PET: Probabilistic Estimating Tree for Large-Scale RFID Estimation

Yuanqing Zheng, *Student Member*, *IEEE*, and Mo Li, *Member*, *IEEE*

Abstract—Estimating the number of RFID tags in the region of interest is an important task in many RFID applications. In this paper, we propose a novel approach for efficiently estimating the approximate number of RFID tags. Compared with existing approaches, the proposed Probabilistic Estimating Tree (PET) protocol achieves $\mathcal{O}(\log\log n)$ estimation efficiency, which remarkably reduces the estimation time while meeting the accuracy requirement. PET also largely reduces the computation and memory overhead at RFID tags. As a result, we are able to apply PET with passive RFID tags and provide scalable and inexpensive solutions for large-scale RFID systems. We validate the efficacy and effectiveness of PET through theoretical analysis as well as extensive simulations. Our results suggest that PET outperforms existing approaches in terms of estimation accuracy, efficiency, and overhead.

Index Terms—RFID systems, tag estimation, aloha networks

1 Introduction

Radio Frequency Identification (RFID) technology [6] has recently attracted dramatic attentions from the research community. A typical RFID system consists of RFID readers, RFID tags, and the middleware software to support proper working of the system [32]. An RFID tag is a small microchip capable of wireless communication, which transmits data in response to interrogation by an RFID reader. RFID tags, each with a small size of memory to store its unique ID number as well as other related information, are usually attached to real objects for explicitly labeling those objects. An RFID reader can thus identify and itemize the objects by verifying the unique IDs of RFID tags attached to them. Due to the simple structure, small size, and low manufacturing cost of RFID tags, it provides us an economic and competitive method to utilize the RFID system for massive object management in a variety of applications, such as localization [23], inventory control [17], [30], [34], [37], object tracking [36], activity monitoring [19], authentication and security [20], [29], [35], etc.

Estimating the number of RFID tags, accordingly the number of objects, is one of the primary tasks in many such applications, e.g., counting the number of conference or exposition attendees with RFID badges [16], verifying the amount of products with RFID labels in cargo shipping at the airport [25].

The problem of estimating RFID tag number can be easily reduced to identifying the IDs of all RFID tags and itemizing them. As the RFID readers and tags in the area usually share one same communication channel, a careful scheduling mechanism must be provided for multiple channel access and collision arbitration. There have been already a number of works proposed for solving the tag identification problem [3], [26], [38] and they can be directly borrowed to compute the exact number of RFID tags when the size of the RFID system is small. Those solutions, however, become infeasible when the RFID system scales up. The processing time rapidly grows as the number of RFID tags increases.

As a matter of fact, counting the exact number of RFID tags is not always necessary. Instead, knowing the approximate amount with some guaranteed accuracy and confidence level will be adequate in many application scenarios. For example, it suffices to know the approximate amount of products instead of the exact number in shipping a large amount of cargoes. In accordance with that, a set of probabilistic counting schemes have been proposed to estimate the approximate number of RFID tags with much reduced time slots for information exchange [12], [15], [16], [24]. Some most recent works achieve processing efficiency with $\mathcal{O}(\log n)$ time slots for each estimation round to the total number of RFID tags n. Nevertheless, as we will later elaborate, most probabilistic approaches require many independent rounds of estimation so as to reach high accuracy and confidence level. Thus, it is yet significant to further improve the processing efficiency such that the system would scale up to support a larger number of RFID tags. Besides, most existing approaches require that the RFID tags react to the reader in a probabilistic manner with uniform or geometric distribution hash functions implemented inside. Generating randomness itself, however, becomes already a heavy burden for resource-limited RFID tags, especially those passive tags without self-support energy source.

In this paper, we propose a novel approach for efficiently estimating the approximate number of RFID tags with arbitrarily required accuracy and confidence level. We divide the tag set and define random estimating paths based on a novel coding structure, Probabilistic Estimating Tree (PET). With the help of PET, we develop an estimation algorithm of $\mathcal{O}(\log\log n)$ processing efficiency to

• *The authors are with the School of Computer Engineering, Nanyang Technological University, 50 Nanyang Avenue, N4-B2A, Singapore 639798. E-mail: yuanqing1@e.ntu.edu.sg, limo@ntu.edu.sg.*

Manuscript received 20 Apr. 2011; revised 11 Sept. 2011; accepted 20 Sept. 2011; published online 3 Nov. 2011.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2011-04-0204. Digital Object Identifier no. 10.1109/TMC.2011.238.

the total number of RFID tags n, which significantly improves the state-of-the-art performance bound. By querying on random estimating paths, PET further shifts the overhead of generating probabilistic hash results on the tags to random path selection on the readers, which remarkably reduces the computational burden of RFID tags, providing better applicability and scalability. Our contributions can be summarized as follows:

- We propose an $\mathcal{O}(\log\log n)$ estimation approach, PET, which significantly improves the state-of-the-art performance bound of RFID estimation, providing the capability to support millions of RFID tags.
- PET remarkably reduces the computation and memory overhead at RFID tags, providing us applicable and scalable alternatives of using much less costly but resource-limited passive RFID tags.
- Similar with some other probabilistic counting approaches, PET is effective in handling dynamic tag set and multiple reader environment, where problems like tag join/leave and duplicate counts among different readers can be easily resolved. The privacy of RFID tags can be preserved as well.

The rest of the paper is organized as follows: we introduce related works in Section 2. In Section 3, we give a formal description of the RFID estimation problem and present the design goal and requirements. In Section 4, we give detailed description on the design and analysis of PET. In Section 5, we do intensive simulations to evaluate PET performance and compare with most recent works. Finally, we conclude this work in Section 6.

2 Related Work

The problem of estimating the number of RFID tags can be directly reduced to identifying the IDs of all RFID tags and itemizing them. Since a large number of RFID tags normally share the same physical communication channel, unordered concurrent communications may result in transmission collisions among tags. To address such a problem [18], [21], [22], [25], [33], many time-domain anticollision methods have been proposed [3], [26], [28], [38], which can generally be classified into two categories: slotted Aloha protocols [26], [28] and tree-based protocols [3], [38]. In an Aloha-based protocol, an RFID tag replies immediately to reader's interrogation. If a collision occurs, the tag replies again after a random delay. The process continues until all tags are successfully recognized by the reader. The Aloha-based protocols mitigate the negative impact of collisions with retransmissions but cannot remove collisions. With Aloha-based protocols, a specific tag may not be identified for an excessively long time. In a Tree-based anticollision protocol, an RFID reader interrogates tags and detects whether or not there are any collisions. Once collisions occur, the reader splits the tag set into two subsets by tag IDs and queries the subsets with fewer tags. The reader continues the splitting procedure and the probability of collisions within each tag subset decreases until each tag can be successfully identified.

In small-scale RFID systems, RFID identification schemes can be directly applied to estimate the exact number of tags by itemizing each tag within the reader's interrogation region. Those solutions, however, become infeasible when the RFID system scales up. The processing time rapidly grows as the number of RFID tags increases. In particular, compared with tree-based anticollision protocols for RFID identification, the PET approach proposed in this paper only estimates the total number of tags. PET does not aim at resolving any tag collisions.

Rather than identifying all the RFID tags, probabilistic counting algorithms have been designed for quickly estimating the number of distinct RFID tags. Kodialam and Nandagopal presented Unified Simple Estimator (USE) and Unified Probabilistic Estimator (UPE) in [15]. One drawback of those schemes is that they are vulnerable to replications when one tag is read by multiple readers, and the schemes require approximate magnitude of the tag number as a prior knowledge. In [16], an Enhanced Zero-Based (EZB) estimator was proposed which provides anonymous estimation and can estimate relatively larger number of tags.

Some most recent approaches advance the estimation efficiency, and achieve $\mathcal{O}(\log n)$ processing efficiency to the number of RFID tags n. In [12], Han et al. present an $\mathcal{O}(\log n)$ estimator by quickly positioning the first nonempty slot with binary search algorithm. They further provide an adaptive shrinking algorithm to adjust the upper bound of the tag number so as to speed up the estimation process. Qian et al. [24] propose LoF estimating algorithm, which leverages a geometric distribution hashing process to code tags with $\mathcal{O}(\log n)$ bits and by so estimates the tags with $\mathcal{O}(\log n)$ time slots. LoF [24] is able to address the multiple reader problem as well. Both approaches require that RFID tags react to the reader with on-chip computations for generating some kind of randomness (uniform or geometric distribution hashing).

