The following publication X. Liu, B. Xiao, S. Zhang, K. Bu and A. Chan, "STEP: A Time-Efficient Tag Searching Protocol in Large RFID Systems," in IEEE Transactions on Computers, vol. 64, no. 11, pp. 3265-3277, 1 Nov. 2015 is available at https://doi.org/10.1109/TC.2015.2394461.

STEP: A Time-Efficient Tag Searching Protocol in Large RFID Systems

Xuan Liu, Bin Xiao, *Senior Member, IEEE,* Shigeng Zhang, *Member, IEEE,* Kai Bu, *Member, IEEE,* and Alvin Chan, *Member, IEEE,*

Abstract—The Radio Frequency IDentification (RFID) technology is greatly revolutionizing applications such as warehouse management and inventory control in retail industry. In large RFID systems, an important and practical issue is tag searching: Given a particular set of tags called wanted tags, tag searching aims to determine which of them are currently present in the system and which are not. As an RFID system usually contains a large number of tags, the intuitive solution that collects IDs of all the tags in the system and compares them with the wanted tag IDs to obtain the result is highly time inefficient. In this paper, we design a novel technique called *testing slot*, with which a reader can quickly figure out which wanted tags are absent from its interrogation region without tag ID transmissions. The testing slot technique thus greatly reduces transmission overhead during the searching process. Based on this technique, we propose two protocols to perform time-efficient tag searching in practical large RFID systems containing multiple readers. In our protocols, each reader first employs the testing slot technique to obtain its local searching result by iteratively eliminating wanted tags that are absent from its interrogation region. The local searching results of readers are then combined to form the final searching result. The proposed protocols outperform existing solutions in both time efficiency and searching precision. Simulation results show that, compared with the state-of-the-art solution, our best protocol reduces execution time by up to 60 percent, meanwhile promotes the searching precision by nearly an order of magnitude.

Index Terms—RFID system; tag searching; time-efficiency; multiple reader; testing slot

1 Introduction

The Radio Frequency IDentification (RFID) technologies are greatly revolutionizing applications in retail industry such as warehouse management and inventory control [1]–[3]. Many modern warehouses and supermarkets install RFID systems to efficiently manage their products. Since the interrogation region of a single reader is usually very limited, large RFID systems need to deploy multiple readers to cooperatively cover all the tags in the system. These readers identify tags by collecting their IDs, which enables the warehouse to make inventory of the products automatically.

Instead of making inventory of all the tags in the system, some applications require searching a particular set of tags (referred to as *wanted tags* hereinafter) with the given tag IDs to confirm which of them are currently present in the system [4]–[6]. Imagine a big warehouse that stores many kinds of products from different manufacturers. Every manufacturer wants to make inventory of only its own products. In this case, a time efficient tag searching protocol will be of great help. There are many other practical applications of tag searching, such as frequent inventory of a specified kind of products in a shopping mall and finding which products contain flaws in order to recall and fix them.

In this paper, we study the practically important *tag searching problem* in large RFID systems that contain multiple readers. We call the wanted tags that are currently present in the system and thus should be included in the searching result as *target tags*, and call the other wanted tags as *nontarget tags*. It may appear that we can collect the IDs of all the tags in the system and find the result by comparing the collected tag IDs with the wanted tag IDs. Such a tag collection solution, however, may incur too long time because it wastes time in collecting IDs of non-wanted tags (i.e., tags not in the wanted tag set), which usually form the majority of the tag population in the system. Another intuitive solution is to broadcast wanted tag IDs and let tags respond to the reader only when they hear their own IDs. This method's performance heavily depends on the number of wanted tags. Moreover, when there are multiple readers, every reader has to broadcast all the wanted tag IDs. When there are many more wanted tags than tags covered by one reader (every reader covers only a small part of tags in the system), this solution may perform even worse than the collection approach.

There were some studies on efficient tag searching in recent years, e.g., [4]–[6]. In [4] the authors proposed the *Compact Approximator based Tag Searching* (CATS) protocol to solve the tag searching problem. CATS employs Bloom filters to compact the information exchanged between the reader and tags, which avoids time-consuming tag ID transmissions and thus reduces searching time.

- *X. Liu, B. Xiao, and A. Chan are with the Department of Computing, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong. E-mail: {csxuanliu, csbxiao, cstschan}@comp.polyu.edu.hk.*
- *S. Zhang is with the School of Information Science and Engineering, Central South University, Changsha, China. E-mail: sgzhang@csu.edu.cn.*
- *K. Bu is with College of Computer Science and Technology, Zhejiang University, China. E-mail: kaibu@zju.edu.cn.*

However, CATS's performance greatly degrades when the wanted tag number increases. When the number of wanted tags is greatly larger than the number of tags in the reader's interrogation region, CATS may even not work [5]. Instead of using a single large Bloom filter, the *Iterative Tag Searching Protocol* (ITSP) proposed in [5] uses a series of short *filtering vectors* to filter out nontarget tags. As nontarget tags are gradually filtered out, the size of the filtering vector decreases, which contributes to the high time efficiency of ITSP. ITSP filters out approximately half nontarget tags in each round, which is not optimal in many cases. In [6] the authors proposed a tag searching protocol for multiple reader RFID systems. However, the protocol proposed in [6] utilizes only replies from tags to filter out nontarget tags, and its performance degrades greatly when the number of tags in the reader's interrogation region is large.

It is very challenging to quickly search a group of wanted tags in an RFID system containing multiple readers. In fact, as tags may enter or leave the system frequently, it is difficult for a reader to know exactly which tags are in its interrogation region. Thus a reader does not know whether there are some wanted tags in its interrogation region and which they are. Each reader needs to quickly distinguish wanted tags from non-wanted tags. As we have mentioned, collecting all the tag IDs is highly time inefficient. We need to design new techniques with which a reader can quickly differentiate between wanted tags and non-wanted tags without time-consuming tag ID transmissions.

In this paper, we design a novel technique called *testing slot* that can quickly test the presence of wanted tags without tag ID transmissions. Based on this technique, we propose a *Searching by iterative Testing and Eliminating Protocol* (STEP), which can quickly search tags in multiple reader systems with high time efficiency. In STEP, every reader obtains its local searching result by employing the testing slot technique to quickly eliminate nontarget tags from the wanted tag set. The local searching results are then combined to form the final searching result. We find that the time efficiency of STEP degrades when the ratio of the nontarget tags to the non-wanted tags in the reader's region decreases. We then propose a novel approach to dynamically adjusting the ratio of the two types of tags, based on which we develop the E-STEP protocol. The proposed protocols outperform existing solutions in both time efficiency and searching precision. Simulation results show that, compared with the state-of-the-art ITSP solution, E-STEP reduces execution time by up to 60 percent and promotes the searching precision by nearly an order of magnitude.

The rest of this paper is organized as follows. In Section 2, we review related work. In Section 3, we describe the system model and define the problem. In Section 4, we describe STEP and analyze its execution time. In Section 5, we propose E-STEP that further improves time efficiency of STEP. Simulation results are reported in Section 6. At last, Section 8 concludes the paper.

2 Related Work

2.1 Tag Identification

The tag searching problem can be naturally solved with *tag identification* protocols, but their time efficiency might be low because they waste time in collecting IDs of nontarget tags. Existing tag identification protocols fall into two categories [7]: Tree-based protocols [8]–[10] and Aloha-based protocols [1], [11]–[18].

In tree-based tag identification protocols, the reader iteratively splits the replying tags into two subsets until at least one of them contains only one tag, in which case the tag can be identified. The *adaptive query splitting* (AQS) protocol and the *adaptive binary splitting* (ABS) protocol proposed in [8] leverage the information collected in previous identification processes to reduce collisions between tags. The *tree hopping* (TH) protocol proposed in [9] estimates the optimal query level before starting a query according to the number of tags in the system, which significantly improves tag identification throughput. The QQTT protocol proposed [10] detects the first colliding bit in the received reply, according to which it smartly separates colliding tags to promote identification efficiency.

In Aloha-based identification protocols, tags are identified in a series of *frames*. The main objective is to tune the frame size to optimize the time efficiency [19], [20] or energy efficiency [11]. The DDC protocol [12] and the FACT protocol [13] further enhance identification throughput by extracting tag IDs from signals received in collision slots. DDC uses a novel random number pattern to extract tag IDs from colliding signals, while FACT utilizes analog network coding to do this. In [16], [17], the authors proposed a set of protocols to enhance identification efficiency by collecting IDs of only new tags. The LOCK mechanism proposed in [21] could be used to perform spot scanning in large RFID systems.

2.2 Tag Searching

The tag searching problem was first defined and studied in [4], where the authors proposed the CATS protocol to solve it. CATS uses Bloom filters to compact the information exchanged between the reader and tags, which avoids tag ID transmissions and hence reduces the searching time. CATS consists of two phases. In the first phase, the reader constructs a Bloom filter that represents all the wanted tag IDs and broadcasts it to tags. Upon receiving the filter, tags in the system check whether they belong to the filter and determine whether to participate in the second phase. In the second phase, the reader collects replies from tags and constructs the second Bloom filter, with which it filters out nontarget tags from the wanted tag set. CATS performs well when the number of wanted tags is significantly less than the number of tags in the reader's interrogation region. However, when the number of wanted tags is large, the performance of CATS degrades dramatically. In practical multiple reader RFID systems, CATS might perform

poorly as the tags covered by a single reader are usually far less than the wanted tags.

The ITSP protocol proposed in [5] improves CATS from two aspects. First, it enhances the time efficiency by using a series of short *filtering vectors* instead of a long Bloom filter. Different from CATS, ITSP uses multiple rounds to iteratively filter out nontarget tags. In each round, the reader broadcasts a filtering vector to filter out nontarget tags in the reader's interrogation region. It also uses replies from tags to filter out nontarget tags from the wanted tag set. As nontarget tags are gradually filtered out, the size of the filter vector decreases, and thus ITSP achieves higher time efficiency than CATS. Second, ITSP can work well even when there are far more wanted tags than tags in the reader's region. However, there are two factors that make the performance of ITSP far from optimized. First, ITSP tries to filter out about a half of nontarget tags in each round, which is not optimal in many cases. Our protocols optimize the length of the filter to minimize the average time in filtering out nontarget tags. Second, ITSP prefers filter vector to replies from tags when filtering nontarget tags, which incurs high transmission overhead when the wanted tag set is large and most of tags in the reader's interrogation region are target tags. In contrast, our protocols use the testing slot to filter out nontarget tags, which can effectively reduce transmission overhead in such cases.

In [6], the authors studied the tag searching problem from the energy efficiency aspect. The protocols proposed in [6] use only replies from tags to filter out nontarget tags, which limits its time efficiency. When most of wanted tags are target tags, the performance of the protocols proposed in [6] is poor. The authors also considered joint optimization of tag searching and reader scheduling in [6].

2.3 Multiple Reader Scheduling

How to schedule readers to work in parallel is important for multiple reader systems. A distributed reader scheduling algorithm based on graph coloring was proposed in [22]. In [23], the authors formalized the reader scheduling problem as a maximum independent set problem, and proposed approximation algorithms based on spatial time division multiple access (STDMA) to solve it. They considered both single channel cases and multiple channel cases. The RASPberry algorithm [24] schedules readers in a way that makes the system work stably in a long term when the arrival rates of tags are in the capacity region of the readers. In [25], the authors studied a general case of reader scheduling and proposed a scheduling algorithm to maximize the number of tags identified per time slot while avoiding interferences among readers. The joint optimization of reader scheduling and tag identification was considered in [26]. They first shelve the interference among readers and schedule all the readers to work, which could identify most tags in the collision-free areas, and then identify other tags by scheduling only a part of readers to work. Recently, in [15] the authors proposed the parallel identification protocol (PIP) that encodes tag IDs into a special pattern and recovers them from colliding signals to improve tag identification throughput.

For simplicity in implementation, in this paper we use a simple graph coloring based algorithm [27] to find a feasible reader scheduling. However, all the aforementioned reader scheduling algorithms could be used in our tag searching protocols.

TABLE 1
Notations used in this paper.

Notation	Meaning
R/T	Set of readers/tags in the system
$r_i/S(r_i)$	The i-th reader/The local searching result of r_i
X	The set of wanted tags
L_i	The set of local tags of reader r_i
Y_l	The set of candidate target tags in the l-th round
Z_l	The set of nontarget tags in the l-th round
N_l	The number of active local tags in the l-th round
C_l	Time to eliminate a nontarget tag in the l-th round
N_δ	The number of false positive tags
Δ	The threshold of N_δ
p_{req}	The probability of $N_\delta \leq \Delta$

3 Problem Description

Table 1 summaries the main notations used in this paper.

3.1 System Model

We consider an RFID system that consists of three components: a back-end server, a set of readers $R = \{r_1, r_2, \ldots, r_M\}$, and a set of tags $T = \{t_1, t_2, \ldots, t_N\}$. Here M and N are the total numbers of readers and of tags in the system, respectively. The server connects the readers via wired or wireless links, and issues orders to schedule their working. Every reader covers a part of tags that are referred to as its *local tags*. The local tag set of reader r_i is denoted as L_i, which is a subset of T. Every tag is covered by at least one reader, i.e., it should satisfy that $T = \bigcup_{i=1}^{M} L_i$.

The system uses the frame-slotted Aloha protocol to transmit data between the reader and tags. In this protocol, a reader first broadcasts a query $<f, s>$ (f is the frame size and s is a random seed), and receives responses from its local tags in a frame that is divided into f slots. Upon receiving the query, a tag randomly selects a slot in the frame with a hash function $H(ID, s)\ mod\ f$ and transmits its response in that slot. According to the number of responses, a slot can be in one of three different states: *empty* (when no tag responds), *singleton* (when only one tag responds) or *collision* (when more than one tags respond). The reader receives responses successfully in only singleton slots. When receiving a response successfully, the reader replies an *ACK* to acknowledge that tag and prevents it from attending the following processes until the next protocol execution. If the reader fails to receive the response, it will reply a

NAK to keep the tags active. At the end of the frame, the reader will issue another query and start a new frame if it detects any active tags. The reader repeats this process until all the tags are acknowledged. In the rest of the paper, when we mention the local tags of reader r_i, we mean *the active tags in r_i's interrogation region that have not been acknowledged*.

In tag identification protocols, the tag needs to transmit its ID to the reader. Our protocols only need to distinguish between empty slots and non-empty slots, which could be achieved by transmitting a one-bit short response. We denote the time duration of an empty slot as t_e, the time to transmit a tag ID as t_{id}, and the time to transmit a short response as t_s. The relationship among them are $t_e < t_s \ll t_{id}$.

3.2 Problem Statement

Given a group of wanted tags $X = \{w_1, w_2, \ldots, w_H\}$, the tag searching problem is to *quickly determine which tags in X are currently present in the system, i.e., find $X \cap T$*. We call tags in $X \cap T$ as *target tags*, and call tags in $X - T$ and in $T - X$ as *nontarget tags* and *non-wanted tags*, respectively. The problem has to be cooperatively solved by multiple readers. The server sends the wanted tag set X to all the readers. Each reader individually finds out target tags in its local region, i.e., $X \cap L_i$. By combining the results of all the readers, the server can get the searching result as $X \cap T = \bigcup_{i=1}^{M}(X \cap L_i)$. Our goal is to *find the target tags for an individual reader r_i, i.e., $X \cap L_i$, as fast as possible*.

Our protocols are probabilistic, and thus the searching result may contain some false positive tags. The objective is to bound the number of false positive tags in r_i's local searching result within a small constant value with high probability. Denoting the local searching result of r_i as $S(r_i)$, the number of false positive tags in $S(r_i)$ is $N_\delta = |S(r_i) - (X \cap L_i)|$. We want to guarantee that

$$Pr\{N_\delta \leq \Delta\} \geq p_{req}, \tag{1}$$

where Δ is a small constant and p_{req} is a predefined probability threshold. For example, if $\Delta = 1$ and $p_{req} = 0.95$, then our protocols can guarantee that there at most one false positive tag in the searching result with a probability higher than or equal to 0.95.

4 STEP: Searching by Iterative Testing and Eliminating Protocol

4.1 Design Guideline

To reduce the transmission time, our protocol design follows a main guideline: Avoid tag ID transmissions during the searching process in which the reader confirms the presence or absence of wanted tags. To achieve this goal, we design a novel technique called *testing slot* with which a reader can quickly test the presence or absence of wanted tags without transmitting tag IDs.

We call a slot as testing slot if *at least* one wanted tag maps to it. In Aloha-based protocols, with the same hash function H, the reader could predict which slot a tag will select if it knows the tag's ID. Thus the reader can compute which slots are testing slots according to the wanted tag set X. We leverage the testing slots to determine which wanted tags *are not present in the system*: When *no response* is received in a testing slot, the corresponding wanted tags *must be absent* from the reader's region. Such tags are nontarget tags and should be eliminated from the wanted tag set.

During the searching process, the reader cares only the state of the slot, i.e., empty or non-empty. We let tags transmit one-bit short responses [5] to notice the reader of their presence. This can effectively avoid tag ID transmissions and reduce the searching time.

4.2 Protocol Overview

To find the target tags, every reader first executes STEP to obtain its local searching result $S(r_i)$. The local searching result may contain false positive tags ($S(r_i) \supseteq (X \cap L_i)$). The back-end server combines all the local searching results to obtain the final searching result as $S = \bigcup_{i=1}^{M} S(r_i)$.

STEP consists of multiple rounds. In each round, the reader uses testing slots to find nontarget tags and eliminates them from the wanted tag set. In the l-th round, the reader maintains a candidate target tag set Y_l that contains all the possible target tags in its interrogation region. ($Y_1 = X$ in the first round.) It computes the testing slots with tags in Y_l. After checking all the testing slots, the reader obtains the updated candidate target tag set Y_{l+1}. It then checks whether the *termination condition* is met. If the termination condition is not met, the reader issues a new round to further eliminate nontarget tags from Y_{l+1}. Otherwise, it treats Y_{l+1} as the local searching result, i.e., $S(r_i) = Y_{l+1}$, and sends $S(r_i)$ to the back-end server to form the final searching result. The termination condition is given in Section 4.5

Non-wanted tags in the reader's local region may interfere with the elimination of nontarget tags from Y_l, because they may select testing slots. Note that a testing slot can help eliminate nontarget tags only when no response is received in it. If some non-wanted tags select testing slots, their responses will disturb the elimination of nontarget tags from Y_l. To solve this problem, we use *non-testing slots* to acknowledge non-wanted tags and mitigate their interference. Non-testing slots are slots that are not selected by any wanted tags. As all the target tags respond in only testing slots, the responses received in non-testing slots must be from non-wanted tags. When the reader receives responses in a non-testing slot, it replies an ACK to acknowledge these non-wanted tags and prevent them from attending the next round.

4.3 Protocol Description

STEP consists of multiple rounds. In the l-th round, the reader first computes testing slots and non-testing slots according to the candidate target tag set Y_l, and then

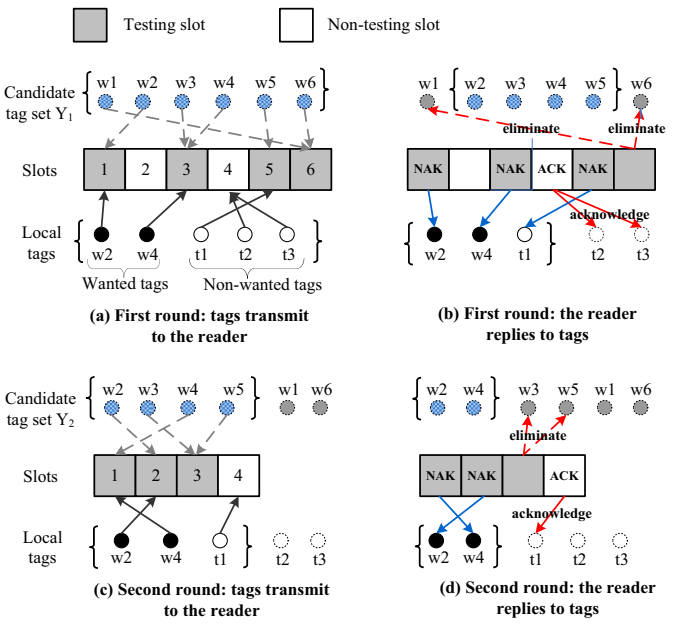


Fig. 1. An example of iteratively testing and eliminating nontarget tags from Y_l.

eliminates nontarget tags from Y_l and acknowledges non-wanted tags in the local region. It repeats the process until the termination condition is met.

At the beginning of the l-th round, the reader broadcasts a query $< f_l, s >$ and computes the testing slots according to Y_l. After receiving the query, local tags select their slots with the same hash function used for testing slots computation, and then transmit short responses to the reader in the selected slots. The reader eliminates nontarget tags from Y_l as follows:

- In a testing slot, if the reader receives some responses, it replies a *NAK* to keep the tag(s) active. If the reader receives no response, it eliminates wanted tags that select this slot from Y_l.
- In a non-testing slot, if the reader receives some responses, it replies an *ACK* to acknowledge these tag(s) to prevent them from participating in the following rounds.

At the end of the round, the reader obtains the updated candidate target tag set Y_{l+1} and checks whether it should terminate. If it does not satisfy the termination condition, it will start a new round and repeat the testing and eliminating process.

Fig.1 illustrates how STEP works. In this example, dotted arrows represent the mapping between candidate target tags and testing slots, and solid arrows represent the transmissions between local tags and the reader. The initial candidate target tag set is $Y_1 = \{w_1, w_2, ..., w_6\}$. There are five local tags, among which w_2 and w_4 are target tags. Fig.1(a) and Fig.1(b) illustrate the first round execution of STEP. The reader first computes testing slots for tags in Y_1 and collects responses from local tags (Fig.1(a)), then eliminates nontarget tags from Y_1 and acknowledges non-wanted tags in the local region (Fig.1(b)). The reader replies *NAK* in testing slots 1, 3, and 5 in which it receives responses. The testing slot 6 is empty, so the reader eliminates the corresponding nontarget tags w_1 and w_6 from Y_1. Meanwhile, it replies *ACK* to acknowledge non-wanted tags t_2 and t_3 in slot 4. Fig.1(c) and Fig.1(d) illustrate the second round execution. The reader re-computes the testing slots with the updated candidate target tag set $Y_2 = \{w_2, w_3, w_4, w_5\}$ and local tags also reselect their slots. Similar as in the first round, the reader eliminates nontarget tags w_3 and w_5 from Y_2 that map to testing slot 3, and acknowledges non-wanted local tag t_1 mapping to the non-testing slot 4. If the protocol terminates after the second round, the local searching result would be $S(r_i)$=$\{w_2, w_4\}$.

4.4 Optimal Frame Size Setting

In this section, we analyze how to set the frame size to minimize the average time cost C_l to eliminate a nontarget tag from Y_l.

Without loss of generality, we consider the l-th round. Let N_l[1] be the number of active local tags of the reader r_i, and let f_l be the frame size. For a slot in the frame, the probability that it is empty equals the probability that none of local tags select that slot, which is

$$p_e = (1 - \frac{1}{f_l})^{N_l} \approx e^{-N_l/f_l}. \quad (2)$$

Obviously, the probability that a slot is non-empty is $(1 - p_e)$. The total duration time of the whole frame is

$$\begin{aligned} T_{total} &= f_l * p_e * t_e + f_l * (1 - p_e) * t_s \\ &= f_l * t_s + f_l * e^{-N_l/f_l} * (t_e - t_s). \end{aligned} \quad (3)$$

Note that t_e is the duration time of an empty slot, and t_s is the duration time of a short response slot.

Let Z_l be the set of all the nontarget tags in the l-th round, i.e., $Z_l = Y_l - (X \cap L_i)$, where $X \cap L_i$ is the target tag set of reader r_i. So the total number of tags that can be eliminated from Y_l is $|Z_l|$. A tag in Z_l will be eliminated if its selected testing slot is actually empty. Because wanted tags select slots uniformly, the expected number of eliminated nontarget tags in this round is

$$N_{eli} \approx |Z_l| * p_e. \quad (4)$$

The average time cost to eliminate one nontarget tag from Y_l is

$$C_l = \frac{T_{total}}{N_{eli}} = \frac{t_s}{|Z_l|} * f_l * e^{N_l/f_l} + \frac{1}{|Z_l|} * f_l * (t_e - t_s). \quad (5)$$

To minimize C_l, we let

$$\frac{\partial C_l}{\partial f_l} = 0, \quad (6)$$

1. In the first round, $N_1 = N_i$ can be estimated by using estimation algorithms like [28], [29]. In the following rounds, the reader can leverage information collected in the l-th round to estimate the number of acknowledged non-wanted tags and update N_l accordingly, as to be discussed in Section 4.6.

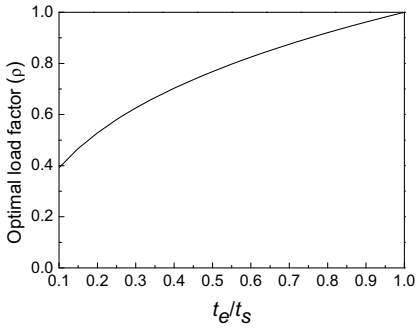


Fig. 2. Optimal load factor (ρ) when t_e/t_s varies.

and know that C_l takes its minimum value when

$$e^{N_l/f_l}(1-\frac{N_l}{f_l})-(1-\frac{t_e}{t_s})=0. \quad (7)$$

Define the load factor as $\rho = N_l/f_l$. Then C_l takes the minimum value when the following equation holds:

$$e^{\rho}(1-\rho)-(1-\frac{t_e}{t_s})=0. \quad (8)$$

We can see that the value of the optimal ρ is solely determined by t_e/t_s. In our system setting (see Section 6), $t_e = 0.184ms$ and $t_s = 0.2ms$. In this case, the optimal value of ρ is 0.9697, and the optimal frame size is $f_l = N_l/0.9697 = 1.03 * N_l$.

As the data transmission rate between the reader and tags varies in a range, t_e and t_s may vary in different applications. Fig. 2 plots the optimal ρ when t_e/t_s varies from 0.1 to 1.

4.5 Termination Condition

We now analyze how to set the termination condition for STEP to achieve the required precision requirement. Note that $(X \cap L_i) \subseteq S(r_i)$. The number of false positive tags is $N_\delta = |S(r_i)| - |X \cap L_i|$. Fig 3 illustrates the relationship among X, L_i, and $S(r_i)$. Our goal is to guarantee that N_δ does not exceed a threshold Δ with probability at least p_{req}, i.e.,

$$P\{N_\delta \le \Delta\} \ge p_{req}. \quad (9)$$

For example, if $\Delta = 2$ and $p_{req} = 0.95$, satisfying Eq. 9 means that there are at most two false positive tags in the local searching result with probability higher than or equal to 0.95.

In STEP, we iteratively eliminate nontarget tags from the candidate target tag set Y_l round by round. If in a certain round (e.g., the l-th round) no nontarget tag is eliminated from Y_l and no non-wanted tag is acknowledged, we can speculate that the number of false positive tags is small. To guarantee that the number of false positive tags is no larger than Δ with a high probability (i.e., satisfying Eq. 9), we need to observe k consecutive rounds. If there are no nontarget tags eliminated in all the k rounds, then the reader terminates the protocol. Otherwise, the reader continues the searching procedure.

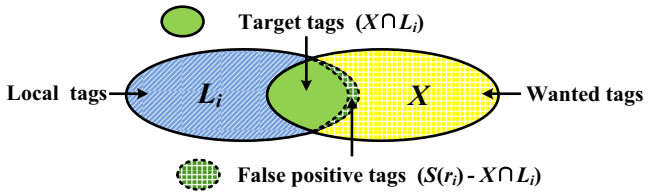


Fig. 3. The relationship among X, L_i, and the searching result $S(r_i)$. $S(r_i)$ is the union of target tags $(X \cap L_i)$ and false positive tags $(S(r_i) - X \cap L_i)$.

We now analyze how to set k to satisfy Eq. 9 for given Δ and p_{req}. Denote by E_k the event that there are no nontarget tags eliminated in all the k consecutive rounds, and denote by E_v the event that there are exactly v false positive tags in the result. Then we have

$$P\{N_\delta \le \Delta | E_k\} = \sum_{v=0}^{\Delta} P\{E_v|E_k\}. \quad (10)$$

According to Bayes' Theorem, the probability $P\{E_v|E_k\}$ can be calculated as

$$P\{E_v|E_k\} = \frac{P\{E_k|E_v\} * P\{E_v\}}{P\{E_k\}}. \quad (11)$$

From the law of total probability it follows that

$$P\{E_k\} = \sum_{v=0}^{\infty} P\{E_k|E_v\}P\{E_v\}. \quad (12)$$

Substituting Eq. 11, 12 into Eq. 10, we have

$$P\{N_\delta \le \Delta|E_k\} = 1 - \frac{\sum_{v=\Delta+1}^{\infty} P\{E_k|E_v\}P\{E_v\}}{\sum_{v=0}^{\infty} P\{E_k|E_v\}P\{E_v\}}. \quad (13)$$

$P\{E_k|E_v\}$ is the probability that there are exactly v false positive tags and none of them are eliminated in all the k consecutive rounds. A nontarget tag cannot be eliminated from the candidate target tag set when it maps to a non-empty testing slot. Thus $P\{E_k|E_v\}$ equals the probability that all the v false positive tags choose non-empty testing slots in all the k rounds, which is

$$P\{E_k|E_v\} = ((1-p_e)^v)^k = (1-p_e)^{vk}, \quad (14)$$

where $(1-p_e)$ is the probability that a testing slot is non-empty.

Because we have no knowledge of the distribution of v, we can assume that v follows the uniform distribution[2], i.e.,

$$P\{E_v\} = \begin{cases} \frac{1}{|S(r_i)|+1}, & 0 \le v \le |S(r_i)| \\ 0, & v \ge |S(r_i)|+1 \end{cases}. \quad (15)$$

2. Actually, v has higher probability to be a small value than to be a large value. However, the uniform distribution assumption makes the estimation of k feasible and actually coincides well with our simulation results reported in Section 6.

Substituting Eqs. 14 and 15 into Eq. 13, we can derive

$$\begin{aligned} P\{N_\delta \le \Delta | E_k\} &= 1 - \frac{\sum_{v=\Delta+1}^{|S(r_i)|}(1-p_e)^{vk}}{\sum_{v=0}^{|S(r_i)|}(1-p_e)^{vk}} \\ &\ge 1 - \frac{\sum_{v=\Delta+1}^{\infty}(1-p_e)^{vk}}{\sum_{v=0}^{\infty}(1-p_e)^{vk}} \\ &= 1-(1-p_e)^{(\Delta+1)k}. \end{aligned} \quad (16)$$

To satisfy Eq. 9, we let

$$1-(1-p_e)^{(\Delta+1)k} \ge p_{req}, \quad (17)$$

from which we have

$$(\Delta+1)k \ge \log_{1-p_e}(1-p_{req}). \quad (18)$$

This requires that

$$k \ge \frac{\log_{1-p_e}(1-p_{req})}{\Delta+1}. \quad (19)$$

As analyzed before, the optimal ρ is 0.9697, with which $p_e = e^{-\rho} \approx 0.3792$. Denote by $\phi = 1 - p_e = 0.6208$. In TABLE 2 we list the minimal value of k for different combinations of Δ and p_{req}.

TABLE 2
The minimum k for different combinations of Δ and p_{req}.

	$\Delta=0$	1	2	3	4	5
$\mathbf{p_{req}=0.95}$	6	3	2	2	1	1
$\mathbf{p_{req}=0.99}$	10	5	3	2	2	2

4.6 Discussions

The average time to eliminate a nontarget tag (C_l) varies in different rounds. Combining Eq. 8 and Eq. 5, C_l can be expressed as

$$\begin{aligned} C_l &= \frac{t_s}{|Z_l|} * \frac{N_l}{\rho} * e^{\rho} + \frac{1}{|Z_i|} * \frac{N_l}{\rho} * (t_e - t_s) \\ &= \frac{N_l}{|Z_l|}\left(\frac{t_s * e^{\rho}}{\rho} + \frac{t_e - t_s}{\rho}\right). \end{aligned} \quad (20)$$

In our system setting, $t_e = 0.184ms$, $t_s = 0.2ms$, and $\rho = 0.9697$. Thus we have

$$C_l = \frac{0.5274 * N_l}{|Z_l|}. \quad (21)$$

Obviously, C_l may vary in different rounds because Z_l and N_l vary round by round. The ratio C_{l-1}/C_l shows the variation trend in two consecutive rounds, which is

$$\frac{C_{l-1}}{C_l} = \frac{N_{l-1}}{N_l} \div \frac{|Z_{l-1}|}{|Z_l|}. \quad (22)$$

$|Z_l|$ denotes the number of nontarget tags after some nontarget tags have been eliminated from Z_{l-1} in the (l-1)-th round. Combining Eq. 2 and Eq. 4, we have

$$\begin{aligned} |Z_l| &= |Z_{l-1}| - |Z_{l-1}| * p_e \\ &\approx |Z_{l-1}| * (1 - e^{-N_{l-1}/f_{l-1}}). \end{aligned} \quad (23)$$

Following a similar process, we can derive the relationship between N_l and N_{l-1}. Different from nontarget tags in Z_{l-1} that will be eliminated in empty slots with probability p_e, the non-wanted tags would be acknowledged in only non-testing slots, i.e., the slots that wanted tags do not select. Thus the probability of acknowledging a non-wanted tag is

$$p'_e = (1 - \frac{1}{f_{l-1}})^{|Z_{l-1}|} \approx e^{-|Z_{l-1}|/f_{l-1}}, \quad (24)$$

with which we have

$$N_l \approx N_{l-1} * (1 - e^{-|Z_{l-1}|/f_{l-1}}). \quad (25)$$

Substituting Eq. 23 and Eq. 25 into Eq. 22, we have

$$\begin{aligned} \frac{C_{l-1}}{C_l} &= \frac{1-e^{-N_{l-1}/f_{l-1}}}{1-e^{-|Z_{l-1}|/f_{l-1}}} \\ &= 1 + e^{-|Z_{l-1}|/f_{l-1}} * \frac{1-e^{(|Z_{l-1}|-N_{l-1})/f_{l-1}}}{1-e^{-|Z_{l-1}|/f_{l-1}}}. \end{aligned} \quad (26)$$

Eq. 26 shows that $\frac{C_{l-1}}{C_l} < 1$ when $|Z_{l-1}| > N_{l-1}$, which indicates that if the number of nontarget tags in Y_l is larger than the number of non-wanted tags, the time efficiency of STEP decreases. In Fig. 4 we plot the value of C_l for the first ten rounds in 500 independent runs of STEP, with the wanted tag number $|X| = 5000$, the local tag number $|L| = 1000$, and the target tag number $|X \cap L| = 200$. Fig. 4 shows that C_l gradually increases along with the increase in rounds, which coincides well with our analysis. In the first several rounds, tags in Z_l are eliminated rapidly. In this case, most slots in the frame might be testing slots, as the nontarget tags in Z_l are significantly more than local tags. In contrast, it is difficult to decrease N_l because non-wanted local tags cannot be acknowledged in testing slots. Since N_l does not decrease a lot, the frame size of the following rounds will be almost the same as in the current round. With the same frame size and much smaller $|Z_l|$, the proportion of empty testing slots will decrease. As a result, it becomes more difficult to eliminate nontarget tags from Y_l, because many testing slots will be also selected by local tags. Eq. 22 indicates that, to keep high tag elimination efficiency, we should *shrink N_l proportionally when eliminating tags from Z_l.*

5 Enhanced Protocol

The E-STEP protocol proposed in this section improves time efficiency by acknowledging non-wanted tags to shrink N_l proportionally when eliminating tags from Z_l.

5.1 Design Guideline

To decrease N_l, we use an indicator vector [30] to suppress non-wanted tags and prevent them from participating in the following rounds. In the l-th round, before broadcasting the query $< f_l, s >$ and eliminating nontarget tags from Y_l, the reader first broadcasts a query $< b_l, s >$ and a b_l-bits indicator vector. Each bit

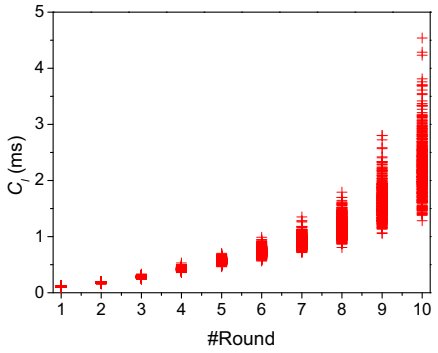


Fig. 4. C_l in different rounds of STEP.

in the vector corresponds to a slot in the frame at the same index location. If the slot is a testing slot, the bit value is '1'; otherwise, the bit value is '0'. The reader divides the vector into segments of 96 bits (i.e., equal to the length of a tag ID) and transmits each segment with time t_{id}. When receiving the query from the reader, local tags select slots in a frame of b_l slots. Then a local tag will check the bit value at the index of its selected slot. If the bit value is '0', the tag suppresses itself and will not participate in the following procedure. Note that in this phase *tags actually do not transmit responses to the reader*, they only receive the indicator vector and suppress themselves accordingly. After broadcasting the indicator vector, the reader broadcasts a new query $<f_l, s>$ and scans the responses to eliminate nontarget tags from Y_l and acknowledge non-wanted tags as does in STEP.

The transmission of the indicator vector incurs additional time cost. For some rounds, using the indicator vector may not effectively improve the time efficiency. We have to determine when we should use an indicator vector, and how to set the optimal length of the vector if we need to use it.

5.2 Protocol Overview

Consider an arbitrary round l. We first calculate the difference $\Delta C = C_l^1 - C_l^2$, where C_l^1 is the average time to eliminate a nontarget tag if the indicator vector is not used, and C_l^2 is the average time if the indicator vector is used. Only when $\Delta C > 0$, the reader will broadcast the indicator vector to suppress non-wanted tags.

There are two cases to consider:

1) **The reader does not broadcast the indicator vector**. In this case, the average time to eliminate a nontarget tag from Y_l is (Eq. 20):

$$C_l^1 = \frac{N_l}{|Z_l|}\left(\frac{t_s * e^{\rho}}{\rho} + \frac{t_e - t_s}{\rho}\right).$$

2) **The reader broadcasts the indicator vector**. In this case, the average time to eliminate a nontarget tag from Y_l consists of two parts:

- For the first part, the reader uses a b_l-bits indicator vector to suppress non-wanted tags. The time cost of the first part is $T_{p1} = \lceil \frac{b_l}{96} \rceil * t_{id}$. A local tag would be suppressed if it maps to a slot whose bit value is '0'. The probability that a local tag maps to such a slot equals the probability that no tags in Y_l choose this slot, which is $p_0 = (1 - \frac{1}{b_l})^{|Y_l|} \approx e^{-|Y_l|/b_l}$. Thus the expected number of suppressed tags is approximately $N_l * p_0 = N_l * e^{-|Y_l|/b_l}$.
- For the second part, the reader tests and eliminates nontarget tags from Y_l. We can derive that the number of remaining local tags after suppressing tags with the indicator vector would be $N_l' \approx N_l * (1 - e^{-|Y_l|/b_l})$. Thus the optimal frame size should be $f_l = N_l'/\rho$. From Eq. 3, the time cost of the second part is $T_{p2} = \frac{N_l'}{\rho} * t_s + \frac{N_l'}{\rho} * e^{-\rho} * (t_e - t_s)$.

The total time of the two parts is

$$\begin{aligned} T_{total} &= T_{p1} + T_{p2} \\ &= \lceil \frac{b_l}{96} \rceil * t_{id} + \frac{N_l'}{\rho} * t_s + \frac{N_l'}{\rho} * e^{-\rho} * (t_e - t_s). \end{aligned} \tag{27}$$

To calculate C_l^2, we need to know how many nontarget tags would be eliminated from Y_l when the indicator vector is used. As we have pointed out in the analysis of STEP (refer to Section 4.4), the expected number of nontarget tags that would be eliminated is

$$N_{eli} \approx |Z_l| * e^{-N_l'/f_l} = |Z_l| * e^{-\rho}, \tag{28}$$

where $Z_l = Y_l - (X \cap L_i)$. In the first several rounds, $|Y_l|$ is usually much larger than $|X \cap L_i|$, thus we can assume that $|Z_l| \approx |Y_l|$.

The average time to eliminate a nontarget tag from Y_l in the second case is

$$\begin{aligned} C_l^2 &= \frac{T_{total}}{N_{eli}} \\ &= \frac{\lceil \frac{b_l}{96} \rceil t_{id} + \frac{N_l'}{\rho} t_s + \frac{N_l'}{\rho} e^{-\rho} * (t_e - t_s)}{|Z_l| e^{-\rho}} \\ &= \frac{1}{|Z_l|}\left(e^{\rho} \lceil \frac{b_l}{96} \rceil t_{id} + \frac{N_l'}{\rho} t_s e^{\rho} + \frac{N_l'}{\rho}(t_e - t_s)\right) \\ &\approx \frac{1}{|Y_l|}\left(e^{\rho} \lceil \frac{b_l}{96} \rceil t_{id} + (\frac{e^{\rho} t_s}{\rho} + \frac{t_e - t_s}{\rho})(1 - e^{-|Y_l|/b_l}) N_l\right). \end{aligned} \tag{29}$$

To compare the time efficiency in the two cases, we calculate their difference, which is

$$\Delta C = \frac{1}{|Y_l|}\left[(\frac{t_s e^{\rho}}{\rho} + \frac{t_e - t_s}{\rho}) N_l e^{-|Y_l|/b_l} - e^{\rho} \lceil \frac{b_l}{96} \rceil t_{id}\right]. \tag{30}$$

In case of the EPC C1G2 tag specification [31], we have $t_{id} = 2.4ms$, $t_e = 0.184ms$, and $t_s = 0.2ms$. Meanwhile, as have been derived in the analysis for STEP, the optimal load factor under this time setting is $\rho = 0.9697$. Thus we have

$$\Delta C = \frac{1}{|Y_l|}\left[0.5274 * N_l * e^{-|Y_l|/b_l} - 4.0668 * \lceil \frac{b_l}{96} \rceil\right]. \tag{31}$$

The value of ΔC represents the expected time reduction when we use an indicator vector. The larger ΔC is, the more time would be reduced. In the l-th round, with the given number N_l and $|Y_l|$, the value of ΔC depends

only on the length of the indicator vector b_l. Let b_l^* denote the optimal b_l that maximizes the value of ΔC, i.e.,

$$b_l^* = \arg_{b_l} \max \Delta C, \tag{32}$$

and let ΔC_{max} denote the value of ΔC when $b_l = b_l^*$. If ΔC_{max} is smaller than 0, which means that using an indicator vector cannot improve the time efficiency, we should not use the indicator vector in the l-th round. Otherwise, we should use an indicator vector of length b_l^* to further improve time efficiency.

5.3 Protocol Description

E-STEP also consists of multiple rounds. Every round is divided into two phases: the suppressing phase and the eliminating phase. Consider the l-th round without loss of generality.

In the suppressing phase, with the number of candidate target tags $|Y_l|$ and the number of local tags N_l, the reader first calculates the value of b_l^* and ΔC_{max}.

- If $\Delta C_{max} \leq 0$, the reader enters the second phase directly.
- If $\Delta C_{max} > 0$, the reader broadcasts a query $<b_l^*, s>$, and broadcasts a b_l^*-bits indicator vector. The reader constructs the vector by mapping candidate target tags in Y_l to a frame containing b_l^* slots: If a slot is a testing slot, e.g., some tags in Y_l select it, the reader will set the bit at the same index location to '1'; otherwise, it sets the bit to '0'.

Tags do not respond to the reader in this phase. When a local tag receives the query and the vector, it first checks the bit at the same index location of its selected slot. If the bit is '0' (i.e., the tag maps to a non-testing slot), the tag will suppress itself and does not participate in the following round. If the bit is '1', the tag will continue to receive the query from the reader in the second phase.

In the eliminating phase, the reader issues a new query $< f_l, s >$ and receives responses from local tags. Note that f_l is calculated according to number of remaining local tags as discussed in the Section 5.2. With these responses, the reader eliminates nontarget tags from Y_l and acknowledges non-wanted tags as in STEP.

At the end of this round, the reader checks whether it should terminate or not. It starts a new round if the termination condition is not met. The termination condition of E-STEP is the same as in STEP.

The indicator vector can effectively reduce per tag elimination time in the first several rounds. Fig. 5 plots C_l (the average time cost to eliminate a nontarget tag) in STEP and E-STEP in the first eight rounds in one random chosen example, when the wanted tag number $|X|$=5000, the local tag number $|L|$=1000, and the target tag number $|X \cap L|$=200. We observe significant reduction of C_l in E-STEP compared with in STEP, especially in the first several rounds. For example, in rounds 2 to 4, E-STEP reduces C_l by almost a half. As most nontarget tags are eliminated in the first several rounds, E-STEP effectively reduces the total execution time.

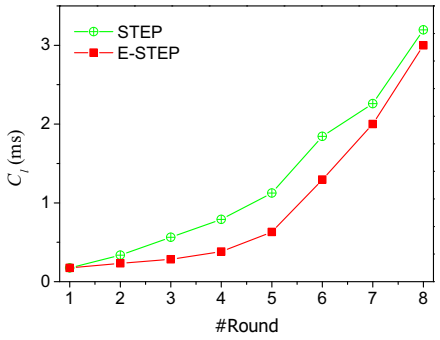


Fig. 5. Comparison of C_l in STEP and E-STEP in one example run.

5.4 Fault Tolerance

The indicator vector transmitted from the reader to tags might be corrupted due to channel errors. In this case, if a tag receives a wrong bit, it may take incorrect action. Recall that when a tag finds that its corresponding bit is '0', it will be suppressed and will not participate in the following rounds. If the actually sent bit is '1' but the received bit is '0', the tag will exit the protocol incorrectly. If this tag is actually a target tag, then it is possible that this tag is incorrectly eliminated from Y_l and thus will not be included in the searching result, which will affect the correctness of E-STEP.

To fix this problem, we can add a cyclic-redundancy check (CRC) code in each segment when transmitting the indicator vector. The tag can thus check whether the received information is correct or not. If the corresponding bit is correctly received, the tag takes its action accordingly as described in our protocol. Otherwise, it will reply to the reader in its selected slot and ignore the corresponding bit in the vector. For example, if the tag finds that the segment containing its bit is ruined, it will reply to the reader in the selected slot, even though the corresponding bit is '0'. With this mechanism, the correctness of E-STEP can be guaranteed in the meaning that there will be no false negative results. The time efficiency of E-STEP might degrade slightly when this scheme is used.

6 Performance Evaluation

We developed a simulator with JAVA and implemented five protocols to compare their performance: STEP, E-STEP, CATS [4], ITSP [5], and the baseline Collection solution. Two metrics are used to compare the performance of different protocols: (a) The *execution time* measured in seconds, and (b) the number of false positive tags in the searching result (N_δ). We consider three parameters that may affect the performance of different protocols: (a) The total number of wanted tags ($|X|$), (b) the ratio of target tags to the wanted tags, which is defined as $\eta = |X \bigcap T|/|X|$, and (c) the number of readers in the system (M). For STEP and E-STEP, we also investigate the impact of Δ, i.e., the threshold of false positive tags,

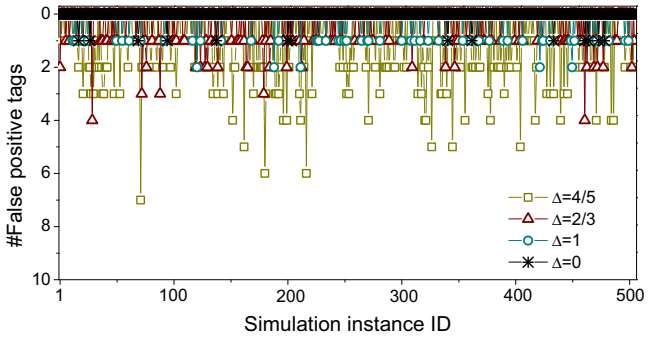


Fig. 6. N_δ of E-STEP in 500 runs when Δ changes.

on their performances. For each parameter setting, we run the considered protocols 100 times and report the average result. The detailed description of the simulator could be found at [32].

6.1 Simulation Scenarios and Time Setting

Simulation scenarios: We consider two different simulation scenarios. The first scenario is a single reader RFID system, in which 2000 tags are deployed. This scenario is used to investigate how different parameters affect the performance of STEP and E-STEP, meanwhile compare their execution time and precision with CATS and the state-of-the-art ITSP protocol. The second scenario considers multiple reader RFID systems. In default, we deploy 64 readers and 50000 tags in the system. The readers are deployed in a grid pattern with distance between adjacent readers set at $\sqrt{2}r$, where r is the interrogation radius of a reader. This results in about 1500 local tags for each reader. The default number of wanted tags ($|X|$) is set at 10000, and the default ratio of target tags (η) is set at 0.2. For multiple reader scenarios, we use the graph coloring algorithm developed in [27] to find a feasible scheduling of readers.

Time setting: We set the time duration of different slots according to the time specification of the EPC C1G2 UHF RFID tags [31]. The data rate between the reader and tags is set at 62.5Kbps. Under this data rate, the time of different slots are as follows: $t_{id} = 2.42$ms, $t_e = 0.184$ms, and $t_s = 0.2$ms. We note that when the data rate changes, the absolute metric data might be different, but similar conclusions could be reached.

6.2 Single Reader Scenario

6.2.1 Impact of False Positive Tag Number Threshold

The false positive tag number threshold Δ affects the performance of STEP and E-STEP. A smaller Δ results less false positive tags in the searching result, but it also requires the reader to observe more rounds before it terminates and consequently increases execution time. Fig. 6 plots the number of false positive tags of E-STEP in 500 runs when Δ varies from 0 to 5. The results validate the speculation that larger Δ results in more false positive tags. Meanwhile, the proposed protocol guarantee the searching precision well. The ratios of runs in which $N_\delta > \Delta$ to the total runs are 0.972, 0.99, 0.99, 0.996, 0.986, and 0.994 when Δ changes from 0 to 5, respectively, all larger than the required probability 0.95.

Fig. 7 plots the average execution time of E-STEP when Δ changes. The execution time decreases when Δ increases, because more false positive tags could be tolerated when Δ is larger. When Δ increases from 0 to 5, the execution time decrease from 1.2s to 0.9s, approximately 25 percent reduction. The execution time of STEP shows the similar trend. Thus Δ could be used to make tradeoff between time efficiency and searching precision: If the application prefers high time efficiency and can tolerate false positive tags, it can set Δ to a relatively large value. In contrast, if the application requires very precise searching result, it should use a small Δ. In the following experiments, we set $\Delta = 0$ and p_{req}=0.95, i.e., there are no false positive tags in the searching result of STEP and E-STEP with a probability no smaller than 0.95.

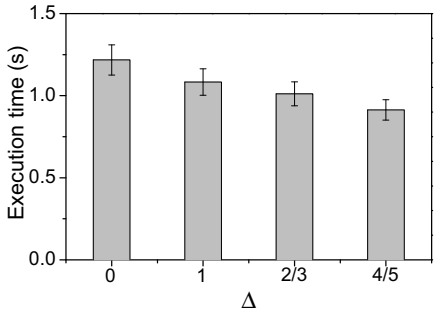


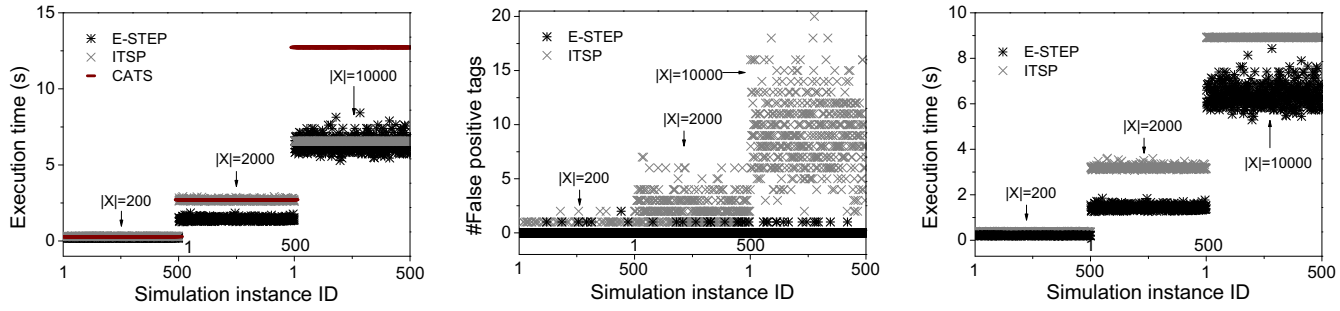
Fig. 7. Execution time of E-STEP when Δ changes.

6.2.2 E-STEP vs. STEP

Fig. 8(a) and Fig. 8(b) plot the execution time of STEP and E-STEP for different η when there are 200 and 2000 wanted tags, respectively. Two conclusions can be drawn. First, E-STEP significantly outperforms STEP when $|X| \ll |T|$. When X is small, most of the reader's local tags are nontarget tags, which can be quickly filtered out by the bit vector used in E-STEP. Compared with STEP, E-STEP reduces execution time by 81% when η=0.1 and 60% when η=0.9, respectively. Second, the improvement of E-STEP over STEP becomes less significant when either η or the wanted tag number ($|X|$) is large. When η or $|X|$ is large, most of the reader's local tags are target tags and cannot be filtered out by the bit vector. When $|X|$=2000, E-STEP reduces execution time by 40% and 6% when η=0.1 and η=0.9, respectively.

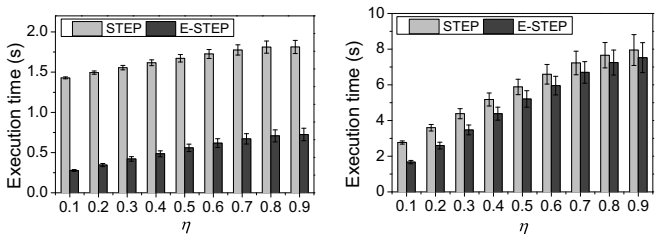
6.2.3 Comparison with CATS and ITSP

Fig. 9(a) plots the execution time of the three protocols in 500 runs when $|X| = 200$, 2000, and 10000, respectively. For ITSP and CATS, we set the false positive tag ratio, defined as $p_f = \frac{|S-X \bigcap T|}{|X-T|}$, as 10^{-3}. All the three



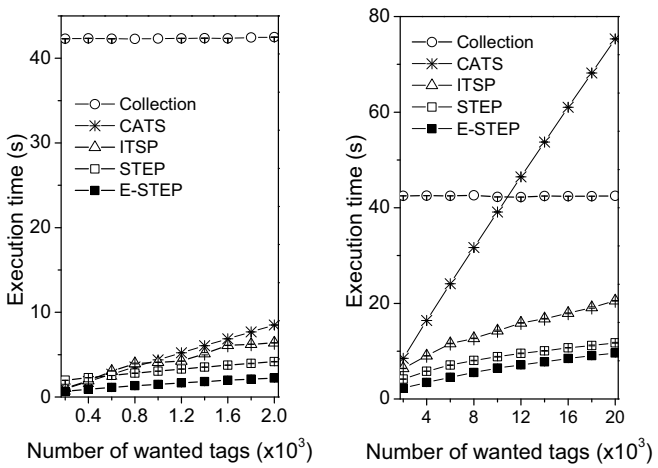
(a) Execution time (b) False positive tag number (c) Execution time

Fig. 9. Comparison between E-STEP, ITSP, and CATS: (a) Execution time, (b) false positive tag number, (c) execution when $p_f = 10^{-4}$ in ITSP.



(a) $|X| = 200$ (b) $|X|$=2000

Fig. 8. Execution time of STEP and E-STEP when η changes ($|T|$=2000): (a) Small $|X|$; (b) large $|X|$.



(a) Small $|X|$ (b) Large $|X|$

Fig. 10. Execution time of different protocols when the number of wanted tags changes: (a) Small $|X|$; (b) large $|X|$. 64 readers, $|T|$=50000, η=0.2.

protocols perform well when $|X|$ is small, due to the high efficiency of the bit vector in filtering out nontarget tags. When $|X|$=200, the average execution time of E-STEP, ITSP, and CATS are 0.213s, 0.305s and 0.295s, respectively. Compared with CATS and ITSP, E-STEP reduces searching time by 30% and 28%, respectively. When $|X|$ is comparable to $|T|$, e.g., $|X| = 2000$, E-STEP performs much better than CATS and ITSP, using 47% and 46% less time, respectively.

Both ITSP and E-STEP perform much better than CATS when $|X|$ is much larger than $|T|$. For example, when $|X| = 10000$, the average execution time of E-STEP, ITSP, and CATS are 0.639s, 0.656s, and 1.273s, respectively. E-STEP and ITSP use only about half of searching time of CATS. However, ITSP generates much more false positive tags than E-STEP does when $|X|$ is large. As shown in Fig. 9(b), there could be up to 20 false positive tags in ITSP. In contrast, the number of false positive tags of E-STEP is independent to $|X|$ and always smaller than 2. In fact, in most runs (1470 out of 1500) there are no false positive tags in E-STEP. Fig. 9(c) plots the execution time of E-STEP and ITSP when p_f is set at $= 10^{-4}$ in ITSP, in which case ITSP can achieve similar searching precision as E-STEP does. In this case, E-STEP obviously outperforms ITSP and reduces execution time by more than 28%.

6.3 Multiple Reader Scenario

6.3.1 Impact of Wanted Tag Number

Fig. 10 plots the execution time of different protocols when the wanted tag number increases. We have two observations. First, in all the considered protocols (except Collection), the execution time increases when there are more wanted tags. CATS's execution time increases much faster than the other three protocols. It performs even better than STEP and ITSP when $|X| \leq 400$, but performs even worse than Collection when $|X| \geq 10000$. Compared with CATS, STEP and E-STEP reduce execution time by up to 84% and 87%, respectively. Second, there is a cross point between STEP's and ITSP's execution time: ITSP uses less time than STEP when $|X| \leq 400$, but uses more time than STEP when $|X|$ is large. E-STEP always outperforms STEP and ITSP. Compared with ITSP and STEP, E-STEP reduces execution time by 60% and 52% in average, respectively.

Fig. 11 plots the number of false positive tags (N_δ)

in E-STEP and ITSP [3] when $|X|$ increases from 2000 to 20000. In ITSP the false positive tags increase proportionally to the wanted tags, while in E-STEP the number of false positive tags is not affected by the wanted tags and is always small. Recall that E-STEP can guarantee $P\{N_\delta < \Delta\} \geq p_{req}$. In our simulation setting, the theoretical expected number of false positive tags in E-STEP is approximately $\Delta * (1 - p_{req}) * M = 1 * (1 - 0.95) * 64$=3.2, which well coincides with the simulation results. In contrast, N_δ increases from 10 to 60 in ITSP. Define the searching precision as the ratio of N_δ to $|X \bigcap T|$. E-STEP promotes the searching precision by nearly an order of magnitude with only half searching time.

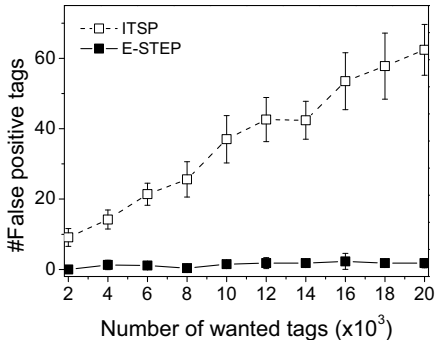


Fig. 11. Number of false positive tags in E-STEP and ITSP when $|X|$ increases from 2000 to 20000. 64 readers, $|T|$=50000, η=0.2.

6.3.2 Impact of Target Tag Ratio

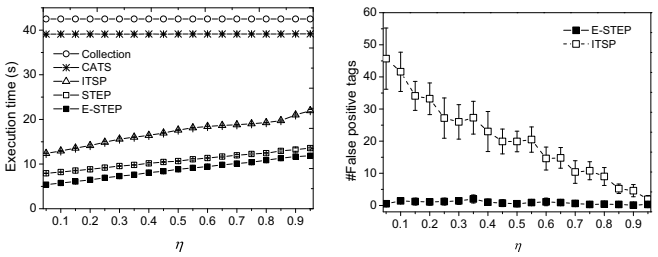
Fig. 12(a) plots the execution time of different protocols when η varies from 0.05 to 0.95. While the execution time of STEP, E-STEP, ITSP all increases when η becomes large, the execution time of CATS is not affected by η. This is because in CATS the size of the Bloom filter is set according to $|X|$, which is independent to η. In contrast, when η increases, there would be more target tags in each reader's region, in which case our protocols and ITSP need to run more rounds to meet the precision requirement. However, STEP and E-STEP always outperform CATS and ITSP. Compared with CATS and ITSP, E-STEP reduces execution time by up to 86% and 57%, respectively.

Fig. 12(b) plots the number of false positive tags in ITSP and E-STEP when η increases. Along with the increase of η, the number of nontarget tags (i.e., $X - T$) becomes smaller, and thus the number of false positive tags in ITSP decreases, as validated by Fig. 12(b). In contrast, the false positive tags in E-STEP are not affected by η and are always less than that in ITSP.

6.3.3 Impact of Reader Number

We change the number of readers in the system by varying the size of the deployment region to investigate how the system scale impacts different protocols' performance. We increase the number of readers ($|M|$) from 16 to 121 and keeps the tag density and distance between adjacent readers unchanged. This means that $|T|$ increases from 12500 (when M=16) to 100000 (when M=121).

3. STEP and CATS have the similar false positive tags as E-STEP and ITSP, respectively.



(a) Execution time (b) N_δ

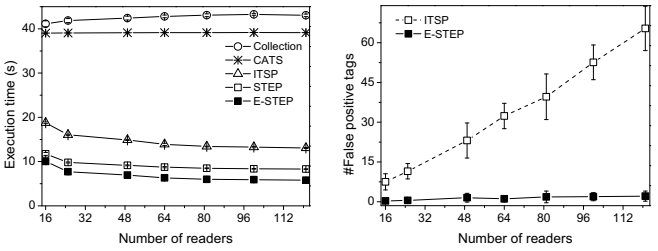
Fig. 12. Execution time and N_δ in different protocols when η changes: (a) Execution time; (b) false positive tag number. 64 readers, $|T|$=50000, $|X|$=10000.

Fig. 13(a) and Fig. 13(b) plot the execution time and the number of false positive tags in different protocols, respectively. Contrary to the intuition that the execution time will increase when the system scales up, it is shown in Fig. 13(a) that the execution times in both our protocols and ITSP decrease. The reason is as follows. Because the number of target tags is fixed, the target tags in each reader's interrogation region become less when M increases. This is equivalent to decreasing η, in which case nontarget tags will be filtered out more efficiently. In contrast, the execution time of CATS is not affected by η and remains unchanged. Moreover, we should point out that although the number of readers increases, the total rounds to schedule all the readers to work does not change. Actually, in all the cases the readers are scheduled in 4 rounds. In average, compared with CATS and ITSP, E-STEP reduces execution time by 82 percent and 53 percent, respectively.

The numbers of false positive tags in E-STEP and ITSP when M increases are plotted in Fig. 13(b). In both our protocol and ITSP, N_δ increases when M becomes larger. However, our protocol achieves much higher searching precision than ITSP does. The false positive tag number in the searching result of E-STEP is smaller than 5 even when there are more than 100 readers in the system, while the false positive tag number of ITSP is larger than 50 in the same case.

7 Discussions on Implementation Issues

The STEP protocol could be implemented on current EPC C1G2 [31] tags and readers after some slight modifications are made on them. First, the reader needs to know the hash function that tags use to select transmission slots in order to calculate the testing slots. To achieve this goal, we can let the readers and tags use the same hash function to select transmission slots. Second,



(a) Execution time (b) N_δ

Fig. 13. Execution time and N_δ in different protocols when M increases: (a) Execution time, (b) false positive tag number. $|T|$ increases from 12500 (M=16) to 100000 (M=121), $|X|$=10,000, η=0.2.

STEP requires tags to transmit short responses rather than their IDs when receiving the searching command. This can be accomplished by adding a new state in the tag-side software implementation and triggering the short-response mode in this state. Third, the reader needs to detect the status of each slot and send different acknowledgement (*ACK* or *NAK*) to tags accordingly. All these modifications are in the software level and could be accomplished by adding some new states and corresponding transition conditions in the software implementation.

E-STEP uses the indicator vector technique that could be implemented on EPC G1G2 compliant tags by using the method given in [33]. To broadcast an indicator vector, the reader first divides the vector into segments of 96 bits and encapsulates each segment into a tag ID. The segments are broadcasted one after another. A tag buffers the segment in which its corresponding bit resides in and acts correspondingly. To further enhance the robustness of the indicator vector, cyclic-redundancy check (CRC) code could be added into each segment to help tags check whether the segment is correctly received. This approach to implement indicator vector has been adopted in many excellent existing researches [4], [5], [30], [33].

8 Conclusion

Searching a particular set of tags in multiple reader RFID systems is very important to many RFID applications but has not been thoroughly investigated yet. In this paper, we propose two highly time-efficient tag searching protocols, STEP and E-STEP, which are applicable to a broad set of application scenarios. Our protocols iteratively eliminate nontarget tags round by round to obtain the searching result, and minimize the average time to eliminate nontarget tags in every round to achieve high time-efficiency. Simulation results exhibit the superior performance of our protocols. Compared with the state-of-the-art ITSP protocol, our best protocol reduces searching time by at most 60%, and greatly reduces the number of false positive tags in the searching result by nearly an order of magnitude.

The merits of our protocols would be more significant in large-scale RFID systems that contain tens of thousands or more tags. The simulation results and findings obtained in this paper represent an important step towards understanding the characteristics and core requirements of an efficient tag searching protocol for large-scale RFID systems. In the future, we would like to deploy our protocols, along with ITSP and CATS, to operate in a real setup of a reasonably large-scale environment to validate the characteristics of our protocols found in this paper.

Acknowledgment

This work is partially supported by HK RGC PolyU 5281/13E, the National Science Foundation of China under Grant Nos. 61103203, 61373181, 61402404, and the Fundamental Research Funds for the Central Universities under Grant No. 2014QNA5012. The authors would also like to sincerely thank the Editors and reviewers for their thoughtful, constructive suggestions and comments.

References

[1] J. Wang, H. Hassanieh, D. Katabi, and P. Indyk, "Efficient and reliable low-power backscatter networks," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 61–72, 2012.

[2] J. Liu, B. Xiao, K. Bu, and L. Chen, "Efficient distributed query processing in large RFID-enabled supply chains," in *Proc. of Infocom*, 2014, pp. 163–171.

[3] X. Liu, B. Xiao, K. Bu, and S. Zhang, "LOCK: A fast and flexible tag scanning mechanism with handheld readers," in *Proc. of IWQoS*, 2014, pp. 360–369.

[4] 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.

[5] W. Lou, Y. Qiao, and S. Chen, "An efficient tag seaching portocol in large-scale RFID systems," in *Proc. of Infocom*, 2013.

[6] S. Zhang, X. Liu, J. Wang, J. Cao, and G. Min, "Energy-efficient active tag searching in large scale RFID systems," *Accepted to apear in Information Sciences*, 2014.

[7] L. Xie, Y. Yin, A. V. Vasilakos, and S. Lu, "Managing RFID data: Challenges, opportunities and solutions," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 3, pp. 1294–1311, 2014.

[8] J. Myung, W. Lee, J. Srivastava, and T. Shih, "Tag-splitting: Adaptive collision arbitration protocols for RFID tag identification," *IEEE Transactions on Parallel and Distributed Systens*, vol. 18, no. 6, pp. 763–775, 2007.

[9] M. Shahzad and A. X. Liu, "Probabilistic optimal tree hopping for RFID identification," in *Proc. of SIGMETRICS*, 2013, pp. 293–304.

[10] Y. Lai, L. Hsiao, H. Chen, C. Lai, and J. Lin, "A novel query tree protocol with bit tracking in RFID tag identification," *IEEE Transactions on Mobile Computing*, vol. 12, no. 10, pp. 2063–2075, 2013.

[11] V. Namboodiri and L. Gao, "Energy-aware tag anticollision protocols for RFID systems," *IEEE Transactions on Mobile Computing*, 2009.

[12] L. Kang, K. Wu, J. Zhang, H. Tan, and L. M. Ni, "DDC: A novel scheme to directly decode the collisions in UHF RFID systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 23, no. 2, pp. 263–270, 2012.

[13] M. Zhang, T. Li, S. Chen, and B. Li, "Using analog network coding to improve the RFID reading throughput," in *Proc. of ICDCS*, 2010, pp. 547–556.

[14] J. Li and Y. Huo, "An efficient time-bound collision prevention scheme for RFID re-entering tags," *IEEE Transactions on Mobile Computing*, vol. 12, no. 6, pp. 1054–1064, 2013.
[15] L. Kong, L. He, Y. Gu, M. Y. Wu, and T. He, "A Parallel Identification Protocol for RFID Systems," in *Proc. of Infocom*, 2014, pp. 154–162.
[16] 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. pp, no. 99, pp. 1–14, 2014, doi:10.1109/TPDS.2014.2326651.
[17] X. Liu, S. Zhang, K. Bu, and B. Xiao, "Complete and fast unknown tag identification in large RFID systems," in *Proc. of MASS*, 2012, pp. 47–55.
[18] J. Yao, T. Xiong, and W. Lou, "Beyond the limit: A fast tag identification protocol for RFID systems," *Accepted to appear in Pervasive and Mobile Computing*, 2014, doi:10.1016/j.pmcj.2014.10.003.
[19] J. Cha and J. Kim, "Novel anti-collision algorithms for fast object identification in RFID system," in *Pro. of IPDPS*, vol. 2, 2005, pp. 63–67.
[20] D. Klair, K. Chin, and R. Raad, "An investigation into thie energy eficiency of pure and slotted aloha based RFID anti-collision protocols," in *Proc. of WoWMoM*, 2007.
[21] X. Liu, S. Zhang, B. Xiao, and K. Bu, "Flexible and time-efficient tag scanning with handheld readers," *IEEE Transactions on Mobile Computing*, vol. pp, pp. 1–14, 2014, doi:10.1109/TMC.2014.2381252.
[22] J. Waldrop, D. Engels, and S. Sarma, "Colorwave: an anticollision algorithm for the reader collision problem," in *Proc. of ICC*, vol. 2, 2003, pp. 1206–1210.
[23] Z. Zhou, H. Gupta, S. Das, and X. Zhu, "Slotted scheduled tag access in multi-reader RFID systems," in *Proc. of ICNP*, 2007, pp. 61–70.
[24] S. Tang, J. Yuan, X. Li, G. Chen, Y. Liu, and J. Zhao, "Raspberry: A stable reader activation scheduling protocol in multi-reader RFID systems," in *Proc. of ICNP*, 2009, pp. 304–313.
[25] S. Tang, C. Wang, X. Li, and C. Jiang, "Reader activation scheduling in multi-reader RFID systems: A study of general case," in *Proc. of IPDPS*, 2011, pp. 1147–1155.
[26] L. Yang, Y. Qi, J. Han, C. Wang, and Y. Liu, "Shelving Interference and Joint Identification in Large-Scale RFID Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. pp, no. 99, pp. 1–11, 2014.
[27] D. Brélaz, "New methods to color the vertices of a graph," *Communications of the ACM*, vol. 22, no. 4, pp. 251–256, 1979.
[28] W. Gong, K. Liu, X. Miao, and H. Liu, "Arbitrarily Accurate Approximation Scheme for Large-Scale RFID Cardinality Estimation," in *Proc. of Infocom*, 2014, pp. 477–485.
[29] M. Shahzad and A. X. Liu, "Every bit counts: fast and scalable RFID estimation," in *Proc. of Mobicom*. ACM, 2012, pp. 365–376.
[30] T. Li, S. Chen, and Y. Ling, "Identifying the missing tags in a large RFID system," in *Proc. of Mobihoc*, 2010, pp. 1–10.
[31] EPCGlobal, "EPC Radio-Frequency Identity Protocols Class-1 Gen-2 UHF RFID Protocol for Communications at 860 MHzC960 MHz," October 2014 [Online], available: http://www.epcglobalinc.org/standards/uhfc1g2.
[32] X. Liu, B. Xiao, S. Zhang, K. Bu, and A. Chan, "Simulator implementation description," 2014, available: http://netlab.csu.edu.cn/personal/zhangshigeng/simdetails.pdf.
[33] S. Chen, M. Zhang, and B. Xiao, "Efficient information collection protocols for sensor-argumented RFID networks," in *Proc. of Infocom*, 2011, pp. 3101–3109.

Xuan Liu received the BSc degree in Information and Computing Mathematics from XiangTan University, China, and MSc degree in Computer Science from National University of Defense Technology, China, in 2005 and 2008, respectively. Currently, she is a PhD candidate in the Department of Computing at The Hong Kong Polytechnic University, Hong Kong. Her research interests include distributed computing systems, mobile computing, focusing on wireless sensor networks and RFID systems.

Bin Xiao received the B.Sc and M.Sc degrees in Electronics Engineering from Fudan University, China, and Ph.D. degree in computer science from University of Texas at Dallas, USA. After his Ph.D. graduation, he joined the Department of Computing of the Hong Kong Polytechnic University as an Assistant Professor. Now he is an associate professor and the director of the Mobile Cloud Computing Lab. His research is mainly on mobile cloud computing, smart phone technology, network security, and wireless networks. He is the editor of 3 books and has published more than 100 technical papers in international journals and conferences. Currently, he is the associate editor of the International Journal of Parallel, Emergent and Distributed Systems and an editor of the International Journal of Distributed Sensor Networks. He is the IEEE Senior member.

Shigeng Zhang received the BSc, MSc, and DEng degrees, all in Computer Science, from Nanjing University, China, in 2004, 2007, and 2010, respectively. He is currently an Assistant Professor in School of Information Science and Engineering at Central South University, China. His research interests include Cloud Computing, Internet of Things, wireless sensor networks, and RFID systems. He has published more than 30 technique papers in international journals and conferences. He is a member of IEEE and CCF.

Kai Bu received the BSc and MSc degrees in computer science from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2006 and 2009, respectively, and the Ph.D. degree in computer science from The Hong Kong Polytechnic University, Hong Kong, in 2013. He is currently an Assistant Professor in the College of Computer Science and Technology at Zhejiang University, China. His research interests include RFID and wireless networks. He is a recipient of the Best Paper Award of IEEE/IFIP EUC 2011. He is a member of the ACM and the IEEE.

Alvin Chan obtained extensive industrial experience before joining the Hong Kong Polytechnic University in 1998, where he is currently an associate professor. After graduating from the University of Leeds, he joined Chartered Industrial of Singapore as a software engineer responsible for the development of software for military radio systems used in combat and hostile environments. Having completed his PhD study at the University of New South Wales in 1995, he was employed as a research scientist by the Commonwealth Scientific and Industrial Research Organization (CSIRO). He returned to Singapore in 1997 and was employed as a Program Manager by the Center for Wireless Communications, at the National University of Singapore (now under A*STAR). Dr. Chan's current research interests focus on several key areas including (but not limited to): adaptive middleware, social relationship and crowd sensing, Internet of Things, mobile computing, context-aware computing and service-oriented computing, among others.