2017 IEEE 37th International Conference on Distributed Computing Systems

Category Information Collection in RFID Systems

Jia Liu†, Shigang Chen‡, Bin Xiao§, Yanyan Wang†, and Lijun Chen†
†State Key Laboratory for Novel Software Technology, Nanjing University, China
‡Department of Computer & Information Science & Engineering, University of Florida, USA
§Department of Computing, The Hong Kong Polytechnic University, China
Email:{jialiu, chenlj}@nju.edu.cn, sgchen@cise.ufl.edu, csbxiao@comp.polyu.edu.hk, cs.yywang@hotmail.com

***Abstract*—In RFID-enabled applications, when a tag is put into use and associated with a specific object, the category-related information (e.g., the brands of clothes) about this object might be preloaded into the tag's memory as required. Since such information reflects the category attributes, all tags in the same category carry the identical category information. To collect this information, we do not need to repeatedly interrogate each tag; one tag's response in a category is sufficient. In this paper, we investigate the new problem of category information collection in a multi-category RFID system, which is referred to as *information sampling*. We propose an efficient two-phase sampling protocol (TPS). By quickly zooming into a category and isolating a tag from this category, TPS is able to sample a category by broadcasting only 7.5-bit polling vector (very efficient when compared to the 96-bit tag ID). We theoretically analyze the protocol performance and discuss the optimal parameter settings that minimize the overall execution time. Extensive simulations show that TPS outperforms the benchmark, greatly improving the sampling performance.**

I. Introduction

Radio frequency identification (RFID) is becoming ubiquitously available in a variety of applications, including library inventory [1], [2], warehouse management [3]–[5], object tracking [6], [7], etc. Among these applications, RFID tags are usually attached to objects that belong to different categories, e.g., subjects of books in a library, classes of medicines in a pharmacy, or brands of clothes in a clothing outlet. When a tag is associated with a specific object, the category-related information about this object[1] is likely to be preloaded into the tag's memory. Since this information reflects the category attributes, each tag in the same category carries identical category-related information.

To collect category information in a multi-category RFID system, we do not need to repeatedly interrogate each tag. One tag's response in a category will suffice. For example, if we want to know the manufacturer of Horizon Organic milk stocked in a warehouse, we just need to query one milk box instead of all of them, as this kind of milk is produced by the same manufacturer. In another example, consider a chilled food storage chamber, where each food is affixed with a sensor-augmented RFID tag (e.g., WISP [8]) equipped with a thermal sensor. The reader periodically samples temperature readings from tags to check whether any area goes beyond the normal temperature. Since tagged objects belonging to the same category are typically packed together or placed closely, the temperature reports from these nearby tags lead to high data redundancy. Hence, it is a waste to collect sensor information from all tags in this case.

In this paper, we study the problem of category information collection in a multi-category RFID system, which is referred to as *information sampling*. To solve this problem, the existing work [9]–[11] needs to either collect all tags' information or take the entire tag set into account each time when isolating an interested tag from others, leading to time-consuming collecting process. The major reason for this is that these solutions are designed for some specific applications but not tailored to the sampling problem that has two new features: (i) Since tags in the same category carry identical category-related information, we do not need to query each individual tag. One tag's response from each category is sufficient to report the corresponding information. (ii) We do not care which tag in a category responds to the reader; anyone in the category can be a candidate for reporting.

By considering above two features, we propose an efficient two-phase sampling protocol (TPS) that consists of two phases. In the first phase, the reader separates a category from others, which helps us quickly zoom into a category instead of the entire tag set. In the second phase, the reader isolates an arbitrary tag from the separated category by using the geometry distribution of tags. Both efficient two steps make TPS far superior to existing solutions. We analyze the protocol performance and discuss the optimal parameter settings that minimize the overall execution time. The theory analysis and simulation results show that TPC is able to sample a category by broadcasting only 7.5-bit polling vector, which is very efficient when compared to the 96-bit tag ID.

The rest of the paper is organized as follows. Section II formulates the sampling problem. Section III proposes a two-phase sampling protocol. Section IV evaluates the performance of the proposed protocols. Section V introduces related work. Finally, Section VI concludes this paper.

