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
Presenter： Dr. Lei Xie
Associate Professor， Nanjing University, China
lxie@nju.edu.cn

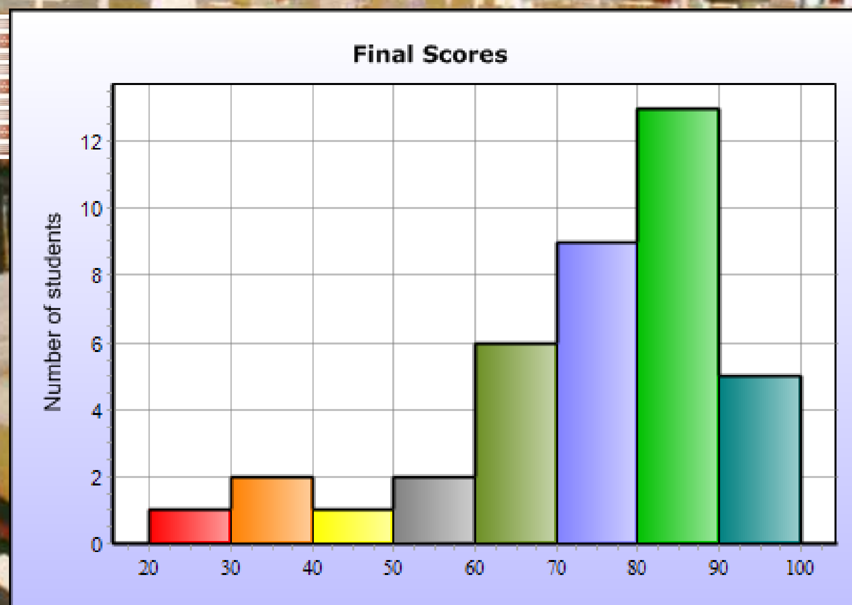
Outline

- Background and Motivation
- Problem Formulation
- Use Ensemble Sampling to Collect Histograms
- Iceberg Query & Top-k Query
- Performance Evaluation
- Conclusion

Background and Motivation

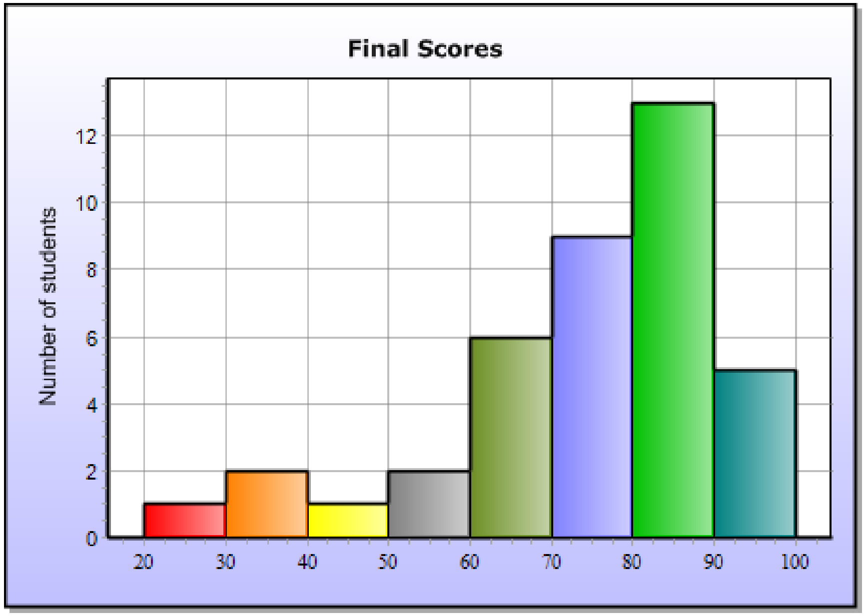
Only some useful statistical information is essential to be collected:

