Revisiting Cardinality Estimation in COTS RFID Systems

Xingyu Chen*  
State Key Laboratory for Novel Software Technology  
Nanjing University, China

Jia Liu*†  
State Key Laboratory for Novel Software Technology  
Nanjing University, China

He Huang  
School of Computer Science and Technology  
Soochow University, China

Yu-E Sun  
School of Rail Transportation  
Soochow University, China

Xu Zhang  
State Key Laboratory for Novel Software Technology  
Nanjing University, China

Lijun Chen  
State Key Laboratory for Novel Software Technology  
Nanjing University, China

ABSTRACT
With 30 billion RFID tags sold worldwide in 2021, a common basic functionality needed by RFID-enabled applications is cardinality estimation — to quickly estimate the number of distinct tags in an RFID system. Although many advanced solutions have been proposed over the past decade, they suffer from one major limitation in practical use: they need to either modify the existing RFID standard or obtain MAC-layer information, both of which however cannot be supported by commercial off-the-shelf (COTS) devices. In this paper, we revisit the counting problem and propose a novel counting scheme called average time duration based counter (ATD) that quickly estimates the number of distinct tags in a standards-compliant manner. Compared with existing work, the competitive advantage of ATD is that it can be directly deployed on a COTS RFID system, with no need for any hardware modifications. In ATD, we found a new and measurable indicator — the time duration between two adjacent singleton slots, which depends on the number of tags. Following this observation, we derive the theoretical relationship between the time indicator and the number of tags and then give the proof of the estimation as well as its parameter settings. Additionally, we propose a flag-flipping solution to address the overlapping problem in the multi-reader case. We implement ATD in a COTS RFID system with 1000 tags. Experimental results show that ATD is 4.2× faster than the baseline of tag inventory; the performance gain will be further increased in a larger RFID system.

CCS CONCEPTS
• Networks → Mobile networks; Network monitoring.

KEYWORDS
Algorithms, RFID, cardinality estimation, tag counting

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 INTRODUCTION
Radio frequency identification (RFID) is becoming ubiquitous in a wide range of applications, ranging from inventory control [4, 12, 19, 33] and warehouse management [5, 7, 20, 24] to object tracking [18, 21, 25, 28, 31] and behavior recognition [11, 27, 30]. After an RFID tag is attached to an object, an RFID reader can communicate with the tag and make the tagged object identifiable and traceable for item-level intelligence. With almost 30 billion tags sold worldwide in 2021 [16], a common basic functionality needed by RFID-enabled applications is cardinality estimation (aka tag counting) — to estimate the number of distinct tags under the
reader’s coverage, which underlies RFID communication protocols and applications [3, 13, 23]. For example, the number of tags determines the \( Q \) parameter specified by the worldwide RFID standard, which is the basis of tag inventory [1]; RFID-based people traffic counters are one of typical applications, which employ RFID readers to detect tags worn by individuals and gather data on pedestrian traffic in diverse environments like retail stores, malls, transportation hubs, exhibition halls, and public spaces [10]; a warehouse staff may need to perform a quick estimation of the number of tagged products left in stock for warehouse management [23]. Often in these scenarios, it is desirable to quickly estimate the number of tags rather than individually query each tag, which helps significantly reduce the processing time.

Given the practical importance of tag counting, a number of counting schemes have been designed to address this problem over the past decade, such as probabilistic framed ALOHA [14, 15], lottery frame protocol [22], average run based estimation [23], and two phase estimation [3]. In spite of these advancements, existing solutions suffer from one major limitation in practical use: they need to either modify the existing widely used RFID protocol or obtain MAC-layer information, both of which however cannot be supported by commercial off-the-shelf (COTS) devices. Namely, existing solutions cannot be deployed on COTS RFID systems. An intuitive solution is to do tag inventory over all tags, one by one. This operation is foolproof but extremely time-consuming, especially in a large-scale RFID system.

In this paper, we revisit the counting problem and propose a standards-compliant scheme called average time duration based counter (ATD) that quickly estimates the number of distinct tags within a required accuracy. Compared with existing work, the biggest difference of ATD is that it can be directly deployed on a COTS RFID system, with no need for any MAC-layer information or hardware modifications. In ATD, we found a new indicator — the time duration between two adjacent singleton slots, which can reflect the number of tags as well as can be measured by COTS devices directly. We derive the theoretical relationship between the time indicator and the number of tags, and extend the counter from single reader case to multi-reader case for accommodating a large and dense RFID system.

We found a new indicator — the time duration between two adjacent singleton slots, which can reflect the number of tags as well as can be measured by COTS devices directly. We derive the theoretical relationship between the time indicator and the number of tags, and extend the counter from single reader case to multi-reader case for accommodating a large and dense RFID system.

We implement ATD through commercial RFID devices. Extensive simulations and experiments show that the time efficiency of ATD is comparable to that of the state-of-the-art which however cannot be deployed on COTS RFID systems. In addition, ATD improves the time efficiency by about 4.2× for counting 1000 tags, compared with the baseline of tag inventory.

The rest of our paper is organized as follows. Section 2 formulates the counting problem. Section 3 presents the basic idea of our estimator. Section 4 details the average time duration based counter. Section 5 evaluates the algorithm. Section 6 introduces the related work. Finally, Section 7 concludes this work.

## 2 PROBLEM FORMULATION

In this work, we study the problem of cardinality estimation (i.e., tag counting) that aims to estimate the number of distinct tags under the reader’s coverage. It generally falls into two categories: single-set counting and multi-set counting. In single-set counting, a reader covers a certain physical area and the task is to count the number of tags within this area. Formally, let \( S \) denote the set of tags, where \( n = |S| \). The goal of single-set counting is to produce an estimate of \( \hat{n} \) for \( n \), which meets \( Pr(|\hat{n} - n| \leq \alpha n) \geq 1 - \beta \), where \( \alpha \) and \( \beta \) reflect the estimation quality and are determined by end users. We refer to \( \alpha \) as the relative error of \( \hat{n} \) and \( \hat{n} \) as an \((\alpha, \beta)\) estimate of \( n \), respectively. In multi-set counting, a mobile reader sequentially visits \( k \) locations or multiple readers located at \( k \) locations cover different inventory zones. At the location \( i \), the tag set covered by the reader is referred to as \( S_i \), where \( n_i = |S_i| \).
The goal of multi-set counting is to produce an \((\alpha, \beta)\)-approximation \(\hat{n}\) for \(n\), where \(n = |S_1 \cup S_2 \cup S_3 \ldots \cup S_k|\). Usually \(n \leq n_1 + n_2 + \ldots + n_k\) since different \(S_k\) might have a non-empty intersection. In this paper, we will implement both single-set counting and multi-set counting in COTS RFID systems, with no need for any MAC-layer information or hardware modifications. Note that a real-world RFID system might suffer from the missing-reading problem caused by environmental factors such as multi-path effects. In this situation, even if we attempt to query each tag individually, we cannot obtain the exact number of all tags due to misreading. Therefore, in this work, we focus on only the tags that can be successfully activated by a reader. Many advanced solutions have been proposed to address the missing-reading problem, including antenna planning and beamforming, which is out of the scope of this work.

