Range Queries for Sensor-augmented RFID Systems

Xiulong Liu*, Jiannong Cao*, Keqiu Li†, Jia Liu‡, Xin Xie§,

*Department of Computing, Hong Kong Polytechnic University, Hong Kong
†School of Computer Science and Technology, Tianjin University, China
‡Department of Computer Science and Technology, Nanjing University, China
§School of Computer Science and Technology, Dalian University of Technology, China

***Abstract*—This paper takes the first step in studying the problem of *range query* for sensor-augmented RFID systems, which is to classify the target tags according to the range of tag information. The related schemes that seem to address this problem suffer from either low time-efficiency or the information corruption issue. To overcome their limitations, we first propose a basic classification protocol called *Range Query (RQ)*, in which each tag pseudo-randomly chooses a slot from the time frame and uses the ON-OFF Keying modulation to reply its range identifier. Then, *RQ* employs a collaborative decoding method to extract the tag information range from even collision slots. The numerical results reveal that the number of queried ranges significantly affects the performance of *RQ*. To optimize the number of queried ranges, we further propose the *Partition&Mergence (PM)* approach that consists of two steps, *i.e.*, top-down partitioning and bottom-up merging. Sufficient theoretical analyses are proposed to optimize the involved parameters, thereby minimizing the time cost of *RQ+PM*. The prominent advantages of *RQ+PM* over previous schemes are two-fold: (i) it is able to make use of the collision slots, which are treated as useless in the previous schemes; (ii) it is immune to the interference from unexpected tags. We use the USRP and WISP tags to conduct a set of experiments, which demonstrate the feasibility of *RQ+PM*. Moreover, extensive simulation results reveal that *RQ+PM* can ensure 100% query accuracy, meanwhile reducing the time cost as much as 40% comparing with the existing schemes.**

***Index Terms*—RFID, Sensor-augmented Tags, Range Query.**

I. Introduction

A. Motivation and Problem Statement

Radio Frequency Identification (RFID) has been widely used in various applications such as inventory management [1]–[7], and object tracking [8]–[11], *etc.* With the development of chip manufacturing technology, RFID tags could be augmented with sensors, *e.g.*, WISP tags [12]. Thus, RFID tags do not only provide static ID information for inventory, but also the real-time information about the state of the tagged object or the environment where these objects reside. A classical problem called information collection aims to quickly collect the *exact* information from a large number of sensor-augmented RFID tags [13], [14]. However, we sometimes only need to know which range a certain tag's information belongs to. For example, consider a large food storage facility, where sensor-augmented RFID tags are attached to the food items. Periodic temperature monitoring helps ensure the quality of the food [13]. Assume that the whole temperature ambit is divided into *low, normal,* and *high* ranges, *e.g.*, $\leq 10°C$, $10°C \sim 20°C$, and $\geq 20°C$. If we know which range each tagged item belongs to, the proper countermeasures can be taken in time for the tagged items that fall in the abnormal ranges. Intuitively, querying the exact tag information consumes more time than just querying the information range. Hence, for the application scenarios in which just knowing the tag information range is satisfactory enough, we prefer the *range query* schemes to the *exact information collection* ones.

This paper takes the first step in studying the problem of range query for sensor-augmented RFID systems, which are formulated as follows. We refer to the tags that we are concerned with as the *target tags*, and use $\mathcal{T}$ to represent the target tag set. Both its size t (*i.e.*, $t = |\mathcal{T}|$) and the exact target tag IDs in $\mathcal{T}$ are known apriori. On the other hand, we refer to the tags whose IDs are not known as the *unexpected tags*, *e.g.*, the tags belong to other users. Let $\mathcal{U}$ represent the set of unexpected tags. Neither its size u (*i.e.*, $u = |\mathcal{U}|$) nor the exact tag IDs in $\mathcal{U}$ are unknown. The whole tag information ambit is divided into λ disjoint ranges $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_\lambda$. We use a reader to query the tags to classify the target tags in $\mathcal{T}$ into λ subsets $\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_\lambda$, which satisfy that, for any $1 \leq i \leq \lambda$, the information of each tag in $\mathcal{S}_i$ is within the range $\mathcal{R}_i$.

B. Limitations of Prior Art

The most related protocols are *Multi-hash Information Collection protocol (MIC)* [13], and *Tag-Ordering Polling protocol (TOP)* [14], which were proposed to collect the exact tag information. They have two major drawbacks when addressing the problem of range query. First, they only make use of the *singleton slot* in which only one tag replies information. The *collision slot* in which two or more tags simultaneously reply information is totally wasted. Hence, the frame utilization is quite low. Second, they make a too ideal assumption that all tag IDs are known apriori, which is not always true in the wild application scenarios. For example, considering a multi-tenant warehouse where each tenant possesses a set of tagged items, a tenant only has the right to know his/her own tag IDs, while knowing nothing about the tag IDs of the other tenants [15]. Due to the interference of unexpected tags, the information sent from the target tags may be corrupted and cannot be correctly received by the reader.

C. Proposed Approach

This paper first proposes the *Range Query (RQ)* protocol. Initially, we generate a λ-bit binary string called *range indicator* for each target tag. The index of bit `1` in a range indicator indicates which range this target tag may belong to.

So far, we know nothing about the information of the target tags, *i.e.*, each target tag may belong to any range. Therefore, the range indicator of each target tag is initialized to all 1s. *RQ* includes multiple rounds of queries. In each round, the reader first broadcasts the information ranges $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_\lambda$ to all tags. Then, an arbitrary tag can learn which range its information belongs to. The tag whose information is within the range $\mathcal{R}_i$ will generate another λ-bit string called *range identifer* in which only the i-th bit is 1 while all the other $\lambda-1$ bits are 0s. Then, the reader broadcasts the parameters f and s to initialize a slotted time frame, where f is the number of slots in the forthcoming frame and s is a random seed. Each tag pseudo-randomly selects a slot in the frame by calculating $c = \mathcal{H}(ID, s) \mod f$, and responds in the c-th slot with its range identifier using the ON-OFF Keying modulation, *i.e.*, a bit 1 is represented by the presence of a carrier wave; a bit 0 is represented by the absence of a carrier wave. Multiple tags may reply their range identifiers in the same slot, and the combined signal received by the reader is logically equal to the result of the bitwise OR operation on these range identifiers. For a target tag, the combined signal received by the reader in its selected slot is called its footprint. Since we know all the target tag IDs and can calculate $c = \mathcal{H}(ID, s) \mod f$ for each of them in advance, we are able to know the mapping between the target tags and the footprints. After obtaining the footprint $\mathcal{F}$ of a target tag, we perform the bitwise AND operation to update its range indicator $\mathcal{I}$, *i.e.*, $(\mathcal{I}$ AND $\mathcal{F}) \Rightarrow \mathcal{I}$. For an arbitrary target within the range $\mathcal{R}_i$, the i-bit in its footprint $\mathcal{F}$ should be always 1. Then, the i-th bit in its range indicator $\mathcal{I}$ should be always 1, too. On the contrary, the other bits in $\mathcal{I}$ tend to become 0s, as the query process repeats round by round. A target tag will be classified into the subset $\mathcal{S}_i$, if only the i-th bit in its range indicator $\mathcal{I}$ is 1. *RQ* repeats this process until all target tags are classified.