II. Problem Statement

A. System Model

We consider an RFID system that consists of a reader and a number of tags. Each tag has a unique tag ID that exclusively represents an individual object it associates with. The tag ID contains two components: *category ID* indicating which category the tag belongs to, and *member ID* identifying

[1]In above examples, the category-related information is the subject of a book, the class of a medicine, or the brand of clothes.

1063-6927/17 $31.00 © 2017 IEEE
DOI 10.1109/ICDCS.2017.320

a specific member in this category. The tags with the same category ID share some common information, which could be static category-related information (such as the brand of the tagged product) that is preloaded when the tag is put into use, or dynamic monitoring information (such as sensor data) that is measured by nearby tags in real time. We refer to this information as *category information*. Notice the difference between the category ID and the category information. The former is a subset of tag ID and the latter is the common attributes of a class of goods. Due to limited on-chip resources, tags cannot communicate amongst themselves, but they can communicate with a reader by one-hop transmissions.

B. Problem Definition

Let $\mathcal{N}$ be a pre-known tag set in the RFID system, where $n = |\mathcal{N}|$. According to category IDs, $\mathcal{N}$ is partitioned into a family of disjoint sets $\mathcal{C} = \{C_1, C_2, ..., C_m\}$, satisfying $\bigcup_{i=1}^{m} C_i = \mathcal{N}$. For simplicity, we use C_i to represent the category ID as well as the set of tags in this category. There are $m = |\mathcal{C}|$ categories in total; each tag in $\mathcal{N}$ belongs to one of them. The sampling problem is to collect one or more bit category information from each category in a time-efficient way. Since the information in a category is identical, it is unnecessary to collect it from all tags. An arbitrary tag's response in each category is sufficient to report the required information.

III. Two-phase Sampling Protocol

A. Baseline Protocol

Basic Polling (BP), as a common anti-collision protocol, provides a request-response way to single out a tag each time in RFID systems. In BP, the reader broadcasts a tag's ID; all tags keep listening and only the uniquely matched tag replies to the reader. The remarkable advantage of BP is that the interrogating request and response are a one-to-one mapping; no collision happens in the open wireless channel. However, broadcasting a tag ID (96 bits long in the EPCglobal Gen2 standard [12]) for polling each tag is time-consuming. We refer to the broadcasting bits (from the reader) to single out a tag as *polling vector*. Clearly, in BP, the polling vector for each tag is the 96-bit tag ID. We treat it as the benchmark for comparison.

B. Basic Idea

A more sophisticated solution of the sampling problem is to randomly select m tags (denoted by M) each from different categories, and then use the advanced work ETOP [11] to collect information from the subset M. However, this design has to take the entire tag set $\mathcal{N}$ into account each time when interrogating a tag, lowering the sampling efficiency. For example, the size of the ordering vector in ETOP has to be set long enough to guarantee tags outside M are not selected. Instead of doing so, we take a deeper look at the two features of the sampling problem and propose a two-phase solution: 1) separating a category from others, and 2) singling out one tag in this category. The first step helps us quickly zoom into a category instead of the entire tag set. The second step uses only a 4-bit index to poll a single tag according to the geometry distribution of tag IDs. Both efficient two steps make up our TPS protocol that is far superior to BP and ETOP.

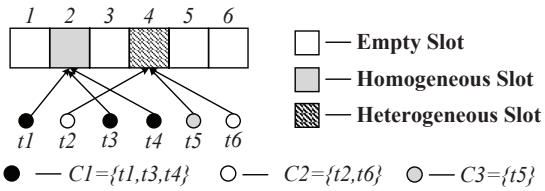


Fig. 1: Three kinds of slots in the ordering phase.

C. Protocol Description

TPS generally consists of several sampling rounds[2], each of which samples nearly 30% categories using two phases: an *ordering phase* and a *polling phase*. The ordering phase tells tags whether and when they will participate in this round, and the polling phase uses a 4-bit index to separate a tag from a category. Details are given below.

1) Ordering Phase: The reader first initializes a frame by broadcasting a request with parameters $\langle f,r \rangle$, where f is the frame size and r is a random seed. The frame is a *virtual frame* that will never be actually played out. It only serves as a vehicle to find out useful slots in this frame; an actual frame comprised of the useful slots only will be carried out later.

