# Counting RFID Tags Efficiently and Anonymously

Hao Han[†§], Bo Sheng[‡], Chiu C. Tan[†], Qun Li[†], Weizhen Mao[†], Sanglu Lu[§]
[†]College of William and Mary, Williamsburg, VA, USA
[‡]Northeastern University, Boston, MA, USA
[§]State Key Laboratory of Novel Software Technology, Nanjing University, China
Email: [†]{hhan, cct, liqun, wm}@cs.wm.edu, [‡]shengbo@ccs.neu.edu, [§]sanglu@nju.edu.cn

***Abstract*—Radio Frequency IDentification (RFID) technology has attracted much attention due to its variety of applications, e.g., inventory control and object tracking. One important problem in RFID systems is how to quickly estimate the number of distinct tags without reading each tag individually. This problem plays a crucial role in many real-time monitoring and privacy-preserving applications. In this paper, we present an efficient and anonymous scheme for tag population estimation. This scheme leverages the position of the first reply from a group of tags in a frame. Results from mathematical analysis and extensive simulation demonstrate that our scheme outperforms other protocols proposed in the previous work.**

## I. Introduction

Radio Frequency IDentification (RFID) technology is widely used in monitoring applications such as inventory control and object tracking [1]–[7]. Small RFID tags, each with a unique ID, are attached to items under monitoring. An RFID reader can remotely collect these IDs later for verification. Due to the large number of deployed RFID tags, collecting all tag IDs for verification is inefficient. Some real-time applications, such as counting the number of tags in a shipping portal, need more efficient techniques to manage tag data. In this paper, we consider the problem of *efficiently* and *anonymously* estimating the cardinality of a large set of RFID tags with a desired accuracy.

Efficient techniques for estimating the number of RFID tags are important for applications when the time window for collecting tag data is small. These applications include real-time monitoring or managing a large quantity of products. For example, a warehouse operator may need to perform a quick estimation of the number of products left in stock. Such applications demand efficient estimating schemes instead of the slow and unnecessary process of reading every tag ID.

Anonymity is another important issue when dealing with RFID tags attached to uniquely identifiable items such as passports [8] or driver's licenses [9]. Either broadcasting tag IDs in the open, or revealing IDs to the RFID reader may leak personal information. For instance, an adversary could capture the communication between the reader and tags or compromise the reader to track users' activities. Identifying each tag ID increases individual security and privacy risks. An alternative way of providing anonymity is to use cryptographic protocols to mask the actual ID [10], [11]. However, the cryptographic techniques require additional modification to the tag hardware, as well as increase the computational complexity on both tags and readers.

Prior work in [12] and [13] considers this problem by using probabilistic estimation based on the framed-slotted ALOHA model. Unfortunately, the scanning time can be considerably long due to the large frame size required. The performance becomes worse when the mobile tags appear dynamically so that counting them at a fixed time instant is not possible. That is because the tags have to be scanned independently with each counting consuming a long time.

In this paper, we propose a novel scheme for the reader to quickly estimate the number of distinct tags within a required accuracy. Our scheme is based on a new distinct element counting method [14], without reading either the actual or pseudo IDs. The main idea of our algorithm is to utilize the position of the first reply from a group of tags in a frame to infer the number of tags. Theoretical analysis and extensive simulation show that our scheme outperforms earlier RFID tag estimation schemes. Moreover, our scheme tries to optimize incremental counting in a mobile environment. Note that our approach has a general purpose of counting RFID tags. Combined with other commands, it can be flexibly adopted in various applications.

Our contributions are summarized as follows.

- We propose a novel anonymous estimating scheme which does not collect the ID from each RFID tag, but is still able to estimate the number of tags accurately.
- We present estimators for both static and dynamic sets of tags. The static set specifies a snapshot of a set of tags, and the dynamic set considers that tags can join or leave the set with time. Both our estimators are more efficient than the existing protocols, even when the cardinality of the tag set varies across many orders of magnitude.
- We propose a novel send-and-reply protocol among the reader and tags to improve performance.

The rest of our paper is as follows. Section II contains the related work. Section III presents our problem definition and system model. Section IV outlines the main idea of our schemes. Section V details the algorithms. Our schemes are evaluated in Section VI, and Section VII concludes.

## II. Related Work

For a reader to successfully identify every tag in proximity, collision arbitration protocols must be considered so that replies from multiple tags will not be garbled due to collision. Collision arbitration protocols are divided into two approaches: ALOHA-based [15]–[17] and tree-based [18]–[20]. In the first

approach, the framed-slotted ALOHA (FSA) protocol, which is an extension of the pure ALOHA protocol [21], is widely used in RFID standards. Built on that, adaptive FSA protocols, where frame size is adaptively adjusted, are explored in [15], [22]–[24].

Recent research work [12], [13], [17], [25] is the closest to this paper. A probabilistic analytical model for anonymously estimating tag population is first proposed in [12]. The main idea is to use the framed-slotted ALOHA protocol and monitor the number of empty and collision slots to count tags. However, the drawbacks of the estimators in [12] are that all the tags must be readable by the reader in a single probe and that the reader must know approximately the magnitude of the number of tags to be estimated. Due to these constraints, an Enhanced Zero-Based (EZB) estimator is presented in [13]. By tuning the parameters for multiple iterations, the number of tags can be estimated with high accuracy, even when the tag population varies a lot. The key improvement in our work over [12] and [13] is that our scheme does not scan the entire frame, which drastically reduces time cost. Finally, another novel estimator for the same problem is proposed in [25] with more focus on the multiple-reader scenario. However, the scheme requires a special geometric distribution hash function, which might not be available in the off-the-shelf RFID systems.

## III. Problem Definition and System Model

### A. Problem Definition

TABLE I
NOTATIONS

| Symbols | Descriptions |
|---|---|
| $\epsilon$ | Confidence interval |
| $\delta$ | Error probability |
| $t$ | Number of distinct tags |
| $t_{max}$ | Upper bound of the number of tags |
| $\tilde{t}$ | Estimation of the number of tags |
| $X$ | Random variable for the number of continuous empty slots before the first non-empty slot in a frame |
| $f$ | Frame size (the number of slots in a frame) |
| $R$ | Random seed |
| $\rho$ | Load factor $t/f$ |
| $k$ | Number of waiting slots |
| $n$ | Number of rounds (frames) |
| $h(\cdot)$ | Hash function |
| $T(\cdot)$ | Theoretical time cost (in number of slots) in a round |
| $m$ | Number of sets of tags |

Given an RFID reader and a set of tags, we want to quickly and accurately estimate the number of distinct RFID tags in the set without identifying each tag individually. Our algorithms allow a user to specify his desired accuracy using two variables, a *confidence interval* $\epsilon$ and an *error probability* $\delta$. Lower values of $\epsilon$ and $\delta$ result in a more accurate estimation. Our algorithms return an estimation $\tilde{t}$ of the actual number of tags $t$, such that $Pr[|\tilde{t} - t| \leq \epsilon t] \geq 1 - \delta$. For example, if the set has 5000 RFID tags, and given $\epsilon = 5\%$ and $\delta = 1\%$, the desired estimator should output the number within [4750, 5250] with probability greater than 99%. Table I summarizes the notations used.

