PLACE: Physical Layer Cardinality Estimation for Large-Scale RFID Systems

Yuxiao Hou, *Student Member, IEEE*, Jiajue Ou, *Student Member, IEEE, ACM*, Yuanqing Zheng, *Member, IEEE, ACM*, and Mo Li, *Member, IEEE, ACM*

***Abstract*—Estimating the number of RFID tags is a fundamental operation in RFID systems and has recently attracted wide attentions. Despite the subtleties in their designs, previous methods estimate the tag cardinality from the slot measurements, which distinguish idle and busy slots and based on that derive the cardinality following some probability models. In order to fundamentally improve the counting efficiency, in this paper we introduce PLACE, a physical layer based cardinality estimator. We show that it is possible to extract more information and infer integer states from the same slots in RFID communications. We propose a joint estimator that optimally combines multiple sub-estimators, each of which independently counts the number of tags with different inferred PHY states. Extensive experiments based on the GNURadio/USRP platform and the large-scale simulations demonstrate that PLACE achieves approximately $3 \sim 4\times$ performance improvement over state-of-the-art cardinality estimation approaches.**

***Index Terms*—RFID, cardinality estimation, physical layer.**

I. Introduction

RADIO FREQUENCY IDENTIFICATION (RFID) technologies [21] have been developing rapidly in recent years. Due to the low cost and small form factor of RFID tags, RFID technology is widely used to label a large number of items and support inventory management [26], item tracking [16], access control [3], etc. In this paper, we mainly focus on passive or semi-passive RFID tags that backscatter reader interrogation signals for communications.

Counting the number of tags is a fundamental operation. Knowing the tag cardinaltiy can facilitate many primary operations in RFID systems such as tag identification [27] and tag searching [29]. An estimation with guaranteed accuracy normally suffices for the practical purposes. As such, many probabilistic counting methods [4], [8], [11], [12], [17], [18], [28], [31] trade the estimation accuracy for the execution time. Previous works typically measure the states of f communication slots, where each tag responds in one random slot. The slot state can be binary if we distinguish busy and idle slots, or it can be ternary if we further differentiate singleton and collision slots. Thus, the tag responses in f slots can be represented with a $f \times 1$ binary or ternary sequence, with zeros representing idle slots. Intuitively, when a larger number of tags participate, we expect more tag responses and consequently fewer idle slots in the response sequence. Despite the subtleties in design details, previous methods estimate the tag cardinality by examining the state of each slot in the response frame and following a probability model to derive the cardinality.

For instance, one previous work EFNEB [8] uses the first busy slot to estimate, while ZOE [31] computes the ratio of zero entries over f slots and derives the tag cardinality. The most recent work [4] advocates the importance of two-phase estimation, and approaches theoretical optimal performance with the binary responses. As only 1 bit or slightly more information is extracted from each slot, previous methods need substantial number of slot measurements to guarantee an estimation accuracy.

In this paper, we present PLACE, a Physical LAyer Cardinality Estimation scheme which extracts more information from each tag response slot, thereby achieving higher estimation efficiency. Unlike previous methods which only distinguish binary or ternary states in each slot, we show that it is possible to detect the number of concurrent tag responses and thus infer integer states from the same slot at RFID physical layer.

To illustrate the possibility of detecting integer slot states, Fig. 1 plots the received signals of real data traces when 0, 1, 2, and 3 tags respond in the same slot: Fig. 1(a)–(d) depict amptitude signals in the time domain and Fig. 1(e)–(h) are the scatter plots of the received physical symbols in quadrature and in-phase components on the I-Q plane. Due to certain modulation schemes, the received signal in the I-Q plane will form certain number of clusters, each of which represents one modulation state. Such a set of clusters is called the constellation map of the received signal. These traces are collected through our measurement testbed including the GNURadio/USRP2 platform and WISP tags (testbed settings detailed in Section II).

In Fig. 1, we find that while straightforward measurement of signal strength levels in time domain can only tell busy or idle state of the slot, we observe from Fig. 1(e)–(h) that if k tags reply at the same time, 2^k symbol clusters are clearly formed in the corresponding constellation map. This is because each tag takes one of the two states by either reflecting or absorbing

Manuscript received January 12, 2015; revised July 04, 2015; accepted September 01, 2015; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor D. Goeckel. Date of publication October 14, 2015; date of current version October 13, 2016. This work was supported by the Singapore MOE AcRF grant MOE2012-T2-1-070, MOE2013-T1-002-005, and NTU NAP grant M4080738.020. A preliminary version of this work was published in the Proceedings of IEEE INFOCOM 2015.

Y. Hou, J. Ou, and M. Li are with the School of Computer Engineering, Nanyang Technological University, Singapore 639798, Singapore (e-mail: s120003@e.ntu.edu.sg; jou1@e.ntu.edu.sg; limo@e.ntu.edu.sg).

Y. Zheng is with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Hong Kong (e-mail: csyqzheng@comp.polyu.edu.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2015.2481999

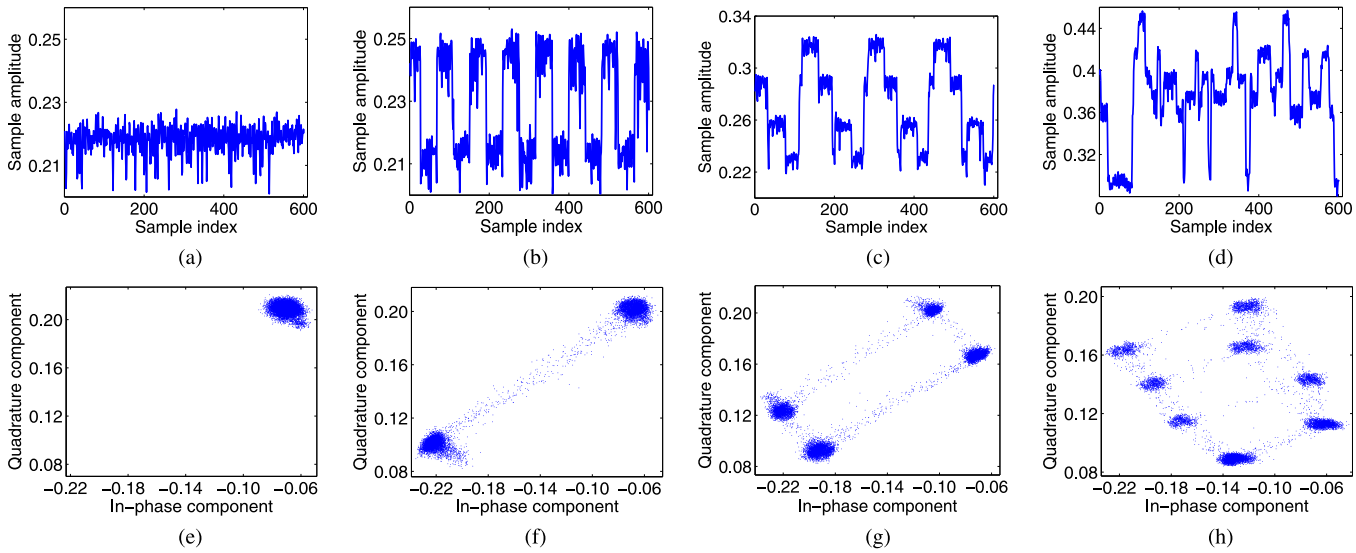


Fig. 1. Tag response signals and their corresponding constellation maps. (a) RN-16 signal of noise; (b) RN-16 signal of 1 tag; (c) RN-16 signal of 2 tags; (d) RN-16 signal of 3 tags; (e) noise, 1 cluster; (f) 1 tag, 2 clusters; (g) 2 tags, 4 clusters; (h) 3 tags, 8 clusters.

radio waves from the RFID reader. Such observation inspires us to detect the exact number of concurrent tag responses in each slot. Ideally, we can infer the number of responding tags from the number of clusters formed, and thereby extend the binary or ternary sequence to an integer sequence.

Although simple in concept, the implementation of physical layer estimation entails many practical challenges. (1) Accurate and efficient estimation of the symbol clusters is non-trivial. In particular, the symbol clustering and counting operation has to be accommodated into the time frame of each RFID slot. In this paper, we design a slot state detection algorithm that divides the I-Q plane into grids and derives the symbol clusters based on the symbol densities of grids. The proposed SSDA algorithm takes only millisecond time in comparison with general clustering algorithms that may take hundreds of seconds. (2) Novel cardinality estimator needs to be designed to make the best use of sequences of integer-state transmission slots instead of previous busy/idle slots. The proposed PLACE scheme combines multiple estimations obtained from each integer slot state and uses an optimal joint estimator such that the overall variance is minimized. (3) Due to noises in practical RFID transmissions, the cluster estimation output inherently contains errors. We run full experiments to understand such errors and analyze the impact on final cardinality estimation accuracy. Our analysis and experiments show that the developed probabilistic estimators in PLACE tolerate the error level from practical measurements.

PLACE is comprehensively evaluated on our testbed built with the GNURadio/USRP platform and WISP tags. We perform large-scale simulations to compare with state-of-the-art cardinality estimation schemes. Experiment and simulation results demonstrate that PLACE achieves approximately $3 \sim 4\times$ performance improvement.

The rest is organized as follows. Section II introduces the background of RFID system and explains our initial observation from the software defined testbed. Section III presents our slot state detection algorithm. Section IV describes mathematically how PLACE utilizes multiple states of slots to achieve high RFID counting efficiency. Section V analyzes how received signal SNR impacts the slot state detection accuracy and suggests an error compensation scheme to enhance the cardinality counting performance. Section VI provides experiments and simulations to evaluate PLACE. Section VII overviews the related works. Section VIII concludes this paper.

II. Background

A. Problem Description

Following previous works [4], [8], [11], [12], [17], [18], [26]–[29], [31], we consider a large-scale RFID system consisting of a number of RFID tags covered by one RFID reader. The RFID systems may use lightweight passive RFID tags or powerful active tags.

We exclusively study the RFID communications working at the 900 MHz UHF band. Current commodity RFID systems adopt the frame-slotted Aloha model, where a frame is divided into a number of slots. Each of RFID tags randomly chooses one slot in the frame to reply. As a result, one slot might be idle, if no tag responds in the slot; or busy, if at least one tag responds. Instead of replying with a 96-bit tag ID [1], which is used in identification-based counting method [27], each tag only needs to reply with a RN16 sequence in probabilistic cardinality estimation approaches.

Suppose the actual tag cardinality is t, and our estimation is $\hat{t}$. A user-specified accuracy requirement (ϵ, δ) can be specified as follows:

$$Pr\left\{|\hat{t} - t| \leq \epsilon t\right\} \geq 1 - \delta. \tag{1}$$

For instance, if the actual number of tags is 1000 and a user specifies the requirement as (5%,1%), then the estimation result is expected to be within the interval [950,1050] with a probability $\geq$99%. An ideal estimation approach is expected to meet the estimation accuracy with the minimum execution time.

Many research efforts have been devoted to improve the cardinality estimation efficiency [4], [8], [11], [12], [17], [18],

Fig. 2. Our testbed with a GNURadio/USRP2 platform and 4 WISP tags.

[28], [31]. Despite the differences in design details, these works estimate the tag cardinality by measuring the slot states and differentiating idle and busy slots, where each tag randomly selects a slot and sends a short message. For instance, EFNEB [8] infers the tag cardinality from the position of the first busy slot. ZOE [31] computes the ratio of idle and busy slots and thereby derives the tag cardinality. ART [18] measures the average run of busy slots to estimate the tag cardinality. Above approaches [8], [18], [31] adopt the two-phase estimation, where in the first phase the system parameters are optimized to ensure high estimation efficiency in the second phase. One most recent work [4] gives an in-depth analysis and explicitly emphasizes the importance of the two-phase design. To the best of our knowledge, all existing works do not leverage the RFID physical layer information. They only extract binary information from each short slot in the frame.

B. Initial Observation From Our Software Defined Testbed

To explore the possibility of cardinality estimation with PHY layer information of RFID transmissions, we set up a testbed with the GNURadio/USRP software defined radio and the WISP tags as depicted in Fig. 2. We use one USRP RFX900 daughterboard working at the 900 MHz UHF band to down-convert the radio signals to the base band. After the down-conversion, the physical layer symbols are transferred to a laptop via a gigabit ethernet link for digital processing. The physical layer sampling rate of the software defined RFID reader is set to 4 million samples per second (MS/s). Thus, the software reader samples 4000 physical symbols every 1 ms. At the physical layer, the USRP reader can retrieve the in-phase and quadrature components of each received symbol, which corresponds to a sample at the I-Q plane.

For each RN16 transmission, a commodity tag needs to send a preamble prior to the RN16 payload. The transmission time of an RN16 varies from 0.02 ms to 8 ms, depending on the backscatter link frequency (BLF) as well as the coding scheme (e.g., FM0, Miller-4) [1].

In our software testbed, the WISP tags are programmed to encode the RN16 messages with Miller-4 and backscatter at 64 kbps, which takes around 2 ms.

We collect more than 500 physical layer traces when different number of tags concurrently transmit RN16 messages. In Fig. 1, we present 4 instances of received RN16 slots with different number of responding tags. In Fig. 1(a)–(d), for illustration purposes, we intercept the first 600 samples (corresponding to preambles of RN16) of the approximately 8000 samples of each trace. Fig. 1(a)–(d) plot the magnitudes of received symbols. Fig. 1(a) measures the background noise when no tag transmits in the slot. When one tag backscatters its RN16 by reflecting or absorbing radio signals, as shown in Fig. 1(b), the received signal strength at the reader may vary depending on the message content. Current commodity readers set an empirical magnitude threshold to decode the backscattered message. When 2 tags transmit simultaneously, we observe the tag collisions as in Fig. 1(c). In such a case, commodity readers cannot reliably decode the tag collisions, since the threshold based method no longer works. We cannot differentiate the number of colliding tags when more than 2 tags transmit together as in Fig. 1(c) and Fig. 1(d), by solely examining the magnitude of received signals.

When we examine the physical symbols in the I-Q plane as depicted in Fig. 1(e)–(h), however, we see that the symbols exhibit distinct clustering patterns, depending on the number of colliding tags. Fig. 1(e) plots the physical layer symbols that are measured when no tag transmits. If there is no noise, all physical layer symbols overlap at one point in the I-Q plane. In practice, due to background noise (which generally follows Gaussian distribution [32]–[34], the symbols spread around and form a cluster as shown in Fig. 1(e). When 1 tag backscatters alone, 2 clusters emerge in the I-Q plane as in Fig. 1(f). Each cluster represents one possible transmission state, i.e., idle or backscattering. We notice that a few samples locate in a narrow band between the two clusters, because the tag takes very short time to transit between the two transmission states. When 2 tags transmit simultaneously, we find that the I-Q plane contains 4 clusters as in Fig. 1(g). This is because we have 4 possible transmission states when 2 tags transmit simultaneously. In Fig. 1(h), we see that the number of clusters doubles as one more tag joins in the transmission. Comparing Fig. 1(g), (h) with Fig. 1(c), (d), we see that the clustering pattern of physical symbols in the I-Q plane contains substantially richer information that allows us to derive the number of colliding tags in each slot.

The theoretical explanation for above observation is as follows. For passive RFID tags, tags transmit their signals by backscattering the readers continuous wave and use On-Off Keying (OOK) modulation scheme. In OOK, each tag has two signal states (1) and (0). When multiple tags reply in the same slot, their responses will add up together and their modulation states can combine in all possible ways. Therefore, when k tags reply concurrently, 2^k signal states exist and 2^k clusters appear in the constellation map. Although our observation is based on GNURadio/USRP2 platform and WISP tags, we believe that similar observation can also be spotted with commercial RFID readers and tags because of the principle in OOK modulation scheme.

In practice, many factors affect the obtained constellation map of the backscattered signal. The major factor is the number of replying tags k that determines the number of physical symbol clusters. Other factors that affect the constellation map include tag type, placement, product providers, etc., which mainly affect the shape of the constellation map (e.g., cluster sizes, positions, the distances in between) but not the number

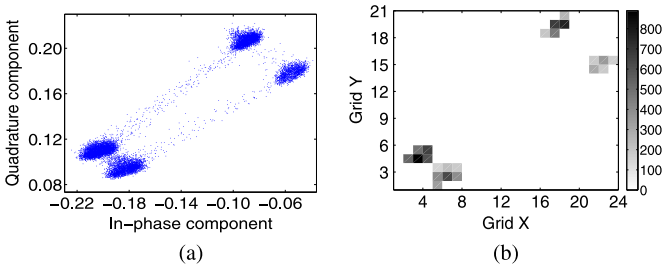


Fig. 3. An illustrative example of Slot State Detection Algorithm (SSDA). (a) Physical layer symbols. (b) Filtered grid density matrix.

of clusters. In our experimental evaluation in Section VI, we test data traces with different tag placements and characterize other factors with different SNRs of tag signals.

III. Slot State Detection Algorithm

From the initial experiments, we see that it is possible to infer the number of colliding tags by clustering the physical symbols in the I-Q plane. Traditional clustering algorithms, however, are inadequate to serve our purpose for at least two reasons. First, many clustering algorithms (e.g., k-means [15]) require a priori knowledge of the number of clusters. Obviously, such algorithms cannot be directly used since the number of clusters is exactly the unknown that we need to derive. Second, although some clustering algorithms (e.g., DBSCAN [6]) do not require the priori knowledge of the cluster number, they typically incur high computational overhead. In particular, DBSCAN incurs a computation overhead of $O(l^2)$, where l denotes the number of input samples. In our system, we need to cluster thousands of symbols in at most 2 ms, i.e., the RN16 slot length.

In this section, we propose a slot state detection algorithm (SSDA) to efficiently process the physical layer symbols and accurately measure the number of colliding tags in the slot. The proposed SSDA method only incurs a computation overhead of $O(l)$.

Intuitively, SSDA leverages the fact that samples in one cluster follow a 2-D Gaussian distribution due to channel noise [32]–[34], and consequently we expect a peak grid for each cluster. Thus, we measure the density of samples in each small grid and count the number of clusters by searching the local maximum in the I-Q plane. The input to SSDA is the physical symbols sampled in one slot. The output of SSDA is the number of responding tags in the slot. The whole process of SSDA contains the following three steps:

Step 1: Calculate the Sample Density in the I-Q Plane: We first find the min and max values of both in-phase and quadrature components over all the physical symbols, represented with complex numbers. We then divide the rectangular area into small grids, whose size is set to 0.01×0.01. We calculate the symbol density of each grid by counting the number of symbols within the corresponding grid.

Step 2: Filter Out the Noise Grids: We set an empirical threshold to differentiate all grids into signal grids and noise grids. If the density of a grid is above the threshold, the grid is considered as a signal grid; otherwise, we filter out the noise grid. Fig. 3 depicts how the raw samples of physical symbols in Fig. 3(a) are filtered to obtain the density matrix in Fig. 3(b).

A constant threshold cannot work well since the grid densities may vary, depending on the number of samples as well as the number of clusters that are formed in the I-Q plane. Thus, we propose to use a percentage threshold PT as follows. Suppose there are l data samples in one slot. We set the grid density threshold to be $PT \times l$ for a percentage threshold of PT.

Step 3: Calculate the Number of Responding Tags: We count the number of clusters C by counting the number of the local maximums of sample density in the I-Q plane. In principle, if k tags collide together in a slot, we should observe $C = 2^k$ clusters. Due to noise, we have $C \leq 2^k$ in practice. Thus, we compute the number of responding tags k as $\lceil \log_2 C \rceil$.

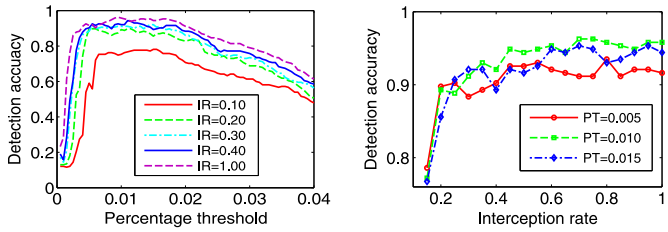
In the following, we study the impact of two key system parameters in SSDA—the percentage threshold, and the number of physical samples collected in one slot. The empirical percentage threshold of SSDA influences detection accuracy in practice: a high threshold may miss the cluster peaks, while a low threshold cannot adequately filter out noise grids.

The number of samples collected in one slot is determined by the slot duration and the sampling rate. As the C1G2 standard supports different combinations of backscatter link frequencies and modulation schemes, we can reduce the slot duration with higher link frequencies and coding rates, so as to reduce the transmission time and the computation overhead involved in SSDA.

We carry out trace-driven evaluations to study the influence of the two system parameters. We sample the physical symbols at 4 MS/s on our testbed and the slot duration is 2 ms. We intercept varied portions of the symbols as input to SSDA. We use an interception rate IR to represent the intercepted portion, e.g., an interception rate IR of 10% means only the first 10% of samples are processed by SSDA. We program the WISP tags and let different number of tags concurrently send RN16 messages in each slot. We record the number of responding tags as the ground truth and measure the detection accuracy. The accuracy is defined as the ratio of correctly detected slots to the number of tested slots.

Fig. 4(a) plots the detection accuracy with the varied percentage threshold ranging from 1% to 4%. We measure the detection accuracy with 5 different interception rates. In the figure, we find that a small percentage threshold (e.g., $<$0.5%) leads to low detection accuracies, because noise grids cannot be filtered out. With the same interception rate, a percentage threshold within [0.5%,1.5%] consistently achieves high detection accuracies. More importantly, we find that the detection accuracy is less sensitive to the change of the percentage threshold within [0.5%,1.5%]. Once the percentage threshold exceeds 2.0%, the detection accuracy decreases as the threshold increases. This is because with a higher threshold, some peak grids with relatively lower grid densities would be accidentally filtered out. Thus, we set the percentage threshold PT to 1%.

We study the impact of interception rates on the detection accuracy. Fig. 4(b) plots the detection accuracy with different interception rates. In the figure, we see that with $PT = 1\%$ and $IR = 100\%$, SSDA can achieve the detection accuracy of above 95%. Moreover, the detection accuracy remains above 95% as long as the interception rate is larger than 50%. In other words, SSDA only needs half of the physical symbols sampled



(a) (b)

Fig. 4. Detection accuracies of SSDA with different system parameters. (a) Detection accuracy with different percentage thresholds (PT). (b) Detection accuracy with different interception rates (IR).

within one slot, to accurately count the number of responding tags. As long as the interception rate is higher than 30%, our detection algorithm can achieve 90% detection accuracy. The experiment results imply that we can potentially reduce the slot duration to further reduce the transmission time.

IV. Estimation Algorithm

As we can differentiate multiple slot states with SSDA, we can devise several estimators for different slot states. For instance, we can estimate the tag cardinality with the fraction of singleton slots, double-tag-collision slots, triple-tag-collision slots, etc. Finally, we design an optimal joint estimator by combining estimations from these subestimators so that the overall variance is minimized.

A. Estimation Protocol

In each slot, each of t tags generates a random integer r using a uniform hash function. We denote the index of the right-most zero in the binary representation of r as R. As in the previous schemes [17], [31], a tag will respond if $R \geq \theta$, where θ is a parameter specified by the reader. Therefore, the probability p that a tag will respond in a slot is as follows

$$p = 2^{-\theta}. \tag{2}$$

Suppose X_k $(k = 0, 1, 2, \ldots)$ is defined as an indicator of k tag responses in a slot, i.e., $X_k = 1$, if k tags are in the slot; $X_k = 0$, otherwise.

For each k, X_k follows the Bernoulli distribution, and the probability of observing k responses in a slot is

$$Pr\{X_k = 1\} = \binom{t}{k} p^k (1-p)^{t-k} \approx \frac{\lambda^k}{k!} e^{-\lambda}, \tag{3}$$

where $\lambda = pt$ is the load factor.

Thus, the expectation $E[X_k]$ and variance $\sigma^2_{X_k}$ are

$$E[X_k] = \frac{\lambda^k}{k!} e^{-\lambda},\ \sigma^2_{X_k} = \frac{\lambda^k}{k!} e^{-\lambda} \left(1 - \frac{\lambda^k}{k!} e^{-\lambda}\right). \tag{4}$$

We define $\bar{X_k} = \frac{1}{m} \sum_{i=1}^{m} X_{k,i}$ as the arithmetic average of m observations. Then, the expectation and variance of $\bar{X_k}$, denoted as $E[\bar{X_k}]$ and $\sigma^2_{\bar{X_k}}$, are as follows

$$E[\bar{X_k}] = E[X_k] = \frac{\lambda^k}{k!} e^{-\lambda},\ \sigma^2_{\bar{X_k}} = \frac{1}{m} \sigma^2_{X_k} \tag{5}$$

Since different k values will produce different estimations of t with different variances, we give the following theorem which provides the optimal combination of multiple sub-estimators.

Theorem 1: Suppose $\hat{t_0}, \hat{t_1}, \ldots, \hat{t_k}$ are $k+1$ estimations for t with variances $\sigma_0^2, \sigma_1^2, \ldots, \sigma_k^2$, respectively. For the weighting scheme $\sum_{i=0}^{k} w_i = 1$ and $0 \leq w_i \leq 1$, the joint estimator $\hat{t} = \sum_{i=0}^{k} w_i \hat{t_i}$ has a variance of $\sigma_{\hat{t}}^2 = \sum_{i=0}^{k} w_i^2 \sigma_i^2$. The optimal weights w_i $(i = 0, 1, \ldots, k)$ for each sub-estimator that minimizes $\sigma_{\hat{t}}^2$ is

$$w_i^* = \frac{\frac{1}{\sigma_i^2}}{\sum_{j=0}^{k} 1/\sigma_j^2},\ i = 0, 1, \ldots, k, \tag{6}$$

and the minimum variance is

$$\sigma^2_{\hat{t},min} = \frac{1}{\sum_{i=0}^{k} 1/\sigma_i^2}. \tag{7}$$

Proof: To minimize $\sigma_{\hat{t}}^2$, we define the following Lagrange multiplier

$$L(w_0, w_1, \ldots, w_k, \beta) = \sum_{i=0}^{k} w_i^2 \sigma_i^2 + \beta \left(\sum_{i=0}^{k} w_i - 1 \right),$$

where the term $\sum_{i=0}^{k} w_i - 1$ incorporates the weight constraint $\sum_{i=0}^{k} w_i = 1$.

We let the partial derivatives of L over w_i $(i = 0, 1, \ldots, k)$ and β be 0. Thus, we have

$$\begin{cases} \frac{\partial L}{\partial w_i} = 2 w_i \sigma_i^2 + \beta = 0,\ i = 0, 1, \ldots, k \\ \frac{\partial L}{\partial \beta} = \sum_{i=0}^{k} w_i - 1 = 0. \end{cases}$$

We solve the equations as follows

$$\begin{cases} w_i^* = \frac{\frac{1}{\sigma_i^2}}{\sum_{j=0}^{k} 1/\sigma_j^2},\ i = 0, 1, \ldots, k. \\ \beta^* = -\frac{2}{\sum_{i=0}^{k} 1/\sigma_i^2}. \end{cases}$$

Thus, we have the minimum variance of $\hat{t}$ as follows

$$\sigma^2_{\hat{t},min} = \sum_{i=0}^{k} w_i^{*2} \sigma_i^2 = \frac{1}{\sum_{i=0}^{k} 1/\sigma_i^2}.$$

□

B. Computing the Number of Rounds m

In practice, m estimation rounds have to be repeated to further reduce $\sigma^2_{\hat{t},min}$ and meet the requirement in (1). In the following, we analyze the minimum value of m.

Let $Y = \frac{\hat{t} - t}{\sigma_{\hat{t},min}}$. Since t is often a large number, according to the law of large number, Y follows the standard normal distribution. Thus, we can derive

$$Pr\left\{ -\frac{\epsilon t}{\sigma_{\hat{t},min}} \leq Y \leq \frac{\epsilon t}{\sigma_{\hat{t},min}} \right\} \geq 1 - \delta. \tag{8}$$

Eq. (8) is equivalent to

$$\frac{\epsilon t}{\sigma_{\hat{t},min}} \geq c, \tag{9}$$

where c meets the following condition

$$1-\delta = erf\left(\frac{c}{\sqrt{2}}\right), \tag{10}$$

and $erf()$ represents the Gaussian error function.

Since $\sigma^2_{\hat{t},min}$ is a function of m, we first compute $\sigma^2_{\hat{t},min}$. According to (7), $\sigma^2_{\hat{t},min}$ is a function of σ^2_k. Because $\hat{t_k}$ is a function of $\bar{X}_k$, the relationship between σ^2_k and $\sigma^2_{\hat{X}_k}$ can be explicitly expressed.

Suppose $\hat{t_k} = f_k(\bar{X}_k)$ is expressed with the Taylor expansion centered on $E[\bar{X}_k]$:

$$f_k(\bar{X}_k) - f_k\left(E[\bar{X}_k]\right) \approx f'_k\left(E[\bar{X}_k]\right)\left(\bar{X}_k - E[\bar{X}_k]\right). \tag{11}$$

Since we have $f_k(E[\bar{X}_k]) = E[f_k(\bar{X}_k)]$, we derive from (11) that

$$Var\left[f_k(\bar{X}_k)\right] \approx \left\{f'_k\left(E[\bar{X}_k]\right)\right\}^2 Var[\bar{X}_k]. \tag{12}$$

We represent (12) as follows

$$\sigma^2_k = \alpha^2_k \sigma^2_{\bar{X}_k}, \tag{13}$$

where

$$\alpha_k = \left.\frac{dt_k}{d\bar{X}_k}\right|_{E[\bar{X}_k]} = \left.\frac{1}{\frac{d\bar{X}_k}{dt_k}}\right|_{E[\bar{X}_k]} = \frac{k!te^{\lambda}}{\lambda^k(k-\lambda)}. \tag{14}$$

Combining (4), (5), (7), (13), and (14), we obtain the expression of $\sigma^2_{\hat{t},min}$ as follows:

$$\sigma^2_{\hat{t},min} = \frac{t^2}{mg(\lambda)}, \tag{15}$$

where $g(\lambda)$ is

$$g(\lambda) = \sum_{i=0}^{k} \frac{(i-\lambda)^2}{i!e^{\lambda}/\lambda^i - 1} \tag{16}$$

After obtaining $\sigma^2_{\hat{t},min}$, we can derive the number of independent measurements m to meet the accuracy requirement by substituting (15) into (9):

$$m \geq \frac{c^2}{\epsilon^2 g(\lambda)}. \tag{17}$$

We see that $g(\lambda)$ depends on k which is the maximum number of detectable colliding tags in one slot. Fig. 5 measures m with different k to meet different (ϵ, δ)-accuracy requirements. We fix ϵ to 0.01 and specify δ to 1%, 10%, and 20%, respectively. In the figure, we find that regardless of the accuracy requirement, PLACE needs to perform fewer rounds of estimation with the increased k. This is because PLACE is able to augment the joint estimator with more independent sub-estimators. Nevertheless, the marginal gain of detecting more tags in each slot gradually decreases as k increases. As the slot state detection accuracy decreases when more tags collide together, we combine 4 sub-estimators in practice.

We further tune λ to maximize $g(\lambda)$ and minimize m. In order to find the optimal λ^* to maximize $g(\lambda)$ and minimize m, we plot $g(\lambda)$ against λ in Fig. 6. We see that $g(\lambda)$ reaches the maximum value with $\lambda^* \approx 5.2$.

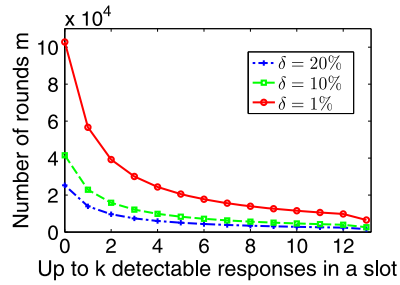


Fig. 5. The number of rounds m over maximum detectable responses k.

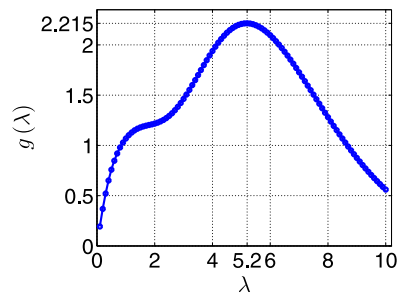


Fig. 6. The function $g(\lambda)$ defined in (16) over the load factor λ.

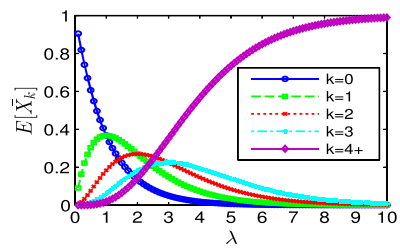


Fig. 7. The expectation of $\bar{X}_k$ over load factor λ, under varying k.

C. Two-Phase Counting Algorithm

We adopt the two-phase estimation design [4]. In the first rough estimation phase, we adjust the threshold θ^* so that the load factor $\hat{\lambda}$ approaches 5.2; in the second phase, we repeat m independent estimation rounds with the optimal threshold θ^*. The weights that are necessary to compute the final estimation $\hat{t}$ are derived from $\hat{\lambda}$ obtained in the first phase.

In the first rough estimation phase, the reader issues a θ value and measures the fraction of each slot state, i.e., $\bar{X}_k$ $(k = 0, 1, 2, 3)$. We denote the fraction of slots with more than 3 concurrent responses as $\bar{X}_{4+}$. From (5), we have

$$E[\bar{X}_{4+}] = 1 - \sum_{k=0}^{3} E[\bar{X}_k] = 1 - e^{-\lambda}\sum_{k=0}^{3}\frac{\lambda^k}{k!}. \tag{18}$$

Fig. 7 plots $E[\bar{X}_k]$ $(k = 0, 1, 2, 3, 4+)$ with different λ. From this figure we observe that when λ is around 5.2, $E[\bar{X}_k]$ $(k = 0, 1, 2, 3)$ can be very small and hence cannot be accurately measured with a small number of slots (e.g., 32 slots). In contrast, $E[\bar{X}_{4+}]$ spans a relatively large range and can be a good indicator of λ. In addition, $E[\bar{X}_{4+}]$ monotonically increases with λ, which allows us to quickly converge to the optimal λ using the binary search method.

In particular, in each query round, we measure $E[\bar{X}_{4+}]$ and compute $\hat{\lambda}$ according to (18). If $\hat{\lambda}$ is smaller than 3, indicating a large θ value, we decrease θ in the next query round; if $\hat{\lambda}$ is larger than 7, indicating a small θ value, we increase θ in the next query round; once $\hat{\lambda}$ is in the range [3,7], we terminate the rough estimation phase and set θ^* and $\hat{\lambda}$ to the corresponding values in the last query round. We set the optimal range for $\hat{\lambda}$ as [3,7] and we can always find the θ^* to let $\hat{\lambda}$ fall into this range. Suppose in a certain query round we compute that $\hat{\lambda} > 7$ or $\hat{\lambda} < 3$, in the following round we can increase or decrease θ accordingly until $\hat{\lambda}$ falls into the range [3,7].

Based on the above rules, the reader adopts a binary search method for θ in the range [0,32] and starts a query round with $\theta = 16$. Since in practice $2^0 < t < 2^{32}$ almost always holds and $\lambda = t/2^\theta$, we can always find the optimal θ^* in [0,32] to let λ approaches λ^*, i.e., 5.2.

Algorithm 1 shows the pseudocode of PLACE algorithm for the reader without considering SSDA detection error. This algorithm includes two phases: the rough estimation phase (Line 1–24) and the second phase which performs tag set cardinality within high accuracy (Line 25–38). The goal of the first phase is to tune the optimal θ so that the corresponding λ, with a form of $t/2^\theta$, approaches the optimal λ^* (i.e., 5.2) at the reader side. We adopt a binary search method. In the beginning, the reader sets the inital value of θ to 16 (Line 1), which is the middle of the range [0,32]. Next the reader broadcasts a random seed s and θ to all tags, waits for tag replies in the coming 32 slots, and collect the values of $\bar{X}_k$ ($k = 0, 1, 2, 3$) (Line 2–13). Based on collected $\bar{X}_k$, $\bar{X}_{4+}$ can be deduced (Line 14), which in turn can be used to compute a $\hat{\lambda}$ according to (18) (Line 15). We check whether the obtained $\hat{\lambda}$ falls in the range [3,7], where λ^* lies. If $\hat{\lambda}$ is not in the range, the reader adjusts θ accordingly and restarts another query round with 32 slots to obtain another $\hat{\lambda}$ (Line 16–19). Otherwise the readers stops the first phase and starts the second phase (Line 20–22).

At the start of the second phase, the needed number of slots m is computed based on accuracy requirements and $\hat{\lambda}$ from the first phase (Line 24). Then the reader issues a query round with the updated θ and a random seed s, waits for tag replies in the following m slots, and collects $\bar{X}_k$ values (Line 25–35). For each $\bar{X}_k$, a $\hat{t}_k$ is computed based on (5) (Line 36). The weights w_k ($k = 0, 1, 2, 3$) are computed based on (5), (13) and (14) (Line 37). Finally $\hat{t}$ is obtained through a weighted sum of $\hat{t}_k$ (Line 38).

Algorithm 2 shows the pseudocode for tags. Each time when it receives a reader command with θ and s (Line 2), it creates a random binary sequence with 32 bits r by hashing s with its own uniform hash function (Line 3). Next it computes the index of the right-most zero in r, which is denoted as R (Line 4). If $R \geq \theta$, the tag replies in this slot with a RN-16 sequence. Otherwise it does not reply (Line 5–9).

Algorithm 1 PLACE algorithm for RFID reader

```
1: θ ← 16
2: while TRUE do
3:    X̄_0 ← 0, X̄_1 ← 0, X̄_2 ← 0, X̄_3 ← 0
4:    i ← 1
5:    while i ≤ 32 do
6:       Generate a random seed s, broadcast a query
         command containing (θ, s), and wait for tag
         responses
7:       if k tags (k = 0, 1, 2, 3) respond in the time slot then
8:          X̄_k ← X̄_k + 1/32
9:       else
10:         Do nothing
11:      end if
12:      i ← i + 1
13:   end while
14:   X̄_4+ ← 1 − X̄_0 − X̄_1 − X̄_2 − X̄_3
15:   Solve for (18) with X̄_4+ and obtain λ̂
16:   if λ̂ < 3 then
17:      θ ← θ/2
18:   else if λ̂ > 7 then
19:      θ ← (θ + 32)/2
20:   else
21:      break
22:   end if
23: end while
24: Set m ← c^2/(ε^2 g(λ̂)) based on (16) and (17)
25: X̄_0 ← 0, X̄_1 ← 0, X̄_2 ← 0, X̄_3 ← 0
26: i ← 1
27: while i ≤ m do
28:   Generate a random seed s, broadcast a query command
      containing the updated θ and s, and wait for tag
      responses
29:   k (k = 0, 1, 2, 3) tags respond in the time slot
30:      X̄_k ← X̄_k + 1/m
31:   else
32:      Do nothing
33:   end if
34:   i ← i + 1
35: end while
36: For each X̄_k (k = 0, 1, 2, 3), compute corresponding t̂_k
    based on (5) and λ = t/2^θ
37: Compute the weights w_k (k = 0, 1, 2, 3) based on (5),
    (13) and (14)
38: Estimate t as t̂ ← Σ_{k=0}^{3} w_k t̂_k
```

D. Discussion

There exist many other ways to utilize multiple slot states information for tag set cardinaltiy estimation. One such way is a frame-based scheme described as follows. The reader provides a long frame consisting of f time slots. Each tag participates in this frame with probability p. If a tag decides to participate, it randomly chooses a slot in the whole frame to reply in. The frame-based scheme is different from what PLACE does, as PLACE requires each tag join each slot independently with certain probability. By counting the number of time slots N_k in the frame where k ($k = 0, 1, 2, 3$) tags reply, the tag set cardinality t can be estimated. Compared with PLACE, this method reduces the interaction overhead between the reader and tags: while PLACE requires the reader to broadcast query command

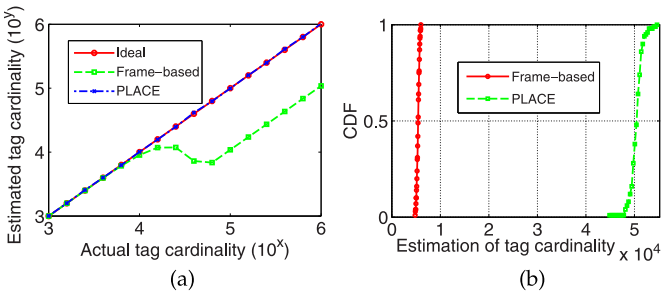


Fig. 8. Performance of the toy estimation scheme. (a) Estimation bias with varying t. (b) CDF of estimation results when $t = 50000$.

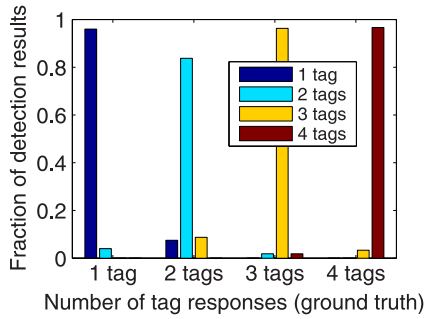


Fig. 9. Detailed accuracy performance of SSDA.

Algorithm 2 PLACE algorithm for RFID tag

```
while TRUE do
    Receive reader query command with parameter θ and random seed s
    Generate a random 32-bit sequence r by hashing s with a uniform hash function
    Compute R, the index of the right-most zero in r
    if R ≥ θ then
        Reply to the reader with RN-16 sequence in the coming time slot
    else
        Do not reply
    end if
end while
```

for each slot, this method allows the reader to broadcast only one query command and wait for all subsequent tag replies.

After obtaining N_k, a combination method is needed for final estimation. One straightforward combination method is to sum up the detectable number of tag replies in the frame and use this value divided by p, i.e., $\hat{t} = \sum_{k=0}^{3} kN_k/p$. By tuning f and p, optimal performance can be achieved for the use-specified accuracy requirement. However, the frame-based scheme can only provide unbiased estimations when f is proportional to t, which indicates that this scheme will incur $O(t)$ time overhead.

We run large-scale simulations to compare the performance of the above frame-based scheme and PLACE. For the frame-based scheme, we fix the frame size parameter f to 9000 and vary the tag set cardinality t, from 10^3 to 10^6. For each t we run the frame-based scheme for 100 times and get the averaged value as the final estimation. For PLACE, simliar settings are applied. The only difference is that we provide 9000 total slots for PLACE operation. We plot both the ideal estimation curve, i.e., $y = x$, and the actual estimation curves of the two schemes in Fig. 8(a). We observe that for the frame-based scheme, the biased estimation occurs if $t > 10^4 > 9000$, indicating that it is not scalable with large tag set cardinality. In other words, we need $O(t)$ slots to perform an accurate estimation of t, which largely increases operation overhead. In constrast, PLACE always has an accurate estimation curve that nearly overlaps with the ideal curve. Fig. 8(b) confirms the performance constrast from the view of CDF. In Fig. 8(b), we fix t to 50000 and f to 1500. Each scheme is operated for 100 times to plot the CDF graph. We find that while the estimation distribution of PLACE is centered around 50000, the median estimation of the frame-based scheme is only 5400, which is seriously biased from the ground truth. This is because the frame size is too small compared with the tag set cardinality.

V. Impact of SSDA Errors and Enhancement

In this section, we analyze how the SSDA detection errors influence the estimation accuracy of PLACE and study the impact of SNR on SSDA detection error.

A. Enhanced PLACE

We denote q_{ij} as the probability of detecting state i as state j, where $i, j = 1, 2, 3, 4+$. Specifically, if $i = j$, q_{ij} indicates the detection accuracy of state i. As empty slots (state 0) can be accurately differentiated from busy slots by measuring the signal strength, we only consider the detection accuracy of state i, where $i, j = 1, 2, 3, 4+$. We use a detection rate matrix $Q = [q_{ij}]_{4\times4}$ to represent the overall detection performance of SSDA.

We use a vector $\vec{X} = (\bar{X}_1, \bar{X}_2, \bar{X}_3, \bar{X}_{4+})^T$ to represent the actual fraction of each state. As the detection results of SSDA may contain some errors, we represent the measurement results as $\vec{X}^E = (\bar{X}_1^E, \bar{X}_2^E, \bar{X}_3^E, \bar{X}_{4+}^E)^T$. Based on the definition of Q, we have $\vec{X}^E = Q\vec{X}$. Thus, we can obtain $\vec{X}$, which can be used to generate an accurate estimation of t, as follows:

$$\vec{X} = Q^{-1}\vec{X}^E, \tag{19}$$

where Q^{-1} is the inverse matrix of Q.

To estimate Q, we perform SSDA with our traces collected from the software defined testbed, which is described in Section II. We set the percentage threshold to be 1% and the interception rate to be 30%. Fig. 9 plots the state detection accuracy of SSDA. The x-axis of Fig. 9 is the ground truth of each tag response state, and the y-axis represents the detection results. We represent the measurement results with Q as follows:

$$Q = \begin{pmatrix} 0.96 & 0.04 & 0 & 0 \\ 0.08 & 0.84 & 0.09 & 0 \\ 0 & 0.02 & 0.96 & 0.02 \\ 0 & 0 & 0.03 & 0.97 \end{pmatrix}.$$

From the above Q, we find the SSDA method achieves high detection accuracies. For the detection errors, we find that state k is more likely to be mistakenly detected as adjacent

states. In practice, Q can vary due to various factors, e.g., reader transmission power, interference to tag responses, etc. To understand the impact of Q on the overall estimation accuracy of PLACE, we approximate Q with Q_0 as follows:

$$Q_0 = \begin{pmatrix} 1-q_0 & q_0 & 0 & 0 \\ q_0 & 1-2q_0 & q_0 & 0 \\ 0 & q_0 & 1-2q_0 & q_0 \\ 0 & 0 & q_0 & 1-q_0 \end{pmatrix},$$

where q_0 can be specified according to empirical measurement results. With Q_0, we can study how different detection performance of SSDA may impact the overall counting accuracy of PLACE. We can recover $\vec{X}$ from $\vec{X}^E$ according to (19) and use $\vec{X}$ for tag cardinality estimation.We name the enhanced PLACE with the error compensation as EPLACE.

B. Impact of SNR on q_0

In practice, the SNR of received tag response signal has significant impacts on q_0. In the following, we first provide a definition for received signal SNR and then study the impact of SNR on q_0.

1) Definition of SNR of Received Signal: Data samples of the received signal form clusters in the I-Q plane, each of which represents a signal state. For each cluster, we can define a centroid and radius: the centroid is the arithmetic mean of all data samples belonging to this cluster; the radius is the averaged Euclidean distance betwee each data sample and the cluster centroid over all data samples in this cluster.

The signal power can be represented by the average amplitude over all data samples of the reply signal. However, there exists a signal state shift in the reply signal due to environmental factors like device thermal noise, multipath, channel degradation, etc. Due to this reason, we need to find another point in the I-Q plane to replace the origin and compensate for the state shift. We choose this point as the centroid of the reference cluster, whose centroid has shortest distance to the origin among all clusters. Actually the reference cluster corresponds to the signal state with lowest power among all signal states. The following is the signal power expression:

$$P_s = \frac{1}{l}\sum_{i=1}^{l} dist(z_i, z_0), \tag{20}$$

where z_0 is the centroid of the reference cluster, z_i $(i = 1, 2, \ldots, l)$ represents each data sample of the tag response signal.

The noise power can be represented by the radius of a cluster since it indicates the spread of signal state. If multiple clusters exist, we should average all cluster radiuses in theory. Later we show that we can obtain the radius of the reference cluster in a fast way. Thus, we measure the radius of the reference cluster as the signal strength.

From the above we find SNR is closely related to the reference cluster. We use an example in Fig. 10 to illustrate how to obtain reference cluster inforamtion in a fast manner. Fig. 10(a) is the constellation map of the tag response signal with $k = 2$. Fig. 10(b) is the constellation map of the noise

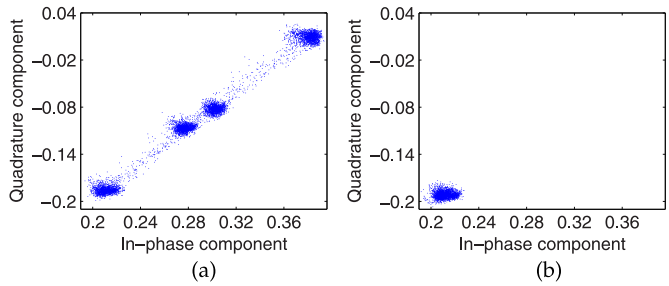


Fig. 10. Illustration on how to utilize the noise before tag response signal to obtain the noise power and reference point. (a) Constellation map of the signal. (b) Constellation map of the noise.

that comes before the signal and after the reader query command. The corresponding trace for Fig. 10 is chosen from our collected traces described in Section II-B.

We find the only cluster in Fig. 10(b) has the same area as the cluster in the lower left corner of Fig. 10(a). Since the lower left cluster in Fig. 10(a) has shortest distance to the origin among all clusters, this cluster corresponds to the signal state where no tags reply with high power level, and hence is the reference cluster by definition.

In real-time cardinality estimation operation, the centroid and radius of the reference cluster can be quickly computed by collecting data samples that come after the reader query command and before the tag response signal. The SNR of tag response signal can be also quickly obtained after collected all data samples of the reply. In contrast, computing SNR by performing clustering operation on the whole tag response signal with DBSCAN takes significant time overhead.

2) Impact of SNR on q_0: Intuitively, a higher SNR leads to further distances between clusters in the constellation map and hence higher SSDA detection accuracy. According to the definition of q_0 in the previous subsection, q_0 is monotonically decreasing when SSDA detection accuracy is increasing. Consequently, a higher SNR leads to lower q_0 value.

In practice, offline training can be used to build a look-up table between SNR and q_0. (An alternative way is to establish the relationship between SNR and SSDA detection accuracy, from which q_0 can also be easily obtained. In our evaluation section, we adopt this method.) From the definition of q_0 in the Q_0 matrix, we know that for different k values, the corresponding SSDA detection accuracy can be either $1-q_0$ or $1-2q_0$. Hence, the prequisite of measuring q_0 is to make clear how many tags are replying concurrently. In the offline training process, we can set, for example, 2 tags in front of the reader and perform the training. In this case, the SSDA detection accuracy would be $1-2q_0$. When real-time tag set cardinality estimation is running, such a look-up table can allow real-time compensation for PLACE estimation results.

VI. Evaluation

In the following, we first compare SSDA with the benchmark clustering algorithm DBSCAN [6] in terms of the slot state detection accuracy and the execution time. We then compare PLACE with previous cardinality estimation schemes including EFNEB [8], LoF [17], ZOE [31] and SRC [4]. Next, we evaluate the impact of SSDA detection errors on the estimation

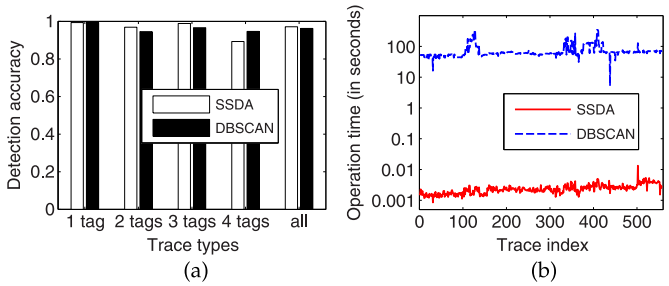


Fig. 11. Performance comparison of DBSCAN and SSDA. (a) Comparison of detection accuracy. (b) Comparison of computational overhead.

accuracy as well as the compensation for the errors. Finally, we study the impact of SNR of tag response signal on SSDA detection accuracy.

A. SSDA Evaluation

Our traces are collected with the GNURadio/USRP testbed and WISP tags as described in Section II. For the experimental purpose, we program the WISP tags and control the number of responding tags in each slot. We record the actual number of responding tags (varying from 1 to 4) as the ground truth in the experiment.

We expect an ideal slot state detection algorithm to efficiently process the samples and accurately count the number of responding tags. We compare the detection accuracy and the execution time of the proposed SSDA with the benchmark scheme DBSCAN. We set the interception rate IR to 30% and the percentage threshold PT to 1% for SSDA.

In DBSCAN, a circular region centered on a point p with radius ε is called ε-neighborhood of p. If at least T points fall into ε-Neighborhood of p, p is called a core point. Otherwise if q falls into the ε-neighborhood of another core point, we call q border point. A noise point is a point that is neither a core point nor a border point. If p is a core point and q is in ε-neighborhood of p, we say q is *directly density-reachable* from p. If q' is directly density-reachable from q, and q is directly density-reachable from p, we say q' is *indirectly density-reachable* from p via q, i.e, $p \rightarrow q \rightarrow q'$. DBSCAN groups all density-reachable points into one cluster. In the experiment, we specify the optimal parameters ($\varepsilon = 0.01, T = 0.01 \times l$), which maximize the detection accuracy of DBSCAN. Although DBSCAN can be used to count the number of responding tags in each slot, it incurs a computation overhead of $O(l^2)$, where l denotes the number of input samples.

Fig. 11(a) compares the detection accuracy of SSDA and DBSCAN. We present the overall detection accuracy as well as the accuracy for each case with different number of responding tags. Fig. 11(a) shows the following results. First, the overall accuracy of SSDA is comparable with that of DBSCAN. Specifically, the overall accuracies of SSDA and DBSCAN are 91.2% and 96.7%, respectively. Second, in the case when 4 tags respond together, SSDA achieves higher accuracy compared with DBSCAN. This is because when 4 tags respond concurrently, the I-Q plane becomes crowded with 16 clusters. As a result, the inter-cluster distances become smaller and the borders between neighboring clusters become blurred. Thus, the border-based DBSCAN may cluster the neighboring clusters together. In contrast, the centroid-based SSDA overcomes this problem and derives the number of clusters by counting the number of local maximums after filtering out noise. Since the local maximums lie in the center of clusters, the distance between the centers of two neighboring clusters tend to be larger than the distance between their borders.

Fig. 11(b) compares the computational overhead of SSDA and DBSCAN. The physical layer symbols are collected with the USRP reader and the symbols are transferred to a laptop for processing. We execute both algorithms on the laptop and measure the execution time of two algorithms. The laptop is equipped with an Intel qual-core 2.9 GHz i7 processor and 15.4 GB memory running 64-bit Ubuntu 13.04. In the figure, the x-axis is the trace index and the y-axis is the operation time in seconds, presented in the log scale. We find that SSDA reduces the operation time compared with DBSCAN by orders of magnitude. Specifically, the average operation time of SSDA and DBSCAN is 1.3 ms and 84.5 s, respectively. SSDA substantially outperforms DBSCAN mainly due to the fact that while DBSCAN incurs $O(l^2)$ computational overhead, SSDA only incurs $O(l)$ overhead. In addition, while DBSCAN has to perform computation-intensive operations such as multiplication and square root calculation to calculate the distance between physical layer symbols, SSDA only needs to perform lightweight operations such as addition and comparison.

B. PLACE Evaluation

We perform extensive simulations to compare PLACE with previous cardinality estimation schemes. As most of these previous schemes do not tolerate noisy channels, we assume no errors in slot state detection in the performance comparison.

We measure the overall execution time as the performance metric, which counts both communication time and the computation time. The communication time mainly consists of the transmission time of reader's command and tags responses. The computation time is mainly consumed in the execution of SSDA for each slot. We ignore the computation time for benchmark schemes. In practice, as SSDA can be executed in real time, the cluster counting operation (which takes 1.3 ms) can be executed in parallel with the signal sampling operation for each RN16 reception (which takes 2 ms) at physical layer. Thus, SSDA incurs little extra time overhead.

Fig. 12 compares the overall operation time to meet different estimation accuracy requirements. The actual tag cardinality is 50000. In Fig. 12(a), we fix δ to 20% and vary ϵ, ranging from 1% to 5%. From Fig. 12(a), we find that like benchmark schemes, PLACE takes less time to meet the estimation accuracy requirement of relaxed confidence intervals. Results in Fig. 12(a) demonstrate that when $\epsilon = 1\%$, PLACE improves the operation time performance over LoF, EFNEB, ZOE and SRC by 18.71×, 17.42×, 3,78× and 3.19× on average. In Fig. 12(b), when we fix ϵ to 1% and vary δ from 1% to 10%, we also find that PLACE substantially outperforms benchmark schemes. In particular, Fig. 12(b) shows that when $\delta = 1\%$, PLACE improves the operation time over LoF, EFNEB, ZOE and SRC by 23.75×, 21.91×, 4.62× and 4.02× on average.

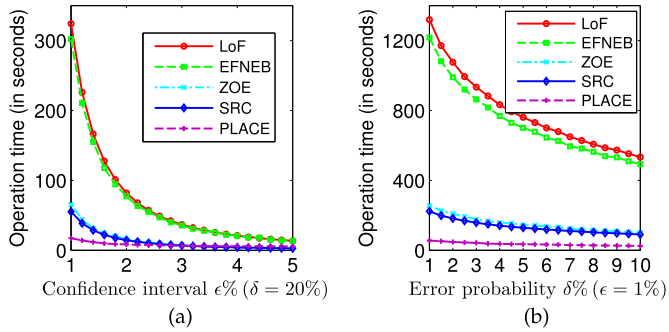


Fig. 12. Comparison of operation time to meet different accuracy requirements among 5 schemes: EFNEB, LoF, ZOE, SRC and PLACE. (a) Fix $\delta = 20\%$ and vary ϵ. (b) Fix $\epsilon = 1\%$ and vary δ.

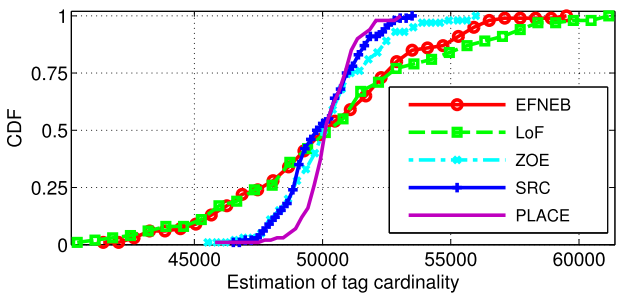


Fig. 13. CDF of estimation results with the same amount of execution time.

We provide each estimation scheme the same amount of execution time to estimate the number of 50000 tags. We repeat the estimation process of each scheme for 100 times. In Fig. 13, we plot the CDF of estimation results for each scheme. From Fig. 13, we find that the estimation results of PLACE are more concentrated on the actual tag cardinality. Moreover, the tail of PLACE is much shorter than those of EFNEB, LoF, ZOE and SRC, indicating smaller estimation variance of PLACE. Specifically, according to the estimation results, provided the same amount of operation time, PLACE has 99 estimation results within the confidence interval [47500,52500], while SRC, which performs best among the benchmarks, has only 91 estimation results within the interval. According to the experiment result, we find that given the same amount of operation time, PLACE can estimate the tag cardinality more precisely and accurately compared with other schemes.

C. The Impact of SSDA Detection Errors on PLACE

To evaluate the impact of SSDA detection errors on the estimation accuracy of PLACE, we measure the estimation accuracy with the ratio of the estimated tag population $\hat{t}$ over the actual population t as in [17], [28], [31]. Ideally, the accuracy should be 1, indicating a perfect estimation result.

We run the basic PLACE and the Enhanced PLACE (EPLACE), which compensates for the errors and adjusts the estimation results. We use q_0 to represent the slot state detection error. We average over 100 runs to obtain each estimation result.

Fig. 15(a)–(c) plot the estimation accuracies of PLACE and EPLACE, with q_0 of 5%, 15%, and 25%, respectively. The y-axis and x-axis represent the estimated number and the actual number of tags, respectively. For illustration purposes, we plot the ideal curve $y = x$. From Fig. 15(a)–(c), we find that without the error compensation, the estimation errors of the basic PLACE increase with both the number of tags and the slot state detection errors. Fortunately, EPLACE is able to compensate for the errors and achieve high accuracy. The experiment results of Fig. 15(a)–(c) demonstrate that EPLACE is able to leverage the knowledge about the detection errors and adjust the estimation results accordingly.

In Fig. 15(d), we vary the error rate from 5% to 25% and measure the corresponding estimation accuracy. We fix the tag cardinality to 50000. We specify the ($\epsilon = 5\%$, $\delta = 1\%$)-accuracy requirement, and provide PLACE the corresponding execution time. From Fig. 15(d), we find that as q_0 increases, the estimation accuracy of basic PLACE decreases dramatically. In contrast, the estimation accuracy of EPLACE remains relatively stable and fluctuates around 1. Although EPLACE cannot achieve the ideal estimation accuracy of 1, the estimation results are all within the targeted accuracy interval of [0.95,1.05].

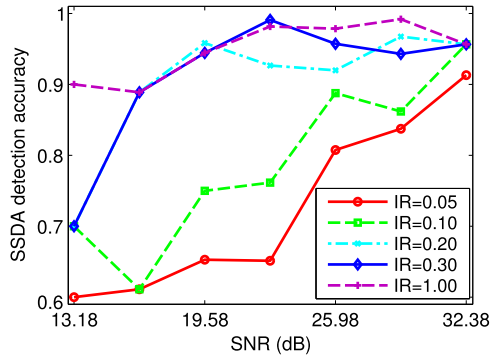


Fig. 14. Impact of SNR of tag response signal on SSDA detection accuracy.

D. The Impact of SNR on SSDA Detection Accuracy

In this subsection we run experiments to evaluate the impact of SNR on SSDA detection accuracy. For each of all traces, we first compute its SNR. Next, we divide the whole SNR range of all traces into small bins with equivalent length. For each bin, we compute a SSDA detection accuracy as the ratio of the number of correctly detected traces over the total number of traces falling into this bin. We plot the results in Fig. 14.

In Fig. 14, the x-axis represents the SNR range. Each SNR value indicates a SNR range, e.g., 13.18 dB represents the range [13.18,16.38]. The y-axis is the SSDA detection accuracy for each bin. To better show the impact of SNR on detection accuracy, we vary the interception rate (IR) of each trace from 0.05, 0.10 to 1. From Fig. 14 we find that in general, the SSDA detection accuracy increases when SNR increases. This rule is especially obvious when the interception rate is low, while when IR is high, e.g., 1, the increasement is not that significant. This is because a higher IR indicates more data samples are used as SSDA input. As a result, more redundancy is available for SSDA and hence it is more robust to low SNR.

VII. Related Work

A series of probabilistic approaches have been proposed to improve the cardinality estimation efficiency [4], [8], [11],

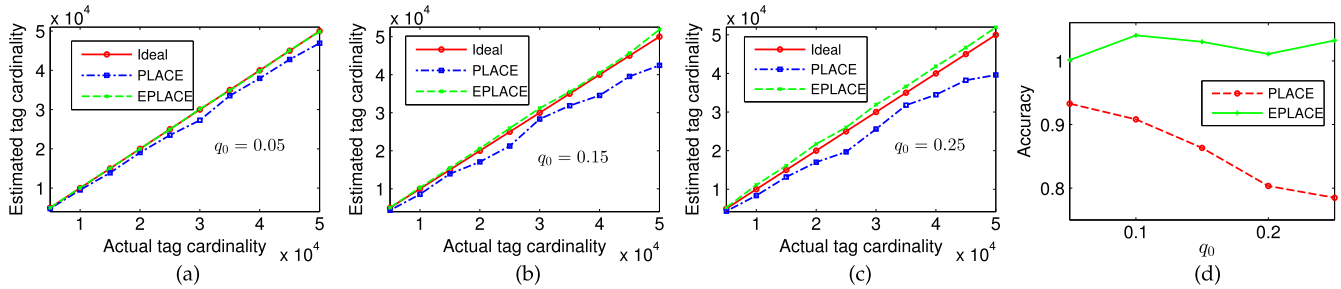


Fig. 15. Impact of q_0 on estimation accuracy of PLACE. (a) $q_0 = 0.05$; (b) $q_0 = 0.15$; (c) $q_0 = 0.25$; (d) Accuracy with varied q_0.

[12], [17], [18], [28], [31]. Kodialam *et al.* propose the first probabilistic counting scheme, Unified Probabilistic Estimator (UPE) [11], which uses the fractions of empty, singleton and collision slots to estimate the tag population. Qian *et al.* propose the Lottery Frame (LoF) scheme [17] to reduce the frame size and avoid the problem of replicated counting. Han *et al.* present the Enhanced First Non-Zero Based estimator (EFNEB) [8], which quickly locates the first busy slot with the binary search. Zheng *et al.* design the Probabilistic Estimating Tree scheme (PET) [28], where a binary tree is used to organize the tags and assist the tag probing. Shahzad *et al.* propose the Average Run based Tag estimator (ART) [18], which estimates the tag population with the average run length of non-empty slots. Zheng *et al.* present the Zero-One Estimator (ZOE) [31], where all tags respond in each slot with a certain probability and the fraction of empty slots is used to estimate the tag cardinality. Li *et al.* [13] present an energy-efficient cardinality estimation algorithm to save power for active RFID tags. Chen *et al.* [4] emphasize the importance of the two-phase design and study the theoretical limits of RFID counting efficiency. Gong *et al.* [7] efficiently estimate the number of counterfeit tags. Liu *et al.* [14] estimate the number of key tags. Unlike those works that only leverage binary or ternary states extracted from each slot, we propose a cardinality estimation scheme which infers the number of colliding tags in each slot at RFID physical layer and thereby improves the estimation efficiency.

Previous works try to read multiple RFID tags by recovering tag collisions at physical layer [2], [10], [19]. Shen *et al.* [19] propose to use software defined radios to recover collisions of HF RFID cards. Khasgiwale *et al.* [10] decode the RN16 message of UHF RFID tags so as to improve the tag arbitration efficiency. Some works [2], [5] present the theoretical analysis on tag collisions and read a small number of tags in parallel. Nevertheless, such deterministic identification schemes cannot efficiently estimate the tag cardinality for large-scale RFID systems. Inspired by those works, we present a probabilistic estimation scheme which is able to extract and synthesize more information from the RFID physical layer.

Many prior works study the problem of collecting data from RFID devices. Yue *et al.* [23] present a data collection scheme using the Bloom filter. BLINK [25] improves the link layer performance with link quality measurement and rate adaptation for RFID devices. Buzz [20] recovers tag collisions at physical layer and collects data from RFID tags in an efficient and reliable manner. Zanetti *et al.* [24] identify RFID tags using the physical layer fingerprints. P-MTI [30] detects missing tags from a set of known tags through PHY information. Different from PLACE, P-MTI works with prior knowledge of combined PHY information from known RFID tags. PLACE on the other hand counts the number of an unknown set of tags. Tagoram [22] tracks mobile tags by leveraging the phase information available at commodity readers.

VIII. Conclusion

Estimating the number of RFID tags is a fundamental operation in RFID systems. In this paper, we introduce a physical layer based cardinality estimator to fundamentally improve the estimation efficiency. We first propose a slot state detection algorithm to accurately count the number of responding tags in each slot. We then devise a joint estimator to combine multiple sub-estimators each of which estimates the tag population with the slot state measurement results. Extensive evaluation results show that PLACE substantially outperforms prior works. The query processes for the RFID reader and tag in our proposed algorithms do not completely follow the C1G2 standard. Nevertheless, our proposed query process incurs less overhead than the standard does. Similar slight modification to the C1G2 standard has been popular in most existing schemes [4], [8], [17], [18], [31].

References

[1] EPC Global, Inc., "EPCglobal C1G2 UHF RFID protocol at 860 MHz–960 MHz," 2008 [Online]. Available: http://www.gs1.org/sites/default/files/docs/epc/uhfc1g2 1 2 0-standard-20080511.pdf

[2] C. Angerer, R. Langwieser, and M. Rupp, "RFID reader receivers for physical layer collision recovery," *IEEE Trans. Commun.*, vol. 58, no. 12, pp. 3526–3537, Dec. 2010.

[3] G. Avoine and P. Oechslin, "A scalable and provably secure hash-based RFID protocol," in *Proc. IEEE PerCom Workshops*, 2005, pp. 110–114.

[4] B. Chen, Z. Zhou, and H. Yu, "Understanding RFID counting protocols," in *Proc. ACM MobiCom*, 2013, pp. 291–302.

[5] M. V. B. Delgodo, C. Angerer, J. V. Alonso, and M. Rupp, "Estimation of the tag population with physical layer collision recovery," in *Proc. 3rd Int. EURASIP Workshop RFID Technol.*, 2010, pp. 106–111.

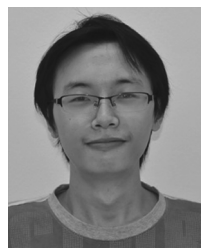
[6] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. ACM KDD*, 1996, pp. 226–231.

[7] W. Gong *et al.*, "Informative counting: Fine-grained batch authentication for large-scale RFID systems," in *Proc. ACM MobiHoc*, 2013, pp. 21–30.

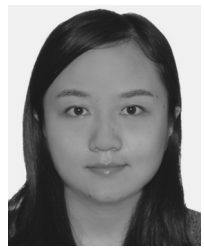
[8] H. Han *et al.*, "Counting RFID tags efficiently and anonymously," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.

[9] Y. Hou, J. Ou, Y. Zheng, and M. Li, "PLACE: Physical layer cardinality estimation for large-scale RFID systems," in *Proc. IEEE INFOCOM*, 2015, pp. 1957–1965.

[10] R. Khasgiwale, R. Adyanthaya, and D. Engels, "Extracting information from tag collisions," in *Proc. IEEE Int. Conf. RFID*, 2009, pp. 131–138.
[11] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in RFID systems," in *Proc. ACM MobiCom*, 2006, pp. 322–333.
[12] M. Kodialam, T. Nandagopal, and W. Lau, "Anonymous tracking using RFID tags," in *Proc. IEEE INFOCOM*, 2007, pp. 1217–1225.
[13] T. Li, S. Wu, and S. Schen. M. Yang, "Generalized energy-efficient algorithms for the RFID estimation problem," *IEEE/ACM Trans. Netw.*, vol. 20, no. 6, pp. 1978–1990, Dec. 2012.
[14] X. Liu, K. Li, H. Qi, B. Xiao, and X. Xie, "Fast counting the key tags in anonymous RFID systems," in *Proc. IEEE ICNP*, 2014, pp. 57–70.
[15] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, 1967, pp. 281–297.
[16] L. Ni, Y. Liu, Y. Lau, and A. Patil, "LANDMARC: Indoor location sensing using active RFID," *Wireless Netw.*, vol. 10, no. 6, pp. 701–710, 2004.
[17] C. Qian, H. Ngan, and Y. Liu, "Cardinality estimation for large-scale RFID systems," in *Proc. IEEE PerCom*, 2008, pp. 30–39.
[18] M. Shahzad and A. Liu, "Every bit counts: Fast and scalable RFID estimation," in *Proc. ACM MobiCom*, 2012, pp. 365–376.
[19] D. Shen, G. Woo, D. Reed, and A. Lippman, "Separation of multiple passive RFID signals using software defined radio," in *Proc. IEEE RFID*, 2009, pp. 139–146.
[20] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk, "Efficient and reliable low-power backscatter networks," in *Proc. ACM SIGCOMM*, 2012, pp. 61–72.
[21] R. Want, "An introduction to RFID technology," *IEEE Pervasive Comput.*, vol. 5, no. 1, pp. 25–33, Jan.–Mar. 2005.
[22] L. Yang *et al.*, "Tagoram: Real-time tracking of mobile RFID tags to high precision using COTS devices," in *Proc. ACM MobiCom*, 2014, pp. 237–248.
[23] H. Yue, C. Zhang, M. Pan, Y. Fang, and S. Chen, "A time-efficient information collection protocol for large-scale RFID systems," in *Proc. IEEE INFOCOM*, 2012, pp. 2158–2166.
[24] D. Zanetti, B. Danev, and S. Čapkun, "Physical-layer identification of UHF RFID tags," in *Proc. ACM MobiCom*, 2010, pp. 353–364.
[25] P. Zhang, J. Gummeson, and D. Ganesan, "BLINK: A high throughput link layer for backscatter communication," in *Proc. ACM MobiSys*, 2012, pp. 99–112.
[26] R. Zhang, Y. Liu, Y. Zhang, and J. Sun, "Fast identification of the missing tags in a large RFID system," in *Proc. IEEE SECON*, 2011, pp. 278–286.
[27] B. Zhen, M. Kobayashi, and M. Shimizu, "Framed ALOHA for multiple RFID objects identification," *IEICE Trans. Commun.*, vol. E88-B, no. 3, pp. 991–999, 2005.
[28] Y. Zheng, M. Li, and C. Qian, "PET: Probabilistic estimating tree for large-scale RFID estimation," in *Proc. IEEE ICDCS*, 2011, pp. 37–46.
[29] Y. Zheng and M. Li, "Fast tag searching protocol for large-scale RFID systems," in *Proc. IEEE ICNP*, 2011, pp. 363–372.
[30] Y. Zheng and M. Li, "P-MTI: Physical-layer missing tag identification via compressive sensing," in *Proc. IEEE INFOCOM*, 2013, pp. 917–925.
[31] Y. Zheng and M. Li, "ZOE: Fast cardinality estimation for large-scale RFID systems," in *Proc. IEEE INFOCOM*, 2013, pp. 908–916.
[32] A. Goldsmith, *Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2005.
[33] D. Pauluzzi and N. Beaulieum, "A comparison of SNR estimation techniques for the AWGN channel," *IEEE Trans. Commun.*, vol. 48, no. 10, pp. 1681–1691, Oct. 2000.
[34] A. Gamal, M. Mohseni, and S. Zahedi, "Bounds on capacity and minimum energy-per-bit for AWGN relay channels," *IEEE Trans. Inf. Theory*, vol. 52, no. 4, pp. 1545–1561, Apr. 2006.



Yuxiao Hou received the BS degree in the Special Class for Gifted Young from the University of Science and Technology of China, China, in 2012. He is currently a PhD student in the School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include RFID and wireless network. He is a student member of IEEE.



Jiajue Ou received the B.E. degree in Communication Engineering from the University of Electronic Science and Technology of China, Chengdu, China, in 2010. She is currently a Ph.D. student in the School of Computer Engineering, Nanyang Technological University, Singapore. Her research interests include networked and distributed sensing, wireless networks and RFID systems. She is a student member of ACM.

Yuanqing Zheng received the BS degree in Electrical Engineering in 2007 and ME degree in Communication and Information System in 2010, from Beijing Normal University, China. He received the PhD degree in School of Computer Engineering from Nanyang Technological University in 2014. He is currently an Assistant Professor with the Department of Computing in Hong Kong Polytechnic University. His research interest includes mobile and wireless computing and RFID. He is a member of IEEE and ACM.

Mo Li received the BS degree in Computer Science and Technology from Tsinghua University, China in 2004, and the PhD degree in Computer Science and Engineering from Hong Kong University of Science and Technology in 2009. He is a Nanyang Assistant Professor with the Computer Science Division, School of Computer Engineering, Nanyang Technological University, Singapore. His research interests include distributed systems, wireless sensor networks, pervasive computing, RFID, and wireless and mobile systems. He is a member of IEEE and ACM.