Upon receiving this request, each tag randomly picks a slot in the frame at the position of $H(cid, r) \mod f$, where cid is the tag's category ID and $H(\cdot)$ is a hash function shared by all tags. Since each tag takes the category ID rather than the tag ID as the hash seed, the tags from the same category must reside in the same slot. Slots picked by no tag, tags only from a single category, tags from more than one category, are called *empty slots*, *homogeneous slots*, and *heterogeneous slots*, respectively. Take Fig. 1 for example. There are three categories in total; the tags in the same category hash to the same slot. The second slot is homogeneous as it is picked by tags from the single category C_1. In contrast, the fourth slot is heterogeneous as its three tags (t_2, t_5, and t_6) are from two different categories. The left are empty slots. Notice that, the frame size f in this phase is related to only the number m of categories, regardless of the number n of tags, greatly lowering the communication overhead. For example, $m = 0.05n$ when each category has 20 tags on average. We will discuss the parameter settings and protocol overhead shortly later.

The categories residing in homogeneous slots are called *homogeneous categories* and the corresponding tags are called *homogeneous tags*. Only homogeneous slots are likely to be useful in the following protocol execution. A slot is *useful* if and only if it is homogeneous and *resolvable* (we will give the definition of resolvable slots in the polling phase). The left slots failing to meet above requirements are referred to as *useless slots*. Each tag does not know whether the slot it picks is useful or not, but the reader does. With the tag ID

[2]The expected number of rounds is about 3.5, which will be discussed later.

(containing category ID) information, the reader can predict which slots in the virtual frame are useful. It will remove the useless slots before carrying out the actual frame. For this purpose, the reader broadcasts an f-bit *ordering vector* V [11]. Each bit in V corresponds to a slot in the virtual frame: '0' indicates useless and '1' indicates useful. If V is too long, the reader can split it into many 96-bit segments and transmit each segment in a time slot of length t_{id} [9].

From a tag's perspective, the ordering vector V carries two pieces of information. For one, the tag can learn whether it has picked a useful slot by examining the corresponding bit in V. Only if it is, the tag will participate in the following polling phase. For the other, V tells the index of a useful slot in the actual frame to be carried out. If a tag finds that there are i ones in V preceding its bit (which is 1), the tag knows that it picks the $(i+1)$th useful slot.

D. Polling Phase

A useful slot is definitely a homogeneous slot. Considering a homogeneous category C_i $(1 \leq i \leq m)$ in the useful slot, we aim to single out a tag in an efficient way. In this phase, each tag individually picks an index $\mathcal{R}(H(id, r) \mod 2^K)$, where id is the tag ID, r is the same hash seed used in the previous ordering phase, K is a constant, and $\mathcal{R}(\cdot)$ is the position of the right-most bit of 1 in binary representation of the input. For example, $\mathcal{R}(81_{10}) = \mathcal{R}(01010001_2) = 1$ and $\mathcal{R}(104_{10}) = \mathcal{R}(01101000_2) = 4$. Indeed, the hash result $(H(\cdot) \mod 2^K)$ of an arbitrary tag is a K-bit binary number. When we check each bit of it from right to left, the value of each bit can be viewed as a Bernoulli trail: '1' is success and '0' is failure. $\mathcal{R}(\cdot)$ operator is actually the number of trails needed to get the first 'success', i.e., $\mathcal{R}(\cdot)$ follows geometric distribution: $Prob[\mathcal{R}(\cdot) = j] = \frac{1}{2^j}$. The feature of geometric distribution makes the number of tags decrease sharply as the index increases. For instance, about a half of tags pick the index 1 and only around $\frac{1}{2^{10}} \approx 0.1\%$ of tags pick the index 10. If an index picked by exactly one tag, we refer to it as a *singleton index*. Intuitively, the index around $\log_2(|C_i|)$ has high probability to be a singleton, where $|\cdot|$ is the cardinality of a set and $|C_i|$ indicates the number of tags in C_i.

Since the reader knows all tags' IDs, it can predict each tag's hashing result. If a slot in the previous virtual frame has one or multiple singleton indices, we call it a *resolvable slot*. As aforementioned, a useful slot is homogeneous as well as resolvable. The reader then plays out an actual frame comprised of useful slots. In each slot, a $\log_2 K$-bit singleton index is broadcast to isolate the corresponding tag (single tag), and the exclusively matched tag transmits the information as required. All tags in a category will keep silent once the information is successfully sampled. The left categories not sampled keep active and will participate in the following sampling rounds. These two phases repeat round by round until all categories are completely sampled. Note that a useful slot may contain more than one singleton index. In this case, we can randomly choose one of them as the polling vector to interrogate the corresponding tag.

