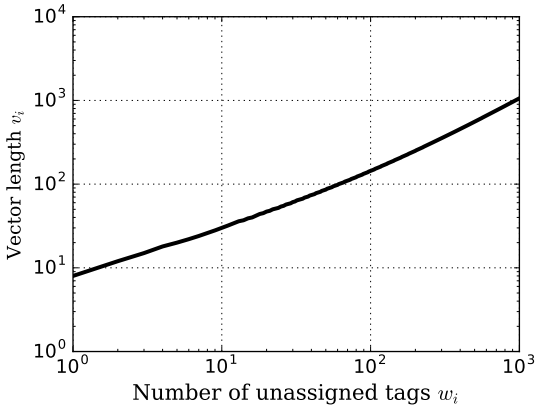



Fig. 6. The length of indicator vector with respect to $w_{\rm i}$

The equation $64x_{\rm i}^2 + w_{\rm i}x_{\rm i} - w_{\rm i} = 0$ have two solutions:

$$x_{\rm i} = \frac{-w_{\rm i} \pm w_{\rm i} \times \left(1+\frac{256}{w_{\rm i}}\right)^{\frac{1}{2}}}{128} \tag{6}$$

Since $x_{\rm i}$ is necessary to be a positive number as $w_{\rm i}$ and $v_{\rm i}$ are both positives, we can confirm one available solution: $x_{\rm i} = \frac{-w_{\rm i}+w_{\rm i}(1+256/w_{\rm i})^{1/2}}{128}$. From Eq. 5 and Eq. 6, we know $\frac{dc}{dx_{\rm i}} < 0$ when $0 < x_{\rm i} < \frac{-w_{\rm i}+w_{\rm i}\times(1+256/w_{\rm i})^{1/2}}{128}$ and $\frac{dc}{dx_{\rm i}} > 0$ when $x_{\rm i} > \frac{-w_{\rm i}+w_{\rm i}(1+256/w_{\rm i})^{1/2}}{128}$. Therefore, with the increase of $x_{\rm i}$, the value of $c$ first increases and then decreases, and $c$ is minimized when:

$$x_{\rm i} = \frac{-w_{\rm i} + w_{\rm i} \times (1+\frac{256}{w_{\rm i}})^{\frac{1}{2}}}{128}. \tag{7}$$

Since $v_{\rm i} = w_{\rm i}/x_{\rm i}$, we can also calculate the value of $v_{\rm i}$ that minimizes the value of $c$:

$$v_{\rm i} = \frac{128}{\left(1+\frac{256}{w_{\rm i}}\right)^{\frac{1}{2}} - 1} = \frac{w_{\rm i}}{128}\left[1+\left(1+\frac{256}{w_{\rm i}}\right)^{\frac{1}{2}}\right] \tag{8}$$

Therefore, to minimize $c$, in each assignment round, we should set $v_{\rm i}$ based on Eq. 8. For clearly representing the relationship of $v_{\rm i}$ and $w_{\rm i}$, we plot $v_{\rm i}$ with respect to $w_{\rm i}$ in Fig. 6. We observe that $v_{\rm i}$ is in proportion to $w_{\rm i}$. When $w_{\rm i}$ is small, *e.g.*, $w_{\rm i} = 10$, $v_{\rm i}$ is set to 2 times of $w_{\rm i}$. But when $w_{\rm i}$ is large, *e.g.*, $w_{\rm i} > 100$, we can find that $v_{\rm i} \approx w_{\rm i}$. This is because when $w_{\rm i}$ is small, parameters account for the majority of transmission overhead $\langle v_{\rm i}, r, a\rangle$, in this case, we need to assign the small set of remaining target tags in as fewer rounds as possible to reduce the parameter transmissions.

### 5.2 Total Transmissions in Filtering&Polling Phases

To completely deactivate non-target tags, we use a hybrid method containing probabilistic filtering phase and precise polling phase. The communication overhead of these two phases are determined by the length $d$ of fingerprint. Specifically, if we use a small $d$, the transmission overhead of MPHF is obviously small. However, a large number of non-target tags will be left to the polling phase due to the high false positive rate of MPHF. Then, the transmission

overhead of polling phase will increase accordingly. On the contrary, if we use a long fingerprint, the transmission overhead of polling phase will be reduced. However, the overhead of the filtering phase will increase. Hence, in this section, we propose theoretical analysis to optimize the length $d$ of the fingerprint to minimize the total transmission overhead of filtering and polling phases.

We assume the fingerprints are pseudo-random values. For a certain $d$, the communication overhead of MPHF is $w \times d$ bits. the false positive probability of MPHF can be expressed as $(1/2)^d$. After filtering phase, the number of left non-target tags can be expressed as $(u-w)(1/2)^d$. To remove these tags, the reader issues ids of them one by one, incurring a polling overhead of $96 \times (u-w) \times (1/2)^d$ bits. Therefore, the total overhead $T_d$ for completely deactivating non-target tags can be expressed as:

$$T_d = w \times d + 96 \times (u-w) \times (1/2)^d \tag{9}$$

To compute the minimum value of $T_d$, we take the first-order derivative of Eq. 9 and set the right side to be 0. Then, we have:

$$w \times d - 96 \times (u-w) \times 2^{-d} \times \ln 2 = 0. \tag{10}$$

Solving the above equation, we have

$$d = -\frac{\ln\left[\frac{w}{96 \ln 2 \times (u-w)}\right]}{\ln 2} \approx -\log_2\left(\frac{w}{u-w}\right) + 6, \tag{11}$$

which provides guidance on how to set $d$ to minimize the communication overhead. We observe that $d$ is determined by the ratio of the number of target tags to the number of non-target tags. Substituting Eq. 11 into Eq. 9, we can obtain the optimal $T_d$ as follows:

$$T_d = w \times \left[-\log_2\left(\frac{w}{u-w}\right) + 7.5\right] \tag{12}$$

## 6 Performance Analysis

In this section, we theoretically analyze the expected execution time of PIC and compare it with the existing protocols. We also elaborate on the reasons why our PIC protocol outperforms the state-of-the-art protocols.

### 6.1 Expected Execution Time of PIC

Since PIC consists four phases, in the following, we present the overhead of each phase respectively.