Cardinality estimation of a large volume of objects has also been studied in other research fields. In [8], Flajolet and Martin introduce the pioneering FM sketch to estimate the cardinalities in a database using a small memory space. There are some other related works for cardinality estimation for applications in database [5], [9], wireless sensor networks [4], etc. Those works, however, cannot be directly borrowed to the RFID environment where we have much resource-limited RFID tags and we need highly efficient approaches to deal with the wireless communication channel shared by thousands or millions of tags.

3 Problem Description

A large-scale RFID system consists of one or more RFID readers and a vast number of RFID tags attached on physical objects. The goal of efficient RFID cardinality estimation is to obtain the approximate number of RFID tags in the region in a fast and accurate manner. Since the number of RFID tags can be extremely large in meeting large-scale application demands, like product amount estimation in shipping cargo containers, the processing approach needs to be designed scalable with the quantity of RFID tags while meeting prerequired accuracy and confidence level.

Similar with [12], [15], the accuracy requirement of estimation is defined by two parameters: a *confidence interval* ε and an *error probability* δ. Assume that an estimating result of the RFID tag number is $\hat{n}$ while the actual number is n, we consider the estimator accurate and precise if $\hat{n}$ satisfies $Pr\{|\hat{n}-n| \le \varepsilon n\} \ge 1-\delta$. For instance, if the actual number of RFID tags in a region is 50,000, and the accuracy requirement is specified as $\varepsilon = 5\%$ and $\delta = 1\%$, an accurate estimation approach is expected to output the estimated number within the interval [47,500, 52,500] with more than 99 percent probability.

The underlying RFID system is assumed to work on a slotted MAC model. The time period is divided into small time slots. In each slot, the reader first transmits continuous waves to energize the RFID tags as well as the command for tags' response and the tags then accordingly respond in the second half of the time slot, which is denoted as Reader Talks First mode and has been widely accepted and used in many RFID systems [3], [12], [26], [38]. Some works also assume that the reader sends out the command at the very beginning of a frame of slots and RFID tags then consequently respond at consecutive time slots. Such an assumption, however, requires that all RFID tags synchronize their frame of slots and have their own energy resource to support their proper working for the entire frame.

Same as the previous works, we assume that the RFID tags can perform stateful computations. As a matter of fact, the current EPC global Class-1 Gen-2 standard [1] requires that RFID tags are able to record intermediate computation states.

One main requirement for a good RFID estimation approach is *efficiency*, which requires a short processing period, i.e., a small number of time slots for reader-tag communication to achieve the desired accuracy. We want to keep it a small order to the total number of RFID tags so as to support scalable RFID systems.

We also want to make the approach *lightweight* for RFID tags. There are two types of RFID tags, active ones and passive ones [27]. Active tags are capable of doing complex computations with self-energy supply but are expensive and bulky. Passive tags are instantly energized by the reader to carry out extremely limited computations but are cheap and easy to be massively used. We want to design the estimation approach lightweight so as to support a variety of applications using passive tags.

Besides, we hope to ensure the RFID estimation process *anonymous*. In some applications, the tag ID carries identity information about the object it is associated with. Revealing such information to the public might lead to leak of private information. We need to design the RFID estimation process *robust* as well, effective with dynamic RFID tag set and multireader environment. In such environment, the tags are attached to mobile objects which may move across the coverage areas of multiple readers. Without careful design consideration, duplicate tag counts may lead to erroneous estimation.

4 PET Design

To estimate the total number of RFID tags in the region of interest, we code those tags and divide them into small subgroups. We show that a probabilistic estimation tree well organizes the coding structure. By defining an arbitrary estimating path on PET, we get an actual querying scheme to estimate tags across different subgroups and approximate the total number of tags. In the following, we first present a basic estimation algorithm of $\mathcal{O}(\log n)$ processing efficiency in Section 4.1. We present theoretical analysis of the basic algorithm in Section 4.2 and then develop a practical estimation protocol in Section 4.3. Based on the basic protocol, in Section 4.4 we optimize the algorithm and present an $\mathcal{O}(\log\log n)$ protocol that significantly improves the processing efficiency. In Section 4.5, we develop techniques to further reduce the computation and memory overhead at RFID tags. In Section 4.6, we discuss some practical issues related to PET protocol and compare with some existing approaches.

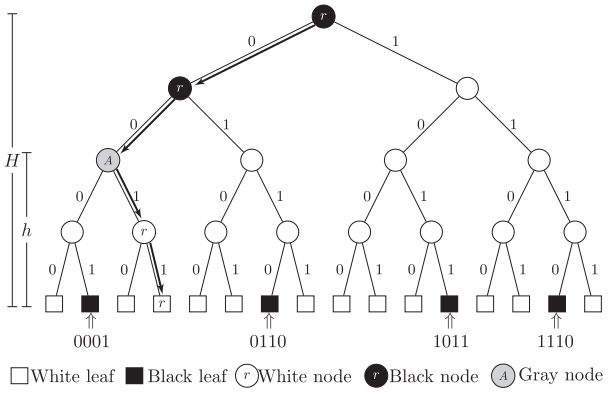


Fig. 1. Probabilistic estimating tree (PET).

4.1 Basic Algorithm

PET is built on top of a binary tree as shown in Fig. 1. We call a nonleaf node as *node*, and a leaf node simply as *leaf*. Each node has two branches, labeled as the *0-branch* and *1-branch*. Each top-down path from the root to a leaf thus gives a bit string with branch labels, and such a bit string codes that leaf. Different RFID tags are mapped to different leaves according to the codes.

For the sake of simplicity, we use an example to illustrate the design of PET. We assume that there are $n = 4$ RFID tags in the system.

First, PET uses a uniform random hash function $\mathcal{H}(tagID) \to [0, 2^H - 1]$ to generate a random code for each RFID tag. In this example, we set $H = 4$, and assume the four RFID tags are assigned random codes 0001, 0110, 1011, and 1110, respectively. As Fig. 1 depicts, they are mapped to the four black leaves. For an arbitrary node in PET, if there are no black leaves within the subtree rooted at it, we say the subtree is *white*, and label the node *white*; otherwise the subtree is *black* and the node is labeled *black*.

We define the *height* of a node as the distance of the path between the node and a leaf in its subtree. The height of PET, denoted as H, in this example is 4.

To estimate the number of tags, an RFID reader generates a random bit string r, say 0011 in this example, indicating an estimating path from the PET root down to a leaf. For each node i, there are two subtrees along its two branches. We denote the subtree of node i along the estimating path r

TABLE 1
Key Notations

Symbols	Descriptions
n	Number of tags
$\hat{n}$	Estimated number of tags
p	Fraction of white leaves on the estimating tree
r	A random estimating path
H	Height of the probabilistic estimating tree
h	Height of the gray node
$\mathcal{H}()$	A uniform hash function
$node_r^i$	Node along r at the height of i
$\mathcal{ST}_r^i$	Subtree of node i along r
$\mathcal{ST}_{\sim r}^i$	Subtree of node i *NOT* along r

as $\mathcal{ST}_r^i$, and the other subtree of node i that does not follow r as $\mathcal{ST}_{\sim r}^i$. If node i is black, either $\mathcal{ST}_r^i$ or $\mathcal{ST}_{\sim r}^i$ is black, and if node i is white, both $\mathcal{ST}_r^i$ and $\mathcal{ST}_{\sim r}^i$ should be white. There is a particular black node i along the path whose $\mathcal{ST}_r^i$ is white (all other black nodes have black $\mathcal{ST}_r$) and such a node is the lowest black node along the path. We define it as a *gray node* (node A in Fig. 1). The height h of the gray node, as our later analysis in Section 4.2, implies the number of RFID tags (black leaves in PET). Intuitively, we can imagine that the bigger fraction of white leaves there are in PET, the higher the gray node is. Thus, we can use h to estimate the number of RFID tags. Table 1 summarizes the notations used across this paper.

