Efficiently Collecting Histograms Over RFID Tags

Lei Xie[†], Hao Han[‡], Qun Li[‡], Jie Wu[§], and Sanglu Lu[†]
[†]State Key Laboratory for Novel Software Technology, Nanjing University, China
[‡]Department of Computer Science, College of William and Mary, USA
[§]Department of Computer Information and Sciences, Temple University, USA
Email: [†]lxie@nju.edu.cn, [‡]{hhan,liqun}@cs.wm.edu, [§]jiewu@temple.edu, [†]sanglu@nju.edu.cn

***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 efficient collection of histograms from the massive number of RFID tags without the need to read all tag data. 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. 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**

I. 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 densely packed into boxes. As the maximum scanning range of a UHF RFID reader is usually 6-10m, 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 [1–3]. Many identification protocols are proposed to uniquely identify the tags 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, 4], popular categories [5], and the histogram. In particular, histograms capture distribution statistics in a space-efficient fashion.

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 histograms 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, as well as preprocessing and mining association rules in data mining [6]. Therefore, collecting histograms over RFID tags is an essential premise for effective queries and analysis in conventional RFID-based applications.

978-1-4799-3360-0/14/$31.00

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 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 skew 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 tags in the order of several thousands, the overall number of categories in the long tail could be in 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. We make the following contributions.

1) To the best of our knowledge, we are the first to consider the problem of collecting histograms over RFID tags, which is a fundamental premise for queries and analysis in RFID-based applications. 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. 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. 2) 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 k-th largest category, and reduce it to the *iceberg query* problem. 3) While achieving time-efficiency, our solutions are completely compatible with current industry standards, i.e., the EPCglobal C1G2 standards, and do not require any modification to tags.

II. 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 s-lotted ALOHA-based anti-collision protocols are designed to efficiently resolve collisions in RFID systems. In order to deal with the collision problems in multi-reader RFID systems, scheduling protocols for reader activation are explored in [7, 8]. Recently, a number of polling protocols [9, 10] 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, 4, 5, 11–13]. The authors mainly consider the problem of estimating the number of tags without collecting the tag IDs. In [4], Murali et al. provide very fast and reliable estimation mechanisms for tag quantity in a more practical approach. In [2], Shahzad et al. propose a new scheme for estimating tag population size called Average Run based Tag estimation. In [5], Sheng et al. consider the problem of identifying popular categories of RFID tags out of a large collection of tags, while the set of category IDs are supposed to be known. 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, including the number of categories and the category IDs.

III. Preliminary

A. 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 *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.

B. The Impact of the Inter-cycle Overhead

The MAC protocol for the C1G2 system is based on s-lotted 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 [14], 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. We have further measured the time interval for various slots with the USRP N210 platform, it is found that the average interval for empty slots is 1.6 ms per slot, and the average interval for singleton/collision slots is 5.1 ms per slot. 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, the inter-cycle duration must be taken into account when considering overall reading performance. It is obvious that reading a larger 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.

IV. 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 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 exists m categories $C = \{C_1, C_2, ..., C_m\}$, and the actual tag size for each category is $n_1, n_2, ..., n_m$.

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:

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

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 a confidence interval of width $2\epsilon \cdot n_i$, i.e., $\frac{\widehat{n_i}}{n_i} \in [1 - \epsilon, 1 + \epsilon]$ with a 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 a probability greater than 95%.

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:

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

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

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:

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

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

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.

V. 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%, and (2) *Separate Counting*: The histogram is collected by estimating the number of tags in each category one by one. Specifically, for each category, the reader broadcasts a *Select* command to activate the tags in the specified category. According to the replies from the specified tags, the estimators like [4][12][15] can be used to estimate the tag size for each category. As the rough tag size for each category cannot be predicted, a fixed initial frame size is used for each category.

Both of 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 or 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.

A. The Estimator SE

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 [4][12][15] 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 $C_1, C_2, ..., C_m$, the overall tag size is n, and the tag size for each category is $n_1, n_2, ..., n_m$. 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,

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

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

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

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:

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

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

B. Accuracy Analysis

1) Accuracy of the SE Estimator: In the ensemble sampling-based estimation, since the estimators such as [4][12][15] can be utilized for estimating the overall number of tags, we use δ

to denote the variance of $\widehat{n}$. 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,

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

Proof: See Appendix A. ■

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 $\sigma_i^2 \leq (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot n_i^2$, $Z_{1-\beta/2}$ is the $1 - \frac{\beta}{2}$ percentile for the standard normal distribution.