3 BASIC IDEA

A typical RFID system consists of an RFID reader and thousands of tags. The communication between the reader and the tags follows C1G2 protocol [1], which is the global UHF RFID communication standard used by COTS RFID devices. To avoid signal collision caused by simultaneous data transmission, C1G2 divides a time frame into multiple time slots (short time bins) and each tag randomly picks a slot to reply. According to the tags’ choices, there are three types of slots: empty slot with no tag, singleton slot with exactly one tag, and collision slot with two or more tags. The tags in singleton slots can be successfully queried by the reader without collision. In collision slots, since multiple tags communicate with the reader concurrently, the reader fails to decode the tag’s information, which results in tag collision. After all tags in the singleton slots are queried, the reader will issue another slotted frame and the collision tags will repeat the above operations until all unread tags are collected. This process is called tag inventory.

As shown in Fig. 1, there are three 8-slot frames and 6 tags randomly pick a slot to reply to the reader. In the first frame, the two tags \(t_1\) and \(t_5\) in singleton slots can be successfully queried by the reader without collision. The left tags collide with others and will move to the next slotted frame. In the second frame, the two tags \(t_1\) and \(t_5\) keep silent and other tags repeat the above inventory process. In this example, the two tags \(t_3\) and \(t_6\) can be successfully queried and only \(t_2\) and \(t_4\) are left unread. They will be interrogated by the reader in the following third frame. Clearly, tag inventory can query (count) all tags in turn, which is an intuitive and standards-complaint solution to the counting problem. However, doing tag inventory over all tags is time-consuming especially in a large-scale RFID system.

Instead of doing tag inventory, another feasible solution is Mark and Recapture (MR) method. MR needs two inventory rounds to obtain two tag subsets, which is denoted as \(s_1\) and \(s_2\). Since the number of tags collected in the first inventory round and the second inventory round should be proportional to the number \(n\) of tags, the estimated tag number \(\hat{n}\) can be expressed as follows:

\[
\hat{n} = \frac{|s_1| |s_2|}{|s_1 \cup s_2|},
\]

where \(|\cdot|\) refers to the number of tags in the given set. This simple method functions but has some limitations. In practice, we do not know the actual number of tags in advance. If \(|s_1|\) and \(|s_2|\) are too small, the intersection of these two sampled subsets are empty, which cannot be used to estimate the number of tags. Large subsets however greatly increase the communication overhead. The above dilemma motivates us to explore a more efficient counting algorithm.

In this work, we use the probability model to improve the time efficiency of the estimator. Given the number of time slots, the numbers of singleton slots, empty slots, and collision slots are dependent on the number of tags in the system. With the ratios of the singleton slots, empty slots, and collision slots to the frame sizes, many advanced solutions have been proposed to do tag counting. Even though these methods are very accurate and efficient in theory, there is still a large gap between the probability model and practical use: the MAC-layer information cannot be obtained by COTS devices, e.g., the frame size, the number of empty slots, and the number of collision slots.

To address this problem, we find that the time interval between two adjacent singleton slots is a potential indicator to estimate the number of tags. If a reader can successfully read a tag, it means that we are executing the singleton slot. The time interval between two adjacent tag replies is equivalent to that between two adjacent singleton slots. If we label empty slots, singleton slots, collision slots as ‘0’, ‘1’, and ‘2’, respectively, we
can obtain a character string that represents a slotted frame. Taking the 10-slot frame in Fig. 1 for example, we have:

\[ 0 | 1 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 2 \]. \hspace{1cm} (2)

Given the character string, we can know that the reader obtains tag replies in the 1st slot and the 6th slot. Suppose that the time duration of an empty slot, a singleton slot, and a collision slot is \( t_0 \), \( t_1 \), and \( t_2 \), respectively. The time duration of a singleton slot is \( t_1 \), the time duration of a collision slot is \( t_2 \). Since there are two ‘0’s and two ‘2’s between the two singleton slots, the time interval between the two singleton slots is \( T = 2t_0 + 2t_2 + t_1 \). As the number of tags varies, the ratios of ‘0’s, ‘1’s, and ‘2’s to the frame size will vary accordingly, which indirectly affect the value of \( T \). As shown in Fig. 2, given different frame sizes, the time duration \( T \) varies as the number of tags increases, which well indicates that we can estimate the number of tags by observing the value of \( T \). Compared with existing work, the competitive advantage of using this time interval \( T \) is that it can be easily obtained by all COTS RFID devices only through measuring the time interval between two adjacent tag replies, without the need for any MAC-layer information or hardware modification.

4 AVERAGE TIME DURATION ESTIMATION (ATD)

In this section, we detail our estimator ATD. For ease of presentation, we first introduce the single-set version of ATD by default and then extend it to the multi-set case in Section 4.4. Table 1 lists the key notations.

4.1 Estimator

In our work, we extract the estimator based on multiple time intervals of tag replies from a group of tags, which consists of multiple inventory rounds. For each round of estimation, the reader starts a slot frame and terminates it after collecting \( m+1 \) (\( m \geq 1 \)) tag replies. Considering the \( i \)th tag collection (\( 1 \leq i \leq m \)), we define its timestamp \( t_i \) as the time when it replies to the reader and a measure of the time interval \( T_i \) can be expressed as \( t_{i+1} - t_i \). We need to build the theoretical relationship between \( T_i \) and the number \( n \) of tags, which will be given in Theorem 4.1. The average time duration can be approximated by \( \bar{T} = \frac{\sum_{i=1}^{m} T_i}{m} \). The mean value is used as the expected value of the time interval between adjacent singletons to estimate the number of tags.

The expectation and variance of \( T \) are shown in Theorem 4.1. The expectation will be used to estimate the number of tags and the variance will be used to compute the optimal value of \( m \) that meets the required counting accuracy.