- Overall tag size
- Popular categories
- Histogram

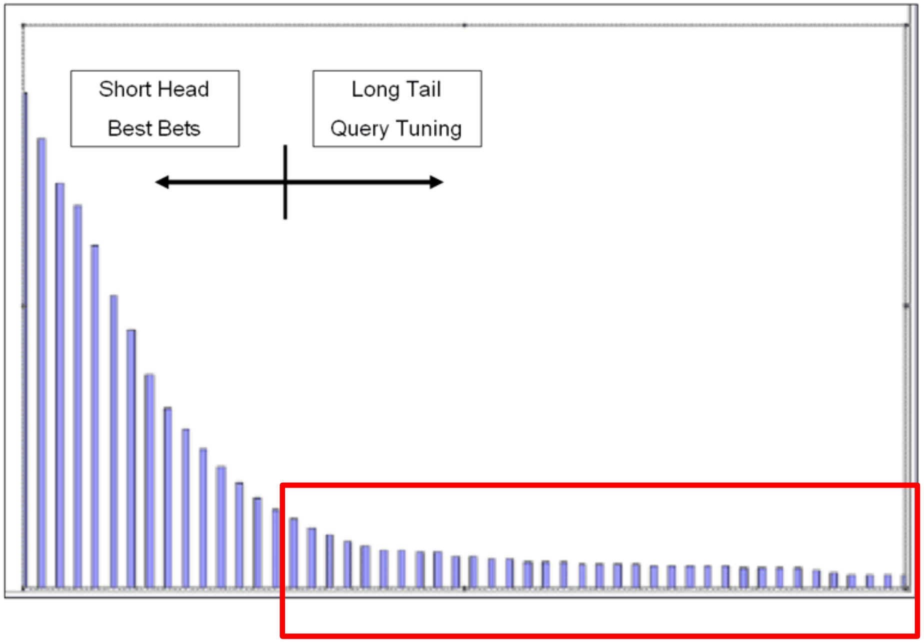


Background and Motivation

- **Histogram Collection**



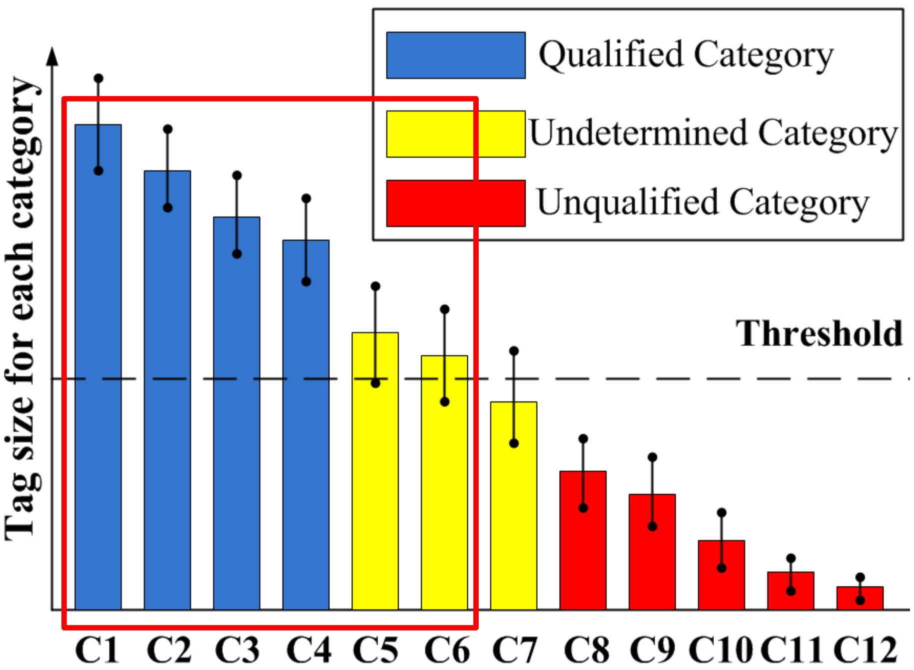
Basic Histogram Collection



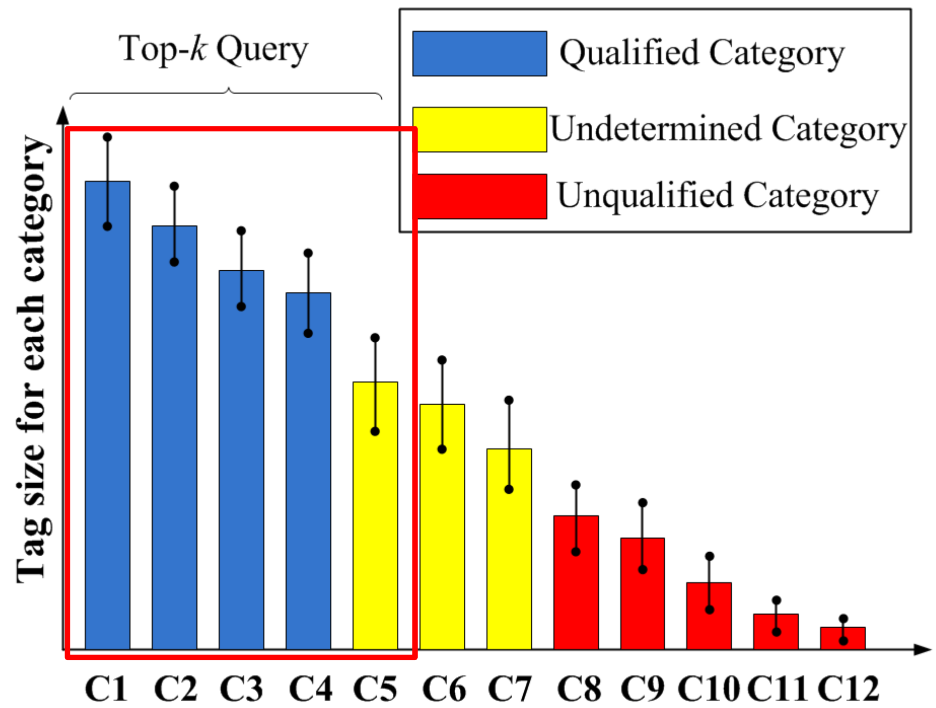
Long Tail

Background and Motivation

- **Advanced Histogram Collection**



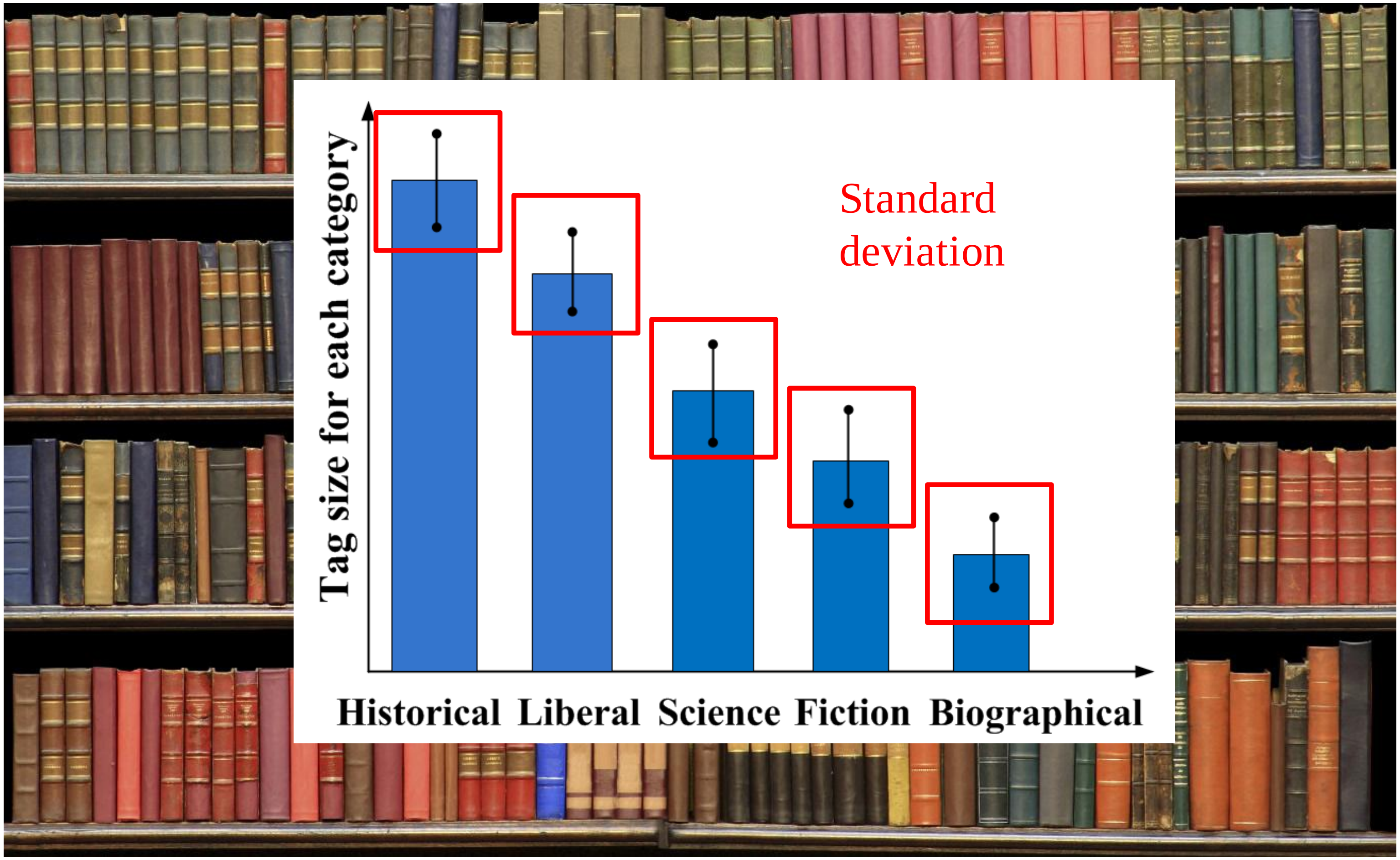
Iceberg Query over Histogram