•**Assignment phase:** In this phase, we assume that the reader broadcasts $k$ indicator vectors $\{V_1, V_2, \cdots, V_k\}$ to assign tags in set $\mathbb{W}$. The length of indicator vectors $V_{\mathrm{i}}$ is denoted as $v_{\mathrm{i}}$. Therefore, the total length of indicator vector is $\sum_{i=1}^{k} v_{\mathrm{i}}$ bits. Let $w_{\mathrm{i}}$ and $\Delta w_{\mathrm{i}}$ denote the number of target tags before broadcasting the $i$th indicator vector and the number of assigned target tags by the $i$-th indicator vector. We have $v_{\mathrm{i}} = \Delta w_{\mathrm{i}} \times c$, where $c$ is the minimum transmission overhead per tag. The total transmission overhead in the assignment phase, denoted as $T_m$, can be expressed as follows.

$$T_m = \sum_{i=1}^{k} \Delta w_{\mathrm{i}} \times c + 64 \times k, \tag{13}$$

where $64 \times k$ represents the overhead of broadcasting the parameters. Because $c$ is a concave function of $w_{\mathrm{i}}$, we can represent it as $c(w_{\mathrm{i}})$. According to Jensen's inequality, we have the following expression:

$$\frac{T_m}{\sum_{i=1}^{k} \Delta w_{\mathrm{i}}} \leq c\left(\frac{\sum_{i=1}^{k} \Delta w_{\mathrm{i}} \times w_{\mathrm{i}}}{\sum_{i=1}^{k} \Delta w_{\mathrm{i}}}\right) + \frac{64 \times k}{\sum_{i=1}^{k} \Delta w_{\mathrm{i}}}. \tag{14}$$

Because $\sum_{i=1}^{k} \Delta w_{\mathrm{i}} = |\mathbb{W}|$, by substituting it into the Eq. 14, we have the following inequality.

$$T_m \leq |\mathbb{W}| \times c\left(\sum_{i=1}^{k} \Delta w_{\mathrm{i}} \times \frac{w_{\mathrm{i}}}{|\mathbb{W}|}\right) + 64 \times k. \tag{15}$$

As $\Delta w_{\mathrm{i}} = w_{\mathrm{i}} \times q_{\mathrm{i}}$ and $q_{\mathrm{i}} \approx 1/e$, we can simplify it as follows:

$$\begin{aligned} T_m &\leq |\mathbb{W}| \times c\left(\sum_{i=1}^{k} q_{\mathrm{i}} \times \frac{w_{\mathrm{i}}^2}{w}\right) + 64 \times k \\ &\approx |\mathbb{W}| \times c\left(\sum_{i=1}^{k} \frac{w_{\mathrm{i}}^2}{e \times |\mathbb{W}|}\right) + 64 \times k \\ &\approx w \times c\left(\frac{\left[1 - \left(1 - \frac{1}{e}\right)^2 k\right] \times w}{\left[1 - \left(1 - \frac{1}{e}\right)^2\right] \times e \times w}\right) + 64 \times k \\ &\leq |\mathbb{W}| \times c\left(\frac{w}{2 - \frac{1}{e}}\right) + 64 \times k \\ &\approx |\mathbb{W}| \times c\,(0.613w) + 64 \times k. \end{aligned} \tag{16}$$

From the above equation, we can conclude that $T_m$ is linear in $w$. The coefficient of $w$ decreases with the increase of $w$. For example, when $w = 200$, the communication overhead is $T_m \approx 4.16w$. When $w = 1,000$, the communication overhead of assignment phase is only $T_m \approx 2.88w$ bits.

•**Filtering and Polling phases:** To deactivate non-target tags, the communication overhead is composed of broadcasting the minimal perfect hashing filter and polling IDs of the non-target tags that are not filtered out. Given the specific values of $u$ and $w$, the minimized transmission overhead of the filter phase has been given in Eq. 12.

•**Collection phase:** In the collection phase, each target tag is able to transmit to the reader without collisions, and the overhead of this phase, denoted as $T_r$, can be expressed as $T_r = |\mathbb{W}| \times l$, where $l$ is the length of the information and $|\mathbb{W}|$ represents the number of target tags. The common length of $l$ includes 1 bit of the Boolean type, an 8-bit integer and a 16-bit float. Let $R_r$ denote the bit rate from the reader to tags and $R_t$ denote the bit rate from the tags to reader, we obtain the expected execution time of PIC:

$$T_{PIC} = (T_m + T_d)/R_r + T_r/R_t. \tag{17}$$

The execution time of PIC is also linear in $w$, and the coefficient is affected by the values of $w$ and $u$. For example, when $w = 1,000$ and $u = 10,000$, we have $T_m \approx 2.9w$ bits, $T_d \approx 10.6w$ bits. As a result, $T_{PIC}$ is given as $T_{PIC} \approx w \times 13.5/R_r + n \times l/R_t$ in this case.

### 6.2 Execution Time of the Polling Protocol

The polling protocol is a direct solution for collecting information from target tags [16]. In polling protocol, the reader only needs to broadcast IDs of target tags one by one. Each tag listens to the channel and sends its information to the reader if it finds that its id is broadcast by the reader. This protocol incurs a communication overhead of $96w$ bits. Therefore, the execution time of polling protocol $T_{PA}$ is given as follows:

$$T_{PA} = n \times 96/R_r + n \times l/R_t. \quad (18)$$

Although, the execution time of polling protocol is also linear in $w$, the coefficient of $w$ equals to be $96 + l$, which is much larger than that of PIC. For example, when $l = 8$, PIC incurs a $20n$ bits broadcasting overhead. Meanwhile, polling protocol costs $104n$ bits, which is five times that of PIC. Obviously, PIC significantly reduces the communication overhead than the polling protocol.