### B. System Model

The MAC protocol for our RFID system is based on the adaptive *framed-slotted ALOHA* model. To read a set of tags, a reader first powers up and transmits continuous wave (CW) to energize tags. Each tag waits for the reader's command before replying. This is known as the *Reader Talks First* mode.

The communication between the reader and tags is composed of multiple frames. Each frame is partitioned into slots. Here, we refer to an individual frame as a round. The reader will first broadcast a *begin round* command containing the frame size $f$ in the forthcoming round, and a random seed $R$. The frame size is the number of slots available for tags to choose in a round. Each tag picks a slot, and this slot determines when a tag will reply. An RFID tag uses a hash function $h(\cdot)$, $f$, $R$, and its ID to pick a slot in the current round, i.e., $h(f, R, id) \rightarrow [0, f-1]$. We assume that the outputs of the hash function have a uniform random distribution such that the tag has the equal probability to select any slot within the round given a seed and ID.

Each RFID tag has a slot counter which will decrease each time the reader indicates that the current slot has ended. The tag will only reply when its slot counter reaches zero. When all the slots in the frame have been accounted for, the reader sends an *end round* command to terminate this round. We assume that the reader can issue an end round command to terminate a round at any time without waiting for the frame to end. The procedure is illustrated in Fig. 1. We call this the original *send-and-reply* protocol.

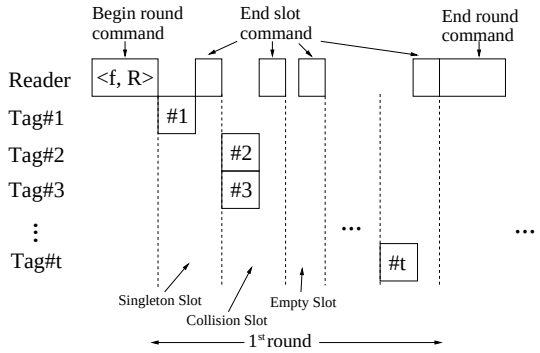


Fig. 1. Collection sequence of passive RFID systems using the adaptive FSA

Since every RFID tag chooses its own slot individually, there will be instances where no tag picks a particular slot. We term this as an *empty slot*. A slot that has only been chosen by one tag is known as a *singleton slot*. A slot that is chosen by more than one tag is called a *collision slot*. We refer to both singleton slot and collision slot as *non-empty slot* in this paper. After collecting all replies, the reader can generate a bitstring, such as

$$\{ \cdots \mid 1 \mid 0 \mid 1 \mid 1 \mid 0 \mid 1 \mid \cdots \},$$

where 0 indicates an empty slot, and 1 represents a non-empty slot.

## IV. Intuition

The previous research [12], [13], [17] takes advantage of the framed-slotted ALOHA protocol to estimate the number of tags. The basic idea is based on the probability model we have described previously. The reader scans all the slots and records the status of each slot: empty, singleton, or collision. By examining the number of empty slots, collision slots, the reader can then estimate the number of tags.

This estimation method while powerful, has some limitations. The main limitation is the large frame size, which translates to a long protocol running time, when there exist a huge amount of tags. Suppose for a large tag population but the frame size is considerably small. All the tags' responses will be packed in a small number of slots, which means that the number of empty slots will become zero and number of collision slots will be equal to the frame size. To make estimation accurate, the frame size should be in proportion to the number of tags. Therefore, scanning the whole frame is inefficient when tag population is large. Furthermore, the performance is even worse in mobile environment, where either RFID tag or reader can move. To count tags over a period of time, we have to use a *very large* frame size at the beginning, such that we can superimpose all the frames and guarantee that the number of empty slots is not zero in the end [13].

To overcome the large frame size problem in the previous protocols, we propose a new idea based on a randomized algorithm for counting. Suppose we have $n$ random numbers uniformly and randomly chosen from (0,1). By examining the smallest number, say $x$, we can estimate $n$. Intuitively, the smaller $x$ is, the larger $n$ would be. If all the numbers are uniformly laid out, $n$ should be approximated by $1/x$. Of course, this estimation is very crude with a very large variance. Fortunately, we can run the same process for a sufficiently large number of times, the estimation will become more accurate. More details will be described later.

Our scheme does not require the reader to scan the whole frame. Instead, the reader only needs to identify the first non-empty slot, and uses the number of consecutive empty slots before that to estimate the number of tags. Again, the fewer the empty slots appear before the first non-empty slot, the more tags there are. In practice, certain number of iterations of such operations are performed, and the mean value is used to achieve an accurate estimation. For example, given,

$$\begin{array}{ll} \{\ 0 \mid 0 \mid 1\ \} & \rightarrow X_1 = 2 \\ \{\ 1\ \} & \rightarrow X_2 = 0 \\ \{\ 0 \mid 0 \mid 0 \mid 0 \mid 0 \mid 0 \mid 0 \mid 0 \mid 1\ \} & \rightarrow X_3 = 9 \end{array}$$

where $X_i$ denotes the number of empty slots before the first reply position in round $i$. From theoretical analysis and extensive simulation, we find even though multiple iterations are required for accuracy, the total time is still much shorter than the schemes in prior work.

## V. Anonymous Estimating Algorithms

In this section, we describe our novel RFID tag estimating scheme, First Non-Empty slots Based (FNEB) estimator.

### A. Basic Algorithm

Again, our algorithm is based on the idea of making observation on the first non-empty slot. However, if the number of tags $t$ is small, the position of the first reply may be located at the end of the frame. Apparently, it is not efficient to use the original *send-and-reply* protocol described in Section III. In that protocol, a reader broadcasts $f$ and $R$ at the beginning of a round, and waits for the first reply from tags. Therefore, when the first reply is toward the end of the round, the reader has to wait for the period of time almost equal to the frame size.

