Efficient Protocols for Collecting Histograms in Large-Scale RFID Systems

Lei Xie, *Member, IEEE*, Hao Han, *Member, IEEE*, Qun Li, *Member, IEEE*, Jie Wu, *Fellow, IEEE*, and Sanglu Lu, *Member, IEEE*

Abstract—Collecting histograms over RFID tags is an essential premise for effective aggregate queries and analysis in large-scale RFID-based applications. In this paper we consider an efficient collection of histograms from the massive number of RFID tags, without the need to read all tag data. In order to achieve time efficiency, we propose a novel, ensemble sampling-based method to simultaneously estimate the tag size for a number of categories. We first consider the problem of basic histogram collection, and propose an efficient algorithm based on the idea of ensemble sampling. We further consider the problems of advanced histogram collection, respectively, with an iceberg query and a top-k query. Efficient algorithms are proposed to tackle the above problems such that the qualified/unqualified categories can be quickly identified. This ensemble sampling-based framework is very flexible and compatible to current tag-counting estimators, which can be efficiently leveraged to estimate the tag size for each category. Experiment results indicate that our ensemble sampling-based solutions can achieve a much better performance than the basic estimation/identification schemes.

Index Terms—Algorithms, RFID, time efficiency, histogram

1 Introduction

With the rapid proliferation of RFID-based applications, RFID tags have been deployed into pervasive spaces in increasingly large numbers. In applications like warehouse monitoring, the items are attached with RFID tags, and are densely packed into boxes. As the maximum scanning range of a UHF RFID reader is usually 6-10 m, the overall number of tags within this three-dimensional space can be up to tens of thousands in a dense deployment scenario, as envisioned in [1], [2], [3]. Many tag identification protocols [4], [5], [6], [7], [8] are proposed to uniquely identify the tags one by one through anti-collision schemes. However, in a number of applications, only some useful statistical information is essential to be collected, such as the overall tag size [2], [9], [10], popular categories [11] and the histogram. In particular, histograms capture distribution statistics in a space-efficient fashion. In some applications, such as a grocery store or a shipping portal, items are categorized according to some specified metrics, such as types of merchandize, manufacturers, etc. A histogram is used to illustrate the number of items in each category.

In practice, tags are typically attached to objects belonging to different categories, e.g., different brands and models of clothes in a large clothing store, different titles of books in a book store, etc. Collecting histogram can be used to illustrate the tag population belonging to each category, and determine whether the number of tags in a category is above or below any desired threshold. By setting this threshold, it is easy to find popular merchandise and control stock, e.g., automatically signaling when more products need to be put on the shelf. Furthermore, the histogram can be used for approximate answering of aggregate queries [12], [13], as well as preprocessing and mining association rules in data mining [14]. Therefore, collecting histograms over RFID tags is an essential premise for effective queries and analysis in conventional RFID-based applications. Fig. 1 shows an example for collecting histogram over the RFID tags deployed in the application scenarios.

While dealing with a large scale deployment with thousands of tags, the traditional tag identification scheme is not suitable for histogram collection, since the scanning time is proportional to the number of tags, which can be in the order of several minutes. As the overall tag size grows, reading each tag one by one can be rather time-consuming, which is not scalable at all. As in most applications, the tags are frequently moving into and out of the effective scanning area. In order to capture the distribution statistics in time, it is essential to sacrifice some accuracy so that the main distribution can be obtained within a short time window–in the order of several seconds. Therefore, we seek to propose an estimation scheme to quickly count the tag sizes of each category while achieving the accuracy requirement.

In most cases, the tag sizes of various categories are subject to some skewed distribution with a "long tail", such as the Gaussian distribution. The long tail represents a large number of categories, each of which occupies a rather small percentage among the total categories. While handling the massive number of tags, in the order of several thousands, the overall number of categories in the long tail could be in

- *L. Xie and S. Lu are with the State Key Laboratory for Novel Software Technology, Nanjing University, China. E-mail: {lxie, sanglu}@nju.edu.cn.*
- *H. Han and Q. Li are with the Department of Computer Science, College of William and Mary, Williamsburg, VA. E-mail: {hhan, liqun}@cs.wm.edu.*
- *J. Wu is with the Department of Computer Information and Sciences, Temple University. E-mail: jiewu@temple.edu.*

Manuscript received 10 Mar. 2014; revised 12 Aug. 2014; accepted 4 Sept. 2014. Date of publication 10 Sept. 2014; date of current version 7 Aug. 2015. Recommended for acceptance by S. Guo.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2014.2357021

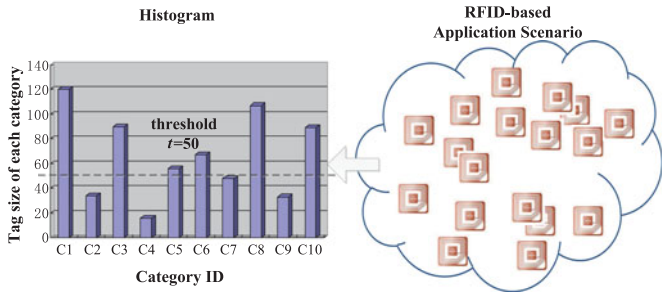


Fig. 1. An example of collecting histogram over RFID tags

hundreds. Therefore, by separately estimating the tag sizes over each category, a large number of query cycles and slots are required. Besides, in applications like the iceberg query and the top-k query, only those major categories are essential to be addressed. In this situation, the separate estimate approach will waste a lot of scanning time over those minor categories in the long tail. Therefore, a novel scheme is essential to quickly collect the histograms over the massive RFID tags. In this paper, we propose a series of protocols to tackle the problem of efficient histogram collection. The main contributions of this paper are listed as follows (a preliminary version of this work appeared in [15]):

1) To the best of our knowledge, we are the first to consider the problem of collecting histograms and its applications (i.e., iceberg query and top-k query) over RFID tags, which is a fundamental premise for answering aggregate queries and data mining over RFID-based applications.
2) In order to achieve time efficiency, we propose a novel, ensemble sampling (ES)-based method to simultaneously estimate the tag size for a number of categories. This framework is very flexible and compatible to current tag-counting estimators, which can be efficiently leveraged to estimate the tag size for each category. While achieving time-efficiency, our solutions are completely compatible with current industry standards, i.e., the EPCglobal C1G2 standards, and do not require any tag modifications.
3) In order to tackle the histogram collection with a filter condition, we propose an effective solution for the *iceberg query* problem. By considering the *population* and *accuracy constraint*, we propose an efficient algorithm to wipe out the unqualified categories in time, especially those categories in the long tail. We further present an effective solution to tackle the *top-k query* problem. We use ensemble sampling to quickly estimate the threshold corresponding to the kth largest category, and reduce it to the *iceberg query* problem.

The remainder of the paper is as follows. Sections 2 and 3 present the related work and RFID preliminary, respectively. We formulate our problem in Section 4, and present our ensemble sampling-based method for the basic histogram collection in Section 5. We further present our solutions for the iceberg query and the top-k query, respectively, in Sections 6 and 7. In Section 8, we provide performance analysis in time-efficiency. The performance evaluation is in Section 9, and we conclude in Section 10.

2 Related Work

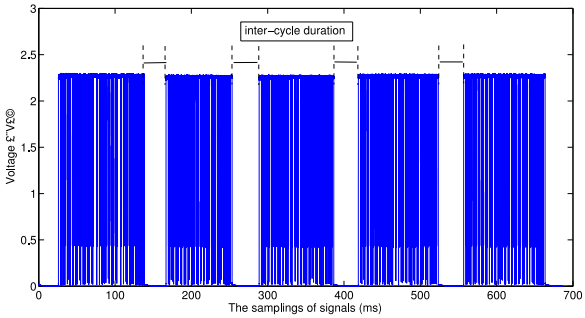
In RFID systems, a reader needs to receive data from multiple tags, while the tags are unable to self-regulate their radio transmissions to avoid collisions; then, a series of slotted ALOHA-based anti-collision protocols [1], [4], [5], [6], [7], [8], [16], [17] are designed to efficiently identify tags in RFID systems. In order to deal with the collision problems in multi-reader RFID systems, scheduling protocols for reader activation are explored in the literature [18], [19]. Recently, a number of polling protocols [20], [21], [22] are proposed, aiming to collect information from battery-powered active tags in an energy efficient approach.

Recent research is focused on the collection of statistical information over the RFID tags [2], [9], [10], [11], [23], [24], [25], [26], [27]. The authors mainly consider the problem of estimating the number of tags without collecting the tag IDs. Murali et al. provide very fast and reliable estimation mechanisms for tag quantity in a more practical approach [9]. Li et al. study the RFID estimation problem from the energy angle [23]. Their goal is to reduce the amount of energy that is consumed by the tags during the estimation procedure. Shahzad et al. propose a new scheme for estimating tag population size called average run based tag estimation (ART) [2]. Chen et al. aim to gain deeper and fundamental insights in RFID counting protocols [27], they manage to design near-optimal protocols that are more efficient than existing ones and simultaneously simpler than most of them. Liu et al. investigate efficient distributed query processing in large RFID-enabled supply chains [28]. Liu et al. propose a novel solution to fast count the key tags in anonymous RFID systems [29]. Luo et al. tackle an interesting problem, called multigroup threshold based classification [25], which is to determine whether the number of objects in each group is above or below a prescribed threshold value. Sheng et al. consider the problem of identifying popular categories of RFID tags out of a large collection of tags [11], while the set of category IDs are supposed to be known. Different from the previous work, in this paper, our goal is to collect the histograms for all categories over RFID tags in a time-efficient approach, without any priori knowledge of the categories. Specifically, we respectively consider the basic histogram collection problem, the iceberg query problem, and the top-k query problem in regard to collecting histograms in large-scale RFID systems. We aim to propose a flexible and compatible framework for current tag-counting estimators based on slotted ALOHA protocol, which can be efficiently leveraged to estimate the tag size for each category.