E. Performance Analysis

We derive the expected execution time of the TPS protocol. Consider an arbitrary sampling round comprised of the ordering phase and the polling phase. The execution time t of this round is:

$$t = \frac{f}{96} \times t_{id} + f \times p \times (t_{poll} + t_{inf}), \tag{1}$$

where f is the frame size (also the length of the ordering vector V) in the ordering phase, p is the probability that a slot in the frame is useful, t_{poll} is the time for the reader to broadcast a $\log_2 K$-bit singleton index, and t_{inf} is the length of a time slot for a tag to transmit the required information. Note that, the control message transmission for launching each round is ignored here as this overhead covers only a couple of bits, which are negligible compared with the following frame transmission and index broadcasting by the reader. We define the *sampling efficiency*, denoted as λ, as the ratio of the number of sampled categories to the execution time of this round. Since a useful slot corresponds to a category to be sampled, the number of successfully sampled categories in this round is equal to that of useful slots, i.e., $f \times p$. We then get the sampling efficiency:

$$\lambda = \frac{f \times p}{t} = \frac{p}{\frac{t_{id}}{96} + p \times (t_{poll} + t_{inf})}. \tag{2}$$

Clearly, the bigger the value of λ is, the more categories will be sampled in each unit of execution time. We thus need to find the optimal p that maximizes λ. According to (2), λ monotonically increases as p increases; the objective is reduced to maximizing p. As aforementioned, a useful slot is both homogenous and resolvable, which are two independent events, we have:

$$p = p_h \times p_r, \tag{3}$$

where p_h is the probability that a slot is homogenous and p_r is the probability that a slot is resolvable. To maximize p, the goal further turns to maximizing both p_h and p_r, respectively. Consider the probability p_h. Since each tag takes the category ID as the input to compute which slot it picks, the tags belonging to the same category must reside in the same slot. Hence, we can simplify the derivation of p_h by treating a category as 'a super tag'. The homogenous slots are thus those picked by only one super tag. We derive the probability:

$$p_h = \binom{m'}{1}(\frac{1}{f})(1 - \frac{1}{f})^{m'-1} \approx \frac{m'}{f} \times e^{-\frac{m'-1}{f}}, \tag{4}$$

where e is the natural constant and m' is the number of categories to be sampled before this round. Letting $\frac{d\lambda(f)}{df} = 0$, we derive the maximal p_h:

$$p_h^* = e^{-1} \quad \text{when} \quad f = m'. \tag{5}$$

For the probability p_r, let us consider an arbitrary category C_i in a useful slot. After the $R(\cdot)$ operation, each tag in C_i picks the index j with the probability of $\frac{1}{2^j}$. Let q_j be the

probability that the index j is a non-singleton index (j is picked by none or multiple tags). We have:

$$q_j = 1 - \binom{|C_i|}{1} \times \frac{1}{2^j} \times (1 - \frac{1}{2^j})^{|C_i|-1}, \quad (6)$$

where $|C_i|$ is the cardinality of C_i, i.e., the number of tags in C_i. Let q_j^* be probability that the maximal singleton index is j. We have:

$$q_j^* = \begin{cases} 1 - q_K, & if\ j = K \\ q_{j+1,K} - q_{j,K}, & if\ j \leq K, \end{cases} \quad (7)$$

where K is the last index and $q_{j,K}$ is the probability that all indices from j to K are non-singleton.

According to (6) and (7), we have the probability $p_r[K, |C_i|]$ that the slot that only C_i picks is resolvable:

$$\begin{aligned} p_r[K, |C_i|] &= \sum_{j=1}^{K} q_j^* \\ &= \sum_{j=1}^{K-1} (q_{j+1,K} - q_{j,K}) + (1 - q_K) \quad (8) \\ &= 1 - q_{1,K} \\ &\approx 1 - \prod_{j=1}^{K} (1 - \frac{|C_i|}{2^j} \times e^{-\frac{|C_i|}{2^j}}). \end{aligned}$$