To resolve this issue to improve the query efficiency, we propose a new *send-and-reply* communication protocol among reader and tags. Compared to the original protocol, our new protocol can identify the first non-empty slot in $O(log_2 f)$ time slots instead of $O(f)$.

The new *send-and-reply* protocols for reader and tags are shown in Algorithm 1 and 2 respectively. In the protocols, the reader sends an extra frame range $r$ to all tags. Initially, the reader splits the whole frame into two, and sets the first half frame as the *candidate range*, the second half frame as the *alternative range*. The reader always sends out its candidate range to the tags. Each tag evaluates $h(f, R, id)$ and replies immediately if the result is inside the range $r$. Otherwise, it keeps silent without doing anything. Then the reader checks the forthcoming slot. If the slot is empty, which indicates there is no tag within the candidate range, the reader splits the alternative range into two and picks the first half as the new candidate range, and the second half as the new alternative range. If the slot is not empty, which indicates there is at least one tag in the candidate range, the reader then splits the candidate range into two, and sets the first half as the the new candidate range, and the second half as the new alternative range. The above procedure is like a binary search tree as shown in Fig. 2. The reader keeps traversing from the root to the leaves and records the path in each iteration. Finally, the reader can identify the first non-empty slot using the equation in line 16 of Algorithm 1, where $z_i$ is a $0/1$ bit indicating the state of the $i^{th}$ iteration.

Fig. 2 illustrates a simple example with frame size of 16. In the first iteration, the reader sends the frame size $16$, search range $[0, 7]$, and a random seed to all tags. No tag replies, so the first slot is empty. Then the reader starts the second iteration with a new range $r = [8, 11]$. At this time, at least one tag replies, so the slot is "1". Repeating the same process twice, the reader identifies the first non-empty slot to be $10$.

It is not difficult to find that if the number of tags is relatively small to the frame size, our new *send-and-reply* protocol is more efficient than the original protocol. Otherwise, the original protocol is better. Therefore, we combine both of them to determine $X$. In the combined *send-and-reply* protocol, we define the number of waiting slots $k$. At every round, the original protocol is tried first. Only when there is no reply within $k$ slots, we turn to use our new protocol. So in the worst case, only $k + \log_2 f$ slots are required.

**Algorithm 1** New *send-and-reply* protocol for the reader

**if** $f$ is not a power of 2 **then**
  $f = 2^{\lceil \log_2 f \rceil}$
**end if**
$a = 0$, $b = f/2 - 1$
Set the search range $r = [a, b]$ and random seed $R$
**for** $i = 1$ to $\log_2 f$ **do**
  Reader broadcasts $r$, $f$, and $R$, and listens in the forthcoming slot for reply (only one slot)
  **if** the slot is EMPTY **then**
    $z_i = 0$
    $a = b + 1$, $b = b + |r|/2$, and updates $r$
  **else**
    $z_i = 1$
    $b = (b - 1)/2$, and updates $r$
  **end if**
**end for**
Return $X = \sum_{i=1}^{\log_2 f} (1 - z_i) \cdot 2^{\log_2 f - i}$

**Algorithm 2** New *send-and-reply* protocol for each tag

Receive range $r$, $f$, and $R$ from reader
Compute slot number $sn = h(f, R, id)$
**if** $sn$ is inside $r$ **then**
  Reply immediately
**else**
  Keep silent
**end if**

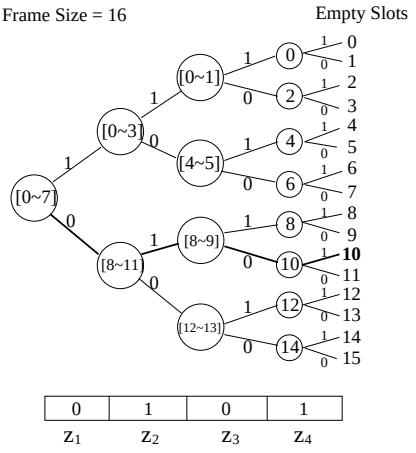


Fig. 2. Illustration of our new send-and-reply protocol

Our combined *send-and-reply* protocol requires a slight modification to existing RFID tags. We add an optional bit mask to indicate the search range $r$ in each *end slot* command sent by the reader. If the parameter is set to a valid range, those tags who pick a response slot inside the range will reply in the forthcoming slot, no matter what value their slot counters are. If the parameter is set to null, the original *send-and-reply* protocol is then used.

With the basic idea described above, the complete algorithm of the FNEB estimator is shown in Algorithm 3. The algorithm takes $t_{max}, \delta$ and $\epsilon$ as inputs, where $t_{max}$ denotes the upper bound of tag population $t$. Initially, the reader computes parameters $f$, $k$, and $n$ by inputs, and then applies the combined send-and-reply protocol $n$ rounds to obtain the average value of $X$, denoted by $Y$. At last, the estimation $\tilde{t}$ is calculated below:

$$\tilde{t} = f \cdot \ln \frac{1+Y}{Y} \tag{1}$$

**Algorithm 3** FNEB estimator for static tag set

INPUT: $t_{max}$, $\delta$, and $\epsilon$
OUTPUT: $\tilde{t}$

1: Compute the frame size $f$ and waiting slots $k$
2: Compute the number of rounds $n$
3: **for** $i = 1$ to $n$ **do**
4:  Generate a new random seed $R_i$
5:  Broadcast $(f, R_i)$ to all tags and wait their replies
6:  Run the original *send-and-reply* protocol
7:  **if** receive reply before $k$th slot **then**
8:  $X_i =$ slot number of first reply - 1
9:  **else**
10:  Run the new *send-and-reply* protocol
11:  $X_i =$ value returned by Algorithm 1
12:  **end if**
13: **end for**
14: Add all $X_i$ and get the average $Y = \sum_{i=1}^{n} X_i/n$
15: Return $\tilde{t} = f \ln \frac{1+Y}{Y}$

In the next two subsections, we will explain why this algorithm can achieve the desired accurate estimation and how to compute parameters $f$, $k$, and $n$ (lines 1 and 2 in Algorithm 3). To ease understanding, we first present the mathematics behind the algorithm and how to pick parameter $n$. We then describe how to determine $f$ and $k$.

### *B. Pick $n$*

The value of $n$ directly determines the performance of our scheme. If $n$ is too small, the estimated $\tilde{t}$ cannot meet the desired accuracy. However, a large $n$ will increase the estimation time. Next, we first present the theoretical underpinnings for the FNEB algorithm, followed by the bounds for n that can satisfy the accuracy requirement.