### 6.3 Execution Time of Enhanced Tag Ordering Protocol

To reduce the communication overhead, the state-of-the art protocol ETOP [18] uses a compact partition Bloom filter, *i.e.*, Tag Ordering Vector (TOV), to represent target tags in set $\mathbb{W}$ and avoids time-consuming transmissions of IDs. TOV consists of several partitions, and each target tag is mapped to one of them by hashing its ID. Each partition is also divided into multiple segments, and the tags in these partition will be further mapped to a bit in each segment by hashing its ID multiple times with different seeds. A tag can know whether it is a target tag by checking if all bits it maps to are 1s. If all of them are bit '1', it is a target tag; otherwise it is a non-target tag. A major contribution of TOV is that the reporting order of target tags is encoded in it. Specifically, if the number of '1'-bit in a segment equals to the number of tags in the correspond partition, we call it ordering segment. Each tag in this partition can respond to the reader based on their locations in the ordering segment. Let $T_o$ be the length of the TOV, the communication overhead of broadcasting TOV is $T_o/R_r$ bits. However, a small set of target tags cannot be filtered out due to the false positive of TOV, which can be expressed as $p_f \approx (1 - e^{-80n/T_o})(1 - e^{-ws/80})$ [18], where $s$ is the number of segments in each partition. Thus, the left non-target tags can be filtered out by broadcasting their IDs, which incurs a communication overhead of $p_f \times 96$ bits. Except polling IDs of the left non-target tags, we also need to poll the IDs of some conflicting tags in the partitions that have no ordering segment. According to [18], the number of conflicting tags can be presented as $g = \frac{w^3 \times 80^2}{T_o}$. Therefore, the total overhead of polling operation incurs a communication overhead of $[g + (u - w) \times p_f] \times 96$ bits. Finally, since all collisions have been removed, to collect information from target tags incurs a communication overhead of $w \times l$ bits. In conclusion, the total transmission overhead of ETOP is calculated as follows:

$$T_{ETOP} = [T_o + 96c + 96(u - w)p_f]/R_r + w \times l/R_t \quad (19)$$

Obviously, the execution time of ETOP is also linear in $w$. As the authors focus on minimizing the energy consumption per tag, how to optimize the $T_{ETOP}$ is not discussed in [18]. Based on our observation, the coefficient of ETOP is smaller than that of polling protocol but larger than that of PIC. For example, when $w = 1,000$, $u = 10,000$, $l = 8$, $s = 4$ and $T_o = 20w$, ETOP incurs a communication overhead of $34.6w$ bits, which is $1.58$ times that of PIC.

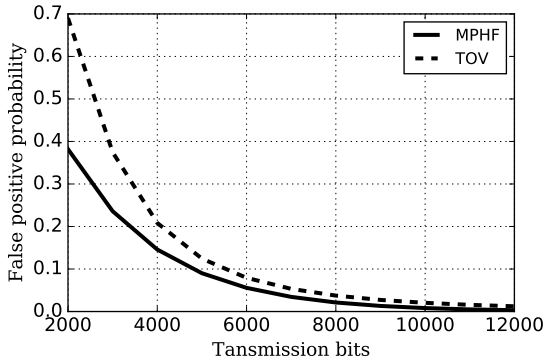



Fig. 7. Comparison of MPHF and the TOV when $w = 1,000$ and $s = 4$

### 6.4 Advantages of PIC

Although the time complexity of the proposed PIC and the state-of-the-art ETOP protocol are both $O(w)$, PIC takes much less time due to the small coefficient of $w$. To investigate the reasons why our PIC outperforms ETOP, we analyze each phase of these two protocols and find two major advantages of PIC. First, the minimal perfect hashing filter used in PIC is more compact than the tag ordering vector used in ETOP, because it leverages the locations of target tags (*i.e.*, assigned slots) to filter out non-target tags. Thus, when broadcasting a same length MPHF and TOV, the false positive probability of MPHF is much smaller than that of TOV. See Fig. 7 for illustration, the false positive probability of MPHF is only 58% of the TOV. Therefore, the number of left non-target tags after broadcasting MPHF is only half of that after broadcasting TOV, which incurs a small polling overhead. Second, MPHF assigns each target tag to an exclusive slot in the time frame, thus each tag can send its information in the assigned slot to avoid collisions. Meanwhile, due to the random mapping, some target tags are conflicted in TOV, it has to further poll their IDs, which incurs additional communication overhead, further prolongs the execution time of ETOP.

### 6.5 Lower Bound on the Required Execution Time

To show the high efficiency of PIC, we also present the lower bound on the execution time of target information collection based on compact approximator. The results revels that the execution time of PIC is very close to the lower bound. First, the absolute lower bound of collecting $l$-bit information from $w$ target tags is exactly to be $w \times l$ bits [25]. To meet this lower bound, the utilization ratio of the time frame should be 100% and each tag in $\mathbb{W}$ should know which slot to send its information to the reader. Since the tags in $\mathbb{W}$ frequently changes according to the user's specification, the reader has to issue some message to tell tags whether they are target tags in this execution. Thus, the actual overhead is always larger than $w \times l$ bits.

As far as we know, the compact approximator, which is an array of bits, is the most space-efficient tool to represent a set. According to a well known conclusion in information theory, in a system with $u$ interrogated tags and $w$ target tags, if $1 < w \ll u$, a compact approximator, that allows for identifying $w$ target tags with a false positive probability $p_f$, is larger than $w \log_2(1/p_f)$ bits [31]. Therefore, broadcasting the compact approximator caused at least $w \log_2(1/p_f)$ bits overhead. Besides, to remove the non-target tags caused by false positive of compact approximator, the reader has to issue $p_f \times (u - w) \times 96$ bits to poll their IDs. As a result, the total overhead of compact approximator-based solution is lower bounded by:

$$T_{LB} = w \times \log_2(1/p_f + (u-w) \times 96 \times p_f + w \times l/R_t. \quad (20)$$

We take the first-order derivative of $p_f$ and set the right side to 0. Then, we have the optimal $p_f$ that minimizes $T_{LB}$ as follows:

$$\begin{aligned} &\frac{dT_{LB}}{dp_f} = w \times \log_2(1/p_f) + (u-w) \times 96 \times p_f = 0 \\ &\Rightarrow p_f = \frac{w}{(u-w) \times 96 \times \ln 2} \approx \frac{0.015w}{u-w}. \end{aligned} \quad (21)$$

By substituting the optimal $p_f$ into Eq. 20, we obtain the optimal communication overhead:

$$T_{LB} = |\mathbb{W}| \times \left[ \log_2 \left( \frac{u-w}{w} \right) + 7.5 \right] / R_r + w \times l / R_t. \quad (22)$$

From Eq. 9 and Eq. 22, we find out that $T_{LB} + T_m/R_r = T_{PIC}$, indicating that without consideration the overhead for assigning tags to exclusive slots in the time frame, PIC achieves the theoretical lower bound on the execution time of target information collection based on the compact approximator. This well explains why PIC has better time-efficiency than prior schemes.