Fig. 2 plots the relationship between p_r and the number of tags in C_i ($|C_i|$). As we can see, when $K = 8$, the probability p_r decreases as $|C_i|$ increases. For example, $p_r \approx 0.8$ when $|C_i| = 20$, whereas $p_r \approx 0$ when $|C_i| = 10,000$. That is because, with the increase of $|C_i|$, the case of $K = 8$ hardly provides tags with sufficient indices to pick, leading to most collisions and lowering the probability p_r of a singleton index. In contrast, when $K = 16$, p_r sees only a slight decrease from 0.82 to 0.78 when $|C_i|$ varies from 10 to 10,000. The case of $K = 32$ almost remains stable at 0.82 regardless of $|C_i|$. The main reason is that the values of K in these two cases are big enough such that few tags in C_i can reach up to the last indices. We have the maximal probability p^* of a useful slot:

$$p^* = p_h^* \times p_r[K, |C_i|] = e^{-1} \times (1 - \prod_{j=1}^{K} (1 - \frac{|C_i|}{2^j} \times e^{-\frac{|C_i|}{2^j}})). \quad (9)$$

When this happens, the category C_i will be sampled in a useful slot and will not participate in the following rounds. In the ordering phase, the average number of bits for generating a useful slot is $\frac{1}{p^*}$. In the polling phase, the reader broadcasts $\log_2 K$-bit singleton index to interrogate a tag. Adding the both overhead, we get the total overhead $O(C_i)$ for singling out a tag in C_i:

$$O(C_i) = \frac{1}{p^*} + \log_2 K. \quad (10)$$

The value of K plays an important role in the overall communication overhead. Table I depicts the setting of the optimal K under different $|C_i|$. As we can see, $K = 16$ covers the interval [207, 52892], which can meet most of applications in practice. In this case, the polling vector is only $\log_2 K = 4$ bits long. Assume $|C_i| = 10,000$, we have the maximum of $O(C_i)$ by letting $p_r[K, |C_i|] = 0.78$, which is equal to $\frac{e}{0.78} + 4 \approx 7.5$ bits. Compared with transmitting 96-bit tag ID in basic polling, it is really a great performance boost.

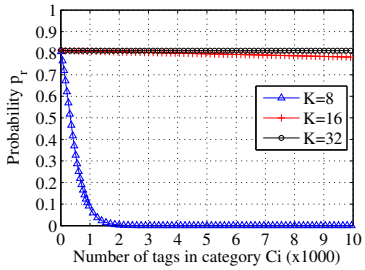


Fig. 2: Relationship between the probability p_r of a resolvable slot and the number of tags in a category C_i.

TABLE I: Optimal K

K	4	8	16	32
$\log_2$ **K**	2	3	4	5
$\|\mathbf{C_i}\|$	[1,13]	[14,206]	[207,52892]	>52892

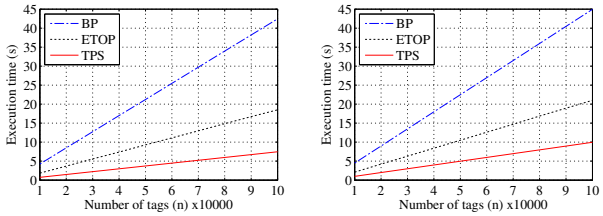
Since an arbitrary category C_i is sampled with the probability of p^* in a sampling round, the expected number of sampling rounds that C_i participates is $\frac{1}{p^*}$. According to (9), when $K = 16$ and $|C_i| = 10,000$, we have $p^* \approx 0.29$. The expected number of sampling rounds is 3.5.

IV. Evaluation

A. Simulation Setting

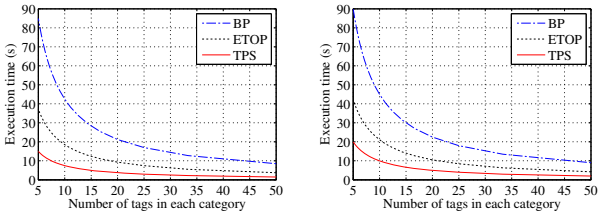
Our simulation settings follow the specification of the C1G2 standard [12]. Any two consecutive communications, from the reader to tags or vice versa, are separated by a time interval. For one, after the reader transmits the commands, all tags have to wait the transmit-to-receive turn-around time T_1 before replying to the reader. For another, upon receiving the reply from tags, the reader needs to wait the receive-to-transmit turn-around time T_2 before talking to tags. According to the specification, T_1 is $\max(RTcal, 20T_{pri})$ and T_2 ranges from $3T_{pri}$ to $20T_{pri}$, where $RTcal$ is the reader-to-tag calibration symbol that equals the length of the data-0 symbol plus the length of the data-1 symbol, and T_{pri} is the backscatter-link pulse-repetition interval. In our simulation, we set $T = T_1 = T_2 = 200\ \mu s$ that complies with the C1G2 standard.