Given the frame size $f$, each tag has the probability $\frac{1}{f}$ to select a specific slot in the frame. For $t$ tags in total, the probability of a certain slot to be empty (denoted as $P_0$) is $P_0 = (1 - \frac{1}{f})^t$. Since $f$ is normally large, $P_0$ can be simplified to $P_0 \approx e^{-\rho}$, where $\rho = \frac{t}{f}$. We call $\rho$ the load factor. Let the random variable $X$ be the number of consecutive empty slots before the first non-empty slot in a frame. We then have

$Pr[X = u] = P_0^u(1 - P_0)$. The expectation of $X$ is

$$\begin{aligned} E(X) &= \sum_{u=0}^{f-1} uPr(X = u) = \sum_{u=0}^{f-1} uP_0^u(1 - P_0) \\ &= \frac{(f-1)P_0^{f+1} - fP_0^f + P_0}{1 - P_0} \\ &= \frac{P_0}{1 - P_0}(1 - P_0^f) - fP_0^f. \end{aligned}$$

Since that $0 < P_0 < 1$, then $P_0^f \to 0$ and $fP_0^f \to 0$ when $f$ is large. So $E(X)$ can be further simplified to

$$E(X) \approx \frac{P_0}{1 - P_0} = \frac{1}{e^\rho - 1}. \quad (2)$$

Correspondingly, the variance of $X$ is

$$\begin{aligned} Var(X) &= \sum_{u=0}^{f-1}(u - E(X))^2 Pr(X = u) \\ &\approx \frac{P_0}{(1 - P_0)^2}. \end{aligned} \quad (3)$$

According to the intuitive relation between $E(X)$ and $t$, the observation of $X$ can be used to estimate $t$. However, there exists variance between the observed value of $X$ and $E(X)$. By the law of large number [26], the estimation becomes more accurate when the number of observations gets larger. We define a random process $Y = \sum_{i=1}^{n} \frac{X_i}{n}$ as the mean of $n$ observations, where $X_i$ is the random variable $X$ for the $i^{th}$ observation. Note that $E(X_i) = E(X)$ and $Var(X_i) = Var(X)$. Since the reader gives a different random seed in each broadcast, $X_i$ ($1 \leq i \leq n$) is independent with each other. Therefore, we have

$$E(Y) = \frac{\sum_{i=1}^{n} E(X_i)}{n} = \frac{nE(X)}{n} = E(X)$$

and

$$Var(Y) = \frac{Var(\sum_{i=1}^{n} Xi)}{n^2} = \frac{nVar(X)}{n^2} = \frac{Var(X)}{n}.$$

Since that $E(Y) = E(X)$, by solving Eq. 2 for $t$, we get

$$t = f \cdot ln\frac{1 + E(Y)}{E(Y)}. \quad (4)$$

Then, according to Eq. 1, by substituting $Y$ for $E(Y)$, we have

$$\tilde{t} = f \cdot ln\frac{1 + Y}{Y}.$$

Next, we will show how to use $Var(Y)$ to compute the tight bound of parameter $n$.

**Theorem 1.** *Given $\delta$, $\epsilon$, and $\rho$, if the number of rounds $n$ is not less than $\frac{c^2 e^{-\rho}(e^\rho - e^{-\epsilon\rho})^2}{(1-e^{-\epsilon\rho})^2}$, the algorithm described above can guarantee the accuracy requirement, that is, $Pr[|\tilde{t} - t| \leq \epsilon t] \geq 1 - \delta$.*

*Proof:* We use $\mu$ and $\sigma$ to denote the expectation and standard variance of $Y$, i.e., $\mu = E(Y)$ and $\sigma = \sqrt{Var(Y)} = \sqrt{Var(X)/n}$. By the central limit theorem, we know

$$Z = \frac{Y - \mu}{\sigma}$$

is asymptotically normal with mean 0 and variance 1; that is, $Z$ satisfies the standard normal distribution and its cumulative distribution function is

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{u^2}{2}}\, du.$$

We can find a constant $c$ which makes

$$\begin{aligned} Pr[-c \leqslant Z \leqslant c] &= \Phi(c) - \Phi(-c) \\ &= erf(c/\sqrt{2}) = 1 - \delta, \end{aligned}$$

where $erf$ is the error function [27]. By solving the formulation above, we get the value of $c$. For example, if $\delta = 1\%$, then $c = 2.576$. Thus, the desired accuracy can be rewritten as

$$\begin{aligned} &Pr[|\tilde{t} - t| \leqslant \epsilon t] = Pr[(1 - \epsilon)t \leqslant \tilde{t} \leqslant (1 + \epsilon)t] \\ &= Pr[(1 - \epsilon)t \leqslant f \ln\frac{1 + Y}{Y} \leqslant (1 + \epsilon)t] \\ &= Pr[\frac{e^{-(1+\epsilon)\rho}}{1 - e^{-(1+\epsilon)\rho}} \leqslant Y \leqslant \frac{e^{-(1-\epsilon)\rho}}{1 - e^{-(1-\epsilon)\rho}}]. \end{aligned}$$

Therefore, if we have

$$\frac{\frac{e^{-(1+\epsilon)\rho}}{1-e^{-(1+\epsilon)\rho}} - \mu}{\sigma} \leqslant -c \text{ and } \frac{\frac{e^{-(1-\epsilon)\rho}}{1-e^{-(1-\epsilon)\rho}} - \mu}{\sigma} \geqslant c,$$

then we can guarantee $Pr[|\tilde{t} - t| \leqslant \epsilon t] \geqslant 1 - \delta$. Combining $\sigma$ and Eq. 3 to solve the inequalities, we get

$$n \geqslant \frac{c^2 e^{-\rho}(e^\rho - e^{-\epsilon\rho})^2}{(1 - e^{-\epsilon\rho})^2}.$$

■

In practice, the number of tags $t$ is not known a priori, making it difficult to predict the exact number of rounds. However, the minimum number of rounds $n$ is a monotonically increasing function against the load factor $\rho$; that is, the number of rounds calculated by $t = t_{max}$ is large enough for the actual $t$. Therefore, $n$ in line 2 of Algorithm 3 is computed by

$$n = \frac{c^2 \cdot e^{-t_{max}/f} \cdot (e^{t_{max}/f} - e^{-\epsilon t_{max}/f})^2}{(1 - e^{-\epsilon t_{max}/f})^2}$$

### C. Determine Optimal Parameters $f$ and $k$

The estimating time of our algorithms is affected by two factors: the number of rounds and the time cost in each round. Here, the time cost is measured by the number of slots. From the discussion above, we find that the number of rounds $n$ is dependent on the frame size $f$. The time cost in a round is either $x + 1$* (if the number of empty slots observed in that round is smaller than $k$) or $k + log_2 f$. That relies on both $f$ and $k$. Hence, if we select inappropriate $f$ and $k$, the performance of our scheme will be adversely affected. Our remaining problem is to determine the "best" value for parameter $f$ and $k$ on a given upper bound $t_{max}$.