## 7 Discussion

In this section, we discuss some practical issues in PIC implementation. We also discuss how to optimize the protocol under some specific application scenarios.

### 7.1 Fingerprint Generator $F(\cdot)$

In the filtering phase of PIC, the reader and tags should share a fingerprint generation function $F(\cdot)$ to implement the minimal perfect hashing filter. To fit for the RFID tag with limited computation capabilities, the generation function $F(\cdot)$ should be lightweight and in accordance with the off-the-shelf RFID tags. In the following, we list three suitable generation functions.

First, we can simply take a slice of 96-bit id as the fingerprint. This method is simple and direct but may not work well when product information is encoded into the ID. For instance, the EPC-ID encodes the category and producer information, thus many tags may have the same ID segments. Thus, the slices of distinct IDs may be easy to collide with others, which affect the filtering efficiency of the minimal perfect-hashing filter. Besides, the slicing operation is not implemented by the off-the-shelf tags. Second, we can also leverage the uniform hashing function $H(\cdot)$, which is usually used for mapping a tag to a slot in the frame slotted Aloha. Although some well-designed hash function, *e.g.*, MD5 and SHA1, may produce a better fingerprint with less collision but they are not supported by off-the-shelf tags. Third, we can use the Cyclic Redundancy Checksum (CRC) function, which is usually used to control the transmission error between the reader and tags. According to EPC C1G2 standard, each RFID tag should carry a CRC precursor to check the received data. As we know, the CRC precursor is occasionally used and works well as a hash function. Thus, we can leverage the produced CRC code as the fingerprint of EPC ID. Although there are two types of supported CRC on C1G2-complaint RFID tags: the 16-bit CRC-16 and the 5-bit CRC-5, we can only apply CRC-16 because CRC-5 produces a 5-bit CRC code, which is too short and easy to be collided with other CRC-5, generated by different ids. Meanwhile, CRC-16 has a much smaller collision probability of $1/65536$, which is long enough to meet our requirements. The detailed computation of CRC-16 is as follows: Let $R(y)$ denote the CRC-16 of $y$; we can obtain $R(y)$ based the equation $y << 16 = G(x)Q(x) + R(y)$, where $Q(x)$ is an integer and $G(x) = 1100000000000011$ [26]. In conclusion, both the uniform hash function and CRC-16 function are suitable for generating fingerprint. They are both lightweight enough and leverage the current functions on commercial RFID tags.

### 7.2 Electronic Product Code (EPC)

In PIC design, we assume the IDs in set $\mathbb{U}$ are randomly generated within $[0, 2^{96} - 1]$. In fact, this is always not true in the real application. As we known, Electronic Product Code (EPC), proposed by EPCglobal, is commonly used ID in the logistic and warehouse management. EPC is designed as a universal identifier that provides a unique identity for every physical object anywhere in the world. It consists of 4 segments, including an 8-bit EPC header, 28-bit manager number, 24-bit object class and 36-bit serial number. If two items belong to the same category, most bits of their IDs are the same. Leveraging this feature, we can further improve the efficiency of PIC.

Instead of constructing a filter based on the IDs in set $\mathbb{W}$, we can add an additional filter phase, which is used to filter tags from unrelated class. Since an item class usually contains multiple items, the number of target classes is usually much smaller than that of the tags. In the added filter phase, the reader broadcasts a filter based on the target classes, which incurs a much smaller overhead compared with broadcasting a filter based on target tags. As a large amount of tags from unrelated class have been deactivated, the reader can construct a shorter bit vector in the forthcoming filtering phase.

### 7.3 Without the Exact Knowledge of IDs in $\mathbb{U}$

In PIC design, we assume the reader exactly knows the IDs of all the tags in set $\mathbb{U}$. However, in a dynamic RFID system where tags move in and move out frequently, it is hard to ensure that the reader always holds the up-to-date records of IDs in $\mathbb{U}$. In this subsection, we present an enhanced PIC protocol, which can work without exact knowledge of IDs in set $\mathbb{U}$. To obtain the up-to-date IDs in set $\mathbb{U}$, we can let the reader execute the identification protocol before running

PIC. However, the ID identification protocol is too time-consuming, which is several times of that of PIC. To reduce the overhead of ID identification protocol, we have to run it once after a long period (such as once an hour/day, for instance). That is, we may not have the exact set of tags at some time.

Fortunately, with some modification, the original PIC can work well even when the reader does not know the exact IDs in the system. The modified PIC drops off the polling phase because we do not know the exact set of tags in the interrogation range. Therefore, it only consists of three phase as follows: In the first phase, the reader assigns each target tag to an exclusive slot in the time frame. This phase is same as the original assignment phase of PIC. In the second phase, the reader broadcasts a minimal perfect hashing filter to filter out non-target tags. As the reader has no knowledge of $\mathbb{U}$, it cannot give an optimal $d$ to optimize the time-efficiency of this phase; therefore, the enhanced PIC chooses a relatively large 16-bit fingerprint, which is large enough to filter most of non-target tags. There is no polling phase in the enhanced PIC protocol. As the reader does not know the exact tags in set $\mathbb{U}$, we cannot know which non-target tags in set $\mathbb{U}$ pass the check of MPHF. Finally, the reader issues a request to query information from all active tags. As the second phase cannot ensure all the non-target tags have been deactivated, we may detect three types of slots during the time frame: singleton slots, collision slots and empty slots. Only the target tag in singleton slot can successfully transmit its information to the reader. The empty slot represents the target tag that maps to this slot and is no longer in the system, and the reader needs to remove it from the tag set. The collision slot represents some non-target tags mapped to this slot. To collect information from the target tag in this slot, the reader needs to poll the ID at the end of this phase.