Top-k Query over Histogram

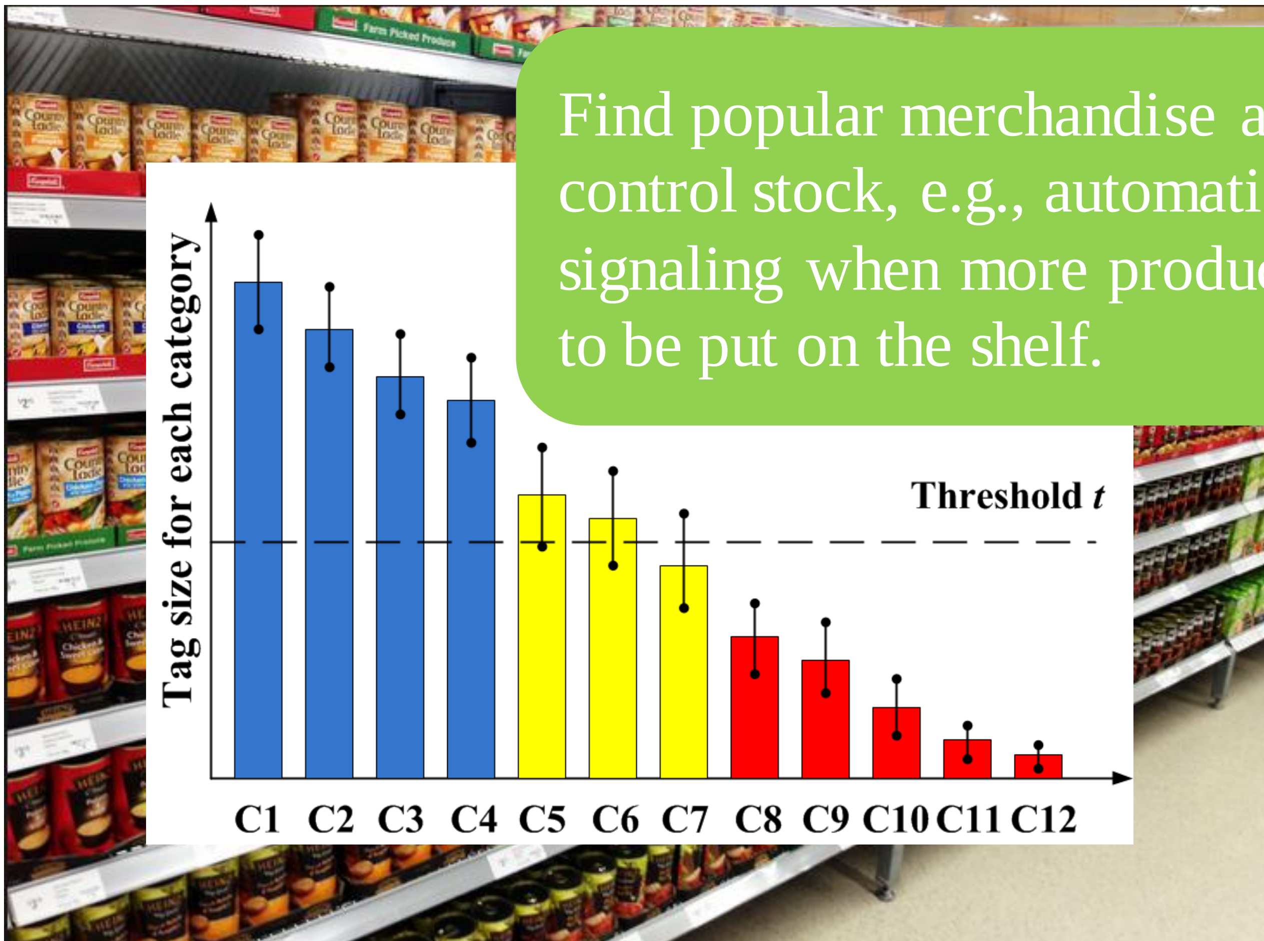
Background and Motivation

- **Application scenarios**
  - Approximately show me the number of books for each category in a large bookshelf?



Background and Motivation

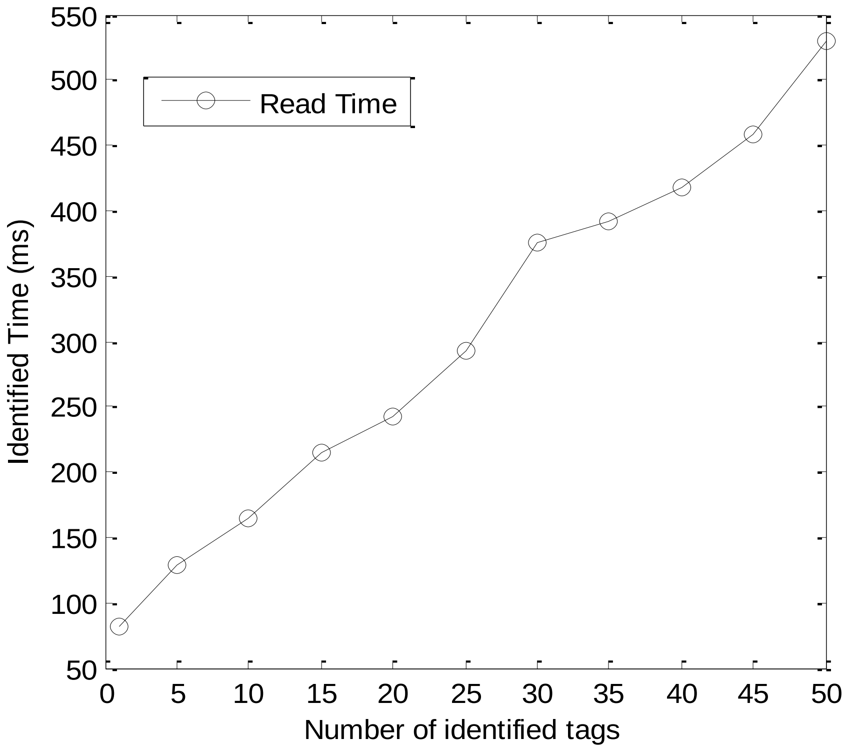
- **Application scenarios**
  - Approximately show me those categories with the quantity above 15 in supermarket shelves?