D. Challenges and Proposed Solutions

The first challenge is to optimize the involved parameters to minimize the execution time of *RQ*. Due to the practical reason [16], the frame size f should be better no more than 512. A solution is to let the reader announce a relatively large frame size f, but send a command to terminate the frame after the first $f' \leq 512$ slots. This paper refers to f and f' as the broadcasted frame size and executed frame size, respectively. To optimize the values of f and f', we first theoretically investigate the probability that an arbitrary target tag could be successfully classified in one round of query, and then calculate how many rounds of queries should be repeated by *RQ* to classify all target tags. Further, we calculate the upper bound $\mathcal{T}_{RQ}^{up}$ on the execution time of *RQ*, which is a function of f and f'. We calculate the first-order partial derivatives of $\mathcal{T}_{RQ}^{up}$ with respect to f and f', respectively. Finally, we prove that $\mathcal{T}_{RQ}^{up}$ achieves the minimum value when $f = M+1$ and $f' = \min\{M+1, 512\}$, where, $M = \max\{n_i | i \in [1, \lambda]\}$ and n_i is the number of tags that belong to the range $\mathcal{R}_i$.

The second challenge is to optimize the number of queried ranges. We observe from the numerical results that, as the number of ranges λ increases, the time cost of *RQ* and its upper bound are both first downwards and then upwards. Hence, it is not efficient to set a too small value or a too large value for λ. To solve this issue, we propose the *Partition&Mergence* approach that consists of two steps: top-down partitioning and bottom-up merging. In the first step, we recursively partition the given λ ranges into $\lambda \times 2^l$ through l layers. Then, we find out the optimal layer w that corresponds to the smallest time cost, *i.e.*, the best partition strategy. Using the $\lambda \times 2^{w-1}$ ranges on layer w, we perform the *RQ* protocol and classify the target tags into $\lambda \times 2^{w-1}$ subsets. In the second step, we recursively merge the tag IDs in two adjacent ranges from layer w to layer 0, and finally classify the target tags into λ subsets.

E. Novelty and Advantages over Prior Art

The major novelty of this paper is two-fold: (i) We formulate and study the new practically important problem of *range query* for sensor-augmented RFID systems; (ii) We propose the *RQ+PM* protocol to quickly classify the target tags. The key technical depth is in proving the upper bound on the execution time of the proposed protocol, optimizing the frame sizes and the number of queried ranges as well. The key advantages of the proposed solution over the previous schemes are also two-fold: (i) *RQ+PM* can make use of the collision slot that contains two or more tag replies, which is treated as useless in previous schemes. Thus, the frame utilization is significantly improved; (ii) Unlike the previous protocols that suffer from information corruption issue, *RQ+PM* is immune to the existence of unexpected tags. Extensive simulation results reveal that *RQ+PM* can ensure 100% query accuracy, meanwhile reducing the time cost by as much as 40% comparing with the existing protocols.

The remainder of this paper is organized as follows. In Section II, we describe the detailed design of the *RQ* protocol, and optimize the involved parameters. In Section III, we propose the *PM* approach to find the optimal number of ranges. In Section IV, we conduct extensive simulations to evaluate the performance of *RQ+PM*. We discuss the related work in Section V. Finally, Section VI concludes this paper.

II. The Range Query Protocol

A. Detailed Design of The RQ Protocol

Assume the user queries the tags using λ ranges. At the very beginning, we generate a λ-bit binary string called *range indicator* for each target tag. The bits 1s in a range indicator $\mathcal{I}$ indicate which ranges the corresponding target tag may belong to. Consider the example shown in Fig. 1. The number of queried ranges λ is 3, hence, the *range indicator* of each target tag has 3 bits. If a target tag's range indicator is 101, it means that this tag may belong to the 1-st and the 3-rd ranges. So far, we know nothing about the information range of any target tags, hence, each range indicator is initialized to all 1s. It means that the corresponding target tag may belong to any ranges. The proposed *RQ* protocol involves multiple rounds of query processes to classify the target tags in $\mathcal{T}$ into λ subsets $\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_\lambda$. In each round of query,

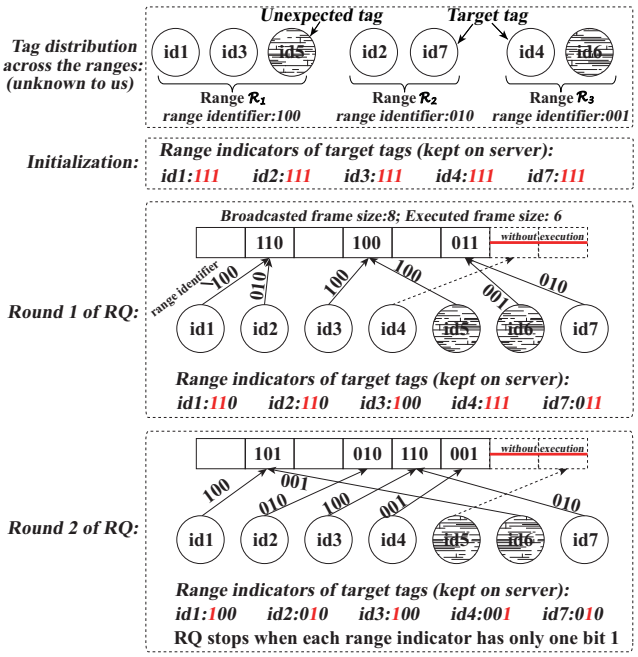

Fig. 1. Exemplifying the proposed *Range Query (RQ)* protocol.

the reader first broadcasts the ranges $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_\lambda$ to all tags, thereby telling them how the whole information ambit is partitioned. For each tag, if it finds that its information is within the range $\mathcal{R}_i$, it will generate a λ-bit binary string called *range identifier*, in which exactly the i-th bit is `1` and all the other bits are `0`s, $i \in [1, \lambda]$. Then, the reader broadcasts the parameters $\langle s, f \rangle$ to initialize a slotted time frame, where s is a random seed and f is the number of time slots in the forthcoming frame. Upon receiving these two parameters, each tag resets its slot counter c by calculating $c = \mathcal{H}(id, s) \mod f$. The reader broadcasts `QueryRep` command at the end of each slot to notify each tag to decrement its slot counter c by one. Once the slot counter c of a tag becomes 0, it will respond to the reader with its range identifier using the ON-OFF Keying modulation: a bit `1` is represented by the presence of a carrier wave; a bit `0` is represented by the absence of a carrier wave. Multiple tags may respond in the same slot. And the *combined signal* received by the reader in this slot is logically equal to the result of the bitwise `OR` operation on all the replied range identifiers. As aforementioned, due to practical reasons [16], the actual frame size should be no more than 512. We let the reader announce a relatively large frame size f, but send a command to terminate the frame after the first $f' \leq 512$ slots. To differentiate between f and f', we call them the *broadcasted frame size* and the *executed frame size*, respectively. As exemplified in Fig. 1, $f = 8$ and $f' = 6$.

Since we know all target tag IDs and all hashing parameters (*i.e.*, s and f), we are able to know which slot each target tag chose to reply its range identifier. For a target tag, we refer to the λ-bit combined signal received in its chosen slot as its *footprint* denoted as $\mathcal{F}$. Note that, a footprint may be shared by multiple tags because they responded in the same slot. If these tags belong to the same range, says $\mathcal{R}_i$, the i-th bit of the footprint will be `1`, *i.e.*, $\mathcal{F}[i] = 1$. In Fig. 1, consider the 4-th slot in the first round of *RQ* as an example. In contrary, if these tags belong to different ranges, *e.g.*, two ranges $\mathcal{R}_i$ and $\mathcal{R}_j$, we should have $\mathcal{F}[i] = 1$ and $\mathcal{F}[j] = 1$. Consider the 2-nd slot in the first round of *RQ* as an example. It is easy to understand that the footprint can reflect what ranges of tags responded in this slot. After executing a frame, we could obtain footprints for some target tags; but could not for the other target tags, because the reader only observes the first f' slots instead of the whole time frame, *e.g.*, $id4$ in the first round of *RQ* in this example. For an arbitrary target tag that occupies a footprint in this time frame, we use its footprint $\mathcal{F}$ to update its range indicator $\mathcal{I}$ by performing $(\mathcal{I}$ `AND` $\mathcal{F}) \Rightarrow \mathcal{I}$. For example, in the first round of *RQ*, the footprint of $id1$ is `110`. Hence, its range indicator is updated by performing `111 AND 110`. For an arbitrary target tag that belongs to the range $\mathcal{R}_i$, only the i-th bit in its footprint is necessary to be `1`. As to the other bits, if this target tag does not share the footprint with the tags that belong to the range $\mathcal{R}_j$ ($j \neq i$), the j-th bit in the footprint will be 0. That is, the other bits have a chance to be `0`s. Therefore, we can assert that, as the query process repeats round by round, the i-th bit in its range indicator $\mathcal{I}$ will be consistently `1`, while the other bits tend to become `0`s. When *only one bit* in its range indicator is `1`, says the i-bit, we can assert that the information of this target tag is within the range $\mathcal{R}_i$, and this target tag will be classified into the subset $\mathcal{S}_i$. *RQ* repeats round by round until all target tags are classified. We summarize the main notations used in this paper in Table I.

TABLE I
MAIN NOTATIONS USED IN THE PAPER.

Notations	Descriptions
$\mathcal{T}$	Set of target tags.
t	Number of target tags, *i.e.*, $t = \lvert\mathcal{T}\rvert$.
$\mathcal{U}$	Set of unexpected tags.
u	Number of unexpected tags, *i.e.*, $u = \lvert\mathcal{U}\rvert$.
$\mathcal{R}_i$	Information range.
λ	Number of queried ranges.
n_i	Number of tags that belong to the range $\mathcal{R}_i$.
M	$\max\{n_i \lvert i \in [1, \lambda]\}$
f	broadcasted frame size.
f'	executed frame size.
$H(\cdot)$	uniform hash function.
τ_d^ν	length of slot that conveys ν-bit data from reader to tags.
τ_u^λ	length of slot that conveys λ-bit data from tag to reader.
$\mathcal{F}$	Footprint of a target tag.
$\mathcal{I}$	Range indicator a target tag.
T_{RQ}	Execution time of the *RQ* protocol.
T_{RQ}^{up}	Upper bound on the execution time of *RQ*.

B. Bit-level Synchronization

Bit-level synchronization is the key foundation to implement the proposed *RQ* protocol. Katabi *et al.* reported in [17] that the synchronization offset for commercial RFID tags is normally no more than $1us$. Recall that transmitting each bit from a tag to a reader requires $18.8us$. Hence, the $1us$ offset is only about 5.3% of a bit duration. In other words, the signal

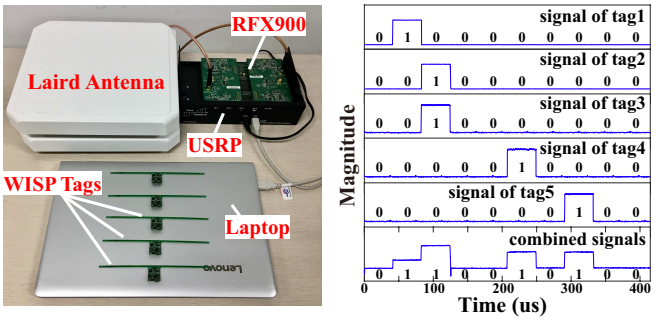

Fig. 2. Verifying the bit-level synchronization of ON-OFF keying modulation.

offset does not have much negative impact on the proposed *RQ* protocol. To verify this statement, we also conduct a set of experiments to observe the signal superposition of RF waves with a USRP1 and five WISP tags, as illustrated in Fig. 2(a). In the experiment, USRP1 works in the 920 MHz UHF band and serves as an RFID reader. Two circular antennas are mounted to two RF daughterboards Flex900 on USRP1; one is for transmitting carrier waves and instructions, and the other is for receiving the data from WISP tags. Fig. 2(b) shows the signal level results from one set of experiments. The first five plots are the individual signals emitted by each tag. Since ON-OFF Keying modulation is used in the experiment, each logical bit can be obtained by observing the value of magnitudes of the signal. For example, the data from tag2 are '0010000000'. The sixth plot is the combined signals when five tags transmit their own data in parallel. As we can see, the logical bits presented by the aggregated symbols are '0110010100', which are actually the bitwise OR of individual signals from five tags. With these experiments, we assert that it is feasible to exploit the mixed signals in our protocol design.

C. Parameter Optimization

In what follows, we first propose Lemma 1 to give the upper bound on the time cost of *RQ*, which is a function of f and f'. Then, we propose Theorem 1 to optimize the values of f and f' by minimizing the obtained upper bound.

Lemma 1. *Let f be the broadcasted frame size, f' be the executed frame size, λ be the number of queried ranges, t be the number of target tags, $M = \max\{n_i | i \in [1,\lambda]\}$, where n_i is the number of tags that belong to the range $\mathcal{R}_i$. To classify all t target tags, the upper bound on the expected execution time of RQ, denoted as T_{RQ}^{up}, is given as follows.*

$$T_{RQ}^{up} = -\frac{f(f'\tau_u^{\lambda} + \tau_d^{\nu} + \lambda\tau_d^{\nu})\ln[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}]}{f'(1-\frac{1}{f})^{M}}, \tag{1}$$

where τ_u^{λ} and τ_d^{ν} are the length of two types of slots, which are specified in Table I.

Proof: Suppose the *RQ* protocol has executed for k rounds. In an arbitrary round of query, since the reader just observes the first f' slots instead of the whole time frame, each target tag has a probability of f'/f to have a footprint. For k rounds of queries, we could collect kf'/f footprints for each target tag on average. For a target tag that belongs to the range $\mathcal{R}_i$, the j-th bit ($j \neq i$) in a certain footprint is 0, if and only if it does not share this footprint with any other tags belonging to the range $\mathcal{R}_j$. The corresponding probability $P\{\mathcal{F}[j] = 0\}$ can be calculated by $(1-1/f)^{n_j}$, where n_j is the number of tags that belong to the range $\mathcal{R}_j$. Then, the probability that the j-th bit in the range indicator is 0, denoted as $P\{\mathcal{I}[j] = 0\}$, is equal to the probability that *at least one* of the kf'/f footprints satisfies $\mathcal{F}[j] = 0$. Hence, $P\{\mathcal{I}[j] = 0\}$ can be calculated as follows.

$$P\{\mathcal{I}[j] = 0\} = 1 - \left[1 - \left(1 - \frac{1}{f}\right)^{n_j}\right]^{\frac{kf'}{f}} \tag{2}$$

The target tag can be successfully classified if only the i-th bit in its range indicator is `1` and all the other bits become `0`s. We use P_i to denote the probability that a target tag belonging to the range $\mathcal{R}_i$ can be classified, which can be calculated by $P_i = \prod P\{\mathcal{I}[j] = 0\}$, where $j \in [1,\lambda]$ and $j \neq i$. By substituting Eq. (2) into P_i, it can be transformed as follows.

$$P_i = \prod_{j\in[1,\lambda]\wedge j\neq i} 1 - \left[1 - \left(1 - \frac{1}{f}\right)^{n_j}\right]^{\frac{kf'}{f}} \tag{3}$$

Since $M = \max\{n_i | i \in [1,\lambda]\}$, it is easy to know that $M \geq n_j$. According to Eq. (3), we have the following inequality.

$$P_i \geq \left\{1 - \left[1 - \left(1 - \frac{1}{f}\right)^{M}\right]^{\frac{kf'}{f}}\right\}^{\lambda-1} \tag{4}$$

The probability that a target tag belonging to the range $\mathcal{R}_i$ cannot be classified is $1-P_i$. Then, the number of target tags that cannot be classified is calculated as follows.

$$\sum_{i\in[1,\lambda]}\{n_i \times (1-P_i)\} \leq t - t \times \left\{1 - \left[1 - \left(1 - \frac{1}{f}\right)^{M}\right]^{\frac{kf'}{f}}\right\}^{\lambda-1} \tag{5}$$

To classify all target tags, we set the right hand side of Eq. (5) to 1. Then, we calculate the round count k as follows.

$$k = \frac{f\ln\left[1 - \left(1 - \frac{1}{t}\right)^{\frac{1}{\lambda-1}}\right]}{f'\ln\left[1 - \left(1 - \frac{1}{f}\right)^{M}\right]} \tag{6}$$

Eq. (6) indicates how many rounds of queries are required on average to classify all t target tags. In each round of query, the reader takes λ slots to broadcast $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_\lambda$ to tags, and another slot to broadcast parameters $\langle s, f\rangle$ to tags for frame initialization. For simplicity, we assume that both the information range $\mathcal{R}_i$ and the parameters $\langle s, f\rangle$ are ν bits. Besides, f' slots in the time frame will be executed. Here, we use τ_d^{ν} to denote the length of slot that conveys ν-bit data from reader to tags (downlink), and τ_u^{λ} to denote the length of slot that conveys λ-bit tag identifier to reader (uplink). Therefore, the expected time cost of *RQ* can be calculated by $T_{RQ} = k(\lambda\tau_d^{\nu} + \tau_d^{\nu} + f'\tau_u^{\lambda})$, and be further transformed as follows.

$$T_{RQ} = \frac{f(\lambda\tau_d^{\nu} + f'\tau_u^{\lambda} + \tau_d^{\nu})\ln[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}]}{f'\ln[1-(1-\frac{1}{f})^{M}]} \tag{7}$$

Based on the fact that $\ln[1-q] < -q$ when $q \in (0,1)$, the time cost T_{RQ} can be magnified to its upper bound as follows.

$$T_{RQ} < -\frac{f(\lambda\tau_d^{\nu} + \tau_d^{\nu} + f'\tau_u^{\lambda})\ln[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}]}{f'(1-\frac{1}{f})^M} \quad (8)$$

Hence, the statement in this lemma has been proved. ■

Note that, the statement in Lemma 1 requires the value of $M = \max\{n_i | i \in [1,\lambda]\}$, where n_i is the number of tags that belong to the range $\mathcal{R}_i$. However, it is not known apriori. Here, we first assume that M is a known value. Later, we will discuss how to obtain this value in practice. Since the parameters f and f' significantly affect the upper bound on the time cost of *RQ*, we will propose Theorem 1 to investigate the optimization of these two parameters to minimize the theoretical upper bound on the execution time.

Theorem 1. *Let $M = \max\{n_i | i \in [1,\lambda]\}$, where n_i is the number of tags that belong to the range $\mathcal{R}_i$. To minimize the upper bound T_{RQ}^{up} on the execution time of the RQ protocol, the broadcasted framed frame size f should be set to $M+1$, and the executed frame size should be set to* $\min\{M+1, 512\}$.

Proof: We calculate the first-order partial derivative of T_{RQ}^{up} with respect to f as follows.

$$\frac{\partial T_{RQ}^{up}}{\partial f} = -\frac{\ln[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}](f'\tau_u^{\lambda} + \lambda\tau_d^{\nu} + \tau_d^{\nu})(f-1-M)}{f'(f-1)(1-\frac{1}{f})^M} \quad (9)$$

In Eq. (9), since $(1-\frac{1}{t})^{\frac{1}{\lambda-1}} \in (0,1)$, $\ln[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}]$ is always less than 0. Then, it is easy to have $\frac{\partial T_{RQ}^{up}}{\partial f} = 0$ when $f = M+1$; $\frac{\partial T_{RQ}^{up}}{\partial f} > 0$ when $f > M+1$; and $\frac{\partial T_{RQ}^{up}}{\partial f} < 0$ when $f < M+1$. Hence, we achieve the minimum value of T_{RQ}^{up} when the broadcasted frame size f is set to $M+1$.

Similarly, we calculate the first-order derivative of T_{RQ}^{up} with respect to f' as follows.

$$\frac{\partial T_{RQ}^{up}}{\partial f'} = \frac{f \ln\left[1-\left(1-\frac{1}{t}\right)^{\frac{1}{\lambda-1}}\right](\lambda\tau_d^{\nu} + \tau_d^{\nu})}{\left(1-\frac{1}{f}\right)^M f'^2} \quad (10)$$

Since $\ln[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}] < 0$, we have $\frac{\partial T_{RQ}^{up}}{\partial f'} < 0$, which means that T_{RQ}^{up} is a monotonously decreasing function with respect to f'. To minimize T_{RQ}^{up}, we should set f' to its maximum value. As a matter of fact, we have $f' \leq 512$ and $f' \leq f = M+1$. Thus, f' should be set to $\min\{M+1, 512\}$. ■

D. Obtaining the Value of M

In this section, we discuss how to estimate the number of tags in each range, and thus obtain the value of M that is required when optimizing f and f'. For the *first* round of query in *RQ*, we modify the protocol in [18], and propose a light-weight estimation protocol. Specifically, the reader issues a time frame, with broadcasted frame size $f = 2$ and executed frame size $f' = 1$. If a tag chooses the first slot, it will respond its range identifier using the ON-OFF Keying modulation. If the first slot is not empty, the reader terminates the current time frame and starts a new time frame that still keeps $f' = 1$ but *exponentially* increases f to 4. Generally, in the ℓ-th time frame, f' is consistently set to 1, and f is set to 2^{ℓ}. As frames go on, the reader will observe an empty slot eventually, because the broadcasted frame size increases rapidly. Suppose after executing m single-one slot frames, the reader observes an empty slot. For clarity, we refer to the footprint obtained in the ℓ-th frame as $\mathcal{F}_{\ell}$. To estimate the number of tags that belong to the range $\mathcal{R}_i$, we find the *maximum* value of ξ that satisfies $(\prod_{1 \leq \ell \leq \xi \leq m} \mathcal{F}_{\ell}[i]) = 1$, which means the number of continuous footprints that are occupied by the tags belonging to the range $\mathcal{R}_i$. Intuitively, the more tags fall in the range $\mathcal{R}_i$, the larger the value of ξ is. The literature [18] had given their quantitative relationship $n_i = 1.2897 \times 2^{\xi}$, based on which we can estimate the number of tags in each range. Note that, the complexity of this approach is as low as $\mathcal{O}(\log M)$, and it just takes tens of slots at most.

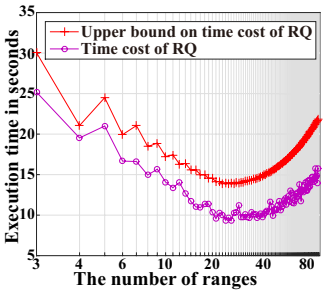

Fig. 3. The time cost of *RQ vs.* the number of queried ranges.

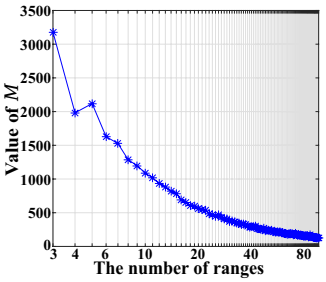

Fig. 4. The value of M *vs.* the number of queried ranges.

For the γ-th round of query ($\gamma \geq 2$), *RQ* is able to estimate the number of tags in each range *with no extra* overhead. Specifically, after executing the last round of query, we are able to construct a bit vector $\mathbb{V}_i$ with f' bits for an arbitrary range $\mathcal{R}_i$, $i \in [1,\lambda]$. For each bit in the vector, we set $\mathbb{V}_i[j] = 1$, *if and only if* we obtain a footprint in the j-th slot and the i-th bit in this footprint is `1`; otherwise, $\mathbb{V}_i[j] = 0$. In fact, the functional relationship between n_i and the number of `0`s in the bit vector $\mathbb{V}_i$ (denoted as $N_{i,0}$) is $n_i = -f \ln(\frac{N_{i,0}}{f'})$ [19]. Then, we can use the number of `0`s in each bit vector to estimate the number of tags in the corresponding range. After estimating the value of each n_i ($1 \leq i \leq \lambda$), it is easy to obtain the value of $M = \max\{n_i | i \in [1,\lambda]\}$.

III. The Partition&Mergence Protocol

A. Motivation of Partition&Mergence (PM)

An intuitive idea is that, *the larger the number λ of queried ranges is, the finer granularity of the query results will be, and the more time our RQ protocol will cost.* To verify this idea, we conducted a set of simulations to investigate the impact of λ on the time cost of *RQ*. Here, we simulated a system that contains 4000 target tags and 500 unexpected tags. The information of each tag is within the ambit $[0, 50]$ and follows the normal distribution $Norm(25, 8)$. All ranges $\mathcal{R}_1, \mathcal{R}_2, \cdots, \mathcal{R}_{\lambda}$ are with the same range width. Fig. 3 plots the execution time of *RQ* and its upper bound with varying λ. We observe from Fig. 3 that the *overall trend* of the time cost of *RQ* and its upper bound are first downwards and then upwards as λ increases.

Such an observation contradicts the above intuitive idea. Next, we will explain the underlying reasons.

By substituting the optimal settings $f = M+1$ and $f' = \min\{512, M+1\}$ into Eq. (1), the upper bound on the execution time T_{RQ}^{up} can be transformed as:

$$T_{RQ}^{up} \approx -e \times M \times \left(\tau_u^{\lambda} + \frac{\tau_d^{\nu} + \lambda\tau_d^{\nu}}{\min\{512, M+1\}}\right) \times \ln\left[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}\right] \quad (11)$$

In the following, we calculate the partial derivatives of T_{RQ}^{up} against M and λ, respectively.

$$\bullet \frac{\partial T_{RQ}^{up}}{\partial M} = \begin{cases} -e \ln\left[1-\left(1-\frac{1}{t}\right)^{\frac{1}{\lambda-1}}\right] \times \left[\tau_u^{\lambda} + \frac{\tau_d^{\nu} + \lambda\tau_u^{\nu}}{(M+1)^2}\right] : M+1 \leq 512 \\ -e \ln\left[1-\left(1-\frac{1}{t}\right)^{\frac{1}{\lambda-1}}\right] \times \left(\tau_u^{\lambda} + \frac{\tau_d^{\nu} + \lambda\tau_u^{\nu}}{512}\right) : M+1 > 512 \end{cases}$$

$$\bullet \frac{\partial T_{RQ}^{up}}{\partial \lambda} = \frac{\partial\left(\tau_u^{\lambda} + \frac{\tau_d^{\nu}+\lambda\tau_d^{\nu}}{\min\{512, M+1\}}\right)}{\partial \lambda} \times \left\{-eM \ln\left[1-\left(1-\frac{1}{t}\right)^{\frac{1}{\lambda-1}}\right]\right\} + \left(\tau_u^{\lambda} + \frac{\tau_d^{\nu}+\lambda\tau_d^{\nu}}{\min\{512, M+1\}}\right) \times \frac{-eM \ln\left(1-\frac{1}{t}\right) \times \left(1-\frac{1}{t}\right)^{\frac{1}{\lambda-1}}}{(\lambda-1)^2[1-(1-\frac{1}{t})^{\frac{1}{\lambda-1}}]}$$

It is easy to find that the partial derivatives $\frac{\partial T_{RQ}^{up}}{\partial M}$ and $\frac{\partial T_{RQ}^{up}}{\partial \lambda}$ are always larger than 0, which infers that T_{RQ}^{up} is a monotonically increasing function against both M and λ. As shown in Fig. 4, when λ is small, the value of M is quite large. And the large value of M will dominate the time upper bound T_{RQ}^{up}. In contrary, when λ is very large, although M is small, the large value of λ will dominate the value of T_{RQ}^{up} instead. Hence, *neither a too small value nor a too large value for λ is preferred, which motivates us to investigate how to choose a proper number λ of queried ranges.* For example, consider the numerical results in Fig. 3. If the user wants to classify the target tags into $\lambda = 3$ ranges, it is more time-efficient to first perform the *RQ* protocol with $\lambda = 6$. After classifying the target tags into 6 ranges, we can merge the adjacent two ranges into one bigger range. Thus, we can still correctly classify the target tags into the desired 3 ranges.

An interesting observation from Fig. 3 is that the execution time of the *RQ* and its upper bound sometimes change in a *zigzag* shape as λ increases (*e.g.*, $\lambda = 3, 4, 5$). We will explain this phenomenon below. The numerical results in Fig. 4 reveal that the *overall trend* of M is downwards with respect to λ. However, sometimes it fluctuates due to the *skew* distribution of the simulated tags across the ranges. For example, the value of M when $\lambda = 5$ is larger than that when $\lambda = 4$. We have proved that T_{RQ}^{up} is an increasing function against M. It is understandable that the time upper bound of *RQ* when $\lambda = 5$ is larger than that when $\lambda = 4$.

B. Design of The Partition&Mergence (PM) Approach

To find a proper number of queried ranges, this section will propose a *Partition&Mergence (PM)* approach that consists of two steps: *top-down partitioning* and *bottom-up merging*. As exemplified in Fig. 5, the top-down partitioning operation is to first keep the λ_0 ranges queried by the user on layer 0. Then, we partition each range on layer 0 into two sub-ranges, and thus generate $2\lambda_0$ ranges, which are kept on layer 1.

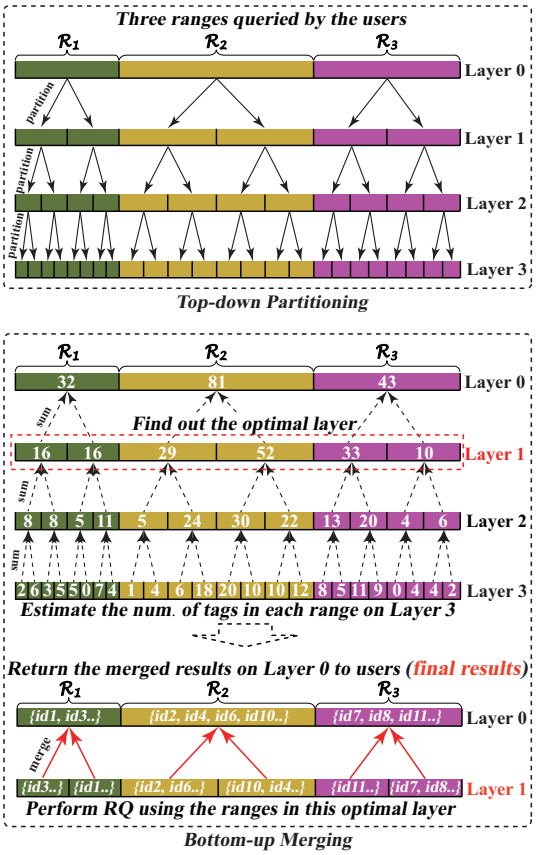

Fig. 5. Exemplifying the *Partition&Mergence (PM)* approach, which consists of two steps: top-down partitioning and bottom-up merging.

Repeating such processes, there will be $\lambda_0 \times 2^l$ sub-ranges on the last layer l. This example contains 4 layers, and the whole information ambit is temporarily partitioned into $3 \times 2^3 = 24$ sub-ranges. The two adjacent ranges which are partitioned from the same range are called *brother ranges* to each other, and the corresponding range on the upper layer is called *father range*. Then, *RQ* queries the tags using the $\lambda_0 \times 2^l$ sub-ranges on layer l for several rounds. With the method proposed in Section II-D, we could estimate the number of tags in each sub-range on layer l. Then, we will perform the *bottom-up merging* operations. Specifically, we sum the two numbers of tags in each pair of brother ranges to get the number of tags in their father range. By repeating such processes recursively from layer l to layer 0 (bottom-up), we could estimate the number of tags in each range on each layer. And then, the values of M_i and λ_i for each layer will be obtained. According to Lemma 1, we can calculate the upper bound on the execution time of *RQ* for each pair of M_i and λ_i, and then find the *optimal layer* that minimizes the value of T_{RQ}^{up}. After that, we keep on using the λ_{opt} ranges on the optimal layer to query the tags. As exemplified in Fig. 5, *Layer 1* is the optimal layer and the number of ranges $\lambda_{opt} = 6$. After classifying all target tags into λ_{opt} ranges, we aggregate the target tags that are within the same pair of brother ranges to obtain the tag distribution on the upper layer. Repeat such a merging

operation recursively until we obtain the tag classification on layer 0, *i.e.*, $\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_{\lambda_0}$, which will be returned to the users. Since a time slot is able to convey 96-bit data at most [20], the number of ranges $\lambda_0 \times 2^l$ should be no more than 96. Hence, the number of layers in *PM* should be just $\lfloor log_2(\frac{96}{\lambda_0}) \rfloor$.

C. Cleaning Tags

If the target tags that have been classified already continue to participate in the rest of querying processes, they will interfere with the classification of the other target tags. Hence, if all the target tags that choose the same slot have been classified already, the reader sends an `NACK` command to deactivate them. Moreover, unexpected tags also need to be deactivated for better time-efficiency. We refer to the slots, in which no target tag will respond, as the *expected empty slots*. If some tags responded in such an expected empty slot, they are necessary to be unexpected tags. Then, the reader sends an `NACK` command to deactivate them.

IV. Performance Evaluation

A. Simulation Settings

We simulate an RFID system that contains t target tags and u unexpected tags. The information of each tag is 16-bit [13], which follows the normal distribution $Norm(\mu, \sigma)$ within the ambit $[a, b]$. The number of ranges queried by the users is λ. We use the default settings $t = 5000$, $u = 5000$, $\lambda = 3$, $\mu = 25$, $\sigma = 8$, and $[a, b] = [0, 50]$ unless specified otherwise. We compare the proposed protocol with four related protocols: the *Multi-hash Information Collection protocol (MIC)* [13], the *Tag-Ordering Polling protocol (TOP)* [14], the *Polling protocol*, and the *Enhanced Dynamic Framed Slotted ALOHA (EDFSA) [21]*. The transmission rate between reader and tags is asymmetric. The uplink rate is $53Kb/s$, *i.e.*, it takes $18.8us$ to transmit 1-bit data from a tag to a reader. The downlink rate is $26.5Kb/s$, *i.e.*, it takes $37.7us$ to transmit 1-bit data from a reader to a tag. Between any two consecutive data transmissions, there is a waiting time $\tau_w = 302us$ [16]. Since the multi-reader case has been widely discussed in the existing literature [22], we follow the benchmark literature and simulate a single reader that has sufficient power to probe all tags.

B. Evaluating the Query Accuracy

In this section, we conduct simulations to evaluate the query accuracy of each protocol. Here, the query accuracy is measured by the *corruption ratio*, which is defined as the ratio of the number of target tags whose information are corrupted to the number of all target tags.

1) Impact of the Target Tag Number t: In this set of simulations, we vary the value of t from 2000 to 20000, and thus investigating its impact on the query accuracy of each protocol. We make several observations from the simulation results in Fig. 6(a) and the underlying reasons will be explained below. The *MIC* and *TOP* protocols suffer from the corruption issue, because a slot chosen by a target tag may also be chosen by an unexpected tag. And [13] and [14] did not consider the existence of unexpected tags, and thus no countermeasure was proposed. Moreover, the corruption ratio of *MIC* and *TOP* decreases as the number of target tags increases. The intuitive reason is that, for the fixed number of unexpected tags, the number of target tags whose information could be corrupted is limited accordingly. Hence, as the total number of target tags increases, the corruption ratio tends to decrease. The *Polling* protocol does not suffer from corruption issue, because only the target tags' IDs are queried by the reader and the unexpected tags will not respond at all. The *EDFSA* protocol does not suffer from corruption issue, because it is able to collect the information along with the corresponding tag ID of each tag. Then, we can easily pinpoint the information of the target tags. The proposed *RQ+PM* protocol also does not suffer from corruption issue at all, because it can make use of the slot that contains the replies from unexpected tags.

2) Impact of the Unexpected Tag Number u: In this set of simulations, we vary the value of u from 0 to 10000, thereby investigating its impact on the query accuracy of each protocol. We observe from the simulation results in Fig. 6(b) that *MIC* and *TOP* do not suffer from corruption issue *only when* there is no unexpected tag (*i.e.*, $u = 0$). As the value of u increases, the corruption ratio increases accordingly. The underlying reason is that, more unexpected tags incur more serious corruption issue.

3) Impact of λ, μ, and σ: In this set of simulations, we vary the values of λ, μ, and σ to investigate their impact on each protocol, respectively. The query accuracy of each protocol under different conditions is plotted in Fig. 6(d)(e)(f), respectively. We make two main observations from these three figures. First, *RQ+PM*, *Polling*, and *EDFSA* are consistently immune to the corruption issue. Second, *MIC* and *TOP* suffer from serious corruption issue, and their corruption ratios keep stable with respect to the varying values of λ, μ, or σ. The underlying reason is that, these variables have nothing to do with the ratio of target tags to unexpected tags. Therefore, the corruption ratio maintains stable.

C. Evaluating the Time-efficiency

The simulation results in Fig. 6 reveal that *MIC* and *TOP* cannot correctly collect the information from target tags, due to the interference of unexpected tags. The *Multi-pairing Unknown tag Identification protocol (MUIP)* [23] was proposed to identify the unexpected tags, which could be easily modified to deactivate the unexpected tags. Then, we can perform the *MIC* protocol or the *TOP* protocol to query the target tags in the environment that contains no unexpected tag. Thus, we can correctly collect the information from target tags without any corruption issue. In this section, we evaluate the time-efficiency of each protocol under different conditions, meanwhile ensuring 100% query accuracy.

1) Impact of the Target Tag Number t: In this set of simulations, we vary the value of t from 2000 to 20000, thereby investigating its impact on each protocol's time-efficiency. We observe from the simulation results in Fig. 7(a) that the time cost of all protocols increases as the number of target tags increases, and the proposed *RQ+PM* protocol

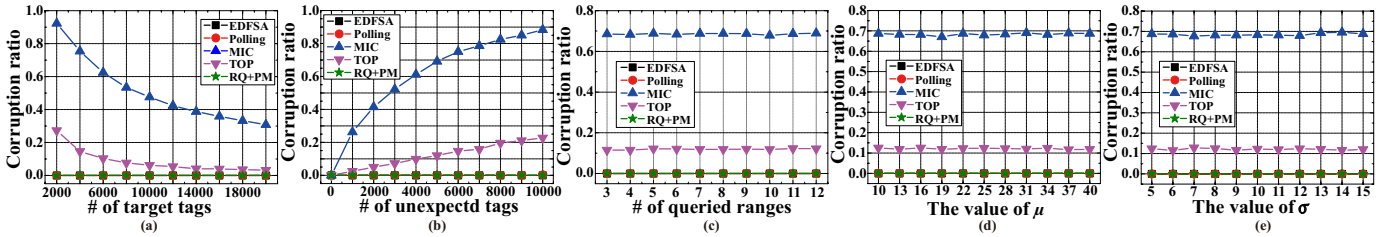

Fig. 6. Investigating the query accuracy of each protocol. (a) corruption ratio *vs.* t; (b) corruption ratio *vs.* u; (c) corruption ratio *vs.* λ; (d) corruption ratio *vs.* μ; (e) corruption ratio *vs.* σ.

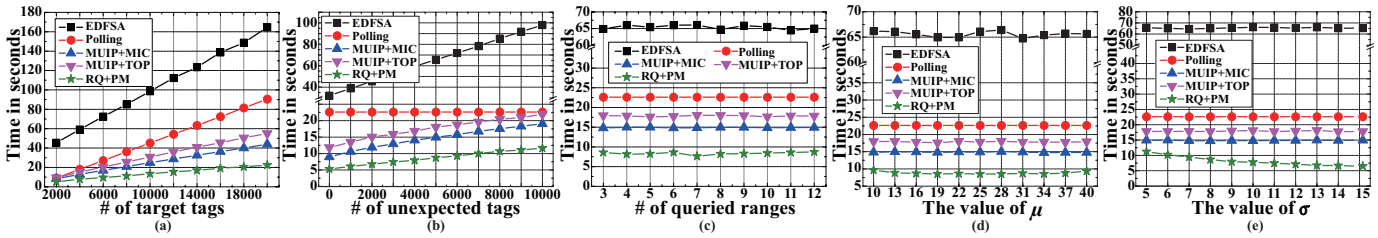

Fig. 7. Investigating the time-efficiency of each protocol. (a) time *vs.* t; (b) time *vs.* u; (c) time *vs.* λ; (d) time *vs.* μ; (e) time *vs.* σ.

consistently performs better than the other protocols. For example, when $t = 20000$, the time cost of *MUIP+MIC*, *MUIP+TOP*, *Polling*, *EDFSA*, is 43.8s, 55.8s, 90.5s, 163.2s, respectively. And that of *RQ+PM* is just 22.7s, *i.e.*, reducing the execution time by 43.8%, 59.3%, 74.9%, 86.1%, than these protocols, respectively.

2) Impact of the Unexpected Tag Number u: In this set of simulations, we vary the value of u from 0 to 10000, thereby investigating its impact of each protocol's time-efficiency. We observe from the simulation results in Fig. 7(b) that the execution time of *Polling* protocol is stable with respect to the number of unexpected tags. The underlying reason is that, the reader just queries the target tags one by one in the *Polling* protocol, and the fixed number of target tags results in the consistent time cost. In contrast, the execution time of the other protocols increases as the number of unexpected tags increases. The intuitive reason is that more interference from unexpected leads to slower query processes. Generally, the proposed *RQ+PM* protocol consistently performs better than the other protocols. For example, when $u = 10000$, the time cost of *MUIP+MIC, MUIP+TOP, Polling, EDFSA*, is 18.8s, 21.8s, 22.6s, 98.2s, respectively. And that of *RQ+PM* is just 11.5s, *i.e.*, reducing the execution time by 38.8%, 47.3%, 49.1%, 88.3%, than these protocols, respectively.

3) Impact of the Queried Range Number λ: In this set of simulations, we vary the value of λ from 3 to 12, thereby investigating its impact on the time-efficiency of each protocol. We observe from the simulation results in Fig. 7(c) that the time cost of all protocols is stable with respect to the number of queried ranges. The execution time of *RQ+PM* is stable against λ, because the proposed *Partition&Mergence (PM)* approach can always dynamically optimize the number of queried ranges. The execution time of the other protocols is stable, because they need to collect the *detailed* tag information, and the corresponding execution time has nothing to do with the queried ranges. It is easy to find that *RQ+PM* consistently performs better than the other protocols with varying number of queried ranges. For example, when $\lambda = 12$, the time cost of *MUIP+MIC, MUIP+TOP, Polling, EDFSA*, is 14.9s, 17.8s, 22.6s, 65.0s, respectively. And that of *RQ+PM* is just 8.7s, *i.e.*, reducing the execution time by 41.6%, 51.1%, 61.5%, 86.6%, than these protocols, respectively.

4) Impact of μ and σ: In this set of simulations, we vary the values of μ and σ to investigate their impact on the time-efficiency of each protocol. In the simulations corresponding to Fig. 7(d), the value of σ is fixed to 8, and the value of μ varies from 10 to 40. We observe that the execution time of *RQ+PM* slightly decreases first and then increases with respect to the value of μ, and achieves the minimum value when μ is near to 25. The intuitive reason is that, if the value of μ deviates from 25 significantly, it will exacerbate the imbalance of information distribution, and further increase the value of M. We have proved that the upper bound on the execution time of our protocol is an increasing function of M. Hence, the larger M is, the larger the time cost of *RQ+PM* will be. That is, if μ deviates from 25, the execution time of *RQ+PM* will increase. In the simulations corresponding to Fig. 7(e), the value of μ is fixed to 25, and the value of σ varies from 5 to 15. We observe that the execution time of *RQ+PM* decreases as the value of σ increases. The underlying reason is that, the larger the value of σ is, the more balanced the information distribution will be. This leads to a smaller value of M, and will decrease the execution time of *RQ+PM*.

V. Related Work

Tag information collection does not only simply gather the information from a given set of tags, but also need to exactly tell which tag corresponds the a collected information. In other words, a solution to the problem of information collection needs to tell us what is the information of a specific RFID tag, which is different from ID identification. We can borrow the traditional tag ID identification protocols [21], [24] to address this problem. Specifically, we can let each tag

reply not only its ID but also the stored information. After successfully identifying all tag IDs, we can also know the information corresponding to each ID. However, such a simple solution is not efficient due to two reasons: (i) a common assumption for the problem of information collection is that we know the IDs of the concerned tags apriori. Recollecting massive IDs is a redundant overhead; (ii) the frame utilization is quite low ($\leq 36.8\%$) when directly borrowing the tag identification protocols. Hence, the problem of information collection requires dedicated effort. Chen *et al.* [13] proposed a multi-hashing method called *Multi-hash Information Collection (MIC)* protocol, which makes use of the known tag IDs and the hash functions embedded in RFID tags to improve the frame utilization and avoid the transmission of tag IDs. Qiao *et al.* investigated how to efficiently collect information from a *subset* of given tags [14]. Hao *et al.* focused on the multi-reader RFID systems, and proposed to use the bloom filter to filter out the tag IDs that may reside in other reader's coverage [11], [25]. The above information collection protocols make a common assumption that all tag IDs are known in advance, which, however, is not true for some practical scenarios [15]. If there are unexpected tags, they will suffer from information corruption issue. Moreover, the slot containing two or more tag replies is treated as a useless slot in these protocols. Hence, the slot utilization needs to be further improved.

VI. Conclusion

This is the first piece of work that studies the practically important problem of *range query* for sensor-augmented RFID systems. We mainly make the following three contributions. First, we propose the *Range Query (RQ)* protocol to completely classify the target tags into multiple ranges. Second, we propose a *Partition&Mergence (PM)* approach, which uses two steps of top-down partitioning and bottom-up merging to find the optimal number of queried ranges for *RQ*. Third, we propose sufficient theoretical analyses to prove the upper bound on the execution time of the proposed protocols, and then optimize the involved parameters. Experiments using the USRP and WISP tags demonstrate the feasibility of the proposed *RQ+PM* protocol. Moreover, extensive simulation results reveal that *RQ+PM* is able to achieve 100% query accuracy, and also significantly reduces the execution time by about 40% than the existing protocols.

Acknowledgment

This work is supported by the National Key Research and Development Program of China No. 2016YFB1000205; the State Key Program of National Natural Science of China under Grants 61432002, 61332004; the funding for Project of Strategic Importance provided by The Hong Kong Polytechnic University (Project Code: 1-ZE24); the NSFC under Grants 61672379, 61772112, 61370199, 61702257, 61772251, and 61702365; the Dalian High-level Talent Innovation Program under Grant 2015R049; the Natural Science Foundation of Tianjin under Grant 17JCQNJC00700; Natural Science Foundation of Jiangsu Province (No.BK20170648).

References

[1] J. Liu, F. Zhu, Y. Wang, X. Wang, Q. Pan, and L. Chen, "RF-Scanner: Shelf Scanning with Robot-assisted RFID Systems," *Proc. of IEEE INFOCOM*, 2017.

[2] 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.

[3] Y. Bu, L. Xie, J. Liu, B. He, Y. Gong, and S. Lu, "3-Dimensional Reconstruction on Tagged Packages via RFID Systems," *Proc. of IEEE SECON*, 2017.

[4] X. Liu, B. Xiao, K. Li, A. X. Liu, J. Wu, X. Xie, and H. Qi, "RFID Estimation with Blocker Tags," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 224–237, 2017.

[5] 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," *Proc. of ACM Ubicomp*, 2016.

[6] X. Liu, K. Li, S. Guo, A. X. Liu, P. Li, K. Wang, and J. Wu, "Top-k Queries for Categorized RFID Systems," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 2587–2600, 2017.

[7] S. Zhang, X. Liu, J. Wang, and J. Cao, "Tag size profiling in multiple reader RFID systems," *Proc. of IEEE INFOCOM*, 2017.

[8] L. Yang, Q. Lin, X.-Y. Li, T. Liu, and Y. Liu, "See Through Walls with COTS RFID System," *Proc. of ACM MobiCom*, 2015.

[9] J. Liu, M. Chen, S. Chen, Q. Pan, and L. Chen, "Tag-Compass: Determining the Spatial Direction of an Object with Small Dimensions," *Proc. of IEEE INFOCOM*, 2017.

[10] L. Shangguan, Z. Zhou, X. Zheng, L. Yang, Y. Liu, and J. Han, "ShopMiner: Mining Customer Shopping Behavior in Physical Clothing Stores with Passive RFIDs," *Proc. of ACM SenSys*, 2015.

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

[12] "http://sensor.cs.washington.edu/WISP.html."

[13] S. Chen, M. Zhang, and B. Xiao, "Efficient Information Collection Protocols for Sensor-augmented RFID Networks," *Proc. of IEEE INFOCOM*, 2011.

[14] Y. Qiao, S. Chen, T. Li, and S. Chen, "Tag-Ordering Polling Protocols in RFID Systems," *IEEE/ACM Transactions on Networking*, vol. 24, no. 3, pp. 1548–1561, 2016.

[15] S. Muhammad and A. X. Liu, "Expecting the Unexpected: Fast and Reliable Detection of Missing RFID Tags in the Wild," *Proc. of IEEE INFOCOM*, 2015.

[16] M. Shahzad and A. X. Liu, "Fast and Accurate Estimation of RFID Tags," *IEEE/ACM Transactions on Networking*, vol. 23, no. 1, pp. 241–254, 2015.

[17] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk, "Efficient and Reliable Low-power Backscatter Networks," *Proc. of ACM SIGCOMM*, 2012.

[18] C. Qian, H. Ngan, Y. Liu, and L. M. Ni, "Cardinality Estimation for Large-scale RFID Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 9, pp. 1441–1454, 2011.

[19] X. Liu, K. Li, A. X. Liu, S. Guo, M. Shahzad, A. L. Wang, and J. Wu, "Multi-category RFID Estimation," *IEEE/ACM Transactions on Networking*, vol. 25, no. 1, pp. 264–277, 2017.

[20] T. Li, S. Chen, and Y. Ling, "Identifying the Missing Tags in a Large RFID System," *Proc. of ACM MobiHoc*, 2010.

[21] S. Lee, S. Joo, and C. Lee, "An Enhanced Dynamic Framed Slotted ALOHA Algorithm for RFID Tag Identification," *Proc. of IEEE MobiQuitous*, 2005.

[22] L. Yang, J. Han, Y. Qi, C. Wang, T. Gux, and Y. Liu, "Season: Shelving Interference and Joint Identification in Large-Scale RFID Systems," *Proc. of IEEE INFOCOM*, 2011.

[23] X. Liu, B. Xiao, S. Zhang, and K. Bu, "Unknown Tag Identification in Large RFID Systems: An Efficient and Complete Solution," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 6, pp. 1775–1788, 2015.

[24] M. Shahzad and A. X. Liu, "Probabilistic Optimal Tree Hopping for RFID Identification," *IEEE/ACM Transactions on Networking*, vol. 23, no. 3, pp. 796–809, 2015.

[25] H. Yue, C. Zhang, M. Pan, Y. Fang, and S. Chen, "Unknown-Target Information Collection in Sensor-Enabled RFID Systems," *IEEE/ACM Transactions on Networking*, vol. 22, no. 4, pp. 1164–1175, 2014.