A major problem of the enhanced PIC is that the responses in singleton slots are regarded as transmitting from the target tags. However, this assumption is not always true. If a target tag is not in the system, and a non-target tag, unknown to the reader, happens to map to the slot assigned to this target tag, the reader cannot detect the absence of this target tag and mistakenly thinks that the information received in this slot is sent by the original target tag. Let $p_e$ denote the probability of this kind of error, we have:

$$p_e = \frac{m}{w} \times \frac{s}{w} \times \left(1 - \frac{1}{w}\right)^s, \tag{23}$$

where $s$ and $m$ denote the number of unknown tags and missing tags, respectively. If $s$ and $m$ are small enough, we can limit the value of $p_e$ under an acceptable level. Thus, we need to periodically execute the ID identification protocol to prevent the values of $s$ and $m$ from becoming too large.

## 8 Performance Evaluation

In this section, we conducted extensive simulations to evaluate the performance of the proposed PIC and compared it with the state-of-the-art protocol ETOP and the lower bound on the execution time of the compact approximator. In what follows, we first present the detailed simulation settings used in this paper. Then, we conducted simulations to investigate the transmission overhead in each phase of PIC. Finally, we compare PIC with ETOP and the lower bound side-by-side.

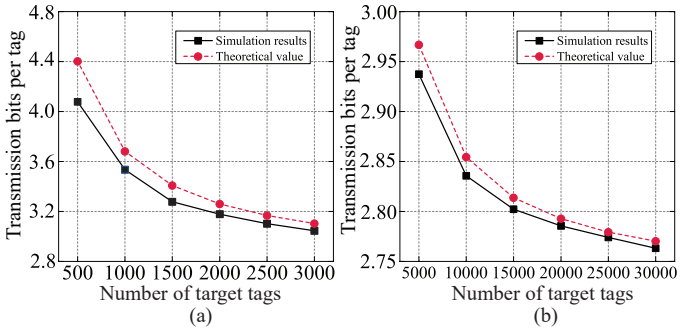



Fig. 8. The assignment overhead per tag with respect to $w$ in simulations

### 8.1 Simulation Setting

The simulation setting is based on the C1G2 standard [26]. which specifies the tag-to-reader transmission rate to be either $40-460kb/s$ in the FM0 encoding format or $5-320kb/s$ in the Miller modulated subcarrier encoding format, and the reader-to-tag transmission rate to be about $26.7-128kb/s$. In our simulation, we consider that the system works on low bit rates and set both the reader-to-tag transmission rate and the tag-to-reader transmission rate to be $64kb/s$. In this case, the execution time of the above protocols is proportional to the number of total transmission bits. Therefore, we can use the transmission bits as the metric to evaluate the performance of the proposed protocol. Without specific illustration, the presented results are averages over 100 simulation runs. In each run, the IDs of interrogated tags $U$ are randomly picked from $[0, 2^{96}-1]$, and target tags are randomly picked from $U$. As ETOP [18] does not provide optimal parameters for minimizing the execution time, when comparing with ETOP, we set the length of the partition Bloom filter to be $24w$ and the number of segments each partition to be $4$ in the simulations, which is a time-efficient setting of ETOP and was used for time comparison with other method in the literature [18].

### 8.2 Overhead of Assigning Target Tags

We first investigate the transmission overhead of minimal perfect hashing mapping, which assigns each target tag in set $\mathbb{W}$ to an exclusive slot in the time frame. Fig. 8(a) shows per tag assignment overhead when we vary $w$ from 500 to $3,000$. We observe that the per tag overhead is decreased with the increase of $w$. For example, when $w = 500$, the per tag overhead is 4.4 bits; when $w = 3,000$, the per tag overhead reduces to 3.2 bits. This is because, when $w$ is large, the parameter transmission overhead will be shared by a large number of tags and thus is negligible. Whereas, when $w$ is small, the parameter transmission overhead accounts for a large proportion of total overhead, which increases the per tag overhead significantly. The results in Fig. 8(b) further support the conclusion that the per tag construction overhead is reduced with the increase of $w$; However, the descending rate decreases with the increase of $w$, and the average overhead per tag approaches to 2.73 bits.

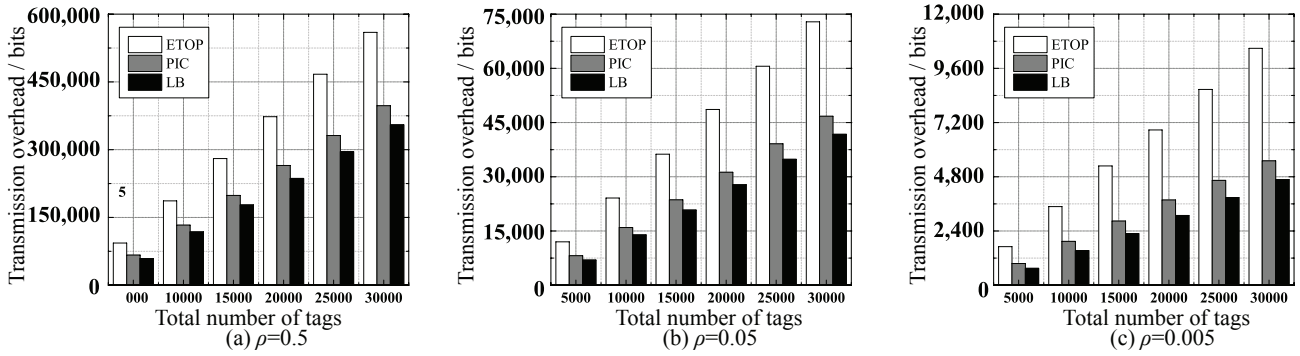



Fig. 10. Comparing the time-efficiency of each protocol. $\rho \in \{0.5, 0.05, 0.005\}$. For each $\rho$, $l$ is 16 bits and $u$ varies from 5,000 to 30,000.