**Theorem 4.1.** Let \( m \) be the number of measured time intervals, the terms \( q_0, q_1, q_2 \) \((q_0 + q_1 + q_2 = 1)\) be the probabilities that a slot is empty, singleton, and collision, respectively, and \( t_1, t_2, t_3 \) be the corresponding time duration of the three types of the slots. The expectation \( E(T) \) and the variance \( Var(T) \) of \( T \) are given as follows:

\[
E(T) = \frac{t_0q_0 + t_2q_2}{q_1} + t_1, \quad (3)
\]

\[
Var(T) = \left( \frac{t_2 - t_0}{q_1(1 - q_1)} + \frac{q_0 t_0 + q_2 t_2}{(1 - q_1)q_1^2} \right)^2. \quad (4)
\]

**Proof.** Suppose a variable \( X \) and \( X = k \) means that the \( k \)th slot is the next singleton slot. Since a probability of a slot to be singleton is \( q_1 \), the probability of

<table>
<thead>
<tr>
<th>Table 1: Key Notations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Symbols</td>
</tr>
<tr>
<td>( n )</td>
</tr>
<tr>
<td>( \hat{n} )</td>
</tr>
<tr>
<td>( T )</td>
</tr>
<tr>
<td>( \tilde{T} )</td>
</tr>
<tr>
<td>( \alpha )</td>
</tr>
<tr>
<td>( \beta )</td>
</tr>
<tr>
<td>( f )</td>
</tr>
<tr>
<td>( f_{opt} )</td>
</tr>
<tr>
<td>( t_0 )</td>
</tr>
<tr>
<td>( t_1 )</td>
</tr>
<tr>
<td>( t_2 )</td>
</tr>
<tr>
<td>( m )</td>
</tr>
<tr>
<td>( E(\cdot) )</td>
</tr>
<tr>
<td>( Var(\cdot) )</td>
</tr>
</tbody>
</table>
the same slot not to be a singleton is $p_1 = 1 - q_1$, the probability that $X = k$ is:

$$Pr(X = k) = (1 - q_1)^{k-1}q_1.$$  

The expectation of $X$ can be represented as:

$$E(X) = \sum_{k=1}^{+\infty} k(1 - q_1)^{k-1}q_1 = q_1 \sum_{k=1}^{+\infty} (p_1^k)' = q_1 \left( \sum_{k=1}^{+\infty} p_1^k \right)'$$

$$= q_1 \left( \sum_{k=0}^{+\infty} p_1^k \right)' = q_1 \left( \frac{1}{1 - p_1} \right)' = \frac{1}{q_1}.$$  

(6)

Next, we answer how to derive the variance of the variable $T$, which will be used to calculate the accuracy of our estimator later. Suppose we know there are $k$ slots between two neighbouring singleton slots. The bit string of these slots consists of $k - 1$ continuous non-singleton slots ('0' or '1' in Eq. (2) with an end of singleton '1'. Since there is only one singleton slot, it does not change the variance of the execution time. Clearly, $Var(T)$ is determined by the time variation of the first $k - 1$ non-singleton slots. Since $Var(T)$ can be calculated by the equation $Var(T) = E(T^2) - E(T)^2$, the problem boils down to deriving $E(T^2)$ in what follows. Consider one non-singleton slot. Since the slot is not singleton, we define the probability that the slot is empty as $p_0 = \frac{1}{1 - q_1}$, the probability that the slot is collision as $p_2 = 1 - p_0 = \frac{q_1}{1 - q_1}$, and the number of empty slots between two singleton slots is $a_0$. Clearly, the number of empty slots $a_0$ follows a typical binomial distribution $B(p_0, k - 1)$.

The expectation and the variance of $a_0$ can be written:

$$E(a_0) = (k - 1)p_0, \quad Var(a_0) = (k - 1)p_0p_2.$$  

(8)

The execution time of the $k - 1$ non-singleton slots is a function of $a_0$, which can be written as follows:

$$T_k = a_0 t_0 + (k - 1 - a_0)t_2$$

$$= (t_0 - t_2)a_0 + (k - 1)t_2.$$  

(10)

Since the coefficient of the variable $a_0$ in Eq. (10) is $t_0 - t_2$, the expectation and variance of the variable $T_k$ can be represented as:

$$E(T_k) = (t_0 - t_2)E(a_0) + (k - 1)t_2$$

$$= (t_0 - t_2)(k - 1)p_0 + (k - 1)t_2,$$  

(11)

$$Var(T_k) = (t_0 - t_2)Var(a_0) = (k - 1)(t_0 - t_2)p_0p_2.$$  

(12)

With $E(T_k)$ and $Var(T_k)$, we can derive $E(T_k^2)$ by the following equation:

$$E(T_k^2) = Var(T_k) + E(T_k)^2$$

$$= (k - 1)(t_0 - t_2)p_0p_2 + (k - 1)^2[(t_0 - t_2)p_0 + t_2]^2.$$  

(13)

Considering all possible value of $k$ and summing the corresponding $E(T_k^2)$ together, we have

$$E(T^2) = \sum_{k=1}^{+\infty} E(T_k^2)Pr(X = k)$$

$$= (t_0 - t_2)p_0p_2 \sum_{k=1}^{+\infty} (k - 1)q_1(1 - q_1)^{k-1}$$

$$+ [(t_0 - t_2)p_0 + t_2]^2 \sum_{k=1}^{+\infty} (k - 1)^2q_1(1 - q_1)^{k-1}$$

$$= (t_0 - t_2)p_0p_2f_1 + [(t_0 - t_2)p_0 + t_2]^2f_2.$$  

(14)

The term $f_1$ in Eq. (14) can be simplified by:

$$f_1 = \sum_{k=1}^{+\infty} (k - 1)q_1p_1^{k-1} = p_1 \sum_{i=1}^{+\infty} iq_1p_1^{i-1}$$

$$= p_1q_1 \sum_{k=1}^{+\infty} (p_1^k)' = p_1q_1 \left( \sum_{k=0}^{+\infty} p_1^k \right)'$$

$$= p_1q_1 \left( \sum_{k=0}^{+\infty} p_1^k \right)' = \frac{1}{1 - p_1}.$$  

(15)

Similarly, the term $f_2$ in Eq. (14) can be simplified by:

$$f_2 = \sum_{k=1}^{+\infty} (k - 1)^2q_1(1 - q_1)^{k-1} = p_1 \sum_{i=1}^{+\infty} i^2q_1(1 - q_1)^{i-1}$$

$$= p_1q_1 \sum_{i=1}^{+\infty} (ip_1^i)' = p_1q_1 \sum_{i=1}^{+\infty} [(i + 1)p_1^i - p_1^i]'$$

$$= p_1q_1 \sum_{i=1}^{+\infty} (p_1^{i+1})' - \sum_{i=1}^{+\infty} (p_1^i)'$$

$$= p_1q_1 \sum_{i=0}^{+\infty} (p_1^i)' - \sum_{i=0}^{+\infty} (p_1^i)'$$

$$= p_1q_1[(\frac{1}{1 - p_1})' - (\frac{1}{1 - p_1})']$$

$$= (1 - q_1)(2 - q_1)\frac{1}{q_1^2}.$$  

(16)
By substituting Eq. (15), Eq. (16) into Eq. (14), we have \( E(T^2) \) as follows:
\[
E(T^2) = \frac{(t_2 - t_0)^2 q_0 q_2}{q_1(1 - q_1)} + \frac{(q_0 t_0 + q_2 t_2)^2 (2 - q_1)}{(1 - q_1) q_1^2}.
\]  
(17)