To find out h, the RFID reader initiates prefix query along the selected estimating path r, for the example in Fig. 1, 0011. First the reader requests those tags whose random codes match prefix 0^{***} to respond. As the four tags (black leaves in PET) are assigned 0001, 0110, 1011, and 1110, respectively, the ones with 0001 and 0110 will respond to the reader at the reply slot. Though the responses result in a collision slot, the reader detects the existence of responsive signal and is aware of the existence of 0^{***} prefix tags. The reader then goes ahead with the estimating path and requests the response of 00^{**} prefix tags, and the tag with 0001 responds. The reader continues such a process till there is no response from RFID tags. In this example, when the reader queries 001^{*} prefix, as no tags are with such a prefix, no response is made and the reader detects an idle slot. The reader can thus infer that there must be some black leaves matching prefix 000^{*} and find out the only gray node A (with path prefix 00^{**}) on the estimating path $r = 0011$ in PET. Consequently, the height h of A is 2. We will show in the next section how the height of A is used to derive the approximate number of RFID tags.

In practice, we use a relatively large H, say 32, to build a large PET that is able to accommodate billions of black leaves. Querying along the 32-bit estimating path will lead us to h and thus the number of RFID tags. In some cases, we may have a priori knowledge about the scale of the RFID tags, and adjust H according to the rough tag cardinality. One thing worth noting is that, the PET structure is neither created nor maintained at the RFID reader. It is only a *conceptual data structure* that illustrates the organization of RFID tag groups as well as the reader query process over such tag groups. As we will see in Sections 4.3 and 4.4, in a practical protocol, the reader simply queries the tags with a randomly selected estimating path and calculates h with tag responses.

4.2 Algorithm Analysis

As suggested in the previous section, the height h of the gray node plays a very important role in estimating the number of RFID tags. We can use h to estimate the number of black leaves in PET, accordingly the number of RFID tags. We denote the fraction of white leaves in PET as p and the fraction of black leaves is $1-p$. We present theoretical analysis for the estimation accuracy of PET algorithm in the following.

We start from two extreme cases, $p = 1$ and $p = 0$, respectively. $p = 1$ corresponds to that all the leaves in PET are white. In such a case, we can infer that the number of tags is 0. $p = 0$ corresponds to that all the leaves in PET are black. In such a case, we can roughly estimate that the number n of tags hashed to the leaves of PET

$$n \geq 2^H. \tag{1}$$

As a matter of fact, in the case of $p = 0$ the hashing process can be modeled as the famous coupon collector problem taking the hashing collision into consideration. This paper does not try to deeply investigate such a case, as we can always choose a sufficiently big H such that we can make $p = (1 - \frac{1}{2^H})^n \approx 1$ for an arbitrary number of n ($H = 32$ can accommodate $n = 40{,}000{,}000$ with $p \approx 0.99$), leading to rare hashing collisions.

For the ease of analysis, we focus on the case where both n and 2^H are sufficiently large, and $p \approx 1$.

Let the random variable h be the height of the gray node i on a randomly selected estimating path r. Then we have

$$Pr(h) = Pr\{\mathcal{ST}_r^i = white, \mathcal{ST}_{\sim r}^i = black\}. \tag{2}$$

As the PET random codes of tags are independently assigned for the 2^{h-1} leaves in either $\mathcal{ST}_r^i$ or $\mathcal{ST}_{\sim r}^i$ with uniformly random distributed hash function. So we have

$$Pr\{\mathcal{ST}_r^i = white\} = p^{2^{h-1}}. \tag{3}$$

We can also obtain

$$Pr\{\mathcal{ST}_{\sim r}^i = black\} = 1 - p^{2^{h-1}}. \tag{4}$$

Hence, we have

$$Pr(h) = p^{2^{h-1}}\left(1 - p^{2^{h-1}}\right). \tag{5}$$

As a result, the expectation of h is

$$E(h) = \sum_{k=1}^{H} kPr(k) = -Hp^{2^H} + \sum_{k=0}^{H-1} p^{2^k}. \tag{6}$$

Since $p = (1 - \frac{1}{2^H})^n \approx e^{-n2^{-H}}$, we have

$$E(h) = -He^{-n} + \sum_{k=0}^{H-1} e^{-n2^{-k-1}} \approx \sum_{k=0}^{H-1} e^{-n2^{-k-1}}. \tag{7}$$

We appeal to Mellin transforms to derive the asymptotic closed form of the harmonic summation [7], [14] as follows:

$$E(h) \approx H - \left[\log_2 n + \frac{\gamma}{\ln 2} - \frac{1}{2} + \mathcal{P}(\log_2 n) + O\left(\frac{1}{\sqrt{n}}\right)\right], \quad (8)$$

where γ is Euler's constant, $\mathcal{P}(x)$ is a periodic and continuous functions of x with period 1 and amplitude bounded by 10^{-5}. We omit the term $\mathcal{P}(\log_2 n) + O(\frac{1}{\sqrt{n}})$, and let $\phi = \frac{e^{\gamma}}{\sqrt{2}} = 1.25941\ldots$, then

$$E(h) \approx H - \log_2(\phi n). \quad (9)$$

Correspondingly, the standard deviation of h is

$$\sigma(h) = \sqrt{Var(h)} = \sqrt{\sum_{k=1}^{H} [k - E(h)]^2 Pr(k)}. \quad (10)$$

Similar to the method we derive $E(h)$, we appeal to Mellin transforms to approximate the standard deviation [7]

$$\sigma(h) \approx \sqrt{\frac{\pi^2}{6(\ln 2)^2} + \frac{1}{12}} = 1.87271\ldots. \quad (11)$$

For detailed analysis, we refer the reader to [13], which presents the probabilistic counting theory and can be used to derive $E(h)$ and $\sigma(h)$.

According to (9), the observation of h, the height of gray node, can be used to estimate n, the number of tags.

However, there may exist variance between the observed height of gray node and its expectation $E(h)$. According to law of large numbers [10], [31], the average of the observation results can be used to approximate the expectation of the value. Besides, as more trials are performed, the average would become closer to the expectation.

We define the random process $\bar{h} = \frac{1}{m}\sum_{i=1}^{m} h_i$ as the average value of m independent observations, where h_i denotes the ith observation of random variable h. Since both the estimating path of the reader and the PET codes at tags are randomly generated in each round of estimation, the trials of $h_i (1 \le i \le m)$ are independent random processes. Therefore, we have

$$E(\bar{h}) = \frac{1}{m}\sum_{i=1}^{m} E(h_i) = E(h), \quad (12)$$

$$\sigma(\bar{h}) = \sqrt{Var(\bar{h})} = \sqrt{\frac{Var\left(\sum_{i=1}^{m} h_i\right)}{m^2}} = \frac{\sigma(h)}{\sqrt{m}}. \quad (13)$$

According to (9) and (12), we can estimate the number of tags as follows:

$$\hat{n} = \phi^{-1} \times 2^{H-\bar{h}} = \phi^{-1} \times 2^{H - \frac{1}{m}\sum_{i=1}^{m} h_i}. \quad (14)$$

Then, according to (13), we will be able to reduce the variance and improve the estimating accuracy by performing m rounds of estimation. On the other hand, if m is too small, the estimation result would be poor because of the precision error.

Next, we show that given a particular accuracy requirement, e.g., $Pr\{|\hat{n} - n| \le \varepsilon n\} \ge 1 - \delta$, how many rounds of estimation PET should take to output satisfying results.

We define a random variable as follows:

$$X = \frac{\bar{h} - \mu}{\sigma}. \quad (15)$$

By the central limit theorem [10], [31], we know the random variable X is asymptotically standard normal distribution, where $\mu = E(\bar{h}) = H - \log_2(\phi n)$ and $\sigma = \sigma(\bar{h}) = \frac{\sigma(h)}{\sqrt{m}}$. So, the cumulative distribution function of variable X is

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{t^2}{2}} dt. \quad (16)$$

Given a particular error probability δ, we can always find a constant range c which satisfies

$$1 - \delta = Pr\{-c \le X \le c\} = erf\left(\frac{c}{\sqrt{2}}\right), \quad (17)$$

where erf is the Gaussian error function [2]. On the other hand, we can rewrite the accuracy requirement as follows:

$$\begin{aligned} &Pr\{|\hat{n} - n| \le \varepsilon n\} \\ &= Pr\{(1-\varepsilon)n \le \hat{n} \le (1+\varepsilon)n\} \\ &= Pr\{(1-\varepsilon)n \le \phi^{-1} \times 2^{H-\bar{h}} \le (1+\varepsilon)n\} \\ &= Pr\{H - \log_2 \phi(1+\varepsilon)n \le \bar{h} \le H - \log_2 \phi(1-\varepsilon)n\}. \end{aligned} \quad (18)$$

Therefore, if we have the following condition:

$$\begin{aligned} \frac{H - \log_2 \phi(1+\varepsilon)n - \mu}{\sigma} &\le -c, \\ \frac{H - \log_2 \phi(1-\varepsilon)n - \mu}{\sigma} &\ge c, \end{aligned} \quad (19)$$

we can guarantee the accuracy requirement $Pr\{|\hat{n} - n| \le \varepsilon n\} \ge 1 - \delta$. Combining (12), (13), and (19), we have

$$m \ge max\left\{\left[\frac{-c\sigma(h)}{\log_2(1-\varepsilon)}\right]^2, \left[\frac{c\sigma(h)}{\log_2(1+\varepsilon)}\right]^2\right\}. \quad (20)$$

Therefore, with such calculated m rounds of estimation, PET can guarantee the accuracy requirement of $Pr\{|\hat{n} - n| \le \varepsilon n\} \ge 1 - \delta$. Note that m is solely determined by the accuracy requirement ε and δ. Given the prerequired accuracy level, we can use a constant m that does not relate to the scale of RFID tags.

4.3 General Protocol

As elaborated in previous sections, the number of RFID tags is estimated based on the height of gray nodes in PET, i.e., the idle slots when the reader query with the selected estimating path. In this section, we formally present the general estimation protocol, regulating both the reader behaviors and RFID tag behaviors.

Algorithm 1 defines the behaviors of the RFID reader during each round of estimation. The RFID reader uses Reader Talk First mode to communicate with tags. At the first, the reader computes the number estimation rounds according to accuracy requirement (line 1). In each round, the reader selects a random estimating path r and a random seed s, and broadcast them to the tags (line 3). The reader queries the tags with the additively increased path prefix in the following 32 time slots (lines 4-11). In particular, at the

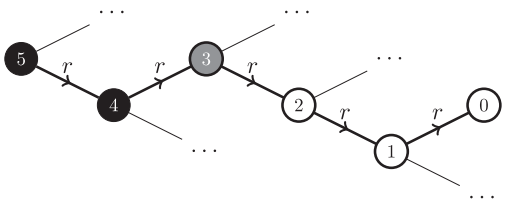


Fig. 2. Gray node on the estimating path r. Node 3 is the gray node in this example.

jth time slot the reader queries with the j-prefix of the selected estimating path r (line 5). At each time slot, the reader broadcasts the prefix $mask$ for each tag's comparison (line 6). The reader listens to the channel and obtains the idle slot j when no response is received and store the value of $j-1$ (lines 7-10). Finally, the reader derives the approximate number of RFID tags according to (14) described in previous section (line 13).

Algorithm 1. PET algorithm for RFID readers

1: $m \leftarrow max\{[\frac{-c\sigma(h)}{\log_2(1-\varepsilon)}]^2, [\frac{c\sigma(h)}{\log_2(1+\varepsilon)}]^2\}$
2: **for** $i \leftarrow 1$ to m **do**
3: Select a random estimating path r and a random seed s; Broadcast r and s
4: **for** $j \leftarrow 1$ to 32 **do**
5: Set high j bits of $mask$
6: Broadcast $mask$; Listen in the following slot
7: **if** there is no response in the slot **then**
8: $h_i \leftarrow j-1$
9: **break**
10: **end if**
11: **end for**
12: **end for**
13: **return** $\hat{n} \leftarrow \phi^{-1} \times 2^{H-\frac{1}{m}\sum_{i=1}^{m} h_i}$

Algorithm 2 defines the behaviors of RFID tags during each round of estimation. Compared with reader behaviors, the task of each tag is simpler. The tag receives the estimating path r as well as the random seed s, and generates the PET random code (lines 1-2). The tag keeps receiving the additively increased $mask$ for the path prefix and compares the path prefix with the prefix of its own generated random code. If they are the same the tag responds to the reader and otherwise the tag simply keeps silent (lines 3-11). With such behaviors fewer RFID tags respond as the querying process goes on and finally all tags will keep silent.

Algorithm 2. PET algorithm for RFID tags

1: Receive the estimating path r and the random seed s
2: Compute PET random code $prc \leftarrow \mathcal{H}(s, tagID)$
3: **while** TRUE **do**
4: Receive $mask$
5: **if** $prc \wedge mask = r \wedge mask$ **then**
6: $\backslash *$ Check whether high $mask$ bits of prc is equal to that of r $*\backslash$
7: Respond immediately
8: **else**
9: Keep silent
10: **end if**
11: **end while**

TABLE 2
Node Classification along an Estimating Path r

Node i	Remarks
$black\ node$	$\mathcal{ST}_r^i = black$
$white\ node$	$\mathcal{ST}_r^i = white$ AND $\mathcal{ST}_{\sim r}^i = white$
$gray\ node$	$\mathcal{ST}_r^i = white$ AND $\mathcal{ST}_{\sim r}^i = black$

Similar with previous estimation approaches like FNEB [12] and LoF [24], the random code of each tag is generated with a random seed sent from the reader at the beginning of each estimation round. In such a case, we need to use active tags such that each tag is capable of executing the random hashing functions to generate the random code.

4.4 *O(loglogn)* Algorithm

Given an estimating path r, we define the node along r with the height of i in PET as $node_i^r$. In the basic algorithm, the reader queries the path prefix additively, which can be mapped to searching the gray node from the root down to the leaf on the estimating path r in PET.

Fig. 2 gives an example to illustrate such a process. In Fig. 2, the estimation process first probes $node_5^r$, and searches along r with additive path prefix query until the gray node $node_3^r$ is found. We need $\mathcal{O}(H)$ time slots for such a sequential path prefix query in searching the height h of the gray node in PET. When the number of RFID tags is large, the height of PET $H \approx \log_2 n$, and thus the basic estimation protocol has $\mathcal{O}(\log n)$ efficiency, which is comparable performance with the state-of-the-art approaches, such as FNEB [12] and LoF [24].

In this section, we improve the estimating efficiency by speeding up the process of path prefix query. As a matter of fact, the nodes along the estimating path r can be classified as shown in Table 2. For arbitrary $i > j$, we have

$$\left(\mathcal{ST}_r^j \bigcup \mathcal{ST}_{\sim r}^j\right) \subseteq \mathcal{ST}_r^i. \tag{21}$$

Thus, we have the following relations between $node_i^r$ and $node_j^r$. For $i > j$,

- If $node_i^r$ is white or gray node, then $node_j^r$ is white node.
- If $node_j^r$ is black or gray node, then $node_i^r$ is black node.
- In either case, only one gray node exists in an estimating path.

Such an observation directly reveals the monotonic feature of the node colors along the estimating path, i.e., the black and white nodes are consecutively aligned and concatenated by the only gray node. In the example shown in Fig. 2, $node_{0,1,2}^r$ are white nodes, $node_{4,5}^r$ are black nodes, and $node_3^r$ is the only gray node in between. Utilizing the monotonic feature of the estimating path, we can use a binary search algorithm to rapidly find the gray node in PET.

Mapped back to the estimation protocol, the reader no longer conducts a sequential path prefix query. Instead, the reader queries the path prefix with binary search. Algorithm 3 gives the improved protocol for the reader. The binary search algorithm is applied to query the path prefix

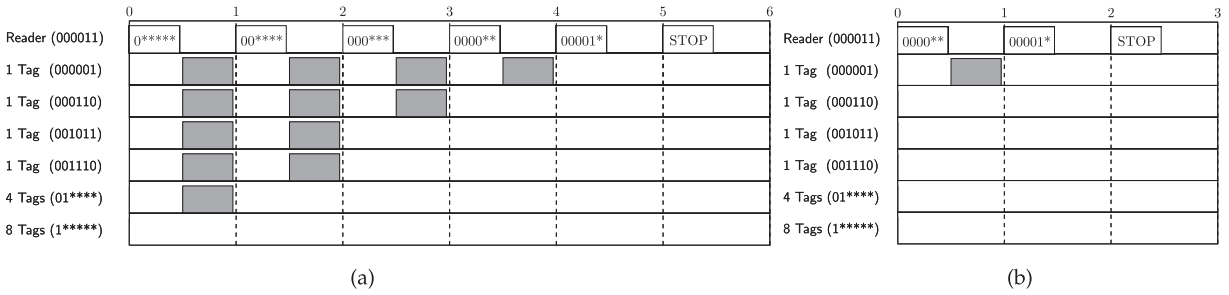


Fig. 3. Protocol execution (a) Basic algorithm. (b) Binary search algorithm. In this example, the height H of PET is set to 6, and the estimating path r is selected to be 000011.

(lines 5-15). Instead of querying the additively increased path prefix, the $mid = \lceil (low + high)/2 \rceil$ bit prefix is chosen at each time slot (line 6). The $high$ end and low end of the prefix range are adjusted according to whether or not the response is received from the tags (lines 9-13). Finally, the $high$ end and low end converge to the height of the gray node on the estimating path, and it is used to derive the approximate number of RFID tags (line 17).

Algorithm 3. PET algorithm for RFID readers with binary search

1: $m \leftarrow max\{\lceil \frac{-c\sigma(h)}{\log_2(1-\varepsilon)} \rceil^2, \lceil \frac{c\sigma(h)}{\log_2(1+\varepsilon)} \rceil^2\}$
2: **for** $i \leftarrow 1$ to m **do**
3: $low \leftarrow 1, high \leftarrow 32$
4: Select a random estimating path r; Broadcast r
5: **while** $low < high$ **do**
6: $mid \leftarrow \lceil (low + high)/2 \rceil$
7: Set high mid bits of $mask$
8: Broadcast $mask$; Listen in the following slot
9: **if** there is no response in the slot **then**
10: $high \leftarrow mid - 1$
11: **else**
12: $low \leftarrow mid$
13: **end if**
14: **end while**
15: $h_i \leftarrow low$
16: **end for**
17: **return** $\hat{n} \leftarrow \phi^{-1} \times 2^{H - \frac{1}{m}\sum_{i=1}^{m} h_i}$

With the above new protocol, the estimation efficiency is further improved. Only $\mathcal{O}(\log H)$ time slots are used to find the gray nodes in PET, which finally gives us $\mathcal{O}(\log \log n)$ estimation efficiency.

Fig. 3 uses an example to demonstrate the performance gain of the improved protocol compared with the basic one. The example contains one RFID reader and 16 RFID tags. The height H of PET is chosen to be 6. Each tag generates a random code with 6 bits and the reader selects a 6-bit random estimating path $r = 000011$.

Fig. 3a depicts the estimation process with the basic protocol. At time slot 0, the reader queries the path prefix 0^{*****}. As a result, the first four tags and the four tags with prefix 01^{****} respond. The reader is aware of that and keeps proceeding at time slot 1 with a path prefix 00^{****}. The query process continues until time slot 4, during which the reader queries the path prefix 00001^{*} and identifies an idle slot. The entire process contains five time slots.

Fig. 3b depicts the estimation process with the improved protocol. At time slot 0, the reader directly queries the mid $(mid = \lceil (low + high)/2 \rceil = \lceil (1 + 6)/2 \rceil = 4)$ path prefix 0000^{**} and one tag responds. Receiving the response, the reader then raises the low end of the query range and at time slot 1 queries path prefix 00001^{*}. At this time, there is a tag response. The reader then lowers the high end of the query range and the estimation converges. The entire process contains only two time slots.

4.5 Shifting the Computation Burden from RFID Tags

With the basic protocol each tag will generate a random PET number for mapping into the PET structure at each round of estimation. Generating the random code at each tag requires a fair amount of computation, which is infeasible for passive tags. An alternative is to preload a number of such random codes on the chip of each RFID tag during manufacturing. At each round of estimation the tag uses one of the preloaded random codes. As a tradeoff, however, extra memory cost is required to store those random codes, which is proportional to the number of estimation rounds m. As there will be generally many rounds of estimations for accurate results, the memory cost for preloaded PET random codes will be high.

With the above concern, we propose to shift such computation burden from RFID tags to the reader. We rely on the random estimating paths generated on the reader rather than refreshing the random codes at tags. Instead of using new random codes at different estimating rounds, a 32-bit PET random code is preloaded on each tag during manufacturing and used across all rounds of estimation. A group of off-the-shelf uniformly distributed hash functions can be used to generate the PET numbers, including Message-Digest algorithm 5 (MD5) and Secure Hash Algorithm (SHA-1). Note that MD5 generates a 128-bit hash value, but we can trivially convert them to shorter length, e.g., a 32-bit hash value, at will.

The reader generates a uniformly distributed random number as an estimating path at each round of estimation. By solely changing the 32-bit estimating path in the 2^{32} combinatorial spaces at each round of estimation, even the PET codes of tags keep unchanged, we are still expecting near independent estimating instances [2], [11], and the algorithm analysis in Section 4.2 still holds. Algorithm 4 defines the new behaviors of RFID tags. The tags use the preloaded random codes through all rounds of estimation (line 1), while in Algorithm 2, tags generate PET random

codes at each round. In such a way, a tag only performs bitwise comparison on the PET code and path prefix during each round of estimation (lines 3-11).

Algorithm 4. PET algorithm for RFID tags with binary search

```
1: PET random code prc = preloaded random number
2: Receive the estimating path r
3: while TRUE do
4:     Receive mask
5:     if prc ∧ mask = r ∧ mask then
6:         \* Check whether high mask bits of prc is equal
           to that of r *\
7:         Respond immediately
8:     else
9:         Keep silent
10:    end if
11: end while;
```

During each round, the reader needs to broadcast a 32-bit random estimating path for each estimation rounds, and has to repeat a large number of estimation rounds for high-accurate estimation. The 32-bit information is capsulated into a packet and sent out together. Thus, the extra overhead imposed by such a component is very limited.

4.6 Discussion

4.6.1 Processing Efficiency and Computational Overhead

According to our analysis, PET achieves the estimation efficiency of $\mathcal{O}(\log \log n)$ which significantly improves the state-of-the-art performance. In PET, RFID tags are only required to perform bitwise operations as well as to store a 32-bit PET random code. The random codes can be preloaded to the tags by RFID manufacturers. Such cost of implementing PET on RFID tags is negligible in terms of both computation and memory requirement in comparison with other probabilistic counting schemes, such as FNEB [12] and LoF [24]. In both approaches, RFID tags need to generate random numbers at each round of estimation, or the random numbers are preloaded and stored at the RFID tags during manufacturing. In order to pursue a high-estimation accuracy, both FNEB and LoF will conduct hundreds or even thousands of rounds of estimation. In such a case, the computation or memory cost would be significant and render it infeasible to work with passive tags.

4.6.2 Command Overhead

With reference to Algorithm 3, PET requires RFID readers to broadcast the *mask*/*prefix* to the tags (a 32-bit random number under the worst situation). The command overhead of broadcasting such a *mask* is nonnegligible. One may have noticed that only high *mid* bits of *mask* are set to ones (line 7), while the low bits remain zeros, which means a 32-bit *mask* actually carries only $\log_2 32 = 5$-bit information. Therefore, it suffices for the RFID readers to broadcast 5-bit *mid*. When receiving the message of *mid*, instead of comparing the preloaded random number prc with the estimating path r (Algorithm 4, line 5), the tags check whether the high mid bits of prc and r are the same, and respond to the readers accordingly.

With another optimization, we can further reduce the command overhead from 5 bits to 1 bit. In Algorithm 3, the readers update the parameters $high$ and low according to the responses from the tags (lines 9-13), and compute a new value of mid (line 6). The responses from the tags only carry 1-bit information representing an empty or nonempty slot. If tags keep $high$ and low locally, they can compute a new value of mid according to 1-bit information which can be used to indicate the previous tag responses, e.g., "0" for an empty slot and "1" for a nonempty slot, respectively. Therefore, instead of broadcasting 5-bit information, it is sufficient to send 1-bit information indicating the responses from the tags. We believe that the cost of managing *high* and *low* (5 bits for each) is small compared with the command overhead.

4.6.3 Multiple Readers and Mobile Tags

Since the propagation range of both RFID readers and RFID tags, especially the passive tags, are limited, we usually deploy multiple readers to enhance the coverage for a large number of tags in the region of interest. In such a scenario, a back end controller coordinates multiple RFID readers. The controller generates an estimating path r, computes the $mask$ for each round of estimation, and sends the estimating path r and $mask$ to the readers. The readers accordingly query RFID tags with r and $mask$, and wait for their replies. The controller aggregates the reports from all readers and takes a slot as idle only when no tag response is reported from any readers. The controller repeats the estimating process until the height of gray node is determined. In the multiple reader scenario, the controller functions as if there were a single RFID reader. Even if a tag is located in the overlapped region and responds to multiple readers, its impact on the controller side is the same as a single response. Thus, the PET protocol well handles the multiple reader scenario due to the duplicate-insensitive nature in tag responses. When RFID tags are attached to mobile objects and moving across interrogation regions of different readers, the responses of the same tag to different readers will converge at the controller as well. Due to the duplicate-insensitive nature, such a scenario is equivalent to that of the multiple readers and can be correctly handled by PET.

4.6.4 Anonymity

In some applications, the tag ID carries private information about the associated object. Revealing such information to the public might lead to the leak of personal privacy. PET fully resists such privacy threats. In PET, during the estimating process, each RFID tag does not participate with its own ID. Instead each tag responds to the reader's query according to a random code which is not directly bound to the tag ID. During the overall estimation process, the preloaded 32-bit random code remains unchanged which is tightly bound to the tag ID. Such a fixed random code does not need to be explicitly transmitted either by the tag or the reader, and the fixed random code is not revealed directly to the public. At each time slot, a number of tags respond to the reader, and their responses cumulate. The readers or any overhearing devices cannot distinguish the exact set of tags which respond at a collision slot. As a result, the

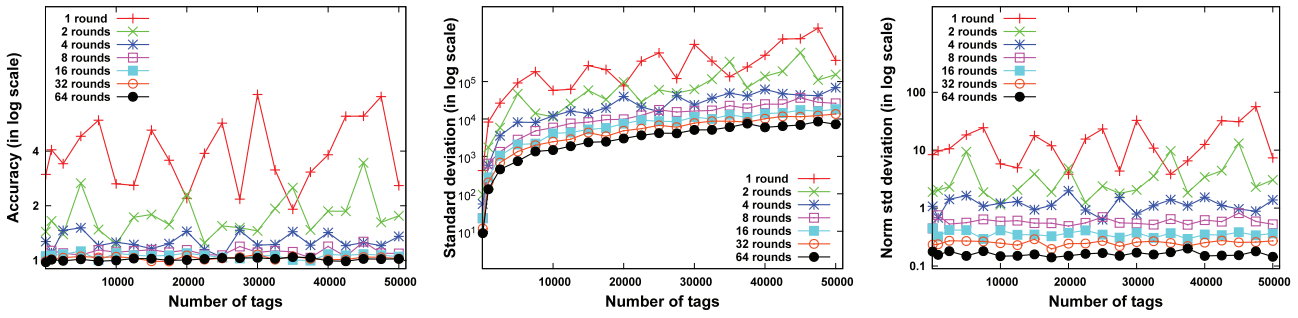


Fig. 4. Evaluation of PET protocol with different numbers of estimating rounds: (a) Estimation accuracy. (b) Standard deviation. (c) Normalized standard deviation.

random PET numbers of RFID tags are not revealed to the readers as well.

5 Performance Evaluation

We evaluate the performance of PET through extensive simulations. First, we investigate the tunable estimation accuracy and processing time of PET with different parameter settings. We then compare PET with two most recent estimation approaches FNEB [12] and LoF [24] in terms of estimation accuracy and efficiency, as well as computational overhead at RFID tags.

5.1 Simulation Setup and Performance Metrics

In the simulations, we assume that there is no transmission loss between RFID tags and the reader. The RFID reader is capable of detecting idle slots from singleton slots as well as collision slots. Before estimation, each RFID tag is assigned a 32-bit PET random code by hashing its unique tag ID. To get each simulation result, we take 300 runs and measure the average.

The estimation accuracy is the most important metric for the estimation protocols. We use the same metric as that defined in LoF [24]

$$Accuracy = \frac{\hat{n}}{n}, \tag{22}$$

where $\hat{n}$ is estimated number of RFID tags while n is the actual number of tags. This parameter indicates the estimating *accuracy*. The closer it is to 1, the more accurate an estimation is. Another metric we are interested in is the standard deviation of estimation

$$\sigma = \sqrt{E[(\hat{n} - n)^2]}. \tag{23}$$

Such a parameter measures the estimating *precision*. Generally, a high-standard deviation means the estimated values are dispersed, and a low-standard deviation means the estimated values are concentrated about the actual tag number. The smaller the standard deviation is, the more precise the estimation protocol performs. Hence, an ideal estimation protocol is expected to have an accuracy of 1 with a low-standard deviation.

Another important metric we consider is the estimating time the protocol takes to meet a particular accuracy and precision requirement. For each round of estimation, the reader takes a number of time slots in interacting with tags. Therefore, the estimating time is proportional to the number of time slots during the estimation. We thus abstract the estimating time as the total number of time slots during the entire estimating process. The estimating time reflects the protocol *efficiency*. The protocol with short estimating time will be able to scale up easily.

Beside accuracy and efficiency, we also take the computation and storage *overhead* on RFID tags into consideration. We measure and compare such overhead of PET and other protocols.

5.2 PET Investigation

First, we demonstrate that PET provides tunable estimation accuracy at the cost of estimating time. As illustrated in Fig. 4a, one can improve the estimation accuracy of PET by performing more rounds of estimation. With 32 to 64 rounds of estimation, PET already maintains the accuracy very close to 1. Such a characteristic of PET enables modulating the estimating accuracy and efficiency according to the distinctive application needs. Fig. 4a also suggests that the change of the number of tags has no significant impact on the estimation accuracy of PET, i.e., we can accurately estimate a wide range of RFID quantities without a priori of the tag number.

We are also interested in the standard deviation of the estimation results. The standard deviation and normalized standard deviation of PET estimation results are depicted in Figs. 4b and 4c, respectively. The figures suggest that by performing more rounds of estimation, the standard deviation of the estimation results can be reduced. Fig. 4c further suggests that if we take a look at the normalized standard deviation, the number of tags will affect little. Sixty-four rounds of estimation lead to nearly 0.2 normalized standard deviation. According to the simulation results, repeating a constant number of estimation rounds suffices to meet the requirement of estimation accuracy regardless of the number of tags.

TABLE 3
Total Time Slots Needed for PET

Rounds	1	4	16	64	256	1024	4096
Time slots	5	20	80	320	1280	5120	20480

TABLE 4
Total Time Slots Needed to Meet the Estimation Accuracy Requirement with Different ε ($\delta = 1\%$)

ε	Enhanced FNEB	LoF	PET
20%	5366	3857	1681
15%	8524	6566	2862
10%	17067	14120	6154
5%	60483	53885	23484

TABLE 5
Total Time Slots Needed to Meet the Estimation Accuracy Requirement with Different δ ($\varepsilon = 5\%$)

δ	Enhanced FNEB	LoF	PET
15%	18759	16734	7406
10%	24699	22004	9590
5%	34465	30705	13382
1%	60483	53885	23484

The processing time of PET is also examined. In the simulation, we use a fixed length of PET number $H = 32$. In such a setting, PET only takes five time slots to complete each round of estimation. Therefore, the total number of time slots needed in m rounds of estimation can be listed in Table 3.

5.3 Performance Comparison

We compare the performance of PET with the two most recent estimation approaches FNEB [12] and LoF [24]. Since both FNEB and LoF assume that the tags are capable of generating large number of random codes, for fairness, we compare the performance of PET with the two recent protocols based on the same assumption. We may compare the estimation accuracies of the approaches given a certain amount of estimating time, or compare the estimating time given a certain estimation accuracy requirement. As we can trade higher estimation accuracy with longer estimating time, we only need to compare with one setting. Hereafter, we mainly compare the estimating time of the three approaches for particular estimation accuracy requirements. We measure the total time slots used in all estimation rounds.

First, we compare the estimating time slots given a particular estimating accuracy requirement $Pr\{|\hat{n} - n| \leq \varepsilon n\} \geq 1 - \delta$, where $\varepsilon = 5\%$, $\delta = 1\%$. We keep δ fixed and change ε from 5 to 20 percent, giving more error tolerance. The number of RFID tags is 50,000.

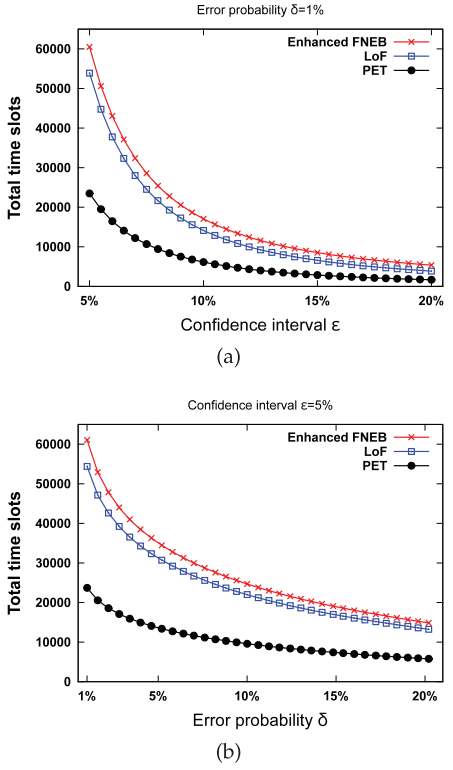


Fig. 5. Performance comparison: (a) Protocol performance with different confidence interval ε, and the same error probability $\delta = 1\%$. (b) Protocol performance with different error probability δ, and the same confidence interval $\varepsilon = 5\%$.

All three approaches shall perform multiple rounds of estimation to achieve the given accuracy requirement. Tables 4 and 5 summarize the total time slots used by each of the three approaches. Fig. 5a gives a more detailed comparison with a better granularity on ε. As suggested by the simulation results, PET outperforms both FNEB and LoF with about 35 to 43 percent of their estimating time. In Fig. 5b, we fix the confidence interval ε and vary error probability δ from 1 to 20 percent. The figure suggests similar results that for arbitrary accuracy requirements PET consumes less than half of the estimating time of FNEB and LoF. From a different point of view, the results also indicate that

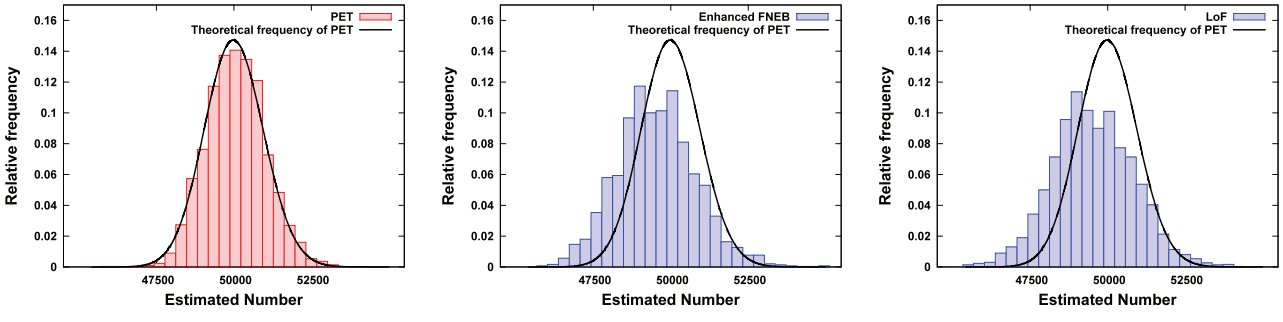
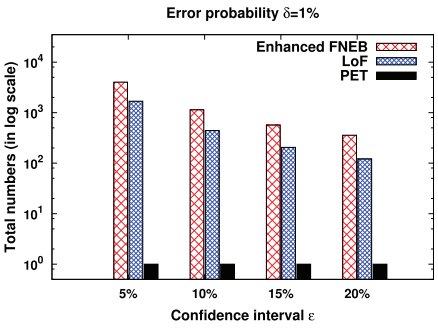
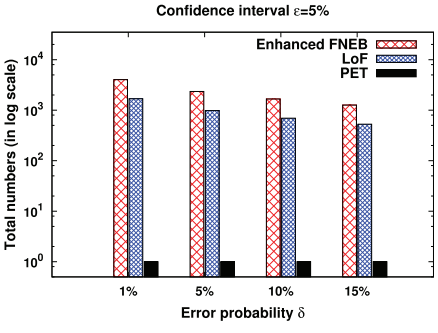


Fig. 6. Estimation accuracy comparison: (a) Theoretical performance of PET versus simulated result of PET. (b) PET versus Enhanced FNEB. (c) PET versus LoF.



(a)



(b)

Fig. 7. Memory consumption in storing the random codes (in *log* scale): (a) With different confidence interval ε, and the same error probability $\delta = 1\%$. (b) With different error probability δ, and the same confidence interval $\varepsilon = 5\%$.

given a certain amount of estimating time, the estimation accuracy of PET will be much higher than FNEB and LoF. As a matter of fact, since PET has $\mathcal{O}(\log \log n)$ estimation efficiency, when the number of RFID tags scales, the performance gain of PET over FNEB or LoF will be larger.

Fig. 6a presents the numerically analyzed performance of PET and the simulated performance of PET in estimating 50,000 RFID tags with estimation accuracy requirement of $\varepsilon = 5\%$, $\delta = 1\%$. Intuitively, we expect that an ideal approach outputs a result concentrated about the actual number (i.e., 50,000 here). From Fig. 6a, we can observe that 1) the simulation results verify the theoretical analysis on PET; 2) most estimated numbers fall into the confidence interval [47,500, 52,500]; 3) for the small portion of estimated numbers $\hat{n} \notin [47,500, 52,500]$, they are still very close to the confidence range.

We let FNEB and LoF consume the same amount of estimating time to estimate the 50,000 tags and compare their performance with the theoretical performance of PET in Figs. 6b and 6c, respectively. The results of Figs. 6b and 6c clearly demonstrate that the estimated values in PET are much more concentrated about the actual number of tags. More importantly, the portion of those falling outside the confidence interval [47,500, 52,500] is much smaller than those of FNEB and LoF. In particular, with the same processing time more than 99 percent estimated results fall into the confidence interval in PET, while FNEB and LoF only guarantee about 90 percent results within such an interval. The results of Fig. 6 give us an intuitive understanding on the performance gain of PET.

Beside the estimation accuracy and efficiency, we compare the computation and storage overhead of the three approaches. For each round of estimation, FNEB or LoF requires each tag to generate a uniformly or geometric distributed random number. For passive tags, such random numbers shall be preloaded and stored at each tag. We examine the storage overhead of storing such random numbers and compare with PET in Fig. 7. In Fig. 7a, we fix the error probability δ and vary the confidence interval ε from 5 to 20 percent. In Fig. 7b, we fix ε and vary δ from 1 to 15 percent. Both figures explicitly suggest that PET outperforms both approaches with much smaller such cost. In contrast, PET shifts computational burden to the more powerful reader side with randomly generating the estimating path.

6 Conclusion

In this paper, we propose PET for efficiently estimating a large number of RFID tags. The theoretical analysis shows that PET is able to estimate the number of tags with arbitrarily required accuracy and confidence level. In particular, PET achieves $\mathcal{O}(\log \log n)$ processing efficiency, to the total number of RFID tags, which significantly improves the state-of-the-art performance bound of RFID estimation. With the estimating path dynamics, PET shifts the computational burden from RFID tags to the reader, which allows us to use much less costly but resource-limited passive RFID tags in more scalable applications. We do extensive simulations to evaluate the performance of PET. The simulation results suggest that PET outperforms two most recent estimation approaches FNEB and LoF in the sense that PET achieves the same accuracy requirement with much less processing time. The computational overhead of PET is also much smaller at RFID tags. PET is effective in dealing with multiple readers and mobile tag environment as well.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive feedback and valuable comments for improving the quality of this paper. They acknowledge the support of COE SUG M58020025, NAP SUG M58020033 from the Nanyang Technological University, and NRF through EWI on project 1002-IRIS-09.

References

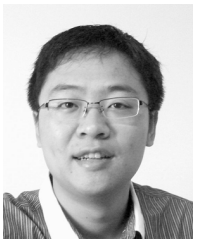
[1] "EPC Radio-Frequency Identity Protocols Class-1 Gen-2 UHF RFID Protocol for Communications at 860MHz-960MHz, EPCglobal," http://www.epcglobalinc.org/standards/uhfc1g2, Apr. 2011.

[2] M. Abramowitz and I.A. Stegun, *Handbook of Mathematical Functions with Formulas Graphs, and Mathematical Tables.* Dover, 1972.

[3] J.I. Capetanakis, "Tree Algorithms for Packet Broadcast Channels," *IEEE Trans. Information Theory*, vol. 25, no. 5, pp. 505-515, Sept. 1979.

[4] J. Considine, F. Li, G. Kollios, and J. Byers, "Approximate Aggregation Techniques for Sensor Databases," *Proc. IEEE 20th Int'l Conf. Data Eng. (ICDE '04)*, 2004.

[5] M. Durand and P. Flajolet, "Loglog Counting of Large Cardinalities (Extended Abstract)," *Proc. 11th Ann. European Symp. Algorithms (ESA),* 2003.
[6] K. Finkenzeller, *RFID Handbook: Radio-Frequency Identification Fundamentals and Applications.* John Wiley & Sons, 2000.
[7] P. Flajolet, X. Gourdon, and P. Dumas, "Mellin Transforms and Asymptotics: Harmonic Sums," *Theoretical Computer Science,* vol. 144, nos. 1/2, pp. 3-58, 1995.
[8] P. Flajolet and G.N. Martin, "Probabilistic Counting Algorithms for Data Base Applications," *J. Computer and System Science,* vol. 31, no. 2, pp. 182-209, 1985.
[9] S. Ganguly, M. Garofalakis, and R. Rastogi, "Processing Set Expressions over Continuous Update Streams," *Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '03),* 2003.
[10] G.R. Grimmett and D.R. Stirzaker, *Probability and Random Processes,* third ed. Oxford Univ., 2001.
[11] P.J. Haas, J.F. Naughton, S. Seshadri, and L. Stokes, "Sampling-Based Estimation of the Number of Distinct Values of an Attribute," *Proc. 21st Int'l Conf. Very Large Data Bases (VLDB '95),* 1995.
[12] H. Han, B. Sheng, C.C. Tan, Q. Li, W. Mao, and S. Lu, "Counting RFID Tags Efficiently and Anonymously," *Proc. IEEE INFOCOM,* 2010.
[13] P. Kirschenhofer and H. Prodinger, "On the Analysis of Probabilistic Counting," *Lecture Notes in Math.,* vol. 1452, pp. 117-120, 1990.
[14] D.E. Knuth, *The Art of Computer Programming: Sorting and Searching,* second ed., vol. 3, Addison-Wesley, 1998.
[15] M. Kodialam and T. Nandagopal, "Fast and Reliable Estimation Schemes in RFID Systems," *Proc. ACM MobiCom,* 2006.
[16] M. Kodialam, T. Nandagopal, and W.C. Lau, "Anonymous Tracking Using RFID Tags," *Proc. IEEE INFOCOM,* 2007.
[17] C.-H. Lee and C.-W. Chung, "Efficient Storage Scheme and Query Processing for Supply Chain Management Using RFID," *Proc. ACM SIGMOD Int'l Conf. Management of Data,* 2008.
[18] J. Lee, T. Kwon, Y. Choi, S.K. Das, and K.-a. Kim, "Analysis of RFID Anti-Collision Algorithms Using Smart Antennas," *Proc. Second Int'l Conf. Embedded Networked Sensor Systems (SenSys '04),* 2004.
[19] Y. Liu, L. Chen, J. Pei, Q. Chen, and Y. Zhao, "Mining Frequent Trajectory Patterns for Activity Monitoring Using Radio Frequency Tag Arrays," *Proc. IEEE Fifth Ann. Int'l Conf. Pervasive Computing and Comm. (PerCom '07),* 2007.
[20] L. Lu, Y. Liu, X. Li, "Refresh: Weak Privacy Model for RFID Systems," *Proc. IEEE INFOCOM,* 2010.
[21] J. Myung and W. Lee, "Adaptive Splitting Protocols for RFID Tag Collision Arbitration," *Proc. ACM MobiHoc,* 2006.
[22] V. Namboodiri and L. Gao, "Energy-Aware Tag Anti-Collision Protocols for RFID Systems," *Proc. IEEE Fifth Ann. Int'l Conf. Pervasive Computing and Comm. (PerCom '07),* 2007.
[23] L.M. Ni, Y. Liu, Y.C. Lau, and A. Patil, "LANDMARC: Indoor Location Sensing Using Active RFID," *Proc. IEEE First Int'l Conf. Pervasive Computing and Comm. (PerCom '03),* 2003.
[24] C. Qian, H. Ngan, and Y. Liu, "Cardinality Estimation for Large-Scale RFID Systems," *Proc. IEEE Sixth Ann. Int'l Conf. Pervasive Computing and Comm. (PerCom '08),* 2008.
[25] C. Qian, Y. Liu, H. Ngan, and L.M. Ni, "ASAP: Scalable Identification and Counting for Contactless RFID Systems," *Proc. IEEE 30th Int'l Conf. Distributed Computing Systems (ICDCS '10),* 2010.
[26] L.G. Roberts, "Aloha Packet System with and without Slots and Capture," *ACM SIGCOMM Computer Comm. Rev.,* vol. 5, no. 2, pp. 28-42, 1975.
[27] T.A. Scharfeld, "An Analysis of the Fundamental Constraints on Low Cost Passive Radio-Frequency Identification System Design," M.S. thesis, Massachusetts Inst. of Technology, 2001.
[28] B. Sheng, Q. Li, W. Mao, "Efficient Continuous Scanning in RFID Systems," *Proc. IEEE INFOCOM,* 2010.
[29] C.C. Tan, B. Sheng, and Q. Li, "Serverless Search and Authentication Protocols for RFID," *Proc. IEEE Int'l Conf. Pervasive Computing and Comm.,* 2007.
[30] C.C. Tan, B. Sheng, and Q. Li, "How to Monitor for Missing RFID Tags," *Proc. 28th Int'l Conf. Distributed Computing Systems (ICDCS '08),* 2008.
[31] H. Tijms, *Understanding Probability: Chance Rules in Every Life.* Cambridge Univ., 2004.
[32] R. Want, "An Introduction to RFID Technology," *IEEE Pervasive Computing,* vol. 5, no. 1, pp. 25-33, Jan.-Mar. 2005.
[33] L. Xie, B. Sheng, C.C. Tan, H. Han, Q. Li, and D. Chen, "Efficient Tag Identification in Mobile RFID Systems," *Proc. IEEE INFOCOM,* 2010.
[34] X. Xu, L. Gu, J. Wang, and G. Xing, "Negotiate Power and Performance in the Reality of RFID Systems," *Proc. IEEE Int'l Conf. Pervasive Computing and Comm. (PerCom),* 2010.
[35] L. Yang, J. Han, Y. Qi, and Y. Liu, "Identification-Free Batch Authentication for RFID Tags," *Proc. IEEE 18th Int'l Conf. Network Protocol (ICNP),* 2010.
[36] D. Zhang, J. Ma, Q. Chen, and L.M. Ni, "An RF-Based System for Tracking Transceiver-Free Objects," *Proc. IEEE Fifth Ann. Int'l Conf. Pervasive Computing and Comm. (PerCom '07),* 2007.
[37] R. Zhang, Y. Liu, Y. Zhang, and J. Sun, "Fast Identification of the Missing Tags in a Large RFID System," *Proc. IEEE Eighth Ann. Comm. Soc. Conf. Sensor, Mesh, and Ad Hoc Comm. and Networks (SECON),* 2011.
[38] 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," *Proc. Int'l Symp. Low Power Electronics and Design (ISLPED),* 2004.



Yuanqing Zheng received the BS degree in electrical engineering and the ME degree in communication and information systems from Beijing Normal University, China, in 2007 and 2010, respectively. He is currently working toward the PhD degree in the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include computer networking, distributed systems, and pervasive computing. He is a student member of the IEEE.

Mo Li received the BS degree from the Department of Computer Science and Technology, Tsinghua University, China, in 2004 and the PhD degree from the Department of Computer Science and Engineering, Hong Kong University of Science and Technology in 2009. He is currently an assistant professor in the School of Computer Engineering at Nanyang Technological University, Singapore. He won the ACM Hong Kong Chapter Professpr Francis Chin Research Award in 2009 and Hong Kong ICT Award Best Innovation and Research Grand Award in 2007. His research interests include wireless sensor networking, pervasive computing, and mobile and wireless computing. He is a member of the IEEE and the ACM.