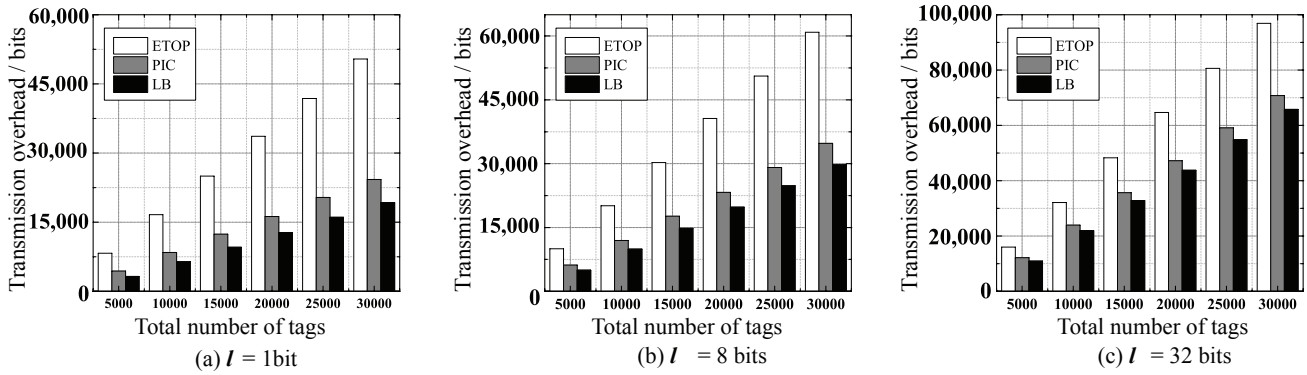



Fig. 11. Comparing the time-efficiency of each protocol. $l \in \{1$ bit, 8 bits, 32 bits$\}$. For each $l$, $\rho$ is 0.05 and $u$ varies from 5,000 to 30,000.

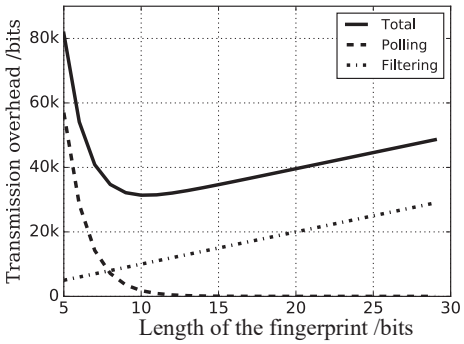



Fig. 9. Transmission overhead vs. the length of fingerprint

### 8.3 Impact of $d$ on the Execution Time of PIC

Then, we investigate how $d$ affects the execution time of our PIC protocol. We set $u = 20,000$, $w = 1,000$, $l = 16$ and vary the value of $d$ from 1 to 16. The results depicted in Fig. 9 show that with the increases of $d$, the total transmission overhead first decreases and then increases. This is because when $d$ is small, the minimal perfect hashing filter can only deactivate a small number of non-target tags. Thus, we have to poll massive IDs to deactivate the left non-target tags, which incurs a significant polling overhead as shown in Fig. 9. On the other hand, when $d$ is large, almost all the non-target tags have been removed. In this case, the further increasing in $d$ has little impact on reducing the polling overhead, instead, it increases the filtering overhead. As a result, when $d$ exceeds a certain threshold, the total overhead begins to increase. We observe that the total transmission overhead is minimized when $d = 10$, whose value approaches 30,000 bits, which is only half of the overhead required when $d = 5$. The observed optimal $d = 10$ matches well with the theoretical optimal value computed based on Eq. 12. The results of this set of simulations highlight the importance of optimizing the fingerprint length $d$.

### 8.4 Overhead in Each Phase of PIC

We then investigate the overhead of each phase under different parameter settings to investigate which phase dominates the overall time-efficiency of PIC. We set $u = 20,000$, $l = 4$ and vary $w$ from $1,000$ to $10,000$ to study the overhead of each phase. Fig. 12 shows the proportion stays stable with the increases in the number of target tags. The filtering phase always accounts for the largest proportion of execution time in our settings. Thus, to further improve the efficiency of PIC, the key challenge is to improve the filtering phase.

### 8.5 Comparison with Prior Work

We evaluate the performance of PIC by comparing it with the ETOP and LB. To allow different application scenarios, we use different parameters combinations to test the

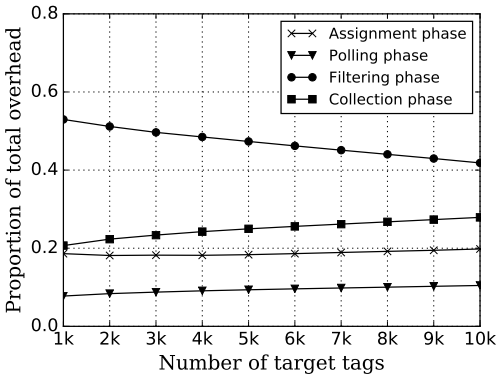



Fig. 12. The proportion of transmission overhead of different phases in the number of tags in the interrogation zone fixed at $20,000$ and the number of target tags range of $[1000, 10000]$.

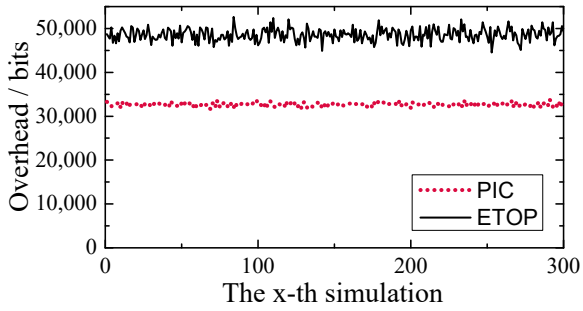



Fig. 13. Transmission bits between the reader and the tags when running PIC and ETOP. The X coordinate denotes the ordinal of the simulations

performance of these protocols. First, we assume that the length $l$ is 16 bits, and set the proportion of target tags $\rho$ to be 0.5, 0.05 and 0.005. For each $\rho$, $u$ varies from 5,000 to 30,000. Fig. 10 illustrates the comparison in terms of transmission bits among ETOP, PIC and LB. We observe that PIC significantly outperforms ETOP in all scenarios by significantly reducing the transmission bits. For example, when $\rho = 0.5$, $u = 10,000$, the number of transmission cost of ETOP is $186,710$ bits, while PIC and LB transmit $132,849$ bits and $118,158$ bits, respectively. Compared with ETOP, PIC reduces the transmission bits by $28.8\%$, and is only 1.13 times that of LB. This improvement is due to the high efficiency of the minimal perfect hashing filter used by PIC. Another observation is the gap between ETOP and PIC increases with the increase of $\rho$. For example, when $\rho = 0.005$ and $u = 10,000$, PIC is $44.3\%$ faster than ETOP and is only 1.29 times of the lower bound. The gap between ETOP and PIC becomes larger with the increases of $u$. And the gap between PIC and ETOP also becomes larger with the decreases of $\rho$.