Remember that the probability of the random variable $X$ equals to $u$ is $P_0^u(1 - P_0)$, where $P_0 = e^{-t/f}$. We use the

*Note that one additional slot is needed for the first non-empty slot

function $T(\cdot)$ to denote the time cost in each round. Given $k$, $t$, and $f$, $T(\cdot)$ can be expressed as

$$\begin{aligned} & T(k,t,f) \\ = & \textstyle\sum_{u=0}^{k-1}(u+1)Pr(X=u)+\sum_{u=k}^{f-1}(k+\log_2 f)Pr(X=u) \\ = & \textstyle\sum_{u=0}^{k-1} P_0^u+\log_2 f\cdot P_0^k \\ = & \frac{1-P_0^k}{1-P_0}+\log_2 f P_0^k, \end{aligned}$$

where the first term describes the cost of using the original *send-and-reply* protocol, if there is a reply within $k$ slots. The second term, indicating the cost of using our new *send-and-reply* protocol, is a constant $k + \log_2 f$. Both of them are multiplied by their probabilities.

Therefore, $n \cdot T(k,t,f)$ is the estimating time of our algorithm for a specified $t$. Our goal is to find parameters $f$ and $k$ to minimize the time cost averaging over all possible values of $t$ from 1 to $t_{max}$. Then, the problem is to minimize

$$\frac{1}{t_{max}} \sum_{t=1}^{t_{max}} n \cdot T(k,t,f)$$

subject to $k, f \in N$, and $0 \leqslant k \leqslant f$, where

$$n = \frac{c^2 e^{-t_{max}/f}(e^{t_{max}/f} - e^{-\epsilon t_{max}/f})^2}{(1-e^{-\epsilon t_{max}/f})^2}$$

$$T(k,t,f) = \frac{1-P_0^k}{1-P_0} + \log_2 f P_0^k.$$

This is a nonlinear programming problem with two unknown integer variables. Although it is difficult to find an expression of $f$ and $k$, the problem is solvable by enumerating all possible parameters to find the optimal values. Given parameters: $t_{max}$, $\epsilon$ and $\delta$, we first fix $f$ and enumerate all values of $k$ from 1 to $f$ to find the best value of $k$ which can minimize the objective function. Then, we repeat the process to search for the optimal $f$. Note that these procedures are all computed by the reader offline.

Table II shows the optimal parameters for some specified $t_{max}$ under that $\epsilon = 5\%$ and $\delta = 1\%$. In the table, $n_{op}$, $f_{op}$, and $k_{op}$ indicate the optimal number of rounds, frame size, and number of waiting slots for each $t_{max}$ respectively. The ratio in the last column is computed by $t_{max}$ over $f_{op}$.

TABLE II
OPTIMAL PARAMETERS FOR DIFFERENT $t_{max}$ WITH $\delta = 0.01, \epsilon = 0.05$

| $t_{max}$ | $n_{op}$ | $f_{op}$ | $k_{op}$ | Ratio ($= t_{max}/f_{op}$) |
|---|---|---|---|---|
| 100 | 3927 | 55 | 6 | 1.818 |
| 500 | 4024 | 264 | 8 | 1.894 |
| 1000 | 4058 | 521 | 9 | 1.919 |
| 5000 | 4014 | 2651 | 12 | 1.886 |
| 10000 | 4024 | 5279 | 13 | 1.894 |
| 50000 | 4042 | 26205 | 15 | 1.908 |

From the table, we have the following observations.

- The ratio of $t_{max}$ to $f_{op}$ is close to 1.9, and $k$ is close to $\log_2 f$. Based on this observation, we can either directly use the quasi-optimal parameters: $f \approx 1.9$ and $k \approx \log_2 f$ for our estimating algorithm without solving the non-linear programming problem, or bound a small search range to exhaustively find the optimal values of $f$ and $k$. Both methods can reduce the computation cost in practice.
- Since the ratio is relatively stable, the optimal number of rounds $n$ will not get obvious increase, when $t_{max}$ becomes large. Therefore, as shown in the evaluation section, our algorithm performs well even if we count a huge amount of tags.

### D. Enhancement: Adjusting Skewed $t_{max}$

In practice, users may overestimate the upper bound $t_{max}$. The actual $t$ may be much smaller than the bound. Thus, the optimal parameters $f$ and $k$ computed by $t_{max}$ may be too large for estimation, since it causes many empty slots before the first reply in each round. We call this the skewed $t_{max}$ problem.

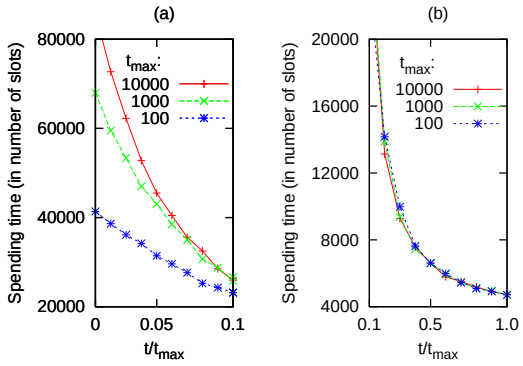


Fig. 3. Time cost versus the normalized number of tags for different $t_{max}$: (a) comparison under $t/t_{max} \leqslant 0.1$; (b) comparison under $t/t_{max} > 0.1$

To show the effect of different $t_{max}$ on the performance of our FNEB estimator, we plot the estimating time in number of slots against $t$ under three different $t_{max}$ (see Fig. 3). To ease comparison, we normalize $t$ by $t_{max}$ and separate the figure into two parts. As we see, when the value of $t/t_{max}$ approaches 1, the time cost decreases significantly. Also, for the same value of $t/t_{max}$, the smaller $t_{max}$ will spend less time, when $t$ is absolutely close to $t_{max}$.