Depending on the physical implementation and the real environment, the transmission rates between the reader and tags are not necessarily symmetric. The tag-to-reader transmission rate varies with the data coding: 40 kbps to 640 kbps for FM0 and 5 kbps to 320 kbps for Miller-modulated subcarrier. We get the intersection set 40 kbps to 320 kbps and adopt the lower bound 40 kbps as the data rate. In other words, it takes a tag 25 μs to transmit one bit. The data rate from the reader



(a) $w = 10$ (b) $w = 20$

Fig. 3: Execution time with respect to the number n of tags.



(a) $w = 10$ (b) $w = 20$

Fig. 4: Execution time with respect to the category size.

to tags ranges from 26.7 kbps to 128 kbps. Similarly, we set the data rate to the lower bound 26.7 kbps, which takes 37.45 μs to transmit one bit by the reader. Besides, the length of the category ID is set to 32 bits throughout the simulations. Note that other parameter settings may change the absolute metric, but the simulation conclusions can be drawn in a similar way.

According to the above parameter settings, the duration t_s of the 1-bit short response from tags is equal to $25+T = 225$ μs; the duration t_{cid} of transmitting a category ID by a tag is $25 \times 12 + T = 500$ μs; the duration t^r_{cid} of broadcasting a category ID by the reader is $37.45 \times 32 + T = 1398.4$ μs; the duration t^r_{id} of broadcasting a tag ID by the reader is $37.45 \times 96 + T = 3795.2$ μs; For transmitting w-bit category information by a tag, the duration t_{inf} is equal to $25w+T$ μs. All results are the average outcome of 100 simulation runs.

B. Execution Time

In this subsection, we evaluate the execution time of our sampling protocol TPS in the case of knowing tag IDs. As aforementioned in Section III, Basic Polling (BP) and ETOP can be modified for the purpose of information sampling. To achieve this goal, the reader first randomly picks a tag from each category and these tags constitute a tag set S. After that, for BP, the reader in turn broadcasts each tag's ID in S and then waits for their replies after each broadcasting. All tags keep listening and only the exactly matched tag transmits the required category information to the reader. The sampling process terminates until all tags in S are interrogated. For another, ETOP is specifically designed for collecting tag information from a tag subset in an efficient way. In our problem, we can treat S as the wanted subset of $\mathcal{N}$ and execute ETOP as-is to collect the category information as required. In the simulations, we keep the same parameter settings of ETOP as that in [11], i.e., the frame size is $24 \times |S|$, the segment is 80 bits long, and each segment consists of 4 partitions.

Fig. 3 compares the execution time of BP, ETOP, and TPS under different numbers of tags. In the simulations, we fix the number of tags in each category at 10 and vary the number of tags from 10,000 to 100,000 (the number of categories ranges from 1000 to 10000). Two kinds of category information with different lengths ($w = 10, 20$) are sampled under varied parameter settings, as shown Fig. 3(a) and Fig. 3(b). Among these figures, we observe that TPS outperforms the other two protocols as it isolates a tag of each category from others by broadcasting only about 7.5-bit polling vector on average. For example, to sample 10-bit category information from 50,000 tags (shown in Fig. 3(b)), BP takes the longest time 21.2s as it needs to transmit tedious tag IDs. ETOP reduces the execution time by 56.1% to 9.3s since it uses segmented Bloom filters to isolate and order tags belonging to S, avoiding most ID transmissions. TPS performs the best and consumes only 3.7s, producing $5.7\times$ and $2.5\times$ performance gains, compared with BP and ETOP respectively. The similar conclusion can also be drawn on other parameter settings in the three figures: TPS is the best, ETOP follows, and BP performs the worst.

In Fig. 4, we compare the execution time of BP, ETOP, and TPS under different numbers of tags in each category. In the simulation, we fix the number of tags at 100,000 and vary the number of tags in each category from 5 to 50. Once the length w of category information is fixed, we observe that the execution time of these three protocols decreases as the number of tags in each category increases. That is because the number of categories decreases with the increase of the number of tags in each category, thereby reducing the number of samples and saving the communication overhead. Similar to Fig. 3, the same conclusion can also be drawn here: TPS is the most time-efficient, ETOP is worse, and BP is the most time-consuming. For instance, when the length of category information is 20 bits long and each category has 25 tags (as shown in Fig. 4(b)), the execution time of BP is 17.0s, which is the longest amongst the three protocols. By contrast, TPS spends the minimal execution time. It takes less than 3.0s to achieve the same sampling task, producing a about $6\times$ performance gain. Although ETOP is far superior to BP, it takes longer time than TPS, i.e., 7.4s.

Note that, the execution time of the three protocols increases as w increases. This is intuitive as the tag needs to transmit more category-related data when w is bigger. Based on above simulation results, we conclude that, by transmitting a few bits to pick a tag in each category, our protocol TPS outperforms BP and ETOP, greatly improving the sampling efficiency.

V. Related Work

A great number of research has been conducted on various issues in RFID systems. Much prior work focuses on the fundamental ID-collection problems. The key ideas are to avoid tag-to-tag collisions in the open wireless channel, which generally fall into two categories: the ALOHA-based [13],

[14] and the tree-based [15], [16]. The former collects a tag's ID by carrying out a slotted frame and isolating the tag in a singleton slot (picked by exactly one tag) in the frame. The tree-based solutions iteratively split a tag set into smaller one by dynamically adjusting and broadcasting an ID prefix. This process repeats until only one tag is left and queried by the reader. In recent years, the research interests in RFID systems have been shifted to some application-oriented functions. For example, cardinality estimation [17], [18] is to count the number of tags; missing tag identification [19], [20] is to identify whether and which tags are absent; searching problems [21]–[23] try to find a group of interested tags from the existing tag set.

Information collection [9]–[11], as another branch of these new functional research, has attracted wide attentions due to its practical importance. Chen et al. [9] first formulate this problem and propose a time-efficient multihash information collection protocol (MIC) to collect sensor information from all tags. By using multiple hash functions, MIC is able to resolve most hash collisions in a slotted frame, greatly improving the protocol performance. In the follow-up work, Yue et al. [10] propose a Bloom filter based Information Collection protocol (BIC) that is tailored to the information collection under the case of multiple RFID readers. By distributively constructing and transmitting a Bloom filter, each reader can quickly identify which tags are under its coverage, speeding up the overall information collection. Unlike prior work, Qiao et al. [11] design an efficient polling-based protocol that collects tag information from only a wanted tag subset. Although these work achieve high performance, they need to collect all tags' information or take the entire tag set into account each time, which is time-consuming for the category information collection in multi-category RFID systems.

VI. Conclusion

In this paper, we study the problem of category information collection in a multi-category RFID system. Considering the new features of the problem, we propose a two-phase sampling protocol (TPS) that first quickly zooms into a category and then isolates an arbitrary tag from the category by using the geometry distribution of tags. We theoretically analyze the protocol performance and discuss the optimal parameter settings that minimize the overall execution time. Extensive simulations show that TPS is able to shorten the length of the polling vector to only 7.5 bits, which is very efficient compared with 96-bit tag IDs.

Acknowledgment

This research is financially supported by the National Natural Science Foundation of China (No. 61272418), the National Science and Technology Support Program of China (No. 2012BAK26B02), the Future Network Prospective Research Program of Jiangsu Province (No. BY2013095-5-02), the Lianyungang City Science and Technology Project (No. CG1420, JC1508), the Fundamental Research Funds for the Central Universities, CNS-1409797, and HK PolyU B-Q38F. This work is partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization, and Project funded by China Postdoctoral Science Foundation.

References

[1] R. Li, Z. Huang, E. Kurniawan, and C. K. Ho, "AuRoSS: an autonomous robotic shelf scanning system," in *Proc. of IEEE/RSJ IROS*, 2015, pp. 6100–6105.

[2] J. Liu, F. Zhu, Y. Wang, X. Wang, Q. Pan, and L. Chen, "RF-Scanner: Shelf scanning with robot-assisted RFID systems," in *Proc. of IEEE INFOCOM*, 2017.

[3] L. Xie, H. Han, Q. Li, J. Wu, and S. Lu, "Efficiently collectiing histograms over RFID tags," in *Proc. of IEEE INFOCOM*, 2014, pp. 145–153.

[4] X. Liu, K. Li, J. Wu, A. X. Liu, X. Xie, C. Zhu, and W. Xue, "Top-k queries for multi-category RFID systems," in *Proc. of IEEE INFOCOM*, 2016, pp. 1–9.

[5] K. Bu, M. Xu, X. Liu, J. Luo, S. Zhang, and M. Weng, "Deterministic detection of cloning attacks for anonymous RFID systems," *IEEE Transactions on Industrial Informatics*, vol. 11, no. 6, pp. 1255–1266, 2015.

[6] L. Yang, Y. Chen, X.-Y. Li, C. Xiao, M. Li, and Y. Liu, "Tagoram: Real-time tracking of mobile RFID tags to high precision using cots devices," in *Proc. of ACM MobiCom*, 2014, pp. 237–248.

[7] L. Xie, J. Sun, Q. Cai, C. Wang, J. Wu, and S. Lu, "Tell me what I see: Recognize RFID tagged objects in augmented reality systems," in *Proc. of ACM UbiComp*, 2016, pp. 916–927.

[8] J. R. Smith, *Wirelessly Powered Sensor Networks and Computational RFID*. Springer-Verlag New York, 2013.

[9] S. Chen, M. Zhang, and B. Xiao, "Efficient information collection protocols for sensor-augmented RFID networks," in *Proc. of IEEE INFOCOM*, 2011, pp. 3101–3109.

[10] H. Yue, C. Zhang, M. Pan, Y. Fang, and S. Chen, "A time-efficient information collection protocol for large-scale RFID systems," in *Proc. of IEEE INFOCOM*, 2012, pp. 2158–2166.

[11] Y. Qiao, S. Chen, T. Li, and S. Chen, "Energy-efficient polling protocols in RFID systems," in *Proc. of ACM MobiHoc*, 2011, pp. 25:1–25:9.

[12] *Epcglobal. EPC radio-frequency identity protocols class-1 generation-2 UHF RFID protocol for communications at 860 mhz-960mhz version 1.2.0*, Tech. Rep., 2008.

[13] S.-R. Lee, S.-D. Joo, and C.-W. Lee, "An enhanced dynamic framed slotted ALOHA algorithm for RFID tag identification," in *Proc. of MobiQuitous*, 2005, pp. 166–172.

[14] H. Vogt, "Efficient object identification with passive RFID tags," in *Proc. of IEEE PerCom*, 2002, pp. 98–113.

[15] C. Qian, Y. Liu, R. Ngan, and L. Ni, "ASAP: Scalable collision arbitration for large RFID systems," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 24, no. 7, pp. 1277–1288, 2013.

[16] L. Pan and H. Wu, "Smart trend-traversal: A low delay and energy tag arbitration protocol for large RFID systems," in *Proc. of IEEE INFOCOM*, 2009, pp. 2571–2575.

[17] X. Liu, X. Xie, K. Li, B. Xiao, J. Wu, H. Qi, and D. Lu, "Fast tracking the population of key tags in large-scale anonymous RFID systems," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 278–291, 2017.

[18] M. Shahzad and A. X. Liu, "Every bit counts: Fast and scalable RFID estimation," in *Proc. of ACM MobiCom*, 2012, pp. 365–376.

[19] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large RFID system," in *Proc. of ACM MobiHoc*, 2010, pp. 1–10.

[20] Y. Zheng and M. Li, "P-MTI: Physical-layer missing tag identification via compressive sensing," in *Proc. of IEEE INFOCOM*, 2013, pp. 917–925.

[21] M. Chen, W. Luo, Z. Mo, S. Chen, and Y. Fang, "An efficient tag search protocol in large-scale RFID systems," in *Proc. of IEEE INFOCOM*, 2013, pp. 899–907.

[22] Y. Zheng and M. Li, "Fast tag searching protocol for large-scale RFID systems," *IEEE/ACM Transactions on Networking*, vol. 21, no. 3, pp. 924–934, 2013.

[23] X. Liu, B. Xiao, S. Zhang, K. Bu, and A. Chan, "STEP: A time-efficient tag searching protocol in large RFID systems," *IEEE Transactions on Computers*, vol. 64, no. 11, pp. 3265–3277.