Find popular merchandise and control stock, e.g., automatically signaling when more products need to be put on the shelf.

Background and Motivation

- **Histogram Collection**
  - In most applications, the tags are frequently moving into and out of the effective scanning area.
  - Traditional tag identification is not suitable for histogram collection
    - Scanning time is proportional to the number of tags, in the order of several minutes.



Background and Motivation

- **Histogram Collection**
  - In order to capture the distribution statistics in time
    - It is essential to sacrifice ***some accuracy***.
    - Obtain the distribution within the order of ***several seconds***.
  - Propose estimation scheme to ***quickly*** count the tag sizes of each category, while achieving the ***accuracy requirement***.
  - We aim to propose a series of novel solutions satisfying the properties:
    - Time-efficient.
    - Simple for the tag side in the protocol.
    - Complies with the EPCglobal C1G2 standards.

Problem Formulation

- **Scenario**
  - A large number of tags (about 5000 tags) with a large number of categories (about 100 categories) in the effective scanning area.
  - The slotted ALOHA-based anti-collision scheme is used.
  - The present set of category IDs cannot be predicted in advance.
- **Objective**
  - Collect the histogram over RFID tags according to some categorized metric, e.g., the type of merchandise.
  - Achieve ***time-efficiency*** while satisfying the ***accuracy /population constraints***.

Basic Histogram Collection

- **Objective**
  - The RFID system needs to collect the histogram for ***all categories*** in a ***time-efficient*** approach.
- **Accuracy constraint**
  - Suppose the estimated tag size for category $C_i (1 \leq i \leq m)$ is $\widehat{n}_i$, then the accuracy constraint is

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

- For example, if $\epsilon = 0.1$, $\beta = 0.05$, then for a category with tag size $ni = 100$, the estimated tag size should be within the range [90, 110] with a probability greater than 95%.

Basic Histogram Collection

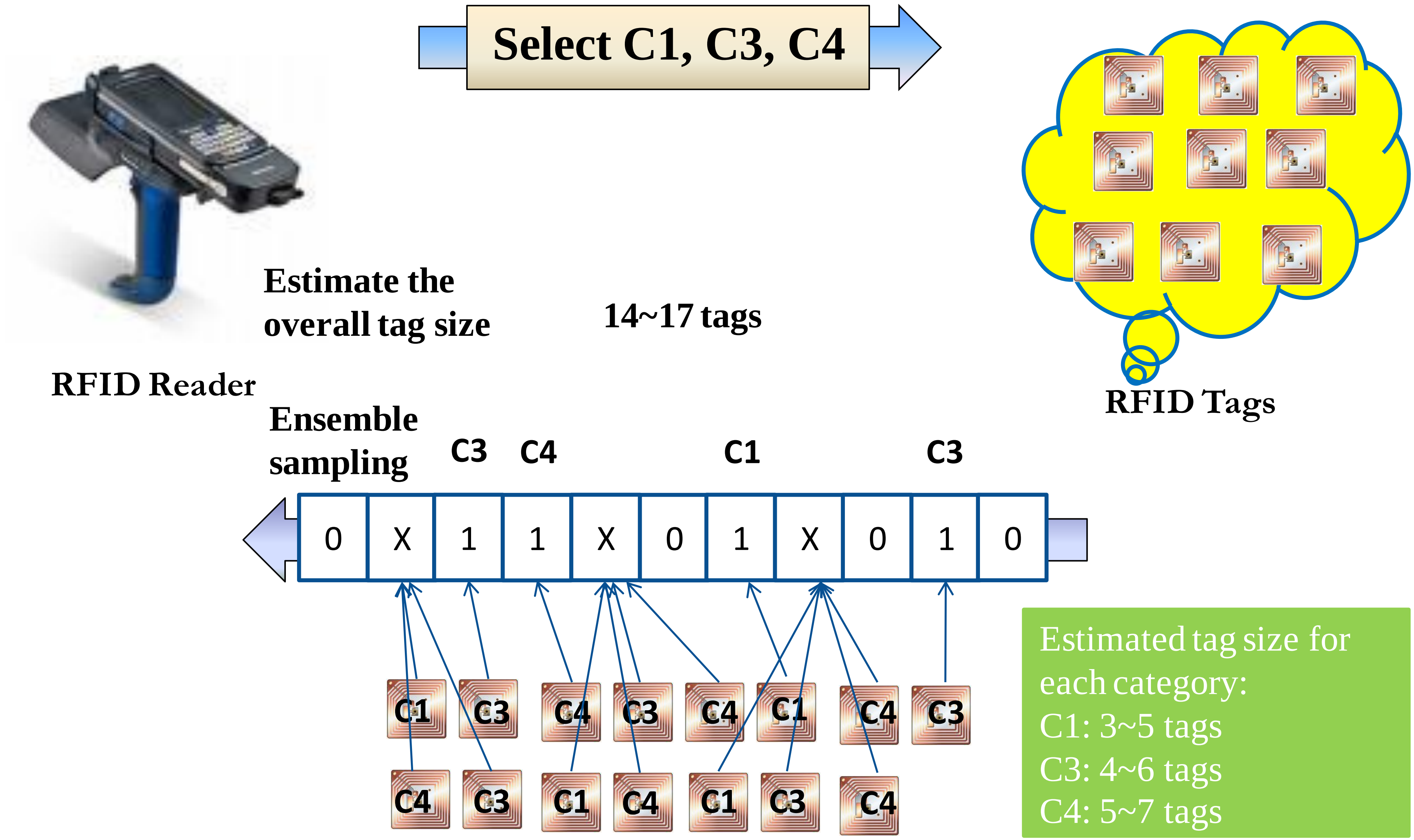
- **Two straightforward solutions**
  - Basic Tag Identification
  - Separate Counting
- Both of the two solutions are not time-efficient.
  - **Basic tag identification**: uniquely identifying each tag in the massive set is not scalable.
  - **Separate counting**:
    - The fixed initial frame size for each category, and the inter-cycle overhead among query cycles, make the scanning time rather large.
    - The reader needs to scan each category with at least one query cycle, not necessarily addressed in the iceberg query or the top-*k* query.