Next, we set $\rho = 0.05$ and set $l$ to 1 bit, 8 bits and 32 bits, respectively. For each value of $l$, we change the number of the total tags from $5,000$ to $30,000$. As shown in Fig. 11, the gap between ETOP and PIC decreases as the increases of $l$. For example, when we set $l = 1$ and $u = 25,000$, PIC is $51.3\%$ faster than ETOP, whereas, when $l$ increases to 32 bits, PIC only saves $26.5\%$ in terms of transmission cost. This observation matches our expectations because our technique mainly improves the efficiency of deactivating non-target tags. With the increases of $l$, the overhead of reporting information is increased, thus accounting for a larger proportion of total overhead. As a result, the overhead of deactivating non-target tags accounts becomes relatively small. The benefit brought by PIC naturally has less effect to the total execution time.

Finally, Fig. 13 compares the stability of PIC and ETOP. We set $u = 20,000$, $\rho = 0.05$ and execute each protocol 300 times. Unsurprisingly, PIC always outperforms ETOP. The variation in time of ETOP is $mean \pm 9.3\%$, and the variation in time of PIC is only $mean \pm 3.8\%$. The worst performance of PIC is $64\%$ of ETOP, that is, PIC has a more stable performance compared with ETOP.

## 9 Conclusion

This paper studies the problem of target tag information collection in large-scale RFID systems, and proposes a novel protocol called Perfect hashing-based Information Collection (PIC). Compared with prior work, PIC can completely prevent the interferences from non-target tags and also avoid the collisions among target tags in a more efficient way. To minimize the transmission cost of PIC, we propose sufficient theoretical analyses to optimize the parameters that have significant impact on its time-efficiency. Extensive simulations are conducted to evaluate the performance of the PIC protocol. The simulation results demonstrate that PIC significantly outperforms the state-of-the-art protocol and its transmission overhead is very close to the lower bound on the required execution time.

## Acknowledgment

This work is supported by the National Key Research and Development Program of China (Grant No. 2016YFB1000205); the State Key Program of National Natural Science of China(Grant No. 61432002); NSFC Grant Nos. 61370199, 61672379, and 61373181; the Dalian High-level Talent Innovation Program (Grant No. 2015R049); and the Fundamental Research Funds for the Central Universities (Grant No. DUT15QY20). This work is partially supported by Tianjin Key Laboratory of Advanced Networking (TANK), Tianjin 300350, China.

## References

[1] C.-H. Lee and C.-W. Chung, "Efficient storage scheme and query processing for supply chain management using rfid," in *Proc. of ACM SIGMOD*, 2008.

[2] X. Liu, K. Li, J. Wu, A. X. Liu, X. Xie, C. Zhu, and W. Xue, "TOP-$k$ Queries for Multi-category RFID Systems," *Proc. of IEEE INFOCOM*, 2016.

[3] X. Liu, B. Xiao, K. Li, J. Wu, A. X. Liu, H. Qi, and X. Xie, "RFID Cardinality Estimation with Blocker Tags," *Proc. of IEEE INFOCOM*, 2015.

[4] M. Shahzad and A. X. Liu, "Every Bit Counts: Fast and Scalable RFID Estimation," *Proc. of ACM MobiCom*, 2012.

[5] X. Liu, K. Li, A. X. Liu, S. Guo, A. L. Wang, X. Xie, and J. Wu, "Multi-category rfid estimation," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 264–27, 2017.

[6] X. Liu, B. Xiao, K. Li, A. X. Liu, J. Wu, X. Xie, and H. Qi, "Rfid estimation with blocker tags," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 224–237, 2017.

[7] T. Liu, L. Yang, Q. Lin, Y. Guo, and Y. Liu, "Anchor-free backscatter positioning for rfid tags with high accuracy," in *Proc. of IEEE INFOCOM*, 2014.

[8] X. Liu, K. Li, H. Qi, B. Xiao, and X. Xie, "Fast Counting the Key Tags in Anonymous RFID Systems," *Proc. of IEEE ICNP*, 2014.

[9] L. Yang, Y. Chen, Xiang-Yang, C. Xiao, M. Li, and Y. Liu, "Tagoram:real-time tracking of mobile rfid tags to high precision using cots devices," in *Proc. of ACM Mobicom*, 2014.

[10] X. Liu, X. Xie, K. Li, B. Xiao, J. Wu, H. Qi, and D. Lu, "Fast tracking the population of key tags in large-scale anonymous rfid systems," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 278–291, 2017.

[11] W. Gong, K. Liu, X. Miao, Q. Ma, Z. Yang, and Y. Liu, "Informative counting: fine-grained batch authentication for large-scale rfid systems," in *Procr. of ACM Mobihoc*, 2013.

[12] J. Liu, B. Xiao, K. Bu, and L. Chen, "Efficient distributed query processing in large rfid-enabled supply chains," in *Proc. of IEEE INFOCOM*, 2014.

[13] R. Jedermann, L. Ruiz-Garcia, and W. Lang, "Spatial temperature profiling by semi-passive rfid loggers for perishable food transportation," *Computers and Electronics in Agriculture*, vol. 65, no. 2, pp. 145–154, 2009.

[14] L. Xie, Q. Li, C. Wang, X. Chen, and S. Lu, "Exploring the gap between ideal and reality: An experimental study on continuous scanning with mobile reader in rfid systems," *IEEE Transactions on Mobile Computing*, vol. 14, no. 11, pp. 2272–2285, 2015.

[15] X. Liu, S. Zhang, K. Bu, and B. Xiao, "Complete and fast unknown tag identification in large RFID systems," in *Proc. of IEEE MASS*. IEEE, 2012.

