PLAT: A Physical-layer Tag Searching Protocol in Large RFID Systems

Feng Zhu*†, Bin Xiao†, Jia Liu*, Xuan Liu‡, Li-jun Chen*
*Department of Computer Science and Technology, Nanjing University, Nanjing, China
†Department of Computing, The Hong Kong Polytechnic University, Hong Kong
‡College of Computer Science and Electronic Engineering, Hunan University, Hunan, China

***Abstract*—Radio Frequency Identification (RFID) technology brings a revolutionary change in warehouse management by automatically monitoring and tracking products. For many RFID-enabled applications, fast searching a particular group of products is practically important in a large-scale RFID system. Different from previous searching protocols, we propose a physical layer tag searching (PLAT) protocol, which makes three fundamental improvements. First, PLAT can exactly pinpoint the search result without false positives at a small delay expense. Second, based on the physical layer signals, PLAT can interpret the accurate number of tags replying in a collision slot, speeding up the execution of tag searching. Third, PLAT can take the global view of the accurate replying information from slots to extract each tag identifier, further improving the protocol performance. We also implement a prototype system based on the USRP and WISP platform. Experimental results validate the feasibility of our protocol. The extensive simulations show PLAT produces the performance gain by a factor of above 2 compared with the state-of-the-art works.**

I. Introduction

Radio Frequency Identification (RFID) is developing rapidly in a range of applications such as inventory control, supply chain logistics and object tracking [1]–[10]. Fast searching a particular set of products in a large-scale RFID system is of practical importance for those applications. For example (as shown in Fig 1), the RFID-enabled warehouse has to deal with massive products from different manufacturers every day. The absence of products often happens due to mistake, theft or other reasons. If one of those manufacturers wants to make inventory of only its products, the warehouse should provide a time efficient searching approach to give an accurate and timely inventory result, which is called the *tag searching problem*.

Many existing efforts are dedicated to solving the tag searching problem due to its practical importance [11]–[13]. They commonly take filtering techniques as a vehicle to compact the communication data between the reader and tags, so that no tedious ID transmission is needed, reducing the search time. However, these protocols expose three limitations. The first limitation is that these protocols are probabilistic due to the intrinsic false positives of filtering techniques. In many RFID monitoring systems, especially when the products of the manufacturers under monitoring are valuable, mis-searching products may not be affordable. The second limitation is that none of the existing protocols explores physical-layer information of each slot to improve the efficiency. They all use busy/idle state of each slot to search the wanted tags, which needs a great deal of extra slots to guarantee the accuracy of searching results. The third limitation is that, in many scenarios the warehouse would naturally store the ID list of all the products, such as the previous example in Fig 1, current protocols cannot efficiently utilize the knowledge of the ID list to stabilize and benefit the searching process.

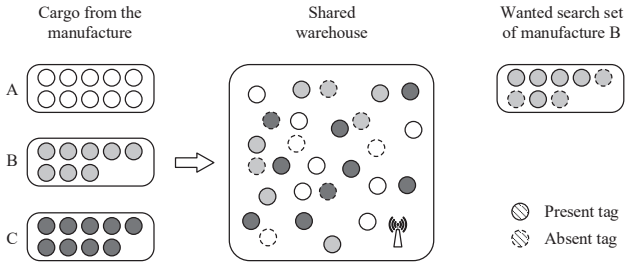


Fig. 1. RFID tag searching problem.

In this paper, we propose a physical layer tag searching (PLAT) protocol to address the searching problem in an efficient way. In PLAT, three novel design ideas are proposed to make a fundamental improvement on the protocol performance. First, with the knowledge of all tag IDs a priori, PLAT can filter out most unwanted tags at a small delay expense by using filtering technique. The tags left with a small scale serve as the input of the following phases, greatly reducing the overhead. Note that the reader has all tag IDs, it can predict the mis-selected ones and can pick them out from the searching result. Second, inspired by the discovery that physical-layer signals can be used to count the number of tags in a slot [14], we propose a heuristic symbols clustering (HSC) algorithm which can interpret the accurate number of tags replying in a slot rather than only three states (empty, singleton, or collision) in the traditional way, further improving the performance. Third, instead of observing each slot independently, PLAT can take the global view of the accurate replying information from multiple slots to extract each tag identifier, further improving the protocol performance. We comprehensively analyse PLAT, including the efficiency of the protocol and the joint optimization for the parameter in each phase of the protocol. By performing a joint optimization, PLAT minimizes the combined overhead.