Basic Histogram Collection

- **Ensemble sampling-based solution**
  - Issue a query cycle by selecting a certain number of categories.
  - Obtain the empty/singleton/collision slots.
  - Estimate the overall number of tags $\widehat{n}$ according to the observed number of empty/singleton/collision slots.
  - Estimate the number of tags $\widehat{n_i}$ for each of the categories according to the sampling in the singleton slots.

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

Basic Histogram Collection



Basic Histogram Collection

- **Ensemble sampling-based solution**
- The ensemble sampling is more preferred than the separate counting in terms of reading performance.
  - 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.
  - The overall scanning time can be greatly reduced.
  - In regard to iceberg query and top-k query, some optimization can be further applied with ensemble sampling to filter unqualified categories and estimate the threshold.

Basic Histogram Collection

- **Ensemble sampling-based solution**
- Accuracy Analysis
  - The variance of the estimator $\widehat{n_i}$

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

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

  - Reduce the variance through repeated tests

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.

Basic Histogram Collection

- **Property of the Ensemble Sampling**
  - During the ensemble sampling, ***the major categories*** occupy most of the singleton slots, those ***minor categories*** cannot obtain enough samplings in the singleton slots for accurate estimation of the tag size.
  - Achieve the accuracy requirement for all categories -> the scanning time 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.
  - **Solution**: *Break the overall tags into several groups for separate ensemble sampling.*

Basic Histogram Collection

- **Ensemble sampling-based solution**
- The optimal granularity for the group size in ensemble sampling
- Each cycle of ensemble sampling should be applied over an appropriate group-> the variance of the tag sizes for the involved categories cannot be too large-> all categories in the same group achieve the accuracy requirement with very close finishing time.
- Solution: ***dynamic programming***

Basic Histogram Collection

- **Dynamic Programming-based Solution (Example)**

A set of tags with 10 categories

(ranked in non-increasing order of the estimated tag size)

Category:	C1,	C2,	C3,	C4,	C5,	C6,	C7,	C8,	C9,	C10
Rough tag size:	100,	80,	75,	41,	35,	30,	20,	15,	12,	8

Dynamic Programming:

We define *T(i,j)* as the minimum scanning time over the categories from Ci to Cj among various grouping strategies

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

$T(i, j)$ is obtained by enumerating each possible combination of $t(i, k)$ and $T(k+1, j)$, and then getting the minimum of $t(i, k) + T(k + 1, j)$.

Basic Histogram Collection

- **Dynamic Programming-based Solution**

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 v in the singleton slots. For each category which appears in the singleton slot with population $n_{s,i} > v \cdot \theta (\theta$ 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**

Set an initial query round to roughly estimate the tag size.

Use dynamic programming to break the tags into smaller groups for ensemble sampling.

Ensemble sampling over each group for multiple cycles for the accuracy requirement.

Iceberg Query Problem

- **Objective**
  - The RFID system needs to collect the histogram for ***those categories with tag size over a specified threshold t*** in a ***time-efficient*** approach.
- **Accuracy constraint** $Pr[|\widehat{n_i} - n_i| \leq \epsilon \cdot n_i] \geq 1 - \beta.$
- **Population constraint**

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

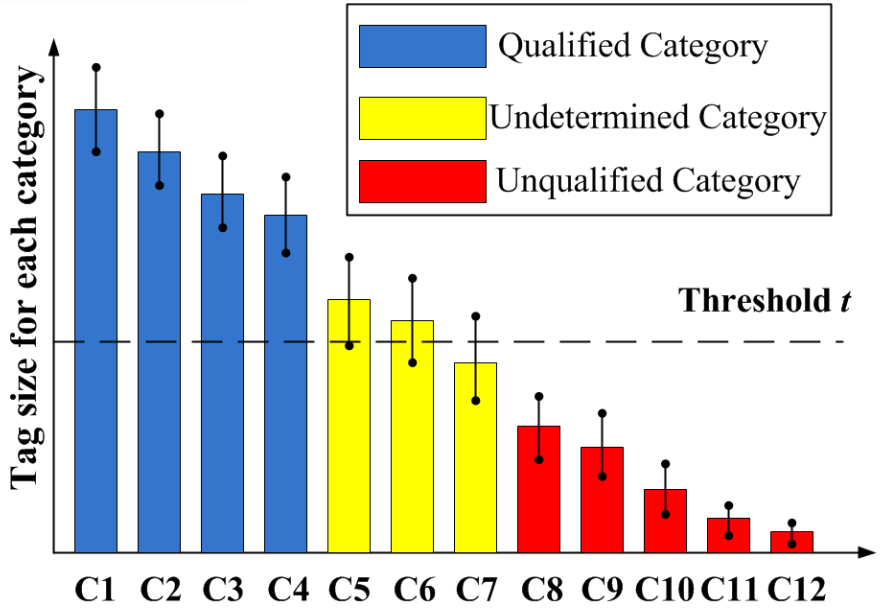
$$Pr[\widehat{n_i} \geq t | n_i < t] < \beta. \quad (3)$$

- **Express population constraint in 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.

Iceberg Query Problem

- **Key issues in Iceberg Query Problem**
  - Quickly determine the ***qualified*** and ***unqualified*** categories according to the threshold.



- **Ensemble sampling-based solution**
  - As the number of repeated tests increases, the averaged variance σi for each category decreases-> the confidence interval for each category is shrinking.
  - After a certain number of query cycles, all categories can be determined as qualified / unqualifed for the population constraint.

Iceberg Query Problem

- **Important clues to optimize the solution**
  - ***Unqualified categories***: When the estimated value $\widehat{n}_i \ll t$, the required variance in population constraint is much larger than accuracy constraint. They can be quickly identified as unqualified, wiped out immediately from ensemble sampling.
  - ***Long tail***: those categories each of which occupies a rather small percentage, but all together they occupy a substantial proportion. 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.