[16] D.-H. Shih, P.-L. Sun, D. C. Yen, and S.-M. Huang, "Taxonomy and survey of rfid anti-collision protocols," *Elsevier Computer communications*, vol. 29, no. 11, pp. 2150–2166, 2006.

[17] S. Chen, M. Zhang, and B. Xiao, "Efficient information collection protocols for sensor-augmented RFID networks," in *Proc. of IEEE INFOCOM*, 2011.

[18] Y. Qiao, S. Chen, T. Li, and S. Chen, "Energy-efficient polling protocols in RFID systems," in *Proc. of ACM Mobihoc*, 2011.

[19] H. Liu, W. Gong, X. Miao, K. Liu, and W. He, "Towards adaptive continuous scanning in large-scale rfid systems," in *Proc. of IEEE INFOCOM*, 2014.

[20] V. Namboodiri and L. Gao, "Energy-aware tag anticollision protocols for rfid systems," *IEEE Transactions on Mobile Computing*, vol. 9, no. 1, pp. 44–59, 2010.

[21] J. Myung and W. Lee, "Adaptive splitting protocols for RFID tag collision arbitration," in *Proc. of ACM Mobihoc*. ACM, 2006.

[22] M. Kaneko, W. Hu, K. Hayashi, and H. Sakai, "Compressed sensing-based tag identification protocol for a passive rfid system," *IEEE Communications Letters*, vol. 18, no. 11, pp. 2023–2026, 2014.

[23] M. Mayer, N. Grtz, and J. Kaitovic, "Rfid tag acquisition via compressed sensing," in *IEEE RFID Technology and Applications Conference (RFID-TA)*, 2014.

[24] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk, "Efficient and reliable low-power backscatter networks," in *Proc. of the ACM SIGCOMM*, 2012.

[25] H. Yue, C. Zhang, M. Pan, Y. Fang, and S. Chen, "Unknown-target information collection in sensor-enabled rfid systems," *IEEE/ACM Transactions on Networking*, vol. 22, no. 4, pp. 1164–1175, 2014.

[26] *EPC Radio-Frequency Identity Protocols Class-1 Gen-2 for UHF Communication at 860MHz-960MHz, EPCglobal*, http://www.epcglobalinc.org/standards/uhfc1g2, Apr 2011.

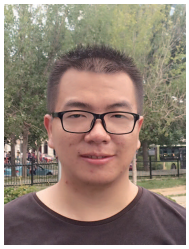
[27] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large rfid system," in *Proc. of ACM MOBIHOC*, 2010.

[28] 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 Transactions on Computers*, vol. 64, no. 1, pp. 87–96, 2015.

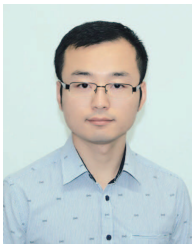
[29] M. Chen, W. Luo, Z. Mo, and S. Chen, "An efficient tag search protocol in large-scale rfid systems with noisy channel," *Proc. of IEEE INFOCOM*, 2013.

[30] X. Liu, H. Qi, K. Li, I. Stojmenovic, A. X. Liu, Y. Shen, W. Qu, and W. Xue, "Sampling Bloom Filter-Based Detection of Unknown RFID Tags," *IEEE Transactions on Communications*, vol. 63, no. 4, pp. 1432–1442, 2015.

[31] S. Lovett and E. Porat, "A lower bound for dynamic approximate membership data structures," in *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2010.

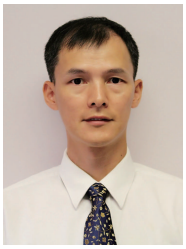



**Xin Xie** received the B.Sc. B.E degree in computer science from Dalian University of Technology, Dalian, China, in 2013. He is currently pursuing the Ph.D in Computer application technology at Dalian University of Technology. His research interests includes RFID technologies and wireless networks.



**Xiulong Liu** received the B.E. degree from the School of Software Technology, Dalian University of Technology, China, in 2010; and the Ph.D. degree from the School of Computer Science and Technology, Dalian University of Technology, China, in 2016. He was a visiting scholar with the Department of Computer and Information Sciences, Temple University, USA, in 2015; and a Postdoctoral Fellow with the School of Computer Science and Engineering, the University of Aizu, Japan, 2016. Currently, he is a Postdoctoral Fellow with the Department of Computing, The Hong Kong Polytechnic University. His research interests include wireless sensing, ubiquitous computing, internet of things, etc.

**Keqiu Li** received the bachelor's and master's degrees from the Department of Applied Mathematics at the Dalian University of Technology in 1994 and 1997, respectively. He received the Ph.D. degree from the Graduate School of Information Science, Japan Advanced Institute of Science and Technology in 2005. He also has two-year postdoctoral experience in the University of Tokyo, Japan. He is currently a professor in the School of Computer Science and Technology, Dalian University of Technology, China. He has published more than 100 technical papers, such as IEEE TPDS, ACM TOIT, and ACM TOMCCAP. He is an Associate Editor of IEEE TPDS and IEEE TC. He is a senior member of IEEE. His research interests include internet technology, data center networks, cloud computing and wireless networks.



**Bin Xiao** received the B.Sc. and M.Sc. degrees in electronics engineering from Fudan University, China, in 1997 and 2000, respectively, and the Ph.D. degree in computer science from the University of Texas at Dallas in 2003. After his Ph.D. graduation, he joined the Hong Kong Polytechnic University as an assistant professor. Currently, he is an associate professor in the Department of Computing at The Hong Kong Polytechnic University, Hong Kong. His research interests include mobile cloud computing, data management, network security, wireless sensor networks, and RFID systems. He is an associate editor for the International Journal of Parallel, Emergent and Distributed Systems. Dr. Xiao is a recipient of the Best Paper Award from IEEE/IFIP EUC 2011.

**Heng Qi** is an associate professor at the School of Computer Science and Technology, Dalian University of Technology, China. He received bachelor's degree from Hunan University in 2004 and master's degree from Dalian University of Technology in 2006. Then he received his Ph.D. degree from Dalian University of Technology in 2012. His research interests include computer network, wireless network and multimedia computing.