With \( E(T^2) \), we can finally obtain \( Var(T) \) by the following equation:
\[
Var(T) = E(T^2) - (E(T) - t_1)^2
\]
\[
= \frac{(t_2 - t_0)^2 q_0 q_2}{q_1(1 - q_1)} + \frac{(q_0 t_0 + q_2 t_2)^2 (2 - q_1)}{(1 - q_1) q_1^2}
- \frac{t_0 t_0 + q_2 t_2}{q_1}.
\]
(18)

Note that \( Var(T) \) is the variance of the execution time of the first \( k - 1 \) non-singleton slots and it does not include the last singleton slot, this is why we need to subtract \( E(T) \) by \( t_1 \) in Eq. 18. This completes the proof.

Then we introduce how we obtain \((q_0, q_1, q_2)\) based on \( f \) and \( n \), which is shown in Theorem 4.2.

**Theorem 4.2.** Let \( n \) be the actual number of tags, \( f \) be the frame size, and \((q_0, q_1, q_2)\) be the probability that a slot in a frame is empty \((q_0)\), singleton \((q_1)\), and collision \((q_2)\), respectively. We have:
\[
q_i = \begin{cases} 
(1 - \frac{1}{f})^n, & i = 0 \\
\frac{n}{2}(1 - \frac{1}{f})^{n-1}, & i = 1 \\
1 - q_0 - q_1, & i = 2
\end{cases}
\]
(19)

**Proof.** The probability that a tag chooses a given slot in a frame is \( \frac{1}{f} \). The probability that it does not choose that slot is \( 1 - \frac{1}{f} \). The probability that \( n \) tags do not pick a slot is \( (1 - \frac{1}{f})^n \), which is actually \( q_0 \). The probability that exactly one tag chooses a slot is \( \binom{n}{1}(1 - \frac{1}{f})^{n-1} \times \frac{1}{f} \), which is the value of \( q_1 \). If a slot is neither an empty slot nor a singleton slot, it must be a collision slot. Hence, the probability of the collision slot is \( 1 - q_0 - q_1 \), which is the value of \( q_2 \).

By substituting Eq. (19) into Eq. (3), we can obtain the theoretical relationship between \( E(T) \) and \( n \), which is written as follows:
\[
E(T) = h(n) = \frac{t_0 q_0 + t_2 q_2}{q_1} + t_1
\]
\[
= \frac{t_0 (1 - \frac{1}{f})^n + t_2 [1 - (1 - \frac{1}{f})^n - \frac{n}{2}(1 - \frac{1}{f})^{n-1}]}{\frac{n}{2}(1 - \frac{1}{f})^{n-1}} + t_1.
\]
(20)

As we can see, given the frame length \( f \), \( E(T) \) is a function of \( n \). By computing the inverse function of \( h(\cdot) \), we can represent \( n \) as a function of \( T \) and use the average time durations of \( m \) measurements to compute the corresponding \( n \) as desired. However, it is not easy to find the inverse function as \( h(\cdot) \) is non-linear. Instead, we use the least square-based method to derive \( \hat{n} \).
\[
\min_n \sum_{i=1}^m ||T_i - h(\hat{n})||^2.
\]
(21)

The best fit happens when \( n \) minimizes the sum of the offsets of the measured data points from the theoretical value estimated by Eq. (20). Since the estimated value is usually not equal to the true number of tags in the system, we denote it as \( \hat{n} \) to make a difference.

### 4.2 Estimation Quality

As aforementioned, \( m \) measurements of time intervals are required to reduce the variation of \( \hat{n} \) and meet the requirement \( Pr(|\hat{n} - n| \leq \alpha n) \geq 1 - \beta \). In the following, we analyze the minimum value of \( m \). Let \( Y = \frac{\hat{n} - n}{\sigma_n} \). Since \( n \) is usually a large number, according to the law of large number, \( Y \) follows the standard normal distribution. Thus, we have:
\[
Pr\{\frac{-\alpha n}{\sigma_n} \leq Y \leq \frac{\alpha n}{\sigma_n}\} \geq 1 - \beta,
\]
(22)

which is equivalent to
\[
\frac{\alpha n}{\sigma_n} \geq c,
\]
(23)

where \( c \) meets the following condition:
\[
1 - \beta = erf\left(\frac{c}{\sqrt{2}}\right),
\]
(24)

where \( erf(\cdot) \) represents the Gaussian error function. Since \( \sigma_n \) is a function of \( m \), we first compute \( \sigma_n \). According to Eq. (20), \( n \) is a function of \( T \) and therefore the relationship between \( \sigma_n \) and \( Var(T) \) can be explicitly expressed. Suppose \( \hat{n} = f(T) \) is expressed with the Taylor expansion centered on \( E[T] \):
\[
f(T) - f(E(T)) \approx f'(E(T))(T - E(T)).
\]
(25)

Since \( f(E(T)) = E(f(T)) \), we have:
\[
Var[f(T)] \approx f'(E(T))^2 Var(T).
\]
(26)

We represent Eq. (26) as follows:
\[
\sigma_n^2 = \frac{c^2 Var(T)}{m},
\]
(27)
where

\[ \epsilon = \frac{dn}{dT} \bigg|_{E(T)} = \frac{1}{dT/dn} \bigg|_{E(T)} \]
\[ = \frac{n^2}{nf(1 - \frac{1}{f})^{n-1}} \log(1 - \frac{1}{f}) - \frac{t_2((f(1 - \frac{1}{f})^{1-n}) - (t_0 + 1)(f - 1))}{f}. \] (28)

By substituting Eq. (26) into Eq. (23), we have:

\[ m \geq \frac{c^2 \epsilon^2 \text{Var}(T)}{\alpha^2 \mu^2}. \] (29)

In Eq. (29), it is worth noting that the number of tags \( n \) should be known a priori to predict the number \( m \) of time intervals. We propose an estimator \( \hat{n} \) to quickly determine the rough estimate of \( n \). We notice that when the frame length \( f \) approaches to the number \( n \) of tags, the reader can collect more tags within a given time window, which can be used to do the estimation. Take Fig. 3 as an example. We vary the frame length \( f \) from 2\(^0\) to 2\(^5\) and plot the corresponding reading rates measured in an RFID system with 80 tags. As we can see, the reading rate increases first and then almost remains stable after peaking at the maximum. The maximum just occurs when \( f = 64 \) that is around 80. We prove this observation with Theorem 4.3.