We also implement a prototype system and validate HSC algorithm based on the Universal Software Radio Peripheral (USRP) platform with the Intel Wireless Identification and Sensing Platform (WISP) tags. When clustering the physical layer signals to exploit the number of replying tags, the noise tends to disrupt the algorithm and makes it difficult to detect the cluster structure. HSC can dive into smaller scale to investigate the density of each clusters without scarifying efficiency and accuracy. The extensive simulations show PLAT produces the performance gain by a factor of above 2 compared with the existing promising works.

The rest of this paper is organized as follows. In Section II, we describe the system model and the problem definition. In Section III, we propose the physical layer tag searching protocol. Section IV analyzes the protocol efficiency. In Section V, we build a prototype system with USRP-based reader and WISP tags, and implement our protocol on it. Section VI evaluates the efficiency of our protocol comparing with other state-of-the-art protocols. Section VII introduces the related work. Section VIII concludes the paper.

II. Preliminary

A. System Model

We consider a large-scale RFID system with a back-end server, multiple readers and a massive number of tags. The back-end server connects with the readers via wired or wireless links, and sends orders to schedule their working. The tags attached to items are allocated at different reader coverage regions. After communicating with the tags, the reader transmits tag information to a back-end server, which provides powerful computation ability to process such data. When multiple readers are synchronized, we can logically treat them as a whole, as the same with [11], [15], [16].

The communications between the readers and tags follow the Reader-Talk-First protocol. Namely, the tag talks only if receiving the reader's commands. A reader initializes each round by sending a request. On receiving the order, each tag randomly chooses a slot. Thus, the reader receives either no tag responds (idle slot) or at least one tag responds (busy slot). Because each tag takes one of the two states by either reflecting or absorbing radio waves from the reader, the exact number of concurrent tag responses in a slot can be detected [14]. Therefore, the reader can distinguish no more than 3 tags through 8 bits, which are 0, 1, 2 and 3+ respectively. According to the Philips I-Code system [17], a short-response slot allows the transmission of one bit information, which is denoted as t_s. A tag slot allows the transmission of the tag ID, which is denoted as t_{ID}. Since our protocol will use 8 bits to distinguish no more than 3 tags, we denote the long-response slot as t_l. Clearly, $t_s < t_l < t_{ID}$.

B. Problem Definition

Consider a set of tags IDs $\mathbb{N} = \{t_1, ..., t_n\}$ that are stored in the system database, where n is the number of entire tags. Because, the absence of products often happens due to mistake, theft or other reasons, $\mathbb{N}$ would be outdated. Let $\mathbb{Y}$ be the set of tags within the coverage area of the system, where y is the number of tags present in the system. Therefore, each tags in $\mathbb{N} - \mathbb{Y}$ are absent in the system. We take $\mathbb{X}$ to represent the wanted tags, where x is the number of wanted tags. The set $\mathbb{X}$ may contain tags on a specified type of items or products under the surveillance by a manufacturer. The target of the search problem for RFID tags is to quickly identify which tags in $\mathbb{X}$ are present in the system, with the prior knowledge of the set of all tag IDs (i.e., $\mathbb{N}$). Namely, $\mathbb{X} \cap \mathbb{Y}$. To meet the requirements, the protocol should reduce the transmitted bits and efficiently utilize the information from each bit.

III. Physical-Layer Tag Searching Protocol

A. Basic Idea

The idea follows three guidelines to achieve high time efficiency. First, with $\mathbb{N}$ a priori, PLAT can filter out most unwanted tags by using filtration technique. Some tags in $\mathbb{N}-\mathbb{X}$ may also be selected to participate following phases. Because the reader has the entire tag IDs, it can predict the mis-selected ones and can pick them out from the searching result. Second, PLAT formulates the linear system of equations to take a global view of multiple slots in multiple frames. Solving the equations can easily obtain the searching result. We reduce the computation overhead by dividing the tags into multiple groups with partition technique. Third, PLAT can dig up more accurate number information in each slot rather than only three states, further improving the performance. By performing a joint optimization to minimize the combined overhead of those phases, the end result is a protocol that is far superior than the promising protocols.

PLAT consists of three phases, which are a filtration phase, a partition phase and a search phase. The filtration phase tells tags whether to participate in the following phases. The partition phase divides the tags into several parts to facilitate the searching process. And the search phase finds out whether the tags in the searching set are absence or not.