Based on these observations, we propose an enhanced approach to solve the skewed $t_{max}$ problem. As mentioned before, a larger $t_{max}$ usually causes more empty slots. Therefore, we can use the position of first reply to decide whether $t_{max}$ is too large for $t$. If it is, we will adaptively shrink $t_{max}$ in the next round. The main algorithm is shown in Algorithm 4, which should be appended at the end of each iteration (between lines 12 and 13) in Algorithm 3.

Recall that $X$ is the random variable indicating the number of empty slots before the first reply from tags, and $X_i$ is the observed value of $X$ in the $i$th round. Let variable $N$ enumerate all possible numbers of tags, decreasing from $t_{max}$ to 1. Then, $Pr[X = X_i | t = N]$ is the probability of observing $X_i$ empty slots on the condition that $t = N$. According to Bayes' theorem, we have

$$Pr[t = N | X = X_i] = \frac{Pr[X = X_i | t = N]}{Pr[X = X_i]}, \qquad (5)$$

**Algorithm 4** Adaptively shrink skewed $t_{max}$

/* After getting $X_i$, we test whether to shrink $t_{max}$ */

1: $p$ = 0
2: **for** $N = t_{max}$ to 1 **do**
3: $\quad p = p + \frac{Pr[X=X_i|t=N]}{\sum_{i=1}^{t_{max}} Pr[X=X_i|t=i]}$
4: $\quad$ **if** $1 - p < 0.1\%$ **and** $N < t_{max}$ **then**
5: $\quad\quad t_{max} = N$
6: $\quad\quad$ Recompute $f$, $k$, and $n$, and restart new rounds
7: $\quad\quad$ **break**
8: $\quad$ **end if**
9: **end for**

where $Pr[X = X_i] = \sum_{i=1}^{t_{max}} Pr[X = X_i|t = i]$. In the algorithm, Eq. 5 is added to variable $p$ as $N$ decreases in each iteration (line 3). So $p$ presents the probability $Pr[N \leq t \leq t_{max}]$ on condition that $X_i$ empty slots have been observed, and $1 - p$ is the probability $Pr[1 \leq t < N]$ correspondingly. Once $1 - p$ is smaller than a very small probability (like $0.1\%$ in our algorithm), it means that $t$ can not be larger than $N$ with high possibility. Therefore, we can shrink $t_{max}$ to the value of $N$. Recall the analysis in Section V-B, $Pr[X = X_i|t = N]$ can be computed by $(e^{-N/f})^{X_i} \cdot (1 - e^{-N/f})$.

However, when the shrinking occurs in the latter rounds, restarting new rounds may incur a large overhead. Therefore, we constrain the number of rounds for shrinking. If $t_{max}$ remains unchanged in certain consecutive rounds, the current $t_{max}$ is deemed stable enough. We will not run Algorithm 4 after those rounds. In the simulation, we set a heuristic value of 30 rounds which is large enough for adjustment.

TABLE III
RESULTS FROM THE ADAPTIVE SHRINK ALGORITHM FOR SINGLE SET OF RFID TAGS WITH $t_{max} = 10000$, $\delta = 0.01$, AND $\epsilon = 0.05$

| No. of tags | No. of shrinks | Final value of $t_{max}$ | Shrinking overhead | Total time (slots) |
|---|---|---|---|---|
| 10 | 5.6 | 14.4 | 350.7 | 5525.9 |
| 50 | 5.5 | 69.9 | 365.1 | 5738.0 |
| 100 | 5.4 | 135.7 | 418.8 | 5732.4 |
| 500 | 5.2 | 667.6 | 444.9 | 5758.8 |
| 1000 | 4.9 | 1307.3 | 441.3 | 5683.2 |
| 5000 | 1.9 | 6467.5 | 371.3 | 5660.8 |

Table III shows the performance of our enhanced FNEB estimator. From that, we find that the final value of $t_{max}$ can be adjusted close to $t$ within several shrinks. As a result, different numbers of tags can lead to almost the same total time.

### *E. Extension: Estimating Multiple Tag Sets*

Previously, we only considered a static tag set. However, for certain applications, we may need to count multiple tag sets in a dynamic environment where either the tags or reader is mobile. For example, a single reader cannot cover all the tags in a large warehouse. Instead, we have to either deploy multiple readers or dispatch a mobile reader moving through the warehouse to cover all tags. In that case, different tag sets queried by readers at different places could have overlapping tags. If we directly apply our previous algorithms on each tag set, these overlapping tags will be counted multiple times, resulting in erroneous overall estimations.

We have extended our FNEB algorithms to estimate multiple tag sets. Due to page limit, we cannot include the details in this paper. The intuition of the protocol is as follows. Suppose we have $m$ tag sets $S_1, S_2, ..., S_m$, and for each set the number of empty slots before the first non-empty slot is $X_1, X_2, ..., X_m$. In a global view, $min(X_1, X_2, ..., X_m)$ infers the total tag size $|S_1 \cup S_2 \cup ... \cup S_m|$. However, each set $i$ ($i \in [1, m]$) does not know whether $X_i$ is minimal. Therefore, we need to track all sets to record the minimal number. In practice, the optimization is used to speed up the above process. If no tag replied before the minimal number of empty slots that we already know, we just terminate reading such a set, because it does not change the minimal value.

The reason why we can minimize slot count from different sets is that the reply slot by each tag is only dependent on the frame size $f$ and random seed $R$. So long as the same parameters are used, a tag will always pick the same slot in the frame. Based on this property, any reply that occurs before the first reply in other sets must belong to a new tag. In other words, even if the same tags have responded in multiple sets, the first non-empty slot will remain the same. The final result is equivalent to having all distinct tags belong to one large single set. Therefore, our extended approach remains accurate while significantly reducing time cost.

## VI. PERFORMANCE EVALUATION

The goal of this paper is to design an estimator to count tags efficiently and anonymously in both static and dynamic environments. Here, we evaluate the performance of our FNEB estimator, the enhanced FNEB estimator for single set of tags, and the extended FNEB estimator for multiple sets of tags. Through extensive simulation, we compare our estimators against several well-known estimators mentioned in the related work. They are the Combined Simple Estimator (CSE) [12], the Unified Probabilistic Estimator (UPE) [12], and the Enhanced Zero-Based (EZB) estimator [13]. These estimators are selected for two reasons. First, they can all provide the desired estimating accuracy (say, $Pr[|\tilde{t} - t| \leq \epsilon t] \geq 1 - \delta$). Second, they are more efficient than other estimators we do not list here.