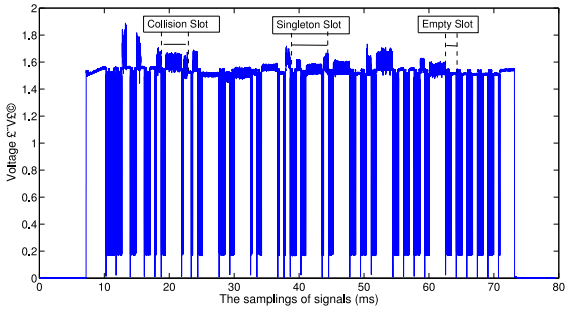
3 Preliminary

3.1 The Framed Slotted ALOHA Protocol

In the Class 1 Gen 2 standard, the RFID system leverages the *framed slotted ALOHA protocol* to resolve the collisions for tag identification. When a reader wishes to read a set of tags, it first powers up and transmits a continuous wave to energize the tags. It then initiates a series of frames, varying the number of slots in each frame to best accommodate the number of tags. Each frame has a number of slots and each active tag will reply in a randomly selected slot per frame. After all tags are read, the reader powers down. We refer to the series of frames between power down periods as a



(a) The raw signal data for 5 continuous cycles



(b) The raw signal data for one frame

Fig. 2. The captured raw signal data of the interrogation between the reader and the tag

Query Cycle. Note that, within each frame, tags may choose the same slot, which causes multiple tags to reply during a slot. Therefore, within each frame there exist three kinds of slots: (1) the empty slot where no tag replies; (2) the single slot where only one tag replies; and (3) the collision slot where multiple tags reply.

In regard to the tag ID, each tag has a unique 96-bit ID in its EPC memory, where the first s binary bits can be regarded as the *category ID* ($1 < s < 96$). According to the C1G2 standard, for each *Query Cycle*, the reader is able to select the tags in a specified category by sending a *Select* command with an s-bit mask in the *category ID* field. If multiple categories need to be selected, the reader can provide multiple bit masks in the *Select* command.

3.2 Basic Tag Identification versus the Estimation Scheme

Assume that there are n tags in total, and that it takes s_i slots to uniquely identify n tags. It is known that for each query round, when the frame size f is equal to the remaining number of tags, the proportion of singleton slots inside the frame is maximized; then, the efficiency is $\frac{n_s}{f} = \frac{1}{e}$. Hence, the essential number of slots is $s_i = \sum_{i=0}^{+\infty} (1 - \frac{1}{e})^i \cdot n = n \cdot e$.

Therefore, assume that it takes s_e slots to estimate the tag size for each category with a certain accuracy. If we want the estimation scheme to achieve a better reading performance than the basic tag identification method, then we need $s_e \cdot l_e \ll s_i \cdot l_i$, where l_e and l_i are the sizes of the bit strings transmitted during the estimation and identification phases, respectively.

3.3 The Impact of the Inter-Cycle Overhead

The MAC protocol for the C1G2 system is based on slotted ALOHA. In order to accurately estimate the size of a specified set of tags, conventionally, the reader should issue multiple query cycles over the same set of tags and take the average of the estimates. The inter-cycle overhead consists of the time between cycles when the reader is powered down, and the carrier time used to power the tags before beginning communication. According to the experiment results in [30], which are conducted in realistic settings, these times are 40 ms and 3 ms respectively, while the average time interval per slot is $1 \sim 2$ ms.

TABLE 1
The Average Time Interval for Various Slots

	after *QueryRep* command	after *Query* command
empty slot	0.9 ms	1.7 ms
singleton slot	4.1 ms	5.1 ms
collision slot	1.3 ms	2.2 ms

We have further measured the time interval for various slots and the inter-cycle duration with the USRP N210 platform. In our experiments, we use the Alien-9900 reader and Alien-9611 linear antenna with a directional gain of 6 dB. The RFID tags used are Alien 9640 general-purpose tags which support the EPC C1G2 standards. We use Alien reader to continuously read 13 tags for 100 query cycles. We use USRP N210 as a sniffer to capture the physical signals. Fig. 2 shows an example of the captured raw signal data of the interrogation between the reader and the tag. According to the realistic experiment results in this setting, the average intervals for various slots are summarized in Table 1. It is found that, in most cases, the slot is started with a *QueryRep* command, then the average interval for empty slots is 0.9 ms per slot, the average interval for singleton slots is 4.1 ms per slot, and the average interval for collision slots is 1.3 ms per slot; when a slot happens to be the first slot of a frame, the slot is started with a *Query* command, then the average interval for empty slots is 1.7 ms per slot, the average interval for singleton is 5.1 ms per slot, and the average interval for collision slots is 2.2 ms per slot. By measuring the time intervals between two adjacent query cycles, it is found that the average interval for inter-cycle duration is 28.3 ms. Note that if the powered-down interval is not long enough, it is possible that some surrounding tags will maintain the former state for the inventoried flag with their local residual power, which causes them to keep silent in the upcoming query cycle.

Therefore, since the average inter-cycle duration (28.3 ms) is much larger than the average time interval of conventional slots (empty slot: 0.9 ms, singleton slot: 4.1 ms, collision slot: 1.3 ms), the inter-cycle duration must be taken into account when considering overall reading performance. It is obvious that reading a large number of tags per cycle amortizes the cost of inter-cycle overhead, resulting in lower per tag reading time, while for small tag sets the inter-cycle overhead is significant. It is essential to sufficiently reduce the inter-cycle overhead when we design a solution and set the corresponding parameters for RFID systems.

4 Problem Formulation

Suppose there are a large number of tags in the effective scanning area of the RFID reader, the RFID system conforms

to EPCglobal C1G2 standards, i.e., the slotted ALOHA-based anti-collision scheme [4], [6] is used in the system model. The objective is to collect the histogram over RFID tags according to some categorized metric, e.g, the type of merchandise, while the present set of category IDs cannot be predicted in advance. As we aim at a dynamic environment where the tags may frequently enter and leave the scanning area, a time-efficient strategy must be proposed. Therefore, the specified accuracy can be relaxed in order to quickly collect the histogram. Assume that the overall tag size is n, there exist m categories $C = \{C_1, C_2, \ldots, C_m\}$, and the actual tag size for each category is $n_1, n_2, \ldots, n_m$.

In the *Basic Histogram Collection*, the RFID system needs to collect the histogram for *all categories*. Due to the inherent inaccurate property for RFID systems, users can specify the accuracy requirement for the histogram collection. Suppose the estimated tag size for category $C_i (1 \leq i \leq m)$ is $\widehat{n_i}$, then the following accuracy constraint should be satisfied:

$$Pr[|\widehat{n_i} - n_i| \leq \epsilon \cdot n_i] \geq 1 - \beta \quad accuracy\ constraint. \tag{1}$$

The accuracy constraint illustrates that, given the exact tag size n_i for a specified category, the estimated tag size $\widehat{n_i}$ should be in an confidence interval of width $2\epsilon \cdot n_i$, i.e., $\frac{\widehat{n_i}}{n_i} \in [1-\epsilon, 1+\epsilon]$ with probability greater than $1-\beta$. For example, if $\epsilon = 0.1, \beta = 0.05$, then in regard to a category with tag size $n_i = 100$, the estimated tag size $\widehat{n_i}$ should be within the range [90,110] with probability greater than 95 percent.

In the *Iceberg Query Problem*, only those categories with a tag size over a specified threshold t are essential to be illustrated in the histogram, while the accuracy requirement is satisfied. As the exact tag size n_i for category C_i is unknown, then, given the estimated value of tag size $\widehat{n_i}$, it is possible to have false negative error and false positive error in verifying the population constraint. Therefore, it is essential to guarantee that the false negative/positive rate is below β, that is:

$$Pr[\widehat{n_i} < t | n_i \geq t] < \beta, \tag{2}$$

$$Pr[\widehat{n_i} \geq t | n_i < t] < \beta. \tag{3}$$

In the *Top-k Query Problem*, we use the definition of the probabilistic threshold top-k query (*PT-Topk query*), i.e., in regard to the tag size, only the set of categories where each takes a probability of at least $1-\beta$ to be in the top-k list are illustrated in the histogram, while the accuracy requirement is satisfied. Much like the iceberg query problem, as the exact tag size n_i for category C_i is unknown, then, given the estimated value of tag size $\widehat{n_i}$, it is possible to have false negative error and false positive error in verifying the population constraint, the following constraint must be satisfied:

$$Pr[C_i \text{ is regarded out of top-}k \text{ list} \,|\, C_i \in \text{top-}k \text{ list}] < \beta, \tag{4}$$

$$Pr[C_i \text{ is regarded in top-}k \text{ list} \,|\, C_i \notin \text{top-}k \text{ list}] < \beta. \tag{5}$$

In this paper, we aim to propose a series of novel solutions to tackle the above problems while satisfying the following properties: (1) Time-efficient. (2) Simple for the tag side in the protocol. (3) Complies with the EPCglobal C1G2 standards. Therefore, in order for the proposed algorithm to work, we only require the tags to comply with the current C1G2 standards: each tag has a unique 96-bit ID in its EPC memory, where the first s binary bits are regarded as the category ID ($1 < s < 96$). According to the C1G2 standard, the reader is able to select the tags in a specified category by sending a *Select* command with an s-bit mask in the category ID field. If multiple categories need to be selected, the reader can provide multiple bit masks in the *Select* command.