Proof: See Appendix B. ■

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 $Z_{1-\beta/2} = 1.96$.

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 [4][12][15] to estimate the overall tag size, then, without loss of generality, if we use the estimator in [4], we can prove that $a < 0$ for any value of $n > 0, f > 0$. 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 2, in order to satisfy the *accuracy constraint*, we should ensure that $\lambda_i \leq (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot n_i$. 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 accurate estimation of the tag size.

C. 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, and 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., $C_1, C_2, ..., C_m$. 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 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}, they are finally divided into 3 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 size of each category inside one group is close to the others. 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, as to sufficiently amortize the fixed time cost for each query cycle.

D. 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 $\bar{n}$, we need to set the initial frame size f by solving $fe^{-\bar{n}/f} = 5$, as suggested in [4]. 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.

Algorithm 1 Algorithm for histogram collection

1: Initialize the set R to all tags. Set $l = 1$.
2: **while** $n_s \neq 0 \wedge n_c \neq 0$ **do**
3: If $l = 1$, compute the initial frame size f by solving $fe^{-\bar{n}/f} = 5$. Otherwise, compute the frame size $f = \widehat{n}$. If $f > f_{max}$, set $f = f_{max}$.
4: 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.
5: 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).
6: Rank the categories in S in non-increasing order of the tag size. Divide the set S into groups $S_1, S_2, ..., S_d$ according to the dynamic programming-based method.
7: **for** each $S_j \in S (1 \leq j \leq d)$ **do**
8: For each category $C_i \in S_j$, compute the frame size f_i from δ_i by solving $\frac{1}{1/\delta_i' + 1/\delta_i} \leq (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot \widehat{n_i}^2$.
9: 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.
10: Estimate the tag size $\widehat{n_i}$ for each category $C_i \in S_j$, illustrate them in the histogram.
11: **end for**
12: $\widehat{n} = \widehat{n} - \sum_{C_i \in S} \widehat{n_i}$. $R = R - S$. $S = \varnothing$. $l = l + 1$.
13: **end while**

VI. Ensemble Sampling for the Iceberg Query

A. 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* [16] 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: 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 1 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 3: 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.

Due to lack of space, we omit the proof of Theorem 3. The detailed proof is given in [17].

In order to better illustrate the inherent principle, Fig.1 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 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/unqualifed, as both the false positive and false negative probability 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/unqualifed for the population constraint.

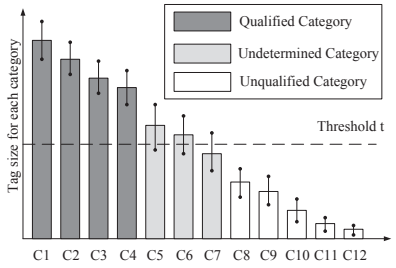

Fig. 1. Histogram with confidence interval annotated

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 4: 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.

Due to lack of space, we omit the proof of Theorem 4. The detailed proof is given in [17].

According to Theorem 4, 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.

B. Algorithm for the Iceberg Query Problem

Algorithm 2 Algorithm for Iceberg Query

1: Initialize R to all categories, set Q, U, V to $\varnothing$. Set $l = 1$.
2: **while** $R \neq \varnothing$ **do**
3: If $l = 1$, set the initial frame size f.
4: Issue a query cycle over the tags, add those relatively major categories into the set S. Set $S' = S$.
5: **while** $S \neq \varnothing$ **do**
6: 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.
7: 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 $S = S - Q - U$.
8: **if** $U \neq \varnothing$ **then**
9: Wipe out all categories unexplored in the singleton slots from S.
10: **end if**
11: **end while**
12: $\widehat{n} = \widehat{n} - \sum_{C_i \in S'} \widehat{n_i}$. $R = R - S'$, $l = l + 1$.
13: **end while**
14: Further verify the categories in V and Q for the accuracy constraint.

We propose the algorithm for the iceberg query problem in Algorithm 2. Steps 1-4 are quite similar to steps 1-4 in Algorithm 1, due to lack of space we omit the detailed statements for these steps. 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: They refer to the categories whose tag sizes are determined to be over the specified threshold t. 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: They refer to the categories whose tag sizes are determined to be below the specified threshold t. 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.