All estimators were implemented in Java. We first investigate the estimators for static set, then the estimators for multiple sets. Unless otherwise specified, we set the maximum number of RFID tags $t_{max}$ to $10000$, the confidence level $\epsilon$ to $0.05$, and the error probability $\delta$ to $0.01$. Each result is the average of $100$ iterations. These experiments test the hypothesis that our estimators can be more efficient than other estimators.

### *A. Time Efficiency*

Prior work in [12] and [13] uses the number of slots that a reader has to scan as an indicator of time efficiency. The reader that scans a few slots will perform faster than the reader that needs to scan many slots. However, the number of slots used is misleading, since different types of slots have variant

durations in practice. According to the current standards (EPC global Class-1 Gen-2 [28]), we assume a reader needs almost 300 $\mu s$ to detect an empty slot, 1500 $\mu s$ to detect a collision slot, and 3000 $\mu s$ to detect a collision slot. Therefore, estimators (like CSE and UPE) that must identify the type of each slot will spend long time on every slot. However, for EZB and our FNEB that only distinguish an empty slot from a non-empty slot, the duration of every slot is equivalent to that of an empty slot.

*1) Single set of RFID tags:* In Table IV, we show the number of slots scanned by every estimator. As we see, if we only compare the number appeared, it seems that CSE and UPE perform well since the sum of slots is small.

However, despite a little more slots needed for estimation, our proposed algorithms do *not* have poor performance (efficiency) relative to CSE and UPE, since the duration of each slot in FNEB and enhanced FNEB is much smaller than that in CSE and UPE. As described above, CSE and UPE have to identify whether a slot is empty, singleton or collision, so additional time is spent to check the CRC (Cyclic Redundancy Check) checksum. Our algorithms otherwise only determine whether a slot is empty or non-empty. Therefore, each slot in our algorithms costs much small time than CSE and UPE. Fig. 4 shows the amount of time required by all estimators with respect to variant slot durations. We see that our enhanced FNEB outperforms any other schemes, especially in large-scale RFID systems. In addition, we understand that the skewed $t_{max}$ is really a serious problem. Without dynamically shrinking $t_{max}$, the FNEB spends much longer time than others, when the number of tags is smaller than 2000.

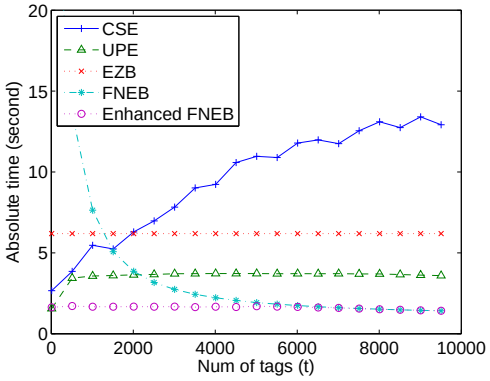


Fig. 4. Time-efficiency comparison of single set estimators

*2) Multiple sets of RFID tags:* Considering multiple sets of tags, only two estimators, EZB and our extended FNEB, can be used to estimate the number of tags among all estimators mentioned early. So we only compare our extended FNEB against EZB here. For simplicity, "FNEB" in Fig. 5 and 6 is refer to the "extended FNEB". Also, since both estimators distinguish between empty slot and non-empty slot, we use the number of slots instead of the absolute time for evaluation.

In the simulation, we set $m = 100$, and use the same model described at the beginning of Section VI to generate multiple

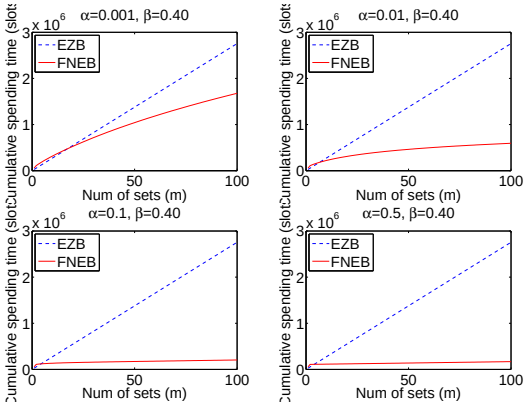


Fig. 5. Cumulative number of slots for estimation versus the number of sets, while increasing $\alpha$ and holding $\beta$

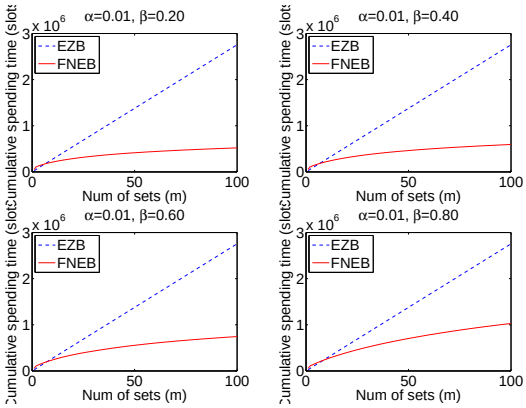


Fig. 6. Cumulative number of slots for estimation versus the number of sets, while increasing $\beta$ and holding $\alpha$

data sets. Let $\alpha$ denote the percentage of the size of each set to $t_{max}$, and $\beta$ denote the percentage of the overlapped tags between two tag sets. In Fig. 5, we hold parameter $\alpha$ and change $\beta$ to conduct the comparison, and vice versa in Fig. 6. From the results, we see that our scheme is more efficient than EZB in all tests.

## B. Additional Discussions

This subsection covers some other issues whose details are omitted due to the page limit.

1) **Accuracy requirements**: In our simulation, we randomly select 1000 possible values for $t$, ranging from 1 to $t_{max}$. The results show that the estimation falling out of the range $[t - \epsilon t, t + \epsilon t]$ only twice. The estimating accuracy holds with more than $1 - \delta$ probability.
2) **Scalability**: The tag population may vary across many orders of magnitude, ranging from tens to thousands of tags. In our simulation, we consider the tag population varies in four scales of $t_{max}$: 100, 1000, 10000, and 100000. The results show the estimating time does not increase obviously. Our estimator scales well.
3) **Signal loss**: Our scheme leverages the first non-empty slot in a frame for estimation. In practice, when the link

TABLE IV
TOTAL TIME (IN NUMBER OF SLOTS) FOR FIVE SINGLE SET ESTIMATORS. SINCE CSE AND UPE NEED TO IDENTIFY THE TYPE OF A SLOT, WE LIST THE DETAIL: EMPTY SLOTS, SINGLETON SLOTS, AND COLLISION SLOTS. FOR OTHERS, WE SIMPLY SHOW THE SUM.

| Number of tags | Total time (in number of slots) | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | CSE | | | | UPE | | | | EZB | FNEB | Enhanced FNEB |
| | empty | singleton | collision | sum | empty | singleton | collision | sum | sum | sum | sum |
| 10 | 2220 | 530 | 305 | 3055 | 1135 | 384 | 71 | 1590 | 21,052 | 98,132 | 5526 |
| 50 | 2264 | 534 | 345 | 3143 | 155 | 269 | 416 | 840 | 21,052 | 91,808 | 5738 |
| 100 | 2277 | 642 | 328 | 3247 | 91 | 239 | 1050 | 1380 | 21,052 | 84,559 | 5732 |
| 500 | 1974 | 972 | 450 | 3396 | 151 | 380 | 1509 | 2040 | 21,052 | 46,525 | 5758 |
| 1000 | 1926 | 1375 | 704 | 4005 | 150 | 388 | 1592 | 2130 | 21,052 | 26,010 | 5683 |
| 5000 | 971 | 1822 | 4358 | 7151 | 147 | 406 | 1697 | 2250 | 21,052 | 6510 | 5661 |

quality is poor, the reader may not be able to detect the signal sent by RFID tags, resulting in the reader possibly observing more empty slots. We can compensate by averaging the results over multiple rounds. In addition, a learning phase can be adopted to characterize the link quality before estimation.

4) **Active attacks**: If an attacker can intentionally generate a reply in an arbitrary slot, there is no feasible solution to solve this problem till now, since all replies from the legitimate tags may be corrupted by the attacker. Therefore, active attacks are excluded in this paper.

## VII. CONCLUSIONS

In this paper, we consider the problem of estimating the number of distinct tags without identifying each tag in a large scale RFID system. We present a new scheme and its variations based on the probability of the position of the first reply from a group of tags. These schemes can be used to estimate tag population in both static and dynamic environments. Theoretical analysis and extensive simulation show our approach drastically improves the time efficiency over prior schemes.

## ACKNOWLEDGMENTS

We would like to thank all the reviewers for their helpful comments. This project was supported in part by US National Science Foundation grants CNS-0721443, CNS-0831904, CAREER Award CNS-0747108, the National High-Tech Research and Development Program of China (863) under Grant No. 2006AA01Z199, the National Natural Science Foundation of China under Grant No. 90718031, No. 60721002, No. 60573106 and the National Basic Research Program of China (973) under Grant No. 2009CB320705.

## REFERENCES

[1] L. Ni, Y. Liu, Y. C. Lau, and A. Patil, "Landmarc: indoor location sensing using active RFID," *Percom '03*.
[2] C. Wang, H. Wu, and N.-F. Tzeng, "RFID-based 3-d positioning schemes," *INFOCOM '07*.
[3] C.-H. Lee and C.-W. Chung, "Efficient storage scheme and query processing for supply chain management using RFID," in *SIGMOD '08*.
[4] A. Nemmaluri, M. D. Corner, and P. Shenoy, "Sherlock: automatically locating objects for humans," in *MobiSys '08*.
[5] L. Ravindranath, V. N. Padmanabhan, and P. Agrawal, "Sixthsense: RFID-based enterprise intelligence," in *MobiSys '08*.
[6] C. C. Tan, B. Sheng, and Q. Li, "How to monitor for missing RFID tags," in *IEEE ICDCS* , 2008.
[7] B. Sheng, C. C. Tan, Q. Li, and W. Mao, "Finding popular categoried for RFID tags," in *ACM Mobihoc*, 2008.
[8] A. Juels, D. Molnar, and D. Wagner, "Security and privacy issues in e-passports," in *SECURECOMM*, 2005.
[9] RFID driver's licenses debated. [Online]. Available: http://www.wired.com/politics/security/news/2004/10/65243
[10] A. Juels, "RFID security and privacy: A research survey," Manuscript, RSA Laboratories, September 2005.
[11] C. C. Tan, B. Sheng, and Q. Li, "Secure and serverless RFID authentication and search protocols," *IEEE Transactions on Wireless Communications*, 2008.
[12] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems," in *MOBICOM*, 2006, pp. 322–333.
[13] M. Kodialam, T. Nandagopal, and W. C. Lau, "Anonymous Tracking using RFID tags," in *INFOCOM*, 2007.
[14] Z. Bar-Yossef, T. S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan, "Counting distinct elements in a data stream," in *RANDOM*, 2002.
[15] J. Zhai and G.-N. Wang, "An anti-collision algorithm using two-functioned estimation for RFID tags," in *ICCSA (4)*, 2005, pp. 702–711.
[16] J. Myung and W. Lee, "Adaptive splitting protocols for rfid tag collision arbitration," in *MOBIHOC*, 2006, pp. 202–213.
[17] H. Vogt, "Efficient object identification with passive RFID tags," in *PERVASIVE*, 2002, pp. 98–113.
[18] C. Law, K. Lee, and K.-Y. Siu, "Efficient memoryless protocol for tag identification," in *DIAL-M*, 2000, pp. 75–84.
[19] D. Hush and C. Wood, "Analysis of tree algorithms for rfid arbitration," in *ISIT*, 1998.
[20] F. Zhou, C. Chen, D. Jin, C. Huang, and H. Min, "Evaluating and optimizing power consumption of anti-collision protocols for applications in rfid systems," in *ISLPED*, 2004.
[21] N. Abramson, "The Aloha system - another alternative for computer communications," in *AFIPS Conference*, 1970.
[22] J.-R. Cha and J.-H. Kim, "Novel anti-collision algorithms for fast object identification in RFID system," in *ICPADS (2)*, 2005, pp. 63–67.
[23] B. Zhen, M. Kobayashi, and M. Shimizu, "Framed ALOHA for multiple rfid objects identification," *IEICE Transactions 2005*.
[24] S.-R. Lee, S.-D. Joo, and C.-W. Lee, "An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification," in *MOBIQUITOUS*, 2005, pp. 166–174.
[25] C. Qian, H. Ngan, and Y. Liu, "Cardinality estimation for large-scale RFID systems," in *PERCOM '08*.
[26] H. Tijims, *Understanding Probability: Chance Rules in Everyday Life*. Cambridge University Press, 2007.
[27] M. Abramowitz and I. A. Stegun, *Handbook of mathematical functions with Formulas, Graphs, and Mathematical Tables*. Dover Publications, 1972.
[28] "EPC radio-frequency identity protocols class-1 generation-2 UHF RFID protocol for communications at 860 mhz - 960 mhz version 1.1.0," EPCglobal, Tech. Rep., 2005.