5 Use Ensemble Sampling to Collect Histograms

When collecting the histograms over a large number of categories, the objective is to minimize the overall scanning time while the corresponding accuracy/population constraints are satisfied. Two straightforward solutions are summarized as follows: (1) *Basic Tag Identification*: The histogram is collected by uniquely identifying each tag from the massive tag set and putting it into the corresponding categories, thus the accuracy is 100 percent, and (2) *Separate Counting*: As the category IDs cannot be predicted in advance, the *tree traversal* method [31] is used to obtain the category IDs. Then, the reader sends a *Select* command to the tags, and it activates the tags in the specified category by providing a bit mask over the category ID in the command. According to the replies from the specified tags, the estimators such as [9], [24], [32] can be used to estimate the tag size for each category. As the rough tag size for each category cannot be predicted in advance, a fixed initial frame size is used for each category.

Both the above two solutions are not time-efficient. In regard to the basic tag identification, uniquely identifying each tag in the massive set is not scalable, for as the tag size grows into a huge number, the scanning time can be an unacceptable value. In regard to the separated counting, the reader needs to scan each category with at least one query cycle, even if the category is a minor category, which is not necessarily addressed in the iceberg query and the top-k query. As the number of categories m can be fairly large, e.g., in the order of hundreds, the *Select* command and the fixed initial frame size for each category, as well as the inter-cycle overhead among a large number of query cycles, make the overall scanning time rather large.

Therefore, we consider an ensemble sampling-based estimation scheme as follows: select a certain number of categories and issue a query cycle, obtain the empty/singleton/collision slots, and then estimate the tag size for each of the categories according to the sampling in the singleton slots. In this way, the ensemble sampling is more preferred than the separate counting in terms of reading performance. Since more tags are involved in one query cycle, more slots amortize the cost of inter-cycle overhead, the *Select* command, as well as the fixed initial frame size. Thus, the overall scanning time can be greatly reduced.

5.1 The Estimator ES

In the slotted ALOHA-based protocol, besides the empty slots and the collision slots, the singleton slots can be obtained. In the ensemble sampling-based estimation, according to the observed statistics of the empty/singleton/collision slots, we

can use estimators in [9], [24], [32] etc. to estimate the overall tag size. Then, according to the response in each singleton slot, the category ID is obtained from the first s bits in the tag ID. Based on the sampling from the singleton slots, the tag size for each category can be estimated. The reason is as follows:

Assume that there exists m categories $C_1, C_2, \ldots, C_m$, the overall tag size is n, and the tag size for each category is $n_1, n_2, \ldots, n_m$. We define an indicator variable $X_{i,j}$ to denote whether one tag of category C_i selects a slot j inside the frame with the size f. We set $X_{i,j} = 1$ if only one tag of category C_i selects the slot j, and $X_{i,j} = 0$ otherwise. Moreover, we use $Pr[X_{i,j} = 1]$ to denote the probability that only one tag of category C_i selects the slot j, then,

$$Pr[X_{i,j} = 1] = \frac{1}{f} \cdot \left(1 - \frac{1}{f}\right)^{n-1} \cdot n_i.$$

If we use $n_{s,i}$ to denote the number of singleton slots selected by tags of category C_i, thus $n_{s,i} = \sum_{j=1}^{f} X_{i,j}$, then, the expected value

$$E(n_{s,i}) = \sum_{j=1}^{f} Pr[X_{i,j} = 1] = \left(1 - \frac{1}{f}\right)^{n-1} \cdot n_i.$$

Furthermore, let n_s denote the number of singleton slots, the expected value $E(n_s) = (1 - \frac{1}{f})^{n-1} \cdot n$. Then, $\frac{E(n_{s,i})}{E(n_s)} = \frac{n_i}{n}$. Thus we can approximate the tag size of category C_i as follows:

$$\widehat{n_i} = \frac{n_{s,i}}{n_s} \cdot \widehat{n}. \tag{6}$$

Here, $\widehat{n}$ is the estimated value of the overall tag size. Let $\widehat{\alpha_i} = \frac{n_{s,i}}{n_s}$, then $\widehat{n_i} = \widehat{\alpha_i} \cdot \widehat{n}$.

5.2 Accuracy Analysis

5.2.1 Accuracy of the ES Estimator

In the ensemble sampling-based estimation, since the estimators such as [9], [24], [32] can be utilized for estimating the overall number of tags, we use δ to denote the variance of $\widehat{n}$. We have the property in Lemma 1.

Lemma 1. *The number of singleton slots n_s and the number of singleton slots $n_{s,i}$ selected by the tags of category C_i, respectively, have the following expectations:*

$$\begin{cases} E(n_s^2) = \left(1 - \frac{1}{f}\right)^{n-1} \cdot n + \frac{f-1}{f} \cdot \left(1 - \frac{2}{f}\right)^{n-2} \cdot (n^2 - n), \\ E(n_{s,i}^2) = \left(1 - \frac{1}{f}\right)^{n-1} \cdot n_i + \frac{f-1}{f} \cdot \left(1 - \frac{2}{f}\right)^{n-2} \cdot (n_i^2 - n_i). \end{cases}$$

Proof. See Appendix A, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2014.2357021. □

We rely on the following theorem to illustrate the accuracy of the estimator SE.

Theorem 1. *Let δ_i represent the variance of the estimator SE $\widehat{n_i}$, the load factor $\rho = \frac{n}{f}$, then,*

$$\delta_i = \frac{n_i}{n} \cdot \frac{e^\rho + n_i - 1}{e^\rho + n - 1} \cdot (\delta + n^2) - n_i^2. \tag{7}$$

Proof. See Appendix B, available in the online supplemental material. □

5.2.2 Reducing the Variance through Repeated Tests

As the frame size for each query cycle has a maximum value, by estimating from the ensemble sampling within only one query cycle, the estimated tag size may not be accurate enough for the accuracy constraint. In this situation, multiple query cycles are essential to reduce the variance through repeated tests. Suppose the reader issues l query cycles over the same set of categories, in regard to a specified category C_i, by utilizing the weighted statistical averaging method, the averaged tag size $\widehat{n_i} = \sum_{k=1}^{l} \omega_k \cdot \widehat{n_{i,k}}$; here $\omega_k = \frac{\frac{1}{\delta_{i,k}}}{\sum_{k=1}^{l} \frac{1}{\delta_{i,k}}}$, $\widehat{n_{i,k}}$ and $\delta_{i,k}$ respectively denote the estimated tag size and variance for each cycle k. Then, the variance of $\widehat{n_i}$ is $\sigma_i^2 = \frac{1}{\sum_{k=1}^{l} \frac{1}{\delta_{i,k}}}$.

Therefore, according to the accuracy constraint in the problem formulation, we rely on the following theorem to express this constraint in the form of the variance.

Theorem 2. *Suppose the variance of the averaged tag size $\widehat{n_i}$ is σ_i^2. The* accuracy constraint *is satisfied for a specified category C_i, as long as $\sigma_i^2 \le (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot n_i^2$, $Z_{1-\beta/2}$ is the $1 - \frac{\beta}{2}$ percentile for the standard normal distribution.*

Proof. See Appendix C, available in the online supplemental material. □

According to Theorem 2, we can verify if the accuracy constraint is satisfied for each category through directly checking the variance against the threshold $(\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot n_i^2$. If $1 - \beta = 95\%$, then $Z_{1-\beta/2} = 1.96$.

5.2.3 Property of the Ensemble Sampling

According to Theorem 1, the normalized variance of the SE estimator $\lambda_i = \frac{\delta_i}{n_i}$ is equivalent to $\lambda_i = \frac{\delta - n \cdot e^\rho + n}{e^\rho + n - 1} \cdot \frac{n_i}{n} + \frac{(\delta + n^2)(e^\rho - 1)}{n \cdot (e^\rho + n - 1)}$. Let $a = \frac{\delta - n \cdot e^\rho + n}{e^\rho + n - 1}$, $b = \frac{(\delta + n^2)(e^\rho - 1)}{n \cdot (e^\rho + n - 1)}$. Then, the normalized variance $\lambda_i = a \cdot \frac{n_i}{n} + b$. Since the SE estimator can utilize any estimator like [9], [24], [32] to estimate the overall tag size, then, without loss of generality, if we use the estimator in [9], we can prove that $a < 0$ for any value of $n > 0, f > 0$. The following theorem shows this property in the normalized variance.

Theorem 3. $\frac{\delta - n \cdot e^\rho + n}{e^\rho + n - 1} < 0$ *for any value of $n > 0, f > 0$.*

Proof. See Appendix D, available in the online supplemental material. □

This property applies to any estimator with variance smaller than δ_0 in *ZE*, which simply estimates the overall tag size based on the observed number of empty slots.

According to Theorem 3, in order to satisfy the *accuracy constraint*, we should ensure $\lambda_i \le (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot n_i$. As $a < 0$ for all values of f, it infers that the larger the value n_i is, the faster it will be for the specified category to satisfy the accuracy constraint. On the contrary, the smaller the value n_i is, the slower it will be for the specified category to satisfy the accuracy

constraint. This occurs during the ensemble sampling, when the major categories occupy most of the singleton slots, while those minor categories cannot obtain enough samplings in the singleton slots for an accurate estimation of the tag size.

5.3 Compute the Optimal Granularity for Ensemble Sampling

As indicated in the above analysis, during a query cycle of the ensemble sampling, in order to achieve the accuracy requirement for all categories, the essential scanning time mainly depends on the category with the smallest tag size, as the other categories must still be involved in the query cycle until this category achieves the accuracy requirement. Therefore, we use the notion *group* to define a set of categories involved in a query cycle of the ensemble sampling. Hence, each cycle of ensemble sampling should be applied over an appropriate group, such that the variance of the tag sizes for the involved categories cannot be too large. In this way, all categories in the same group achieve the accuracy requirement with very close finishing time. In addition, according to Eq. (7), as the number of categories increases in the ensemble sampling, the load factor ρ is increased, then the achieved accuracy for each involved category is reduced. Therefore, it is essential to compute an optimal granularity for the group in regard to the reading performance. Suppose there exists m categories in total, the objective is to divide them into $d(1 \leq d \leq m)$ groups for ensemble sampling, such that the overall scanning time can be minimized while achieving the accuracy requirement.