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 $R = \varnothing$.

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 moved 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/unqualifed, 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.

VII. Ensemble Sampling for the Top-k Query

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 unsuitable. 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.

Hence, according to the definition of *PT-Topk query*, 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, if the threshold $\widehat{t}$ can be accurately estimated, then the top-k query problem is reduced to the iceberg query problem. Then it is essential to quickly determine the value of the threshold $\widehat{t}$ while satisfying the constraint $Pr[|\widehat{t} - t| \leq \epsilon \cdot t] \geq 1 - \beta$. We rely on the following theorem to express the above constraint in the form of the variance.

Theorem 5: The constraint $Pr[|\widehat{t} - t| \leq \epsilon \cdot t] \geq 1 - \beta$ is satisfied as long as $Var(\widehat{t} - t) \leq \epsilon^2 \cdot t^2 \cdot \beta$.

Due to lack of space, we omit the proof of Theorem 5. The detailed proof is given in [17].

According to Theorem 5, 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, say g, 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 g is below the threshold $\sqrt{\epsilon^2 \cdot t^2 \cdot \beta}$, the estimated value $\widehat{t}$ can be close enough to the exact threshold t. Then, we can apply the *iceberg query* with threshold $\widehat{t}$ over the undetermined categories R and the qualified categories Q. Due to lack of space, we omit the detailed algorithm for *PT-Topk* Query Problem. The detailed algorithm is given in [17].

VIII. Performance Analysis in Time-Efficiency

As mentioned in the problem formulation, the most critical factor for the histogram collection problem is time efficiency. In regard to 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 time efficiency that is achieved. In regard to the iceberg query and top-k query, the time delay mainly depends on the number of categories with tag size close to the threshold t. Due to the variance in tag size estimation, relatively large numbers 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.

IX. Performance Evaluation

We have conducted simulations in Matlab, 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 $\tau_s = 1$ms, the inter-cycle overhead is $\tau_c = 43$ms. We compare our solutions with two basic strategies: the basic tag identification (BI) and the separate counting (SC) (explained in Section V). All results are the averaged results of 500 independent trials.

A. 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.2(a), 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

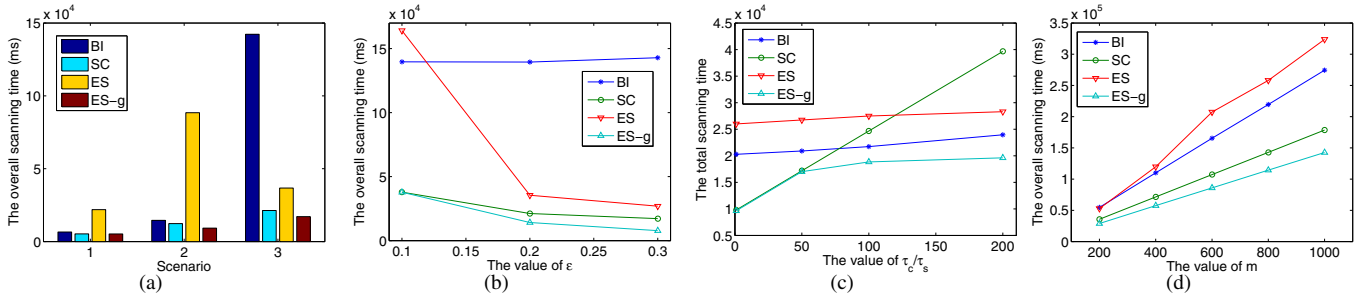

Fig. 2. Simulation results in basic histogram collection: (a)The overall scanning time in various scenarios;(b)The overall scanning time with various ϵ;(c)The overall scanning time with various τ_c/τ_s;(d)The scanning time with various value of m.