Iceberg Query Problem

- **Ensemble sampling based solution for Iceberg Query**
  - Qualified categories Q
  - Unqualified categories U
  - Undetermined categories R

Ensemble sampling.

Wipe out unqualified categories quickly.

Wipe out unqualified categories in the "long tail".

Algorithm 2 Algorithm for Iceberg Query

```
Initialize R to all categories, set Q, U, V to ∅. Set l = 1.
while R ≠ ∅ do
    If l = 1, set the initial frame size f.
    Issue a query cycle over the tags, add those relatively
    major categories into the set S. Set S' = S.
    while S ≠ ∅ do
        Compute the frame size f_i for each category C_i ∈ S
        such that the variance σ_i = |t − n̂_i| / Φ^{-1}(1−β). If f_i > n̂_i · 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.
        Select all tags in S, issue a query cycle with the
        frame size f, compute the estimated tag size n̂_i and
        the averaged standard deviation σ_i for each category
        C_i ∈ S. Detect the qualified category set Q and
        unqualified category set U. Set S = S − Q − U.
        if U ≠ ∅ then
            Wipe out all categories unexplored in the singleton
            slots from S.
        end if
    end while
    n̂ = n̂ − Σ_{C_i ∈ S'} n̂_i. R = R − S', l = l + 1.
end while
Further verify the categories in V and Q for the accuracy
constraint.
```

Top-k Query Problem

- **Objective**
  - The RFID system needs to collect the histogram for ***those categories in the top-k list*** in a ***time-efficient*** approach.
- **Accuracy constraint** $Pr[|\widehat{n_i} - n_i| \leq \epsilon \cdot n_i] \geq 1 - \beta.$
- **Population constraint**

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

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

- Both **Basic Tag Identification** and **Separate Counting** are not suitable.

Top-k Query Problem

- **Important clues to optimize the solution**
  - As the exact value of tag size ni is unknown, in order to define $Pr[Ci \in \text{top-}k \text{ list}]$, it is essential to determine a threshold t so that $Pr[Ci \in \text{top-}k \text{ list}] = Pr[ni \geq t]$.
  - Ideally, t should be the tag size of the kth largest category; however, it is difficult to compute an exact value of t in the estimation scheme.
  - If the threshold t can be accurately estimated, then the top-k query problem is reduced to the iceberg query problem.
- The key problem is to quickly determine the value of the threshold $\widehat{t}$ while satisfying the constraint.

$$Pr[|\widehat{t} - t| \leq \epsilon \cdot t] \geq 1 - \beta$$

Top-k Query Problem

- **Ensemble sampling based solution for Top-k Query**

Use ensemble sampling to estimate the threshold *t*: rapidly make it converge to *t*.

Apply the Iceberg Query method.

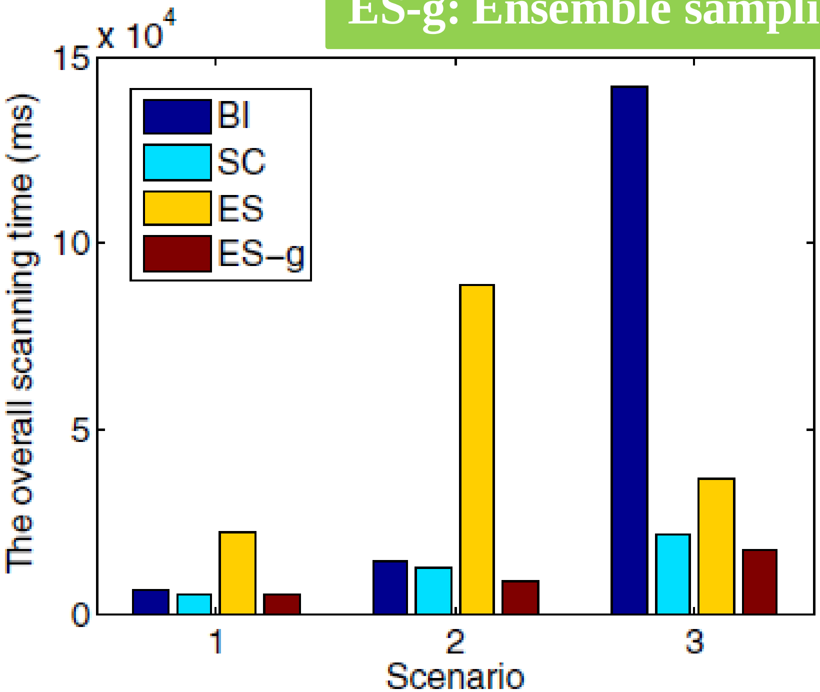
Algorithm 3 Algorithm for *PT-Topk* Query Problem

1: Initialize R to all categories, set $l = 1$, $\eta = \Phi^{-1}(1 - \frac{p}{2})$.
2: **while** true **do**
3: 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.
4: 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}.
5: 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}.
6: Wipe out the qualified/unqualified categories from R. $R = R - Q - U$. Suppose the number of qualified categories in current cycle is q, set $k = k - q$.
7: 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 $g = t_{up} - t_{low}$. $l = l + 1$.
8: **if** $g^2 \leq \epsilon^2 \cdot \beta \cdot \widehat{t}^2$ **then**
9: Break the while loop.
10: **end if**
11: **end while**
12: Apply the *iceberg query* with threshold $\widehat{t}$ over the undetermined categories R and the qualified categories Q.

Performance Evaluation

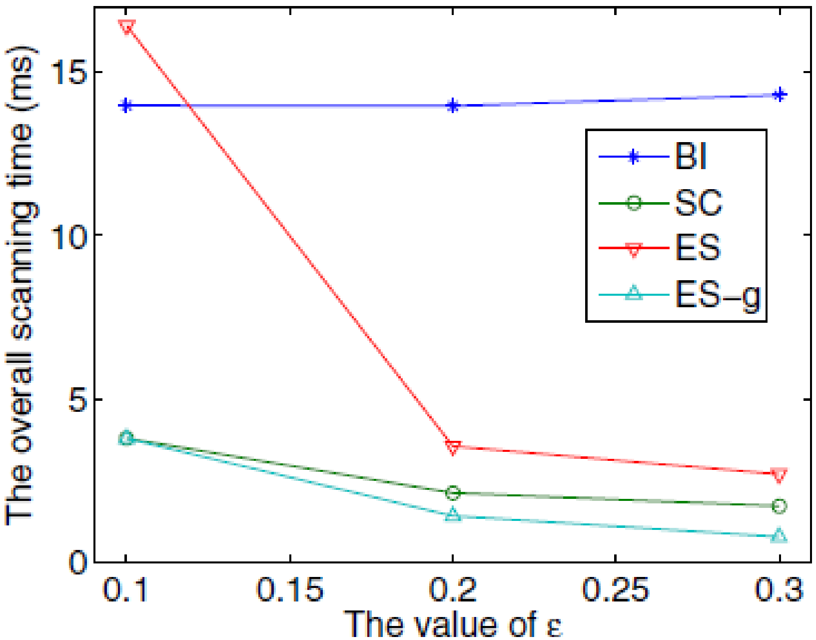
- **The Performance in Basic Histogram Collection**

BI: Basic Identification
SC: Separate Counting
ES: Ensemble sampling with one group
ES-g: Ensemble sampling with optimized grouping



(a)

Varying the number of categories and tag size

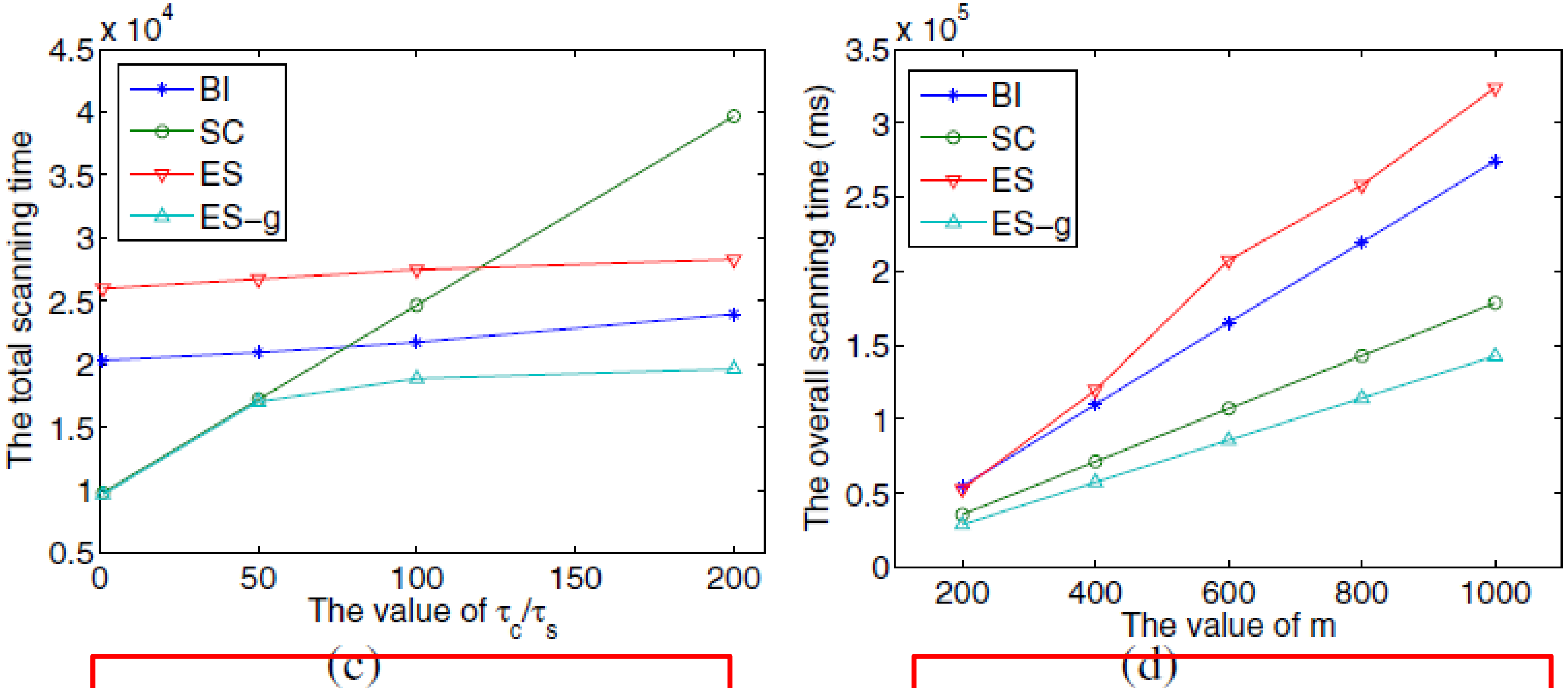


(b)

Varying the accuracy requirement

Performance Evaluation

- **The Performance in Basic Histogram Collection**

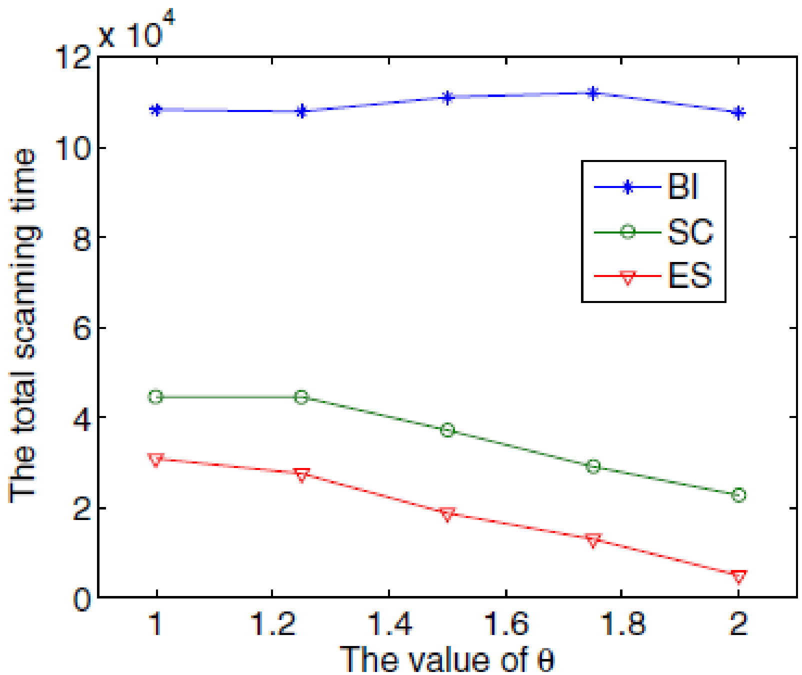


Varying the inter-cycle overhead.

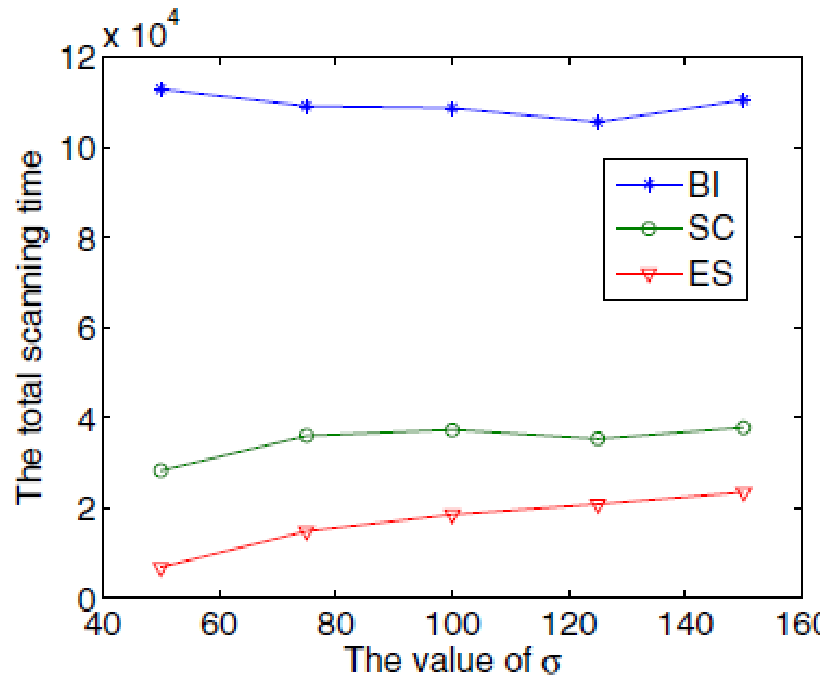
Varying the overall number of categories.

Performance Evaluation

- **The Performance in Advanced Histogram Collection**



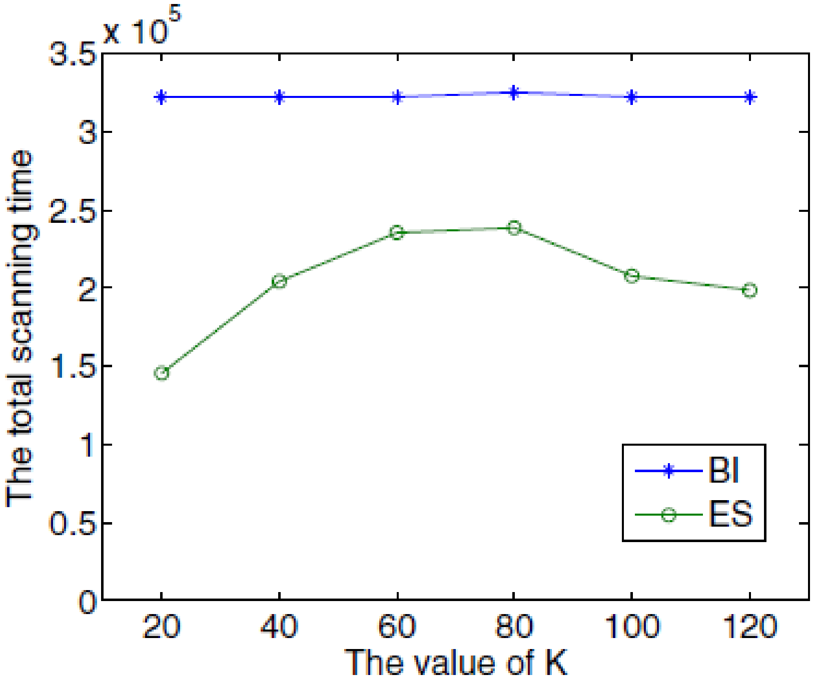
Varying the values of threshold ratio θ in iceberg query.



Varying the standard deviation σ of tags size for various categories.

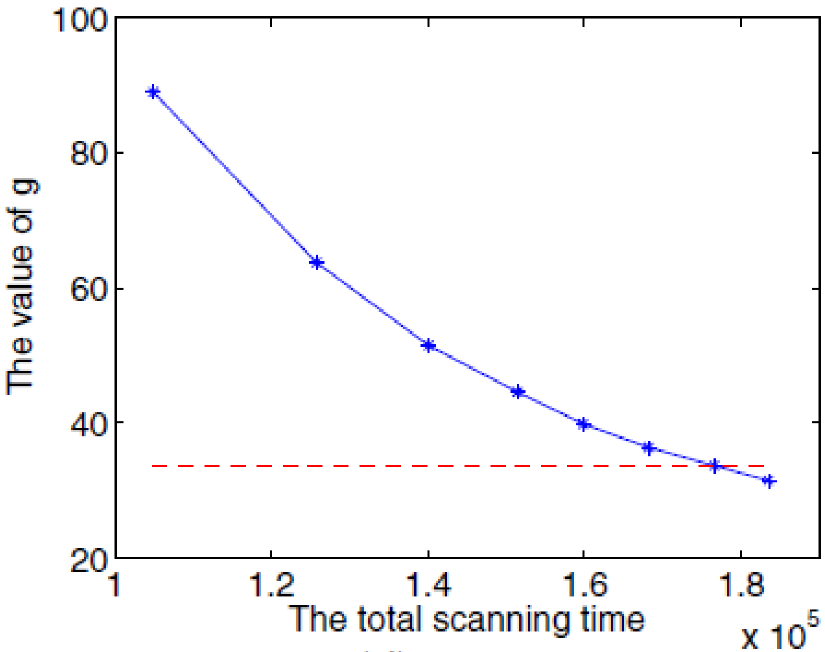
Performance Evaluation

- **The Performance in Advanced Histogram Collection**



(c)

Varying the values of k in top-k query.



(d)

Evaluate the convergence for estimating the threshold t in top-k query.

Conclusion

- We propose a series of protocols to tackle the problem of efficient histogram collection
  - Basic histogram collection
  - Iceberg Query
  - Top-k Query
- To the best of our knowledge, we are the first to consider collecting histograms over RFID tags, a fundamental premise for queries and analysis in RFID applications.
- We propose a novel, ensemble sampling-based method to simultaneously estimate the tag size for a number of categories.
- Our solutions are completely compatible with current industry standards and do not require any modification to tags.

Q & A

Thanks for your attention!