For a specified group, in order for all involved categories to satisfy the accuracy requirement, it is essential to compute the required frame size for the category with the smallest tag size, say n_i. Let $t_i = (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot n_i$, then according to Theorem 2, we can compute the essential frame size f such that $\lambda_i(f) \leq t_i$. Assume that the inter-cycle overhead is τ_c, the average time interval per slot is τ_s. Therefore, if $f \leq f_{max}$, then the total scanning time $T = f \cdot \tau_s + \tau_c$. Otherwise, if the final estimate is the average of r independent experiments each with an estimator variance of $\lambda_i(f_{max})$, then the variance of the average is $\frac{\lambda_i(f_{max})}{r}$. Hence, if we want the final variance to be t_i, then r should be $\frac{\lambda_i(f_{max})}{t_i}$, the total scanning time is $T = (f_{max} \cdot \tau_s + \tau_c) \cdot r$.

We propose a dynamic programming-based algorithm to compute the optimal granularity for ensemble sampling. Assume that currently there are m categories ranked in non-increasing order according to the estimated tag size, e.g., $C_1, C_2, \ldots, C_m$. We need to cut the ranked categories into one or more continuous groups for ensemble sampling. In regard to a single group consisting of categories from C_i to C_j, we define $t(i,j)$ as the essential scanning time for ensemble sampling, which is computed in the same way as the aforementioned T. Furthermore, we define $T(i,j)$ as the minimum overall scanning time over the categories from C_i to C_j among various grouping strategies. Then, the recursive expression of $T(i,j)$ is shown in Eq. (8):

$$T(i,j) = \begin{cases} \min_{i \leq k \leq j}\{t(i,k) + T(k+1,j)\}, & i < j, \\ t(i,i), & i = j. \end{cases} \quad (8)$$

In Eq. (8), the value of $T(i,j)$ is obtained by enumerating each possible combination of $t(i,k)$ and $T(k+1,j)$, and then getting the minimum value of $t(i,k) + T(k+1,j)$. By solving the overlapping subproblems in $T(i,j)$, the optimization problem is then reduced to computing the value of $T(1,m)$.

For example, suppose there are a set of tags with 10 categories, these categories are ranked in non-increasing order of the estimated tag size, say, {100, 80, 75, 41, 35, 30, 20, 15, 12, 8}, then they are finally divided into three groups for ensemble sampling according to the dynamic programming, i.e., {100,80,75}, {41,35,30}, and {20,15,12,8}. In this way, the tag sizes of each category inside one group are close to each other, during the ensemble sampling all categories in the same group can achieve the accuracy requirement with very close finishing time, very few slots are wasted due to waiting for those, comparatively speaking, minor categories. On the other hand, these categories are put together with an appropriate granularity for ensemble sampling to sufficiently amortize the fixed time cost for each query cycle.

5.4 The Ensemble Sampling-Based Algorithm

In Algorithm 1, we propose an ensemble sampling-based algorithm for the *basic histogram collection*. In the beginning, as the overall number of tags n cannot be predicted, in order to accomodate a large operating range up to $\overline{n}$, we need to set the initial frame size f by solving $fe^{-\overline{n}/f} = 5$ as suggested in [9]. Then, during each cycle of ensemble sampling, we find the category with the largest population υ in the singleton slots, and set a threshold $n_{s,i} > \upsilon \cdot \theta (0 < \theta < 1)$ to filter out those minor categories which occasionally occupy a small number of singleton slots. For example, suppose it is observed from the singleton slots that the number of slots occupied by various categories are as follows: {35, 25, 10, 5, 3, 1}, if θ is set to 0.1, then the categories with the number of slots equal to 3 and 1 are filtered out from the next ensemble sampling. Therefore, during the ensemble sampling, we can avoid estimating tag sizes for those minor categories with a rather large variance. Then, the involved categories are further divided into smaller groups based on the dynamic programming. Therefore, as those major categories are estimated and wiped out from the set R phase by phase, all categories including the relatively minor categories can be accurately estimated in terms of tag size. The query cycles continue to be issued until no singleton slots or collision slots exist.

6 Ensemble Sampling for the Iceberg Query

6.1 Motivation

In some applications, the users only pay attention to the major categories with the tag sizes above a certain threshold t, while those minor categories are not necessarily addressed. Then, the *iceberg query* [33] is utilized to filter out those categories below the threshold t in terms of the tag size. In this situation, the *separate counting* scheme is especially not suitable, since most of the categories are not within the scope of the concern, which can be wiped out together immediately.

According to the definition in the problem formulation, three constraints for the *iceberg query* must be satisfied:

$$\begin{aligned} &Pr[|\widehat{n_i} - n_i| \leq \epsilon \cdot n_i] \geq 1 - \beta && \textit{accuracy constraint}, \\ &Pr[\widehat{n_i} < t | n_i \geq t] < \beta && \textit{population constraint}, \\ &Pr[\widehat{n_i} \geq t | n_i < t] < \beta && \textit{population constraint}. \end{aligned}$$

Algorithm 1. Algorithm for Histogram Collection

1: INPUT: 1. Upper bound $\overline{n}$ on the number of tags n
2: 2. Confidence interval width ϵ
3: 3. Error probability β
4: Initialize the set R to all tags. Set $l = 1$.
5: **while** $n_s \neq 0 \wedge n_c \neq 0$ **do**
6: If $l = 1$, compute the initial frame size f by solving $fe^{-\overline{n}/f} = 5$. Otherwise, compute the frame size $f = \widehat{n}$. If $f > f_{max}$, set $f = f_{max}$.
7: Set S to $\varnothing$. Select the tags in R and issue a query cycle with the frame size f, get n_0, n_c, n_s. Find the category with the largest population υ in the singleton slots. For each category which appears in the singleton slot with population $n_{s,i} > \upsilon \cdot \theta(\theta$ is constant, $0 < \theta < 1)$, add it to the set S. Estimate the tag size n_i for each category $C_i \in S$ using the *SE* estimator. Compute the variances δ_i' for each category $C_i \in S$ according to Eq. (7).
8: Rank the categories in S in non-increasing order of the tag size. Divide the set S into groups $S_1, S_2, \ldots, S_d$ according to the dynamic programming-based method.
9: **for** each $S_j \in S(1 \leq j \leq d)$ **do**
10: For each category $C_i \in S_j$, compute the frame size f_i from δ_i by solving $\frac{1}{1/\delta_i' + 1/\delta_i} \leq (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot \widehat{n_i}^2$.
11: Obtain the maximum frame size $f = \max_{C_i \in S_j} f_i$. If $f < f_{max}$, select all categories in S_j, and issue a query cycle with frame size f. Otherwise, select all categories in S_j, and issue r query cycles with the frame size f_{max}. Wipe out the categories with satisfied accuracy after each query cycle.
12: Estimate the tag size $\widehat{n_i}$ for each category $C_i \in S_j$, illustrate them in the histogram.
13: **end for**
14: $\widehat{n} = \widehat{n} - \sum_{C_i \in S} \widehat{n_i}$. $R = R - S$. $S = \varnothing$. $l = l + 1$.
15: **end while**

The first constraint is the *accuracy constraint*, while the second and third constraints are the *population constraints*. In regard to the accuracy constraint, we have demonstrated in Theorem 2 that it can be expressed in the form of the variance constraint. In regard to the population constraint, the second constraint infers that, in the results of the iceberg query, the false negative probability should be no more than β, while the third constraint infers that the false positive probability should be no more than β. We rely on the following theorem to express the population constraint in another equivalent form.

Theorem 4. *The two population constraints, $Pr[\widehat{n_i} < t | n_i \geq t] < \beta$ and $Pr[\widehat{n_i} \geq t | n_i < t] < \beta$, are satisfied as long as the standard deviation of the averaged tag size $\sigma_i \leq \frac{|n_i - t|}{\Phi^{-1}(1-\beta)}$, $\Phi(x)$ is the cumulative distribution function of the standard normal distribution.*

Proof. See Appendix E, available in the online supplemental material. □

In order to better illustrate the inherent principle, Fig. 3 shows an example of the histogram with the $1 - \beta$ confidence interval annotated, the y-axis denotes the estimated tag size for each category. In order to accurately verify the population constraint, it is required that the variance of the estimated tag size should be small enough. Note that when

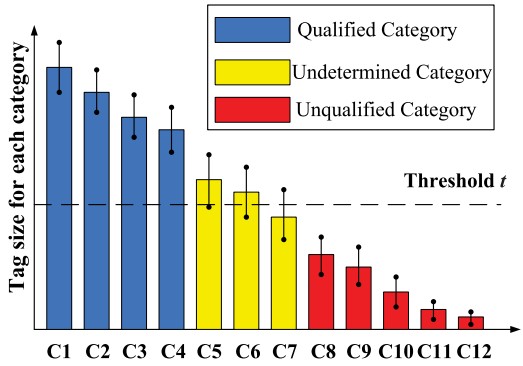


Fig. 3. Histogram with confidence interval annotated.

the $1 - \beta$ confidence interval of the tag size $\widehat{n_i}$ is above/below the threshold t, the specified category can be respectively identified as qualified/unqualified, as both the false positive and false negative probabilities are less than β; otherwise, the specified category is still undetermined. According to the weighted statistical averaging method, as the number of repeated tests increases, the averaged variance σ_i for each category decreases, thus the confidence interval for each category is shrinking. Therefore, after a certain number of query cycles, all categories can be determined as qualified/unqualified for the population constraint.

Note that when the estimated value $\widehat{n_i} \gg t$ or $\widehat{n_i} \ll t$, the required variance in the population constraint is much larger than the specifications of the accuracy constraint. In this situation, these categories can be quickly identified as qualified/unqualified, and can be wiped out immediately from the ensemble sampling for verifying the population constraint. Thus, those undetermined categories can be further involved in the ensemble sampling with a much smaller tag size, verifying the population constraint in a faster approach.

Sometimes the tag sizes of various categories are subject to some skew distributions with a "long tail". The long tail represents those categories each of which occupies a rather small percentage among the total categories, but all together they occupy a substantial proportion of the overall tag sizes. In regard to the iceberg query, conventionally the categories in the long tail are unqualified for the population constraint. However, due to the small tag size, most of them may not have the opportunity to occupy even one singleton slot when contending with those major categories during the ensemble sampling. They remain undetermined without being immediately wiped out, leading to inefficiency in scanning the other categories. We rely on the following theorem to quickly wipe out the categories in the long tail.

Theorem 5. *For any two categories C_i and C_j that $n_{s,i} < n_{s,j}$ satisfies for each query cycle of ensemble sampling, if C_j is determined to be unqualified for the population constraint, then C_i is also unqualified.*

Proof. See Appendix F, available in the online supplemental material. □

According to Theorem 5, after a number of query cycles of ensemble sampling, if a category C_j is determined unqualified for the population constraint, then for any category C_i which has not appeared once in the singleton slots, $n_{s,j} > n_{s,i} = 0$, it can be wiped out immediately as an unqualified category.

6.2 Algorithm for the Iceberg Query Problem

We propose the algorithm for the iceberg query problem in Algorithm 2. Assume that the current set of categories is R, during the query cycles of ensemble sampling, the reader continuously updates the statistical value of $\widehat{n}_i$ as well as the standard deviation σ_i for each category $C_i \in R$. After each query cycle, the categories in R can be further divided into the following categories according to the population constraint:

- Qualified categories Q: If $\widehat{n}_i \geq t$ and $\sigma_i \leq \frac{\widehat{n}_i - t}{\Phi^{-1}(1-\beta)}$, then category C_i is identified as qualified for the population constraint.
- Unqualified categories U: If $\widehat{n}_i < t$ and $\sigma_i \leq \frac{t - \widehat{n}_i}{\Phi^{-1}(1-\beta)}$, then category C_i is identified as unqualified for the population constraint.
- Undetermined categories R: The remaining categories to be verified are undetermined categories.

Algorithm 2. Algorithm for Iceberg Query

1: INPUT: 1. Upper bound $\overline{n}$ on the number of tags n
2: 2. Confidence interval width ϵ
3: 3. Threshold t
4: 4. Error probability β
5: Initialize R to all categories, set Q, U, V to $\varnothing$. Set $l = 1$.
6: **while** $R \neq \varnothing$ **do**
7: If $l = 1$, compute the initial frame size f by solving $fe^{-\overline{n}/f} = 5$. Otherwise, compute the frame size $f = \widehat{n}$. If $f > f_{max}$, set $f = f_{max}$.
8: Set S to $\varnothing$. Select the tags in R and issue a query cycle with frame size f, get n_0, n_c, n_s. Find the category with the largest population υ in the singleton slots. For each category which appears in the singleton slot with population $n_{s,i} > \upsilon \cdot \theta (\theta$ is constant, $0 < \theta < 1)$, add it to the set S. If $\upsilon \cdot \theta < 1$, then add all remaining categories into S. Set $S' = S$. $l = 1$.
9: **while** $S \neq \varnothing$ **do**
10: Compute the frame size f_i for each category $C_i \in S$ such that the variance $\sigma_i = \frac{|t - \widehat{n}_i|}{\Phi^{-1}(1-\beta)}$. If $f_i > \widehat{n}_i \cdot e$, then remove C_i from S to V. If $f_i > f_{max}$, set $f_i = f_{max}$. Obtain the frame size f as the mid-value among the series of f_i.
11: Select all tags in S, issue a query cycle with the frame size f, compute the estimated tag size $\widehat{n}_i$ and the averaged standard deviation σ_i for each category $C_i \in S$. Detect the qualified category set Q and unqualified category set U. Set $S = S - Q - U$.
12: **if** $U \neq \varnothing$ **then**
13: Wipe out all categories unexplored in the singleton slots from S.
14: **end if**
15: **end while**
16: $\widehat{n} = \widehat{n} - \sum_{C_i \in S'} \widehat{n}_i$. $R = R - S'$, $l = l + 1$.
17: **end while**
18: Further verify the categories in V and Q for the accuracy constraint.

Therefore, after each query cycle of ensemble sampling, those *unqualified categories* and *qualified categories* can be immediately wiped out from the ensemble sampling. When at least one category is determined as unqualified, all of the categories in the current group which have not been explored in the singleton slots are wiped out immediately. The query cycles are then continuously issued over those *undetermined categories* in R until $R = \varnothing$.

For example, suppose the threshold is set to 30, after a query cycle of ensemble sampling, the estimated number of tags for each category is as follows: {120, 80, 65, 35, 28, 10, 8}, according to the standard deviation of estimation for various categories, then the categories with estimated tag size of 120, 80 and 65 can be immediately determined as qualified, the categories with estimated tag size of 10 and 8 can be also immediately determined as unqualified, for those categories with estimated tag size 35 and 28, due to the current estimation error, we cannot yet determine if they are exactly qualified or unqualified, thus another cycle of ensemble sampling is required for further verification.

During the ensemble sampling, if there exist some categories with tag sizes very close to the threshold t, then the required number of slots to verify the population constraint can be rather large. Thus, we compute the essential frame size f_i for each category C_i and compare it with the expected number of slots $\widehat{n}_i \cdot e$ in basic tag identification. If $f_i > \widehat{n}_i \cdot e$, then the category is removed from the set S to V. We heuristically set the frame size f to the mid-value among the series of f_i, such that after a query cycle, about half of the categories can be determined as qualified/unqualified, and thus wiped out quickly. Therefore, after the while loop, for each category $C_i \in V$, basic identification is used to obtain the exact tag size n_i. If $n_i \geq t$, C_i is illustrated in the histogram. For each category $C_i \in Q$, the reader verifies if it has satisfied the accuracy requirement; if so, C_i is illustrated in the histogram and wiped out from Q. Then, ensemble sampling is further applied over the categories in Q to satisfy the accuracy requirement by using the optimized grouping method.

7 Ensemble Sampling for the Top-k Query

7.1 Motivation

In some applications, when the number of categories is fairly large, the users only focus on the major categories in the top-k list in regard to the tag size. Then the *top-k query* is utilized to filter out those categories out of the top-k list. In this situation, the *separate counting* scheme is especially not suitable. If the specified category is not in the top-k list, it is unnecessary to address it for accurate tag size estimation. However, since the threshold t for the top-k list cannot be known in advance, the *separate counting* scheme cannot quickly decide which categories can be wiped out immediately.

Moreover, when the distribution around the kth ranking is fairly even, i.e., the size of each category is very close, it is rather difficult to determine which categories belong to the top-k categories. Based on this understanding, we utilize the probabilistic threshold top-k query (*PT-Topk query*) to return a set of categories Q where each takes a probability of at least $1 - \beta (0 < \beta \leq 1)$ to be in the top-k list. Therefore, the size of Q is not necessarily going to be exactly k.

Hence, as the exact value of tag size n_i is unknown, in order to define $Pr[C_i \in \text{top-}k \text{ list}]$, i.e., the probability that category C_i is within the top-k list in terms of tag size, it is essential to determine a threshold t so that $Pr[C_i \in \text{top-}k \text{ list}] = Pr[n_i \geq t]$. Ideally, t should be the tag size of

the kth largest category; however, it is rather difficult to compute an exact value of t in the estimation scheme due to the randomness in the slotted ALOHA protocol. Therefore, according to the problem formulation in Section 4, we attempt to obtain an estimated value $\widehat{t}$ such that the following constraints are satisfied:

$$\begin{aligned} &Pr[|\widehat{n_i} - n_i| \le \epsilon \cdot n_i] \ge 1-\beta \quad accuracy\ constraint, \\ &Pr[|\widehat{t} - t| \le \epsilon \cdot t] \ge 1-\beta \quad accuracy\ constraint\ of\ \widehat{t}, \quad (9) \\ &Pr[\widehat{n_i} < \widehat{t} | n_i \ge \widehat{t}] < \beta \qquad population\ constraint, \end{aligned}$$

$$Pr[\widehat{n_i} \ge \widehat{t} | n_i < \widehat{t}] < \beta \qquad population\ constraint. \quad (10)$$

Therefore, if the threshold $\widehat{t}$ can be accurately estimated, then the top-k query problem is reduced to the iceberg query problem. The population constraints (9) and (10) are respectively equivalent to the population constraints (4) and (5). Then it is essential to quickly determine the value of the threshold $\widehat{t}$ while satisfying the constraint $Pr[|\widehat{t} - t| \le \epsilon \cdot t] \ge 1-\beta$. We rely on the following theorem to express the above constraint in the form of the variance.

Theorem 6. *The constraint $Pr[|\widehat{t} - t| \le \epsilon \cdot t] \ge 1-\beta$ is satisfied as long as $Var(\widehat{t} - t) \le \epsilon^2 \cdot t^2 \cdot \beta$.*

Proof. See Appendix G, available in the online supplemental material. □

7.2 Algorithm

According to Theorem 6, we utilize the ensemble sampling to quickly estimate the threshold $\widehat{t}$. The intuition is as follows: after the first query cycle of ensemble sampling, we can estimate a confidence interval $[t_{low}, t_{up}]$ of the threshold t according to the sampled distribution. Then, by wiping out those categories which are obviously qualified or unqualified to be in the top-k list, the width of the confidence interval can be quickly reduced. As the approximated threshold $\widehat{t}$ is selected within the confidence interval, after a number of query cycles of ensemble sampling, when the width is below a certain threshold, the estimated value $\widehat{t}$ can be close enough to the exact threshold t.

Based on the above analysis, we propose an algorithm for the top-k query problem in Algorithm 3. In the beginning, a while loop is utilized to quickly identify an approximate value $\widehat{t}$ for the threshold t. Suppose that the averaged estimated tag size and standard deviation for each category C_i are respectively $\widehat{n_i}$ and σ_i, if we use p to denote a small constant value between 0 and 1, let $\eta = \Phi^{-1}(1-\frac{p}{2})$. Then, given a fixed value of p, the $1-p$ confidence interval for n_i is $[\widehat{n_i} - \eta \cdot \sigma_i, \widehat{n_i} + \eta \cdot \sigma_i]$. For each iteration, we respectively determine an upper bound t_{up} and a lower bound t_{low} for the threshold t, according to the kth largest category in the current ranking. Then, we respectively wipe out those qualified and unqualified categories according to the upper bound t_{up} and a lower bound t_{low}. The value of k is then decreased by the number of qualified categories. In this way, the threshold t is guaranteed to be within the range $[t_{low}, t_{up}]$ with a probability of at least $1-p$. When $p \to 0$, then $t \in [t_{low}, t_{up}]$ with the probability close to 100 percent. Moreover, an estimated threshold $\widehat{t}$ is also selected within this range. Therefore, let the width $g = t_{up} - t_{low}$, then the variance of $\widehat{t} - t$ is at most g^2. In order to guarantee that $Var(\widehat{t} - t) \le \epsilon^2 \cdot t^2 \cdot \beta$, it is essential to ensure $g^2 \le \epsilon^2 \cdot t^2 \cdot \beta$. As the ensemble sampling is continuously issued over the categories in R, the standard deviation σ_i for each category $C_i \in R$ is continuously decreasing. Furthermore, as the qualified/unqualified categories are continuously wiped out, the upper bound t_{up} is continuously decreasing while the lower bound t_{low} is continuously increasing. The width of the range $[t_{low}, t_{up}]$ is continuously decreasing. The while loop continues until $g^2 \le \epsilon^2 \cdot t^2 \cdot \beta$. Then, after the estimated threshold $\widehat{t}$ is computed, the iceberg query is further applied over those categories with the threshold $\widehat{t}$.

Algorithm 3. Algorithm for *PT-Topk* Query Problem

1: INPUT: 1. Upper bound $\overline{n}$ on the number of tags n
2: 2. Confidence interval width ϵ
3: 3. The value of k
4: 4. Error probability β
5: Initialize R to all categories, set $l = 1$, $\eta = \Phi^{-1}(1-\frac{p}{2})$.
6: **while** true **do**
7: Issue a query cycle to apply ensemble sampling over all categories in R. Compute the statistical average value and standard deviations as $\widehat{n_i}$ and σ_i.
8: Rank the categories in R according to the value of $\widehat{n_i} + \eta \cdot \sigma_i$ for each identified category C_i. Find the k-th largest category C_i, set $t_{up} = \widehat{n_i} + \eta \cdot \sigma_i$. Detect the qualified categories Q with threshold t_{up}.
9: Rank the categories in R according to the value of $\widehat{n_i} - \eta \cdot \sigma_i$ for each identified category C_i. Find the k-th largest category C_i, set $t_{low} = \widehat{n_i} - \eta \cdot \sigma_i$. Detect the unqualified categories U with threshold t_{low}.
10: Wipe out the qualified/unqualified categories from R. $R = R - Q - U$. Suppose the number of qualified categories in current cycle is q, set $k = k - q$.
11: Rank the categories in R according to the value of $\widehat{n_i}$ for each identified category C_i. Find the k-th largest category C_i, set $\widehat{t} = \widehat{n_i}$. Set $g = t_{up} - t_{low}$. $l = l + 1$.
12: **if** $g^2 \le \epsilon^2 \cdot \beta \cdot \widehat{t}^2$ **then**
13: Break the while loop.
14: **end if**
15: **end while**
16: Apply *iceberg query* with threshold $\widehat{t}$ over the undetermined categories R and the qualified categories Q.

For example, suppose the value of k is 5, after a query cycle of ensemble sampling, the estimated number of tags for various categories is ranked in decreasing order as follows: {C_1:120, C_2:85, C_3:67, C_4:50, C_5:48, C_6:45, C_7:20, C_8:15 }, the threshold t_{up} and t_{low} are respectively set to 68 and 28 according to the fifth largest category, then the categories with tag size 120 and 85 can be determined as qualified categories since their tag sizes are above the threshold t_{up}, the categories with tag size 20 and 15 can be also determined as unqualified categories since their tag sizes are below the threshold t_{low}. Therefore, the remaining categories are as follows: $C3, C4, C_5$ and C_6, we hence need another cycle of ensemble sampling to further verify the threshold according to the third largest category.

8 Discussion on Practical Issues

8.1 Time-Efficiency

As mentioned in the problem formulation, the most critical factor for the histogram collection problem is the time

efficiency. In regard to the basic histogram collection, the time delay is mainly impacted by two factors: 1) the number of categories m, 2) the category with the smallest tag size, say n_i, inside the group for ensemble sampling. Generally, as the number of categories m increases, the number of groups and the essential number of slots for each ensemble sampling is increasing, causing the time delay to increase. Besides, the category with the smallest tag size n_i directly decides the essential frame size inside the group, the larger the gap among the tag sizes of each category in the same group, the lower the time efficiency that is achieved.

In regard to the iceberg query and the top-k query, the time delay mainly depends on the number of categories with the tag size close to the threshold t. Due to the variance in tag size estimation, a relatively large number of slots are required to verify whether the specified categories have tag sizes over the threshold t. For the top-k query, additional time delay is required to estimate the threshold t corresponding to the top-k query.

8.2 Interference Factors in Realistic Settings

In realistic settings of various applications, there might exist several interference factors which hinder the actual performance of histogram collection. These practical issues mainly include path loss, multi-path effect, and mutual interference. In the following we elaborate on the detail techniques to effectively tackle these problems.

Path loss. Path loss is common in RFID-based applications, which may lead to the probabilistic backscattering [7] in RFID systems, even if the tags are placed in the reader's effective scanning range. In such scenario, the tags may reply in each query cycle with a certain probability instead of 100 percent. Therefore, in regard to the tag-counting protocols in our solutions, we need to essentially estimate the probability via statistical tests in the particular application scenarios. In this way, we can accurately estimate the number of tags according to the probability obtained in advance.

Multi-path effect. Multi-path effect is especially common for indoor applications. Due to multi-path effect, some tags cannot be effectively activated as the forwarding waves may offset each other, even in the effective scanning range of RFID systems. To mitigate the multi-path effect, we can use the mobile reader to continuously interrogate the surrounding tags such that the multi-path profile can be continuously changing. In this way, the tags are expected to have more chances to be activated for at least once during the continuous scanning [8].

Mutual interference: If the tags are placed too close, they may have a critical state of mutual interference [34] such that neither of the tags can be effectively activated. This is mainly caused by the coupling effect when the reader's power is adjusted to a certain value. Hence, in order to mitigate the mutual interference among RFID tags, we should skillfully tune the transmission power of the reader so as to avoid the critical state among tags. A suitable power stepping method should be leveraged to sufficiently reduce the mutual interference among all tags.

8.3 Overhead from Tag Identification

In our ensemble sampling-based solution, we conduct efficient sampling over the singleton slots to estimate the number of tags for various categories. However, since the proposed scheme needs to identify the tag in singleton slots and read 96-bit EPC from the tag, it may incur high communication overheard for ensemble sampling. We thus conduct real experiments with the USRP N210 platform to evaluate the ratio of tags that are identified during the whole process of collecting histograms. We respectively test the *slot ratio* (the ratio of the number of singleton slots to total number of slots) and *time ratio* (the ratio of the overall time interval for the singleton slots to total time duration). In the experiment, we use the Alien reader to interrogate 50 tags and use USRP N210 as a sniffer to capture the detailed information in the physical layer, we average the experiment results via 50 repeated test. According to the real experiment results, we find that the average slot ratio is 33 percent, which is lower than 36.8 percent in ideal case when the frame size is set to an optimal value. We further find that the average time ratio is 62 percent, it implies that the singleton slots occupy a considerable proportion of the overall scanning time.

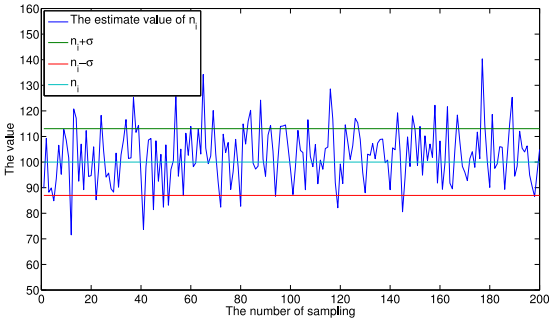
In order to sufficiently reduce the identification overhead in singleton slots, we can make a slight modification for the C1G2 protocol as follows: each tag can embed the category ID into the RN16 response, in this way, during the process of collecting histograms, each tag only need to reply the RN16 random number in the selected slot instead of the exact EPC ID, the high overhead for identification can be effectively avoided. We further evaluate the average time ratio for this new method, we find that the average time ratio can be reduced from 62 to 44 percent, which is much closer to the slot ratio.

9 Performance Evaluation

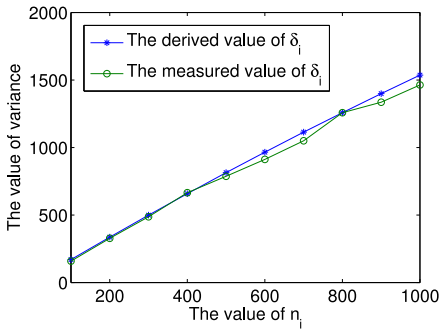
We have conducted simulations in Matlab, and the scenario is as follows: there exist m categories in total, and we randomly generate the tag size for each category according to the normal distribution $N(\mu, \sigma)$. We set the default values for the following parameters: in regard to the accuracy constraint and the population constraint, we set $1 - \beta = 95\%$, and $\epsilon = 0.2$. The average time interval for each slot is $\tau_s = 1\,\text{ms}$, and the inter-cycle overhead is $\tau_c = 43\,\text{ms}$. We compare our solutions with two basic strategies: the basic tag identification (BI) and the separate counting (SC) (explained in Section 5). All results are the averaged results of 500 independent trials.

9.1 Evaluate the Actual Variance in Ensemble Sampling

In order to verify the correctness of the derivation in the variance of the SE estimator, i.e., δ_i in Eq. (7), we conduct simulations and evaluate the actual variances in ensemble sampling, thus quantifying the tightness between the derived value of δ_i and the measured value in simulation studies. We conduct ensemble sampling on 5,500 tags for 200 cycles. For each query cycle, the frame size f is set to 5,500. We look into a category C_i with tag size $n_i = 100$. In Fig. 4a, we plot the estimated value of n_i in each cycle, while the expected values of $n_i - \sigma_i$ and $n_i + \sigma_i$ are respectively illustrated in the red line and the green line. We observe that the estimated value $\widehat{n_i}$ majorly vibrates between the interval $(n_i - \sigma_i, n_i + \sigma_i)$. In Fig. 4b, we further compare the



(a) The vibration of the estimated value $\widehat{n_i}$



(b) δ_i: the derived/measured value

Fig. 4. Evaluate the actual variance in ensemble sampling.

measured value of δ_i with the derived value, varying the tag size of category C_i from 100 to 1,000. As the value of n_i increases, we observe that the gap between the two values are very tight, which infers that the derived value of δ_i used in those performance guarantees can well depict the measured value in a statistical manner.

9.2 The Performance in Basic Histogram Collection

We compare the ensemble sampling with one group (ES) and the ensemble sampling with optimized grouping (ES-g) with the basic strategies. In Fig. 5a, we compare the overall scanning time under three various scenarios. In scenario 1 we set the number of categories $m = 50$, the average tag size $\mu = 50$ and its standard deviation $\sigma = 30$. We observe that the ES strategy has the longest scanning time while the others have fairly small values in scanning time. This is because the variance σ is relatively large as compared to the tag size. The minor categories become the bottleneck in regard to the estimation performance, thus greatly increasing the scanning time. In scenario 2 we set $m = 100$, $\mu = 50$ and $\sigma = 30$. As the number of categories is increased, the scanning time of the separate counting (SC) is apparently increased due to the large inter-cycle overhead and the constant initial frame size in each category. Still, the ES strategy has the longest scanning time. In Scenario 3, we set $m = 100$, $\mu = 500$ and $\sigma = 100$, we observe that the BI has the longest scanning time as the current overall tag size is rather large. The ES strategy now requires a fairly short scanning time as the variance σ is relatively small as compared to μ. Note that in all cases, our optimized solution ES-g always achieves the best performance in terms of scanning time. In Fig. 5b, we compare the scanning time with various values of ϵ in the accuracy constraint. We set $m = 100, \mu = 500, \sigma = 100$. As the value of ϵ is increasing, the scanning time of all solutions, except the BI strategy, is decreasing. Among the four strategies, the ES-g solution always achieves the best performance in scanning time.

In Fig. 5c, we evaluate the impact of the inter-cycle overhead in the strategies. We set $m = 150, \mu = 50, \sigma = 10$. It is known that, by reducing the transmitted bits in singleton slots, the average slot duration can be further reduced, while the inter-cycle overhead is not easily greatly reduced due to the necessity to calm down the activated tags. So we test the overall scanning time with various ratios of τ_c/τ_s. We observe that the BI strategy and the ES strategy have a fairly stable scanning time, as the number of query cycles is relatively small. The separate counting (SC) has a relatively short scanning time when τ_c/τ_s is less than 50. As the value of τ_c/τ_s increases, its scanning time linearly increases and surpasses the other strategies. The ES-g strategy always has the shortest scanning time. In Fig. 5d, we evaluate the scalability of the proposed algorithms while varying the overall number of categories. We set $\mu = 100, \sigma = 20$. Note that while the number of categories increases, the scanning time of each solution grows in a linear approach. Still, the ES-g solution always achieves the minimum scanning time.

9.3 The Performance in Advanced Histogram Collection

We evaluate the performance of our iceberg query algorithm. We use ES to denote our optimized solution based on ensemble sampling. In Fig. 6a we compare the scanning time with various values of threshold ratio θ. We set $m = 200$, $\mu = 200, \sigma = 100$, the exact threshold is set to $t = \theta \cdot \mu$. We observe that as the threshold increases, the scanning time of the SC strategy and the ES strategy is continuously decreasing, while the scanning time for the BI strategy is not affected. In Fig. 6b we compare the scanning time with various standard deviation σ. We set $m = 200, \mu = 200$, and the threshold ratio $\theta = 1.5$. We observe that as the value of σ increases, the

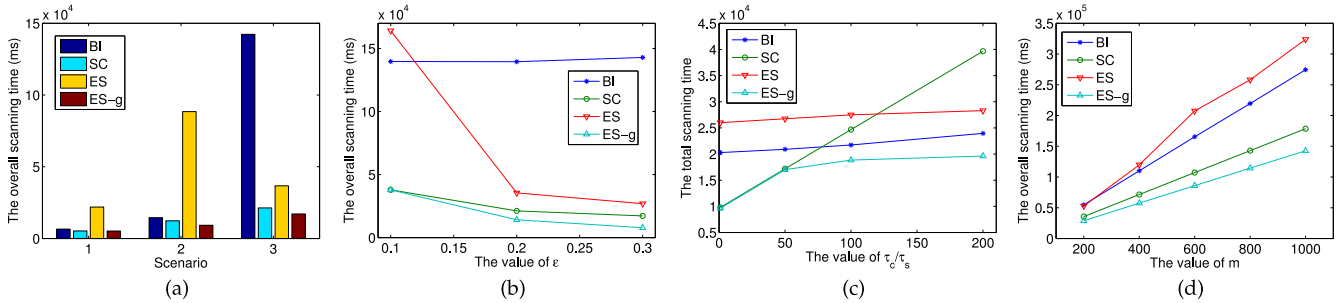


Fig. 5. Simulation results in basic histogram collection: (a) The overall scanning time in various scenarios. (b)The overall scanning time with various ϵ. (c)The overall scanning time with various τ_c/τ_s. (d) The scanning time with various value of m.

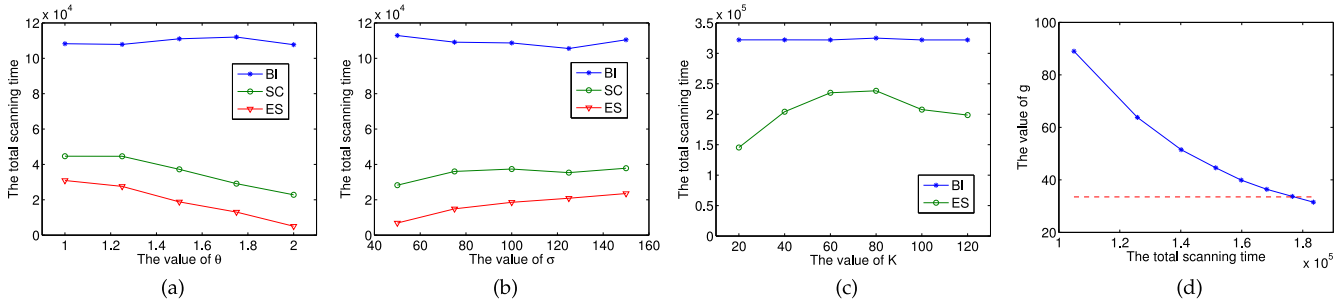


Fig. 6. Simulation results in advanced histogram collection: (a) The scanning time with various threshold θ. (b) The scanning time with various variance σ. (c) The scanning time with various values of k.(d) The variation of g with the scanning time.

scanning time of the SC strategy and the ES strategy grows slowly. The reason is as follows: as the standard deviation σ increases, the number of qualified categories is increasing, thus more slots are essential to verify the categories for accuracy; besides, fewer categories have tag sizes close to the threshold, thus fewer slots are required to verify the population constraint. In all, the overall scanning time increases rather slowly.

We evaluate the performance of our PT-Top$k$ algorithm. In Fig. 6c, we compare the scanning time with various values of k. We observe that as k increases from 20 to 120, the scanning time of the ES strategy increases from 1.5×10^5 to 2.5×10^5, and then decreases to 2×10^5. The reason is that, as the value of k increases, the exact threshold is reduced, and more categories are identified as qualified, thus more slots are essential to verify the categories for accuracy. Then, as the value of k further increases, more qualified categories with large tag sizes can be quickly wiped out in the threshold estimation, and thus fewer slots are required in the threshold estimation, and the overall scanning time is decreased. In Fig. 6d, we evaluate the convergence for estimating the threshold t. We set $m = 200, \mu = 500, \sigma = 200, k = 20$. We observe that the width of the range $[\widetilde{t}, \overline{t}]$, i.e., g, is continuously decreasing as the scanning time increases. When the scanning time reaches 1.8×10^5, the value of g is below the required threshold in the dash line, then the iteration ends.

10 Conclusion

Collecting histograms over RFID tags is an essential premise for effective aggregate queries and analysis in large-scale RFID-based applications. We believe this is the first paper considering the problem of collecting histograms over RFID tags. Based on the ensemble sampling method, we respectively propose effective solutions for the basic histogram collection, iceberg query problem, and top-k query problem. Simulation results show that our solution achieves a much better performance than others.

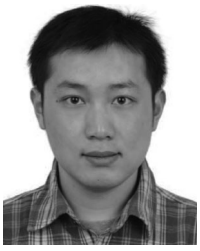
Acknowledgments

This work was supported in part by National Natural Science Foundation of China under Grant No. 61100196, 61472185, 61321491, 91218302, 61373129; JiangSu Natural Science Foundation under Grant No. BK2011559; Key Project of Jiangsu Research Program under Grant No. BE2013116; EU FP7 IRSES MobileCloud Project under Grant No. 612212. The work of Qun Li was supported in part by US NSF Grants CNS-1117412, CNS-1320453, and CAREER Award CNS-0747108. The work of Jie Wu was supported in part by US NSF grants ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167. Lei Xie is the corresponding author.

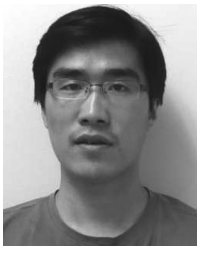
References

[1] C. Qian, Y. Liu, H.-L. Ngan, and L. M. Ni, "Asap: Scalable identification and counting for contactless RFID systems," in *Proc. IEEE Int. Conf. Distrib. Comput. Syst.*, 2010, pp. 52–61.
[2] M. Shahzad and A. X. Liu, "Every bit counts - fast and scalable RFID estimation," in *Proc. ACM MobiCom*, 2012, pp. 365–376.
[3] Y. Zheng and M. Li, "Fast tag searching protocol for large-scale RFID systems," in *Proc. IEEE Int. Conf. Netw. Protocols*, 2011, pp. 362–372.
[4] H. Vogt, "Efficient object identification with passive RFID tags," in *Proc. Pervasive*, 2002, pp. 98–113.
[5] B. Zhen, M. Kobayashi, and M. Shimuzu, "Framed aloha for multiple RFID objects identification," *IEICE Trans. Commun.*, vol. E88-B, pp. 991–999, 2005.
[6] S. Lee, S. Joo, and C. Lee, "An enhanced dynamic framed slotted aloha algorithm for RFID tag identification," in *Proc. MobiQuitous*, 2005, pp. 166–172.
[7] L. Xie, B. Sheng, C. Tan, H. Han, Q. Li, and D. Chen, "Efficient tag identification in mobile RFID systems," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[8] L. Xie, Q. Li, X. Chen, S. Lu, and D. Chen, "Continuous scanning with mobile reader in rfid systems: An experimental study," in *Proc. ACM MobiHoc*, 2013, pp. 11–20.
[9] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in rfid systems," in *Proc. ACM MobiCom*, 2006, pp. 322–333.
[10] C. Qian, H.-L. Ngan, and Y. Liu, "Cardinality estimation for large-scale rfid systems," in *Proc. IEEE PerCom*, 2008, pp. 1441–1454.
[11] B. Sheng, C. C. Tan, Q. Li, and W. Mao, "Finding popular categoried for RFID tags," in *Proc. ACM MobiHoc*, 2008, pp. 159–168.
[12] V. Poosala, V. Poosala, V. Ganti, and Y. E. Ioannidis, "Approximate query answering using histograms," *IEEE Data Eng. Bull.*, vol. 22, 1999, pp. 5–14.
[13] Y. E. Ioannidis and V. Poosala, "Histogram-based approximation of set-valued query answers," in *Proc. 25th VLDB Conf.*, 1999, pp. 174–185.
[14] D. Brauckhoff, X. Dimitropoulos, A. Wagner, and K. Salamatian, "Anomaly extraction in backbone networks using association rules," in *Proc. ACM IMC*, 2009, pp. 1788–1799.
[15] L. Xie, H. Han, Q. Li, J. Wu, and S. Lu, "Efficiently collecting histograms over rfid tags," in *Proc. IEEE INFOCOM*, 2014, pp. 145–153.
[16] Y. Yin, L. Xie, J. Wu, A. V. Vasilakos, and S. Lu, "Focus and shoot: Efficient identification over rfid tags in the specified area," in *Proc. MobiQuitous*, 2013, pp. 1–12.
[17] X. Liu, B. Xiao, K. Bu, and S. Zhang, "Lock: A fast and flexible tag scanning mechanism with handheld readers," in *Proc. IEEE/ACM IWQoS*, 2014, pp. 1–9.
[18] S. Tang, J. Yuan, X. Y. Li, G. Chen, Y. Liu, and J. Zhao, "Raspberry: A stable reader activation scheduling protocol in multi-reader rfid systems," in *Proc. IEEE Int. Conf. Netw. Protocols*, 2009, pp. 304–313.
[19] L. Yang, J. Han, Y. Qi, C. Wang, T. Gu, and Y. Liu, "Season: Shelving interference and joint identification in large-scale RFID systems," in *Proc. IEEE INFOCOM*, 2011, pp. 3092–3100.

[20] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large RFID system," in *Proc. ACM MobiHoc*, 2010, pp. 1–10.
[21] S. Chen, M. Zhang, and B. Xiao, "Efficient information collection protocols for sensor-augmented RFID networks," in *Proc. IEEE INFOCOM*, 2011, pp. 3101–3109.
[22] Y. Qiao, S. Chen, T. Li, and S. Chen, "Energy-efficient polling protocols in RFID systems," in *Proc. ACM MobiHoc*, 2011, pp. 25–34.
[23] T. Li, S. Wu, S. Chen, and M. Yang, "Energy efficient algorithms for the RFID estimation problem," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[24] W. Chen, "An accurate tag estimate method for improving the performance of an RFID anticollision algorithm based on dynamic frame length aloha," *IEEE Trans. Autom. Sci. Eng.*, vol. 6, no. 1, pp. 9–15, Jan. 2009.
[25] W. Luo, Y. Qiao, and S. Chen, "An efficient protocol for RFID multigroup threshold-based classification," in *Proc. IEEE INFOCOM*, 2013, pp. 890–888.
[26] Y. Zheng and M. Li, "Zoe: Fast cardinality estimation for large-scale RFID systems," in *Proc. IEEE INFOCOM*, 2013, pp. 908–916.
[27] B. Chen, Z. Zhou, and H. Yu, "Understanding RFID counting protocols," in *Proc. ACM MobiCom*, 2013, pp. 291–302.
[28] J. Liu, B. Xiao, K. Bu, and L. Chen, "Efficient distributed query processing in large RFID-enabled supply chains," in *Proc. IEEE INFOCOM*, 2014, pp. 163–171.
[29] X. Liu, K. Li, H. Qi, B. Xiao, and X. Xie, "Fast counting the key tags in anonymous RFID systems," in *Proc. IEEE Int. Conf. Netw. Protocols*, 2014, pp. 1–9.
[30] M. Buettner and D. Wetherall, "An empirical study of uhf rfid performance," in *Proc. ACM MobiCom*, 2008.
[31] L. Pan and H. Wu, "Smart Trend-Traversal: A Low Delay and Energy Tag Arbitration Protocol for Large RFID Systems," in *Proc. IEEE INFOCOM, Mini-Conf.*, 2009, pp. 223–234.
[32] H. Han, B. Sheng, C. C. Tan, Q. Li, W. Mao, and S. Lu, "Counting rfid tags efficiently and anonymously," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[33] M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman, "Computing iceberg queries efficiently," in *Proc. 24th VLDB Conf*, 1998, pp. 299–310.
[34] J. Han, C. Qian, X. Wang, D. Ma, J. Zhao, P. Zhang, W. Xi, and Z. Jiang, "Twins: Device-free object tracking using passive tags," in *Proc. IEEE INFOCOM*, 2014, pp. 469–476.



Lei Xie received the PhD degree in computer science from Nanjing University, Nanjing, China. He is currently an associate professor in the Department of Computer Science and Technology at Nanjing University. His research interests include RFID systems, pervasive and mobile computing, and Internet of things. He has published more than 30 papers in the *IEEE Transaction on Parallel and Distributed Systems*, ACM MobiHoc, IEEE INFOCOM, IEEE ICNP, IEEE ICC, IEEE GLOBECOM, MobiQuitous, etc. He is a member of the IEEE.



Hao Han received the PhD degree in computer science from the College of William and Mary, Williamsburg, VA, in 2013. He is currently a research scientist at the Networks and Security Group in Intelligent Automation, Inc, Rockville, MD. His research interests include wireless networks, mobile computing, cloud computing and RFID systems. He is a member of the IEEE.

Qun Li received the PhD degree in computer science from Dartmouth College, Hanover, NH. He is an associate professor in the Department of Computer Science at the College of William and Mary, Williamsburg, VA. His research interests include wireless networks, sensor networks, RFID, and pervasive computing systems. He received the US National Science Foundation (NSF) Career award in 2008. He is a member of the IEEE.

Jie Wu is currently the chair and a Laura H. Carnell professor in the Department of Computer and Information Sciences at Temple University. He is also an Intellectual Ventures endowed visiting chair professor at the National Laboratory for Information Science and Technology, Tsinghua University, Beijing, China. Prior to joining Temple University, he was a program director at the National Science Foundation and was a Distinguished Professor at Florida Atlantic University, Boca Raton, FL. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. He regularly publishes in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including *IEEE Transactions on Service Computing* and the *Journal of Parallel and Distributed Computing*. He was a general co-chair/chair for IEEE MASS 2006, IEEE IPDPS 2008, and IEEE ICDCS 2013, as well as a program co-chair for IEEE INFOCOM 2011 and CCF CNCC 2013. Currently, he is serving as a general chair for ACM MobiHoc 2014. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and a chair for the IEEE Technical Committee on Distributed Processing (TCDP). He received the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award. He is a CCF Distinguished speaker and a fellow of the IEEE.

Sanglu Lu received the BS, MS, and PhD degrees from Nanjing University, Nanjing, China in 1992, 1995 and 1997, respectively, all in computer science. She is currently a professor in the Department of Computer Science and Technology at Nanjing University. Her research interests include distributed computing and pervasive computing. She is a member of the IEEE.