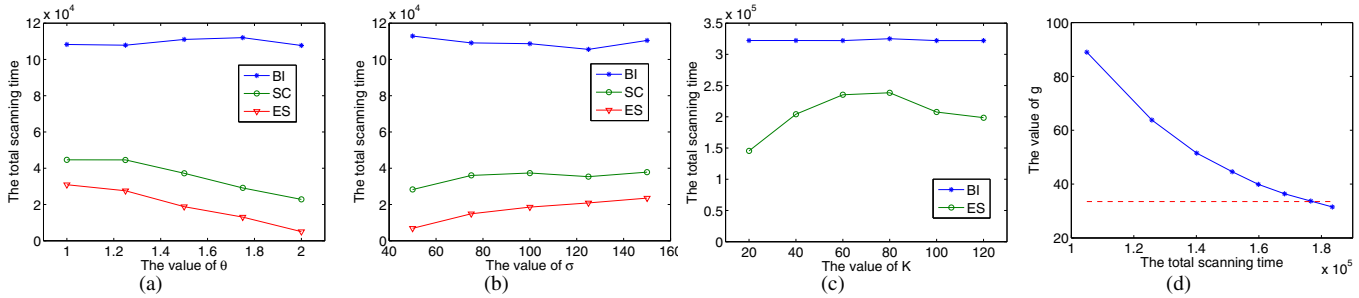

Fig. 3. Simulation results in advanced histogram collection: (a)The scanning time with various threshold θ;(b)The scanning time with various variance σ;(c)The scanning time with various value of k;(d)The variation of g with the scanning time.

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 basic identification (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.2(b), 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.2(c), 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 easy to be 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.2(d), 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.

B. 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.3(a) 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.3(b) 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 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 is increasing rather slowly.

We evaluate the performance of our PT-Top$k$ algorithm. In Fig.3(c), 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, thus fewer slots are required in the threshold estimation, and the overall scanning time is decreased. In Fig.3(d), 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 $[\widetilde{t}, \bar{t}]$, 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.

X. Conclusion

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 much better performance than others.

Acknowledgments

This work is supported in part by National Natural Science Foundation of China under Grant No. 61100196, 61073028, 61321491, 91218302; JiangSu Natural Science Foundation under Grant No. BK2011559. Qun Li is supported in part by US NSF grants CNS-1117412, CNS-1320453, and CAREER Award CNS-0747108. Jie Wu is supported in part by US NSF grants ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167.

References

[1] C. Qian, Y. Liu, H.-L. Ngan, and L. M. Ni, "Asap: Scalable identification and counting for contactless rfid systems," in *Proc. of IEEE ICDCS*, 2010.
[2] M. Shahzad and A. X. Liu, "Every bit counts - fast and scalable rfid estimation," in *Proc. of MobiCom*, 2012.
[3] Y. Zheng and M. Li, "Fast tag searching protocol for large-scale rfid systems," in *Proc. of ICNP*, 2011.
[4] M. Kodialam and T. Nandagopal, "Fast and reliable estimation schemes in rfid systems," in *Proc. of MobiCom*, 2006.
[5] B. Sheng, C. C. Tan, Q. Li, and W. Mao, "Finding popular categoried for RFID tags," in *Proc. of ACM MobiHoc*, 2008.
[6] Y. E. Ioannidis and V. Poosala, "Histogram-based approximation of set-valued query answers," in *Proc. of the 25th VLDB Conference*, 1999.
[7] 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. of ICNP*, 2009.
[8] 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. of INFOCOM*, 2011.
[9] S. Chen, M. Zhang, and B. Xiao, "Efficient information collection protocols for sensor-augmented rfid networks," in *Proc. of INFOCOM*, 2011.
[10] Y. Qiao, S. Chen, T. Li, and S. Chen, "Energy-efficient polling protocols in rfid systems," in *Proc. of MobiHoc*, 2011.
[11] T. Li, S. Wu, S. Chen, and M. Yang, "Energy efficient algorithms for the rfid estimation problem," in *Proc. of INFOCOM*, 2010.
[12] W. Chen, "An accurate tag estimate method for improving the performance of an rfid anticollision algorithm based on dynamic frame length aloha," *IEEE Transactions on Automation Science and Engineering*, vol. 6, no. 1, pp. 9–15, 2009.
[13] W. Luo, Y. Qiao, and S. Chen, "An efficient protocol for rfid multigroup threshold-based classification," in *Proc. of INFOCOM*, 2013.
[14] M. Buettner and D. Wetherall, "An empirical study of uhf rfid performance," in *Proc. of MobiCom*, 2008.
[15] H. Han, B. Sheng, C. C. Tan, Q. Li, W. Mao, and S. Lu, "Counting rfid tags efficiently and anonymously," in *Proc. of INFOCOM*, 2010.
[16] M. Fang, N. Shivakumar, H. Garcia-Molina, R. Motwani, and J. D. Ullman, "Computing iceberg queries efficiently," in *Proc. of the 24th VLDB Conf*, 1998.
[17] L. Xie, H. Han, Q. Li, J. Wu, and S. Lu, "Efficiently collecting histograms over rfid tags," Nanjing University, Tech. Rep., 2013, http://cs.nju.edu.cn/lxie/publication/histogram.pdf.

Appendix

A: Proof of Theorem 1

Proof: According to the definition $\widehat{\alpha_i} = \frac{n_{s,i}}{n_s}$, $E(\widehat{\alpha_i}) = E(\frac{n_{s,i}}{n_s})$, $E(\widehat{\alpha_i}^2) = E(\frac{n_{s,i}^2}{n_s^2})$. Despite that $n_s = \sum_{i=1}^{m} n_{s,i}$, the relationship between n_s and $n_{s,i}$ is rather weak as long as m is fairly large. Hence we can assume $n_{s,i}$ and n_s are independent of each other. Then, according to the property of independence, $E(\widehat{\alpha_i}) = \frac{E(n_{s,i})}{E(n_s)} = \frac{n_i}{n}$, $E(\widehat{\alpha_i}^2) = \frac{E(n_{s,i}^2)}{E(n_s^2)}$. Then, as it can be proved that

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

we have

$$E(\widehat{\alpha_i}^2) = \frac{n_i}{n} \cdot \frac{1+(1-\frac{1}{f-1})^{n-2} \cdot (n_i-1)}{1+(1-\frac{1}{f-1})^{n-2} \cdot (n-1)} \approx \frac{n_i}{n} \cdot \frac{1+e^{-\rho} \cdot (n_i-1)}{1+e^{-\rho} \cdot (n-1)}.$$

Thus, $\delta_i = E(\widehat{n_i}^2) - E^2(\widehat{n_i}) = E(\widehat{\alpha_i}^2 \cdot \widehat{n}^2) - n_i^2$. Note that $\widehat{n}$ is estimated from the estimator which relies on the number of empty/singleton/collison slots as well as the frame size f, while $\widehat{\alpha_i}$ is equal to $\frac{n_{s,i}}{n_s}$, i.e., the proportion that the category C_i occupies in the singleton slots. Therefore, we can assume the independence between $\widehat{\alpha_i}$ and $\widehat{n}$, then $\delta_i = E(\widehat{\alpha_i}^2) \cdot E(\widehat{n}^2) - n_i^2$. As the variance $\delta = E(\widehat{n}^2) - E(\widehat{n})^2$ and $E(\widehat{n}) = n$, thus $E(\widehat{n}^2) = \delta + n^2$. Therefore,

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

■

B: Proof of Theorem 2

Proof: Suppose identical frame sizes are used for each query cycle in the repeated tests. Then, in regard to a specified category C_i, the series of estimated tag sizes $\{\widehat{n_{i,k}}\}$ for l query cycles are independent and identically distributed. Since the variance $\delta_{i,k}$ is equal for each query cycle, according to the weighted statistical averaging method, the estimated tag size is $\widehat{n_i} = \frac{1}{l}\sum_{k=1}^{l} \widehat{n_{i,k}}$. As the expected value and standard deviation for the averaged value $\widehat{n_i}$ are respectively n_i and σ_i, then according to the *Central Limit Theorem*, $X = \frac{\widehat{n_i}-n_i}{\sigma_i}$ is asymptotically normal with mean 0 and variance 1, that is, X satisfies the standard normal distribution.

Therefore, the definition of the accuracy constraint is equivalent to: $Pr[\frac{-\epsilon \cdot n_i}{\sigma_i} \le \frac{\widehat{n_i}-n_i}{\sigma_i} \le \frac{\epsilon \cdot n_i}{\sigma_i}] \ge 1-\beta$. We use $Z_{1-\beta/2}$ to denote the $1-\frac{\beta}{2}$ percentile for the standard normal distribution. Hence, in order to satisfy the accuracy constraint, it is essential to ensure $\frac{\epsilon \cdot n_i}{\sigma_i} \ge Z_{1-\beta/2}$, that is, $\sigma_i^2 \le (\frac{\epsilon}{Z_{1-\beta/2}})^2 \cdot n_i^2$. ■