**Theorem 4.3.** Let \( n \) be the actual number of tags and \( f \) be the frame size. The portion \( p_1 \) of singleton slots in the frame reaches maximum when \( f \approx n \).

**Proof.** As aforementioned, the probability \( p_1 \) that a time slot is singleton is:

\[ p_1 = \frac{n}{f} (1 - \frac{1}{f})^{n-1}. \] (30)

By taking the derivation of \( p_1 \) with respect to \( f \), we could obtain

\[ \frac{dp_1}{df} = n \left( \frac{1}{f^2} \right) \left( 1 - \frac{1}{f} \right)^{n-2} \frac{n - f}{f}, \] (31)

We could obtain the optimal frame \( f \) that maximizes \( p_1 \), i.e., \( p_1 = (1 - \frac{1}{f})^{n-1} \approx \frac{1}{e} \) and \( f \approx n \).

Given an RFID reader, since its maximal reading rate \( r_{max} \) is fixed, we can obtain \( r_{max} \) first and then increasingly adjust \( f \) to a proper value that makes the reading rate close to \( r_{max} \). Namely, the probability \( p_1 \) of a singleton slot approaches to the maximum. In this case, we can approximately treat the value of \( f \) as a rough estimator of \( n \).

![Figure 3: The relationship between Q and reading rate.](image)

### 4.3 Parameter Settings

So far, we have introduced the estimator \( E(T) \) and the number \( m \) of time intervals that guarantee the estimation quality. These two results are jointly determined by the parameters of the frame size \( f \) and slot durations \( (t_0, t_1, t_2) \). Next, we introduce how we determine these parameters.

#### 4.3.1 Optimal Frame Size \( f_{opt} \)

The estimating time of ATD depends on two factors: the number \( m \) of rounds and the time cost \( E(T) \) in each round, where the time cost is measured by the number of slots. As mentioned above, the number \( m \) of rounds is dependent on the frame size \( f \). If an \( f \) can minimize the estimating time, we treat it as the the optimal frame size \( f_{opt} \). According to the RFID standard C1G2, an RFID reader leverages a non-negative integer \( Q (0 \leq Q \leq 15) \) to control the frame size. Once \( Q \) is specified before an inventory round, the corresponding number of slots in the frame will be set to \( 2^Q \). Thus, the problem is to find the \( Q \) value that can minimize the estimation time, which can be formulated as follows:

\[ \min_{0 \leq Q \leq 15} mE(T), \] (32)

where the product of \( m \) and \( E(T) \) refers to the expectation of the sum of \( m \) waiting times. Since \( Q \) only has 16 candidate values, we can iteratively try each of the candidate \( Q \) values and select the one that minimizes the overhead. Note that this process is executed virtually by simulations, which will not increase the communication overhead.

#### 4.3.2 Measurement of \( t_0 \), \( t_1 \), and \( t_2 \)

Given a slotted frame, the time durations of each slot are not always the same, which is actually determined by the slot type. For example, in an empty slot, the reader has nothing received from tags and will quickly moves to the next slot. In a singleton slot, a tag will transmit the required
data (e.g., tag ID) to the reader, which consumes higher time overhead than the empty slot and the collision slot. In a collision slot, the reader just needs to send a small number of bits (RN16) to detect the collision. Considering the above facts, the time durations of different types of slots should be determined.

Our basic idea is to find the optimal setting by optimizing the difference between the measured data in practice and expectation value in theory. More specifically, we set up an RFID system with the tag number and frame size \( f \) known in advance. After that, we utilize them to estimate \( E(T) \) and then use the least squares method to derive \( t_0, t_1, \) and \( t_2 \):

\[
\min_{t_0, t_1, t_2} \sum_{i=1}^{m} \|T_i - E(T)\|^2, \tag{33}
\]

where \( T_i \) is the \( i \)th waiting time measured in practice. The best fit happens when \( t_0, t_1, \) and \( t_2 \) jointly minimize the sum of the offsets of the measured data points from the theoretical value. Alg. 1 sketches the estimation algorithm of ATD.

### 4.4 Multi-set Case

So far, we have introduced ATD in the single-set case. With the proliferation of RFID-enabled applications, a mobile reader or multiple readers might be required to cover larger surveillance zone for interrogating more tags. In this subsection, we detail how to make ATD tailored to the multi-set case. The key challenge in the multi-set case is that the tags in the overlapped zone will be counted multiple times by a mobile reader or by different readers. If we simply treat each inventory zone as a single-set case and sum up all estimated numbers of tags, the final estimated number will be larger than the actual one. For example, in Fig. 4, three RFID tags are under the interrogation zones of the reader \( V_1 \) and the reader \( V_2 \). If these two readers count tags individually, these three tags will be counted twice, leading to estimation errors. To address this problem, our basic idea is to use a binary flag to silence counted tags. We set the tags to the state ‘0’ if it has been counted and the state ‘1’ otherwise. By this means, the tags in the overlapped zone will not be repetitively queried by different readers.

We now introduce how to implement flag-based silence within the scope of C1G2 protocol. C1G2 protocol provides us with a command called Select that is able to select a particular tag subset to reply. According to C1G2, a Select command can be written as follows:

\[
S(\#_1, a, b, p, l, k). \tag{34}
\]

As we can see, there are six fields in a select command. The first two fields determine the function of a select command and the last four fields determine which tag will be affected. The details are given below.

- **MemBank, Mask, Length, Pointer.** These four fields jointly determine which tags are matching. MemBank specifies which memory bank is chosen for comparison. As aforementioned, four memory banks are available, MemBank-0, MemBank-1, MemBank-2, and MemBank-3, which are indicated by 0, 1, 2, and 3, respectively. Pointer indicates the starting position in the chosen memory bank. Length determines the length of Mask, which is a customized bit string according to the user demands. If Mask is the same as the string that begins at Pointer and ends Length bits later in the memory of MemBank, then the corresponding tag is matched.

- **Target, Action.** The field Target indicates the object that Select will operate, which is either a tag’s SL flag or an inventoried flag in any one of four sessions. The sessions are specified by the C1G2 protocol to fit the case of exclusive reading among multiple readers. In our work, we use the SL flag, so this field is set to 4. The

---

**Algorithm 1: Average Time Duration Estimation**

**Input:** Reliability \( \alpha \), confidence interval \( \beta \)

**Output:** An estimate \( \hat{n} \) of the number of tags

1. Obtain a rough estimation \( \hat{n} \) of \( n \);
2. Compute the optimal frame size \( f_{\text{opt}} \);
3. Compute the number \( m \) of time intervals with \( \hat{n} \) and \( f_{\text{opt}} \);
4. Issue a \( f_{\text{opt}} \)-slot frame;
5. Record the first \( m + 1 \) reply timestamps \( \{t_1, t_2, \ldots, t_{m+1}\} \);
6. Terminate the slotted frame;
7. for \( i = 1 \) to \( m \) do
8. \[ T_i = t_{i+1} - t_i; \]
9. \[ \hat{n} = \arg \min_{\hat{n}} \sum_{i=1}^{m} \|T_i - h(\hat{n})\|^2; \]
selection function is actually achieved by masking the interested tags, setting the matching tags’ inventoried flags or SL flag to a specific state while not-matching tags to opposite, and finally operating the tags with the same flag state. How to set the inventoried flag and the SL flag is determined by the Action field. As shown in Table 2, eight actions are available, where matching and not-matching tags set their SL flag to $A$ or $B$. By combining Target and Action, the reader is able to modify the state of the inventoried flags or the SL flag for a group of tags. For example, when the Action is 0, the matching tags are set to $A$ (SL) while the not-matching tags are set to $B$ ($\sim$SL). The term “do nothing” means the tags keep their flags unchanged.

We assume that the tag set is $\tau$ and $\tau' \subseteq \tau$ is subsets of tags that are covered by other readers. The solution is to set the target tags $\tau'$ to SL while other $\tau - \tau'$ are set to $\sim$SL. In the inventory stage, we let only the tags with SL participate in the response. By this means, if the tag in $\tau - \tau'$ has been queried by one of the readers, it will keep silent to other readers. Specifically, we assume $v_1$ is the first reader that counts the number of tags in the overlapped zone. After the reader finishes its inventory process, the reader broadcasts the select command:

$$\mathrm{⊤} t_1 \leftarrow AB : S(t = 4, 4, 1, 32, 0, \text{null}),$$

(35)

where the Target field being set to 4 ($t = 4$) means that the operating object of the select is the SL flag. The mask filed is set to null so all tags are matching tags and their SL flags will be set to $\sim$SL. Afterwards, the reader can use a query command to acknowledge tags with one of the two states to reply. Since their SL flags have been set to $\sim$SL while other tags are still in SL state, they can be distinguished by the reader and will not be queried again. Let us take Fig. 4 as an example. With the three tags under the two readers’ inventory zone, the number $n$ of tags can be estimated by the following three steps. First, the two readers set tags within their coverage zone to SL state, respectively. Second, the first reader $v_1$ starts a frame and estimates the number $n_1$ of tags with flag SL. After that, $v_1$ sets tags within its coverage zone to $\sim$SL. Third, the second reader $v_2$ estimates the number $n_2$ of tags with flag SL. Since the tags within the overlapped zone are set to SL, they will not participate in $v_2$’s estimation and the final tag estimate is $n_1 + n_2 - 3$, which is equal to $n$ in theory. Similarly, the solution can be easily generalized to RFID systems with more than two readers. The total number of tags can be derived from $\sum_{i=1}^{m} n_i$, where $n_i$ is the number of tags estimated by the $i$th reader.

### 5 EVALUATION

#### 5.1 Implementation

We implement ATD in a COTS RFID system that consists of an RFID reader, up to four reader antennas, and 2000 RFID tags. As shown in Fig. 5, the reader model is ALR-F800 [2] from Alien technology that is one of the industry’s most experienced RFID suppliers. The reader is connected to a directional antenna Lariad S9028PCR [17] that is with 8.5 dBi gain and operates at around 920 MHz for communicating with tags. We use a total of 2,000 COTS tags (AZ-9634 [2]) in our experiments. These tags are densely attached to 5 cartons, each of which contains 400 tags. The software development of the system is Java. The host computer is a laptop with an Intel Core i5-8250U 1.8GHz CPU and 8GB RAM. All results are the average outcome of 100 trials in a single reader (antenna) case by default. Multi-set cases will be discussed in Section 5.4.

The standards-compliant implementation of ATD involves two industrial standards: EPCglobal Gen2 air interface protocol (C1G2) [1] and Low Level Reader Protocol (LLRP). As aforementioned, the former specifies physical interactions and logical operating procedures between the readers and tags, and LLRP defines an interface between RFID readers and a client (laptop in our experiments). A client computer connects one or more readers via either a wired or a wireless link. By enabling

![Figure 5: System setup.](image-url)
LLRP, the client is able to manipulate a reader to issue C1G2 commands as we need. There are two kinds of operations: reader operation (RO) and access operation (AO). RO defines the parameters for tag inventory, including Select and Query. AO specifies parameters for data access (except for EPC) on a tag. Since ATD does not need to read data from a tag’s memory, this study just focuses on RO. An RO comprises a list of AISpecs, each of which is used for antenna setting. If multiple AISpecs are set up, the order of AISpec execution follows the order in which they appear.

As aforementioned, ATD needs a prior acquaintance with the frame length to do tag counting. According to C1G2, the frame length $f$ is determined by a non-negative integer $Q$, i.e., $f = 2^Q$, where $0 \leq Q \leq 15$. The RFID reader is required to dynamically increase or decrease the value of $Q$ for accommodating different tag cardinalities. However, we tested six models of the most widely used RFID readers from three experienced suppliers, i.e., ALR-F800 and 9900+ from Alien Inc [2], R220 and R420 from Impinj [9], Mercury6 and M6e from ThingMagic [26], and unexpectedly found that these commercial RFID readers have the ability to change $Q$ under the hood as specified by the C1G2 standard but they do not expose this level details to users. Namely, we cannot set and obtain $Q$ directly from the commercial RFID readers as desired, which becomes a key hurdle that knows the frame length and limits the applicability of ATD in COTS RFID systems. In spite of this constraint, we found that the device (e.g., ALR-F800 and 9900+) provides us with an alternative: an interface to set an upper limit $Q_{\text{max}}$ of $Q$. With this ability, we can dynamically adjust $Q_{\text{max}}$ to obtain a proper value of $Q$ by observing the sampling rate of tags. We have the following basic idea: Because $Q_{\text{max}}$ is the upper limit of $Q$, the self-adjustment of $Q$ must not be greater than $Q_{\text{max}}$. Given many tags, small $Q$ will generate many tag collisions and experience a rise trend towards $Q_{\text{max}}$. If the number of tags is larger than $2^{Q_{\text{max}}}$, $Q$ will increase gradually and be equal to $Q_{\text{max}}$ finally, in which we have $Q = Q_{\text{max}}$. The setting of $Q_{\text{max}}$ depends on the sampling rate. More specifically, as $Q_{\text{max}}$ increases, the sampling rate experiences a rise trend first. After peaking at a maximum (when $2^{Q_{\text{max}}}$ is close to the number of tags), the sampling rate decreases. In this way, we can easily find a value of $Q_{\text{max}}$ making $2^{Q_{\text{max}}}$ roughly less than the number of tags. Since the self-adjustment of $Q$ (under the hood) is not greater than $Q_{\text{max}}$, $Q$ is highly likely to be set to $Q_{\text{max}}$ in this case. Therefore, we can obtain $Q$ for tag counting in COTS systems.

5.2 Estimation Accuracy

In Section 4.2, we have proved that our method ATD can guarantee any required accuracy with $m$ measurements of the average time durations in theory. In this subsection, we study the accuracy of ATD and compare our work with mark and recapture method (MR). Since the accuracy of an estimator is determined by the confidence interval $\alpha$ and the error probability $\beta$, we use two different settings (i.e., $\{\alpha = 0.1, \beta = 0.1\}$ and $\{\alpha = 0.15, \beta = 0.15\}$) and vary the number $n$ of tags from 500 to 1000 (about the maximal number of tags covered by a single antenna) at a step of 100. Each plot is the average outcome of 100 independent tests. As shown in Fig. 6, ATD can bound the estimation accuracy with the given the estimation quality ($\alpha, \beta$). For example, when $\alpha = 0.1$ and $\beta = 0.1$, the mean estimation errors are 0.092, 0.086, 0.093, 0.095, 0.088, and 0.081 for counting 500, 600, 700, 800, 900, and 1000 tags respectively. In other words, the mean errors are bounded within $\alpha = 0.1$. When the quality ($\alpha, \beta$) relaxes to (0.15, 0.15), the mean errors are larger but still smaller than $\alpha = 0.15$. The difference is that ATD uses less communication overhead to meet the later estimation quality. Another important observation is that estimation errors of our method almost remain stable once the quality ($\alpha, \beta$) is given, which well indicates the stability and scalability of our method.

Taking a closer look at accuracy, we plot the CDF of estimation errors, which is shown in Fig. 7. As we can
see, 80% of the estimation errors are below 0.122 when the quality \((\alpha, \beta)\) is set to \((0.1, 0.1)\), and 80% of estimation errors are less than 0.148 when the quality \((\alpha, \beta)\) is set to \((0.15, 0.15)\). The actual probability is smaller than the given \(\beta\) due to two reasons. First, we found that the tag inventory by commercial readers is not very stable. Given \(n\) tags, the reader cannot get \(n\) tag replies all the time. In other words, some tags will not be collected at a random and small probability. This will affect the measure of the time duration of two adjacent singleton slots, and lower the accuracy of our method. Second, the estimation errors of durations of three different slots will give rise to some errors. In spite of the uncertain of tag inventory and duration errors, we assert that ATD can do tag counting in a relatively accurate way once estimation quality is given.

After that, we compare the estimation accuracy of our method with MR. In the experiment, we fix the execution time and compare the corresponding estimation errors, where \((\alpha, \beta)\) is set to \((0.1, 0.1)\) and the number \(n\) of tags ranges from 500 to 1000 at a step of 100. As we can see in Fig. 8, ATD is much better than MR when estimating different number of tags. For example, when the number of tags is 1000, the estimation error of ATD is 8.1% while that of MR is as large as 44.6%. What is worse, the estimation error of MR further increases with number of tags. In other words, MR cannot function well in a large RFID system with tens of thousands of tags. In contrast, ATD does not have this problem: the estimation errors almost keep stable, regardless of the number of tags to be counted.

### 5.3 Time Efficiency

In this section, we investigate the time efficiency of our method. Since none of existing work can be deployed on COTS RFID systems, we first compare ATD with existing advanced solutions (UPE [14], FNEB [6], ZOE[34], and SRC[3]) through simulations. Amongst the four work, UPE is the first RFID counting protocol and uses the number of empty and collision slots in the frame for estimation, FNEB uses the indices of the first non-empty slots for estimation, ZOE lets all tags randomly choose whether or not to reply in a slot and leverages the probability of that slot to be empty for estimation, SRC[3]) summarizes the advantage of the above works and designs a two-stage counting protocol for better results. To guarantee the comparison is fair, we use the same parameter settings as [3]: a slot in UPE takes 0.8 ms, a slot in all other protocols takes 0.4 ms, and for all protocols a trail incurs an extra overhead of 1 ms. Besides, it is worth noting that all the above methods leverage different methods to measure the rough estimation for parameter optimization, which also increases the overhead. Since all of them cannot be implemented by existing COTS RFID devices, we assume all methods derive the rough estimation by the scheme proposed in this paper (Theorem. 4.3) and do not consider this part of time in our simulations. The overall time overhead including the time used to obtain the rough estimator will be validated through experiments in practice, which will be shown shortly later in Fig. 12.

In Fig. 9, we fix the estimation quality of \((\alpha, \beta)\) to the value \((0.05, 0.2)\) and vary the number \(n\) of tags from 1000 to 100,000. As we can see, ATD and SRC have the similar execution time, which is much better than the other three methods. UPE can achieve good estimation accuracy only when the number of tags is small, which is not suitable for large-scale RFID systems. The performances of FNEB and ZOE are worse than ATD, because these two methods need to use \(m\) tails to do an estimation. In contrast, ATD measures multiple time durations in one time frame (trail) and thus the overhead is much smaller. Additionally, the performance of our method remains stable, regardless of the number of tags. For example, the execution times of ADT varies from 0.358 s to 0.442 s when the number of tags increases from 1000 to 10,000. This good stability can help users quickly count a large RFID system with tens of thousands of tags.
is worth noting that ATD experiences a slightly periodic oscillation because of the setting of the frame length $f$ in commercial RFID readers. As aforementioned, the estimation time achieves the minimum when $f = n$ holds. However, the frame length can only be set exponentially with 16 candidate values ranging from $2^0$ to $2^{15}$. As the number of tags increases from $2^i$ to $2^i + 1$ ($0 \leq i \leq 15$), the running time will increase first and then see a slight decline until $n$ reaches $2^i + 1$. However, this fluctuation is small and can be neglected in practical use.

In Fig. 10, we fix $\alpha$ to 0.05 and the number $n$ of tags to 10,000, and compare the time efficiency of different methods with respect to $\beta$. As we can see, ATD performs the best under different $\beta$, UPE follows, FENB and ZOE still consume longer time for the same estimation task. Additionally, we find that the execution time of these three methods sees a sharp decline (especially for ATD and FNEB) as $\beta$ increases from 4% to 20%. This is because large $\beta$ relaxes the accuracy, which requires smaller number of rounds for ATD to achieve the tag counting task. In Fig. 11, we fix $\beta$ to 0.2 and the number $n$ of tags to 10,000, and compare the time efficiency of different methods with respect to $\alpha$. Similarly, ATD generally is superior to the other three methods over different $\beta$. Additionally, we find that the execution time of these four methods declines with $\alpha$. This is because large $\alpha$ relaxes the accuracy, which requires smaller number of rounds for ATD to achieve the tag counting task. Note that SRC is not taken into account in the experiments because it leverages a look-up table to obtain its optimal parameters, which does not contain different values of $\alpha$ and $\beta$ used in our experiments.

Above simulations well indicate that the time efficiency of our approach ATD is comparable to the state-of-the-art. However, unlike existing work, our method can be directly deployed on a COTS RFID system with no need for any hardware modifications, which is where ATD shines. We next study the time efficiency of ATD in real RFID systems. We take tag inventory (exclusive collection — querying each tag and obtaining its ID individually) as the benchmark and compare the time overhead of these two cardinality estimation methods. In the experiment, we fix the estimation quality of $(\alpha, \beta)$ to $(0.1, 0.1)$ and increase the number of tags from 500 to 1000 with a step of 100. As shown in Fig. 12, ATD is much more efficient than exclusive collection over different number of tags. For example, exclusive collection requires 8.31 s to query 1000 tags and ATD reduces this time to 1.98 s, which improves the time efficiency by a factor of 4.2×. The performance gain is attributed to ATD’s ability to estimate the number of tags accurately with only 30–50 tag responses in this case, much smaller than exclusive collection. Besides, the performance gap will be further widened because the execution time of exclusive collection is proportional to the number of tags while that of ATD almost remains stable. For example, if there are 10,000 tags, the performance gain will be roughly more than 40×, which is a great improvement. We assert that in a passive RFID system, a reader (antenna) can cover about 1000 tags at most. However, if it is an active or semi-passive RFID system, the number of tags under a reader’s coverage will be much larger.

5.4 Multi-set Case
In these experiments, we evaluate ATD in multi-set scenarios, in terms of the accuracy and the time efficiency. As discussed in Section 4.4, the main departure from the single reader (antenna) case is that there are tags in the overlapped zone. If no action is taken, these tags will be counted multiple times by different readers. In the evaluation, we fix the estimation quality of $(\alpha, \beta)$ to $(0.1, 0.1)$ and vary the number of readers from 1 to 4, each of which covers about 500 tags. In Fig. 13, we study the estimation accuracy of ATD under different number of reader antennas. As we can see, ATD bounds the estimation accuracy with the given the estimation
quality \((\alpha = 0.1, \beta = 0.1)\), regardless of the number of antennas. Thanks to the selective masking, the tags in the overlapped zone are counted by only once, which makes the complicated multi-set case boil down to a single reader case. In Fig. 14, we further study the time efficiency of ATD in multi-set case. As shown in this figure, the execution time experiences a linear rise with the number of antennas because each reader antenna takes a fixed execution time to perform an estimation (once a certain estimation quality is given) regardless of the number of tags. Besides, in our experiments, we find that the additional select command that is used to filter out overlapped tags consumes only about 20 ms which is negligible. The above experimental results indicate that ATD has the potential to be easily deployed in a dense or large RFID system. It is worth noting that existing multi-set methods assume that multiple readers can work concurrently, which is equivalent to a single-set case. However, this assumption does not hold in a practical scenario due to reader-to-reader collision.

6 RELATED WORK
Cardinality estimation is to estimate the number of distinct tags under the reader’s interrogation zone, which underlies RFID protocols and applications, having received considerable attention from the research community over the past decade. The first tag estimation scheme, called Unified Probabilistic Estimator (UPE), was proposed by Kodiallam and Nandagopal in 2006 [14]. UPE uses the slotted framed Aloha protocol and makes estimation based on either the number of empty slots or that of collision slots in a frame. The zero-based estimator (EZB) [15] takes the average number of zeros in the frame as the vehicle to do the estimation. The lottery frame estimation scheme (LOF) employs the geometric distribution hash functions to itemize all tags in order. H. Han et al. propose the first non-empty-based estimator (FENB) [6], which leverage the size of the first run of 0s in the frame. The average run based tag estimation (ART) uses the average run size of 1s to estimate the tag cardinality [23]. Zheng et al. propose another efficient estimate scheme zero-one estimator (ZOE), which only needs \(O(\log \log n)\) time slots to perform an estimation [34]. Chen et al. summarize prior work and design a two phase RFID counting scheme, which is more time-efficient and accurate than existing schemes [3].

In spite of these advancements, all these solutions are implemented through simulations rather than real-world experiments. In recent years, some work attempts to deploy tag counting algorithm in real system. Huo et al. attempt to implement the traditional probabilistic method through USRP [8]. They use the average number of tags reply in a slot as the feature and build a prototype to derive it. However, the interrogation range of USRP is less than 1 m and the cost of USRP is high, which is not a good choice for practical use (especially in large RFID systems). The most related work is Tash [32] and single-slot count [29]. Tash designs a C1G2 compatible hash function that is able to be used to estimate the cardinality of tags. However, this hash function is not tailored to tag counting and it needs to write the some information into each tag’s memory in turn, which actually takes more communication overhead than the basic tag inventory. The single-slot count queries all tags in singleton slots, which is actually the basic tag inventory. Hence, in this paper, we aim to design a novel tag counter that can estimate the tag number in COTS RFID system, with no need for any MAC-layer information or hardware modification.

7 CONCLUSION AND LIMITATION
We revisit the problem of cardinality estimation in RFID systems and propose a novel counter that can quickly estimate the number of distinct tags in a standards-compliant manner. The key technical novelty is that we find a new indicator — the time duration between two adjacent singleton slots, which can reflect the number of tags as well as can be measured by COTS devices. We derive the theoretical relationship between the time indicator and the number of tags, and we implement the counter in commercial RFID devices without any hardware modification. Extensive experiments show that our method is far superior to the baseline of tag inventory.

Limitations: First, we have shown that our estimator ATD is near-optimal compared with existing advanced work but we fail to derive the variance of ATD in theory. Second, due to limited hardware resources, we set up a medium-sized RFID system (1000 tags), which is not large enough to fully reflect the performance of our system. In future work, we would implement a larger RFID system, e.g. with over 10 thousand tags (including passive tags as well as semi-passive tags or active tags), to further study the performance improvements.

ACKNOWLEDGMENTS
We would like to thank anonymous reviewers and shepherd for their insightful comments on this paper. This research is financially supported by the National Natural Science Foundation of China (No. 62072231), Fundamental Research Funds for the Central Universities, and Collaborative Innovation Center of Novel Software Technology and Industrialization.
REFERENCES