B. Phase I: Filtration

In the filtration phase, the reader runs multiple rounds to broadcast a filter that silences the irrelevant tags. Only the tags passing this filter will participate the following phases. This phase roughly filters the most of tags in $\mathbb{N} - \mathbb{X}$ by leveraging a filtering vector which is denoted as $\mathbf{V}_F$.

In a round, the reader first constructs a filtering vector $\mathbf{V}_F$ by hashing the ID of each tag in $\mathbb{X}$ to an f-bit vector. Here, the optimal f is set with respect to $|\mathbb{X}|$, which will be discussed in the following section. If only the tags from $\mathbb{N} - \mathbb{X}$ select the bit, the reader sets the bit to '1'. Otherwise, the reader sets the bit to '0'. After constructing $\mathbf{V}_F$, the reader broadcasts a request with parameters $\langle f, r_f \rangle$, where r_f is a random seed, and then $\mathbf{V}_F$. If $\mathbf{V}_F$ is too long, the reader can split it into 96-bit segments and transmit each of them in a time slot of length t_{id} [18]. On receiving $\mathbf{V}_F$, each tag in $\mathbb{N}$ hashes its own ID to a position in the f-bit filter as $H(id, r_f)$ mod f. Each tag from both $\mathbb{N} - \mathbb{X}$ and $\mathbb{X}$ checks its representative bit in $\mathbf{V}_F$. If the corresponding bit equals '1', the tag will keep

silent and not participate the following phases. Therefore, the tags from $\mathbb{N}-\mathbb{X}$ can be roughly filtered in multiple rounds.

The total number of the rounds in filtration phase will be determined by the parameters $|\mathbb{X}|$ and $|\mathbb{N}-\mathbb{X}|$ which are available to the reader. We delegate the detailed derivation on optimal number of rounds in Section IV. After the pre-calculated number of rounds, the reader roughly filters tags in $\mathbb{N}-\mathbb{X}$. We denote the set of those tags as Γ, which is $\mathbb{X} \subseteq \Gamma$. Then the reader heads to the partition phase.

C. Phase II: Partition

In the partition phase, the reader divides the tags in Γ into multiple partitions, which could dramatically reduce the computation overhead of solving the constructed linear system in the search phase. Each tag will be assigned a partition index, which the tag takes to enroll the search phase.

Consider a round, the reader first broadcasts a request with parameters $\langle p, r_p \rangle$, where p is the total number of partitions and r_p is a random seed. Note that the number of partitions p can be calculated as $p = \lceil \frac{\gamma}{w} \rceil$, where γ is the cardinality of the set Γ and w is the number of tags in each partition. Thus the tags are divided into p partitions by mod operation. Note w is determined by the computation ability of the server. In this paper, we set $w = 100$. Upon receiving this request, each tag picks a partition whose index is $H(id, r_p)$ mod p+1. The tag records the partition index which will be used in the search phase. We denote the set of tags which select the i-th partition as P_i. Since the partition depends on the random selection of each tag, $|P_i|$ fluctuates around w.

D. Phase III: Search

In the search phase, the reader manages to search the IDs in the target set $\mathbb{X}$ upon the reception of each slots. The search phase consists of multiple rounds, each of which has 4 steps. First, the reader construsts the indicating vector $\mathbf{V}_I$ to resolve the searching set. Second, the tags reply according to the indicate vector $\mathbf{V}_I$. Then, the reader silences the identified tags in the set Γ. Finally, the server formulates the system of linear equations based on the status of each slot. The solution to the system reveals the state (absence or presence) of each tag in the set Γ ($\mathbb{X} \in \Gamma$), i.e. $\mathbb{X}$.

1) Broadcast Step: In the broadcast step, the reader constructs an indicating vector $\mathbf{V}_I$ according to the search state of each partition set P_i ($i \in [1, p]$). The length of the vector equals the number of partitions (i.e. p). Consider a round, if all the tags in P_i have been identified, the reader sets the i-th bit of $\mathbf{V}_I$ to '0' ($V_i = 0$). On the contrary, the reader sets $V_i = 1$ to collect more information of P_i to identify the set.

After constructing $\mathbf{V}_I$, the reader first broadcasts a request with parameter $\langle l, r_s \rangle$, where l is the length coefficient and r_s is a random seed which is different in each round. Note that the parameter $\langle l, r_s \rangle$ is used for selecting slot by each tag. The reader then broadcasts the indicating vector $\mathbf{V}_I$.

2) Reply Step: Upon receiving the reader requests, each tag selects a slot to reply. Since the tags have the knowledge of the partition index they belong to, checking the corresponding bit of the indicating vector $\mathbf{V}_I$ from the reader makes the tag silent or replying. If the i-th bit of $\mathbf{V}_I$ equals '0', the tags from P_i keep silent in the following steps. On the contrary, if the i-th bit of $\mathbf{V}_I$ equals '1', the tags from P_i are still needed to participate the following steps.

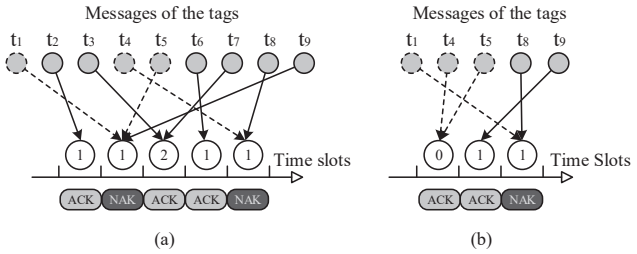


Fig. 2. Illustration of the search phase.

Consider a round, the tags from the set P_i count the total number of '1' bits before the i-th bit in the indicating vector $\mathbf{V}_I$, which is denoted as κ. If the $V_i = 0$, the tag keeps silence. On the contrary, the tag calculates the corresponding slot index as $l \times \kappa + (H(id, r) \bmod l)$. After selecting the slots, the tags transmit its responses to the reader in those slots. Obviously, the slots that the tags from the set P_i select are bounded at the range $[l \times \kappa, l \times \kappa + l - 1]$.

3) Acknowledgement Step: In the acknowledgement step, the reader sends acknowledgement (ACK or NAK) to eliminate the identified tags. When receiving a response, the reader chooses to reply an ACK to acknowledge that tag and prevents it from participating the following steps until the next execution of PLAT, or to reply a NAK to keep the tags active. Consider a round, the reader broadcasts requests as described in the broadcast step. After receiving the query, the tags select their slots and transmit their response to the reader in the selected slots as described in the reply step. We denote the number of tags are supposed to reply in i-th slot as S_i and the number of tags actually reply in i-th slot as ω_i ($\omega_i = 0, 1, 2, 3+$). The reader eliminates identified tags as follows:

- If $\omega_i = 0$, the reader replies an ACK.
- If $\omega_i \leq 3$ and $\omega_i = |S_i|$, the reader replies an ACK.
- If $\omega_i \neq 0$ and $\omega_i \neq |S_i|$, the reader replies a NAK.

Fig 2 illustrates how the reader in the acknowledgement step works. For presentation clarity, we suppose only one partition in our example. Fig 2(a) illustrates the first round execution of search phase. In the first slot, only t_2 replies, which means $\omega_1 = |S_1| = 1$. The reader replies an ACK to eliminate t_2. The same replies are for the third and fourth slot. In the second slot, t_1, t_5 and t_9 are supposed to reply, which means $|S_2| = 3$. But the reader receives only $\omega_2 = 1$ replies. The reader replies a NAK. The same replies are for the fifth slot. In Fig 2(b), the reader receives $\omega_1 = 0$ reply and then replies an ACK.

4) Computation Step: In this step, sever searches the set Γ. First, the server infers the absence or presence of tags in Γ through two kinds of slots. Then the server leverages the system of linear equations to fully utilize the information,

accelerating the search process. Until all the tags from Γ have been identified, the reader finishes the execution of PLAT. Due to $\mathbb{X} \subseteq \Gamma$, all the tags from the target set $\mathbb{X}$ also have been completely identified.

As described in previous step, the reader sends ACK in two kinds of slot: 1) $\omega_i = 0$; 2) $\omega_i \leq 3$ and $\omega_i = |S_i|$. For the first kind of slot, the server can easily identify that tags in S_i are all absent. In Fig 2(b), both t_4 and t_5 are supposed to reply in the first slot based on database. Since the reader receives $\omega_1 = 0$, it infers that both t_4 and t_5 are absent in the coverage of the reader. For the second kind of slot, the server can easily identify that tags in S_i are all present. In Fig 2(a), both t_3 and t_7 are supposed to reply in the third slot based on the database. Since the reader receives $\omega_3 = 2$, it infers that both t_3 and t_7 are present in the coverage of the reader.

Through the simple inferences above, the reader can search the set Γ in multiple rounds. But it wastes some slots which can accelerate the search process. After two rounds in Fig 2, the reader infers the set Γ except for t_1 and t_8. Thus the reader should run another round to identifies t_1 and t_8. Here, we leverage the system of linear equations to accelerate the search of the set Γ. By this approach, the reader can finish search in only the two rounds of Fig 2.

We introduce a_i to represent the existence of t_i in the system. If t_i is absent in the system, let $a_i = 0$. On the contrary, let $a_i = 1$. Therefore, we can construct the system of linear equations to formulate the information of each slot. On gaining the slot information, the equations exist as:

$$\begin{aligned} \sum_{j \in S_i} a_j &= b_i \quad (b_i = 0, 1, 2), \\ \text{or } \sum_{j \in S_i} a_j &\geq 3 \qquad (b_i = 3+). \end{aligned} \tag{1}$$

To solve the system efficiently, we deprive the inequations and only utilize the equations to construct the linear system. Note the reader can distinguish no more than 3 tags through 8 bits, which are 0, 1, 2 and 3+ respectively. For each slot, we can add the equations to the system as:

$$\begin{aligned} \sum_{j \in S_i} a_j &= b_i \qquad\qquad (b_i = 0, 1, 2), \\ \text{or } \sum_{j \in S_i} a_j &= 3 \quad (|S_i| = 3,\ b_i = 3+). \end{aligned} \tag{2}$$

Take Fig 2 for an example, the initial candidate tag set is $\Gamma = \{t_1, t_2, ..., t_9\}$. The target of the search is to identify whether the tag in Γ is absent or present. Therefore, the server introduces a_i to represent the existence of t_i. After two rounds, it constructs the system of linear equations as shown in the left part of Eqn (3).

$$\begin{cases} a_2 = 1 \\ a_1 + a_5 + a_9 = 1 \\ a_3 + a_7 = 2 \\ a_6 = 1 \\ a_4 + a_8 = 1 \\ a_4 + a_5 = 0 \\ a_9 = 1 \\ a_1 + a_8 = 1 \end{cases} \Rightarrow \begin{cases} a_2 = 1 \\ a_1 + a_5 + a_9 = 1 \\ \boxed{a_3 = 1} \\ \boxed{a_7 = 1} \\ a_6 = 1 \\ a_4 + a_8 = 1 \\ \boxed{a_4 = 0} \\ \boxed{a_5 = 0} \\ a_9 = 1 \\ a_1 + a_8 = 1 \end{cases} \tag{3}$$

Because the value of a_i is 0 or 1, we can easily break an equation to multiple equations to increase more rank for the system of equations in the two cases ($|S_j| \neq 0$) as: (1) $\sum_{j \in S_i} a_j = 0 \Rightarrow \forall j \in S_i, a_j = 0$; (2) $\sum_{j \in S_i} a_j = |S_i| \Rightarrow \forall j \in S_i, a_j = 1$. We can re-construct the system of linear equations as shown in the right part of Eqn (3). Here, we denote the number of equations as ν.

We then convert the system of linear equation to matrix form as $Ax = b$, where A is an $\nu \times \gamma$ coefficient matrix, x is a $1 \times \gamma$ row vector, and b is a $1 \times \nu$ row vector. We can convert the system as shown in the right part of Eqn (3) to the matrix form as follows:

$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \tag{4}$$

To solve the matrix equation, we can easily obtain the result $x^T = [0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 1]$. The reader infers that three tags (t_1, t_4 and t_5) in Γ are absent. If we take the tag IDs which have been already identified by the simple inferences to simplify the linear matrix. We can get the matrix as:

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_8 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}. \tag{5}$$

By solving it, we can also easily obtain the search result, which also decreases the computation overhead.

IV. Protocol Analysis

In this section, we first analyse the setting of optimal parameters for each phases. Then we give the joint optimization to minimize the total execution time of three phases.

A. The Optimal Parameter for the Filtration Phase

The target of the filtration phase is to eliminate irrelevant tags from the set $\mathbb{N} - \mathbb{X}$ with a defined accuracy. In a round, let ξ be the expected number of the tags, which are eliminated to keep silent in the following phases. And let τ_f be the execution time of a round in the filtration phase. Thus the expected number of eliminated tags per unit time (denoted as φ) can

be represented as $\varphi = \frac{\xi}{\tau_f}$. We take the following lemmas to analyze ξ, τ_f and φ.

Lemma 1. *In a round, the expected number of the eliminated tags in $\mathbb{N}-\mathbb{X}$ (ξ) is $\xi = |\mathbb{N}-\mathbb{X}|e^{-|\mathbb{X}|/f}$, where $|\mathbb{N}-\mathbb{X}|$ is the cardinality of $\mathbb{N}-\mathbb{X}$, and $|\mathbb{X}|$ is the cardinality of $\mathbb{X}$.*

Proof. In a round, a tag from the set $\mathbb{N}-\mathbb{X}$ can be eliminated when it selects a bit in the filtering vector $\mathbf{V}_F$ to which no tag from the set $\mathbb{X}$ is mapped. Here we denote the probability of that a tag from the set $\mathbb{N}-\mathbb{X}$ can be eliminated in this round as $Pr(A)$. We then can get $Pr(A) = (1-\frac{1}{f})^{|\mathbb{X}|} \approx e^{-|\mathbb{X}|/f}$. Given the fact that f is normally very large, $Pr(A)$ can be simplified to $e^{-|\mathbb{X}|/f}$, where e is the natural constant. We know the total number of the set $\mathbb{N}-\mathbb{X}$ is $|\mathbb{N}-\mathbb{X}|$. Each of them is expected to be eliminated with the probability $Pr(A)$. Therefore, the expected number of the eliminated tags in $\mathbb{N}-\mathbb{X}$ is $|\mathbb{N}-\mathbb{X}|Pr(A) = |\mathbb{N}-\mathbb{X}|e^{-|\mathbb{X}|/f}$. □

Lemma 2. *When the parameter of the vector size is set to f, the expected execution time in a round is $\tau_f = ft_{ID}/96$.*

Proof. The reader first broadcasts the parameters $\langle f, r_f\rangle$ in a tag slot time t_{ID}. Then the reader broadcasts the filtering vector $\mathbf{V}_F$, which is divided into multiple segments of 96-bit to be transmitted in multiple slot time. The time for transmitting $\mathbf{V}_F$ is $\lceil\frac{f}{96}\rceil t_{ID}$. Due to the large value of f, combining the execution times yields $t_{ID} + \lceil\frac{f}{96}\rceil t_{ID} \approx \frac{ft_{ID}}{96}$. □

Thus, upon taking Lemma 1 and 2, the expected number of eliminated tags per unit time (denoted as φ) can be rewritten as $\varphi = \frac{96|\mathbb{N}-\mathbb{X}|}{ft_{ID}}e^{-\frac{|\mathbb{X}|}{f}}$ The derivative of φ is given as:

$$\frac{\partial\varphi}{\partial f} = \frac{96|\mathbb{N}-\mathbb{X}|e^{-\frac{|\mathbb{X}|}{f}}}{f^3 t_{ID}}(|\mathbb{X}|-f). \tag{6}$$

To get the maximum efficiency of φ, we let $\frac{\partial\varphi}{\partial f} = 0$. Therefore, φ achieves the maximum when f is set to $|\mathbb{X}|$.

We then analyse the total number of rounds the filtration phase needs. In the filtration phase, the reader sets the accuracy of tag elimination as $\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|}$, where α is the coefficient ranges from 0 to 1. Here we denote the number of the round that the eliminating process is repeated for as ζ. We take the following lemmas to analyze ζ.

Lemma 3. *After ζ rounds, the probability that a tag from $\mathbb{N}-\mathbb{X}$ can be eliminated at least once (denoted as $Pr(B)$) is as $Pr(B) = 1-(1-\frac{1}{e})^{\zeta}$.*

Proof. The length of the filtering vector $\mathbf{V}_F$ is set to $|\mathbb{X}|$ as described in Subsection IV-A. The probability (denoted as $Pr(C)$) that a tag from $\mathbb{N}-\mathbb{X}$ can be eliminated in a round is $Pr(C) = (1-\frac{1}{|\mathbb{X}|})^{|\mathbb{X}|} \approx e^{-1}$. After ζ rounds, the probability that a tag from $\mathbb{N}-\mathbb{X}$ can be eliminated at least once can be calculated as $Pr(B) = 1-(1-Pr(C))^{\zeta} = 1-(1-\frac{1}{e})^{\zeta}$ □

Due to the target of the filtration phase is to eliminate the tags from $\mathbb{N}-\mathbb{X}$ with an accuracy $\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|}$. Thus the probability that a tag from $\mathbb{N}-\mathbb{X}$ can be eliminated at least once is no less than $1-\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|}$. Then we can get $Pr(B) = 1-(1-\frac{1}{e})^{\zeta} \geq 1-\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|}$. With respect to ζ, we can have:

$$\zeta \geq \frac{\ln(\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|})}{\ln(1-e^{-1})}. \tag{7}$$

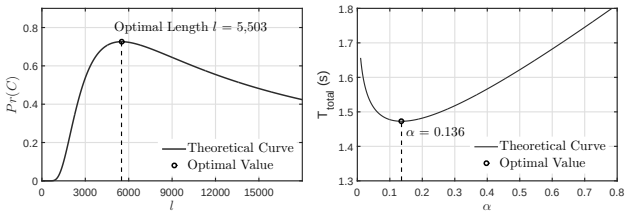
The filtration phase should repeat for $\lceil\frac{\ln(\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|})}{\ln(1-e^{-1})}\rceil$ rounds to eliminate the tags from $\mathbb{N}-\mathbb{X}$ with a given accuracy.

B. The Optimal Parameter for the Search Phase

In the search phase, tags from each partition P_i will only select the slot ranges from $[l\times\kappa, l\times\kappa+l-1]$. Actually, each partition has its optimal value of l_i. But we only set $l = \max(l_1, l_2, ..., l_p)$ for the parameter. Here, we focus on one partition to investigate the optimal l_i. As a result, it is easy to get the maximum l among these partitions. Our problem can be defined as given γ, we want to pick the smallest l_i for the search phase. Since the scanning time is proportional to the length coefficient l_i, our problem is formulated as to minimize l_i with the constrain of $\gamma > 0$.

Lemma 4. *Given l_i, the probability (denoted as Pr(C)) of the effective slot which can be used to construct the linear equations is $Pr(C) = \sum_{k=1}^{3}\binom{\gamma}{k}(\frac{1}{l_i})^k(1-\frac{1}{l_i})^{\gamma-k}$.*

Proof. We know the length of the vector is l_i, thus the probability for each tag select a slot to reply is $\frac{1}{l_i}$. If the reader wants to identify tags through the current slot, there must be no more than 3 tags. Let $\mathbb{N}$ represent the number of tags select the slots to reply. When $\mathbb{N}=1$, we have $Pr(\mathbb{N}=1) = \binom{\gamma}{1}\frac{1}{l_i}(1-\frac{1}{l_i})^{\gamma-1}$. Therefore, the probability of the effective slot which can be used to construct the linear equations in a round is $Pr(C) = \sum_{k=1}^{3}Pr(\mathbb{N}=k) = \sum_{k=1}^{3}\binom{\gamma}{k}(\frac{1}{l_i})^k(1-\frac{1}{l_i})^{\gamma-k}$. □



(a) (b)

Fig. 3. Theoretical curve and the optimal value for the function of $Pr(C)$ and T_{total}. (a) The value of $Pr(C)$ with respect to l_i when $\gamma = 10,000$. (b) The value of T_{total} with respect to α, when $|\mathbb{X}| = 2,000$ and $|\mathbb{N}| = 50,000$.

From Lemma 4, we know $Pr(C)$ is a function of l_i. Fig 3(a) shows the value of $Pr(C)$ with respect to l_i when $\gamma = 10,000$. To find the maximum of $Pr(C)$, we let $\frac{\partial Pr(C)}{\partial l_i} = 0$. Solving it numerically, we get the optimal l_i. For example, the optimal length of the vector in Fig 3(a) is 5,503.

C. Joint Optimization

To search for a wanted set $|\mathbb{X}|$, the reader must execute three phases, which are filtration phase, partition phase and search

phase. Therefore, we can consider the three individual parts as a whole to make the joint optimization. Here we denote the total time of the three phases as $T_{total} = T_1 + T_2 + T_3$, where T_1, T_2 and T_3 represent the execution time of each phases.

For the first phase, the reader uses an f-bit vector to filter the irrelevant tags in ζ rounds. Then the execution time of the filtration phase is $T_1 = \zeta\tau_f$. From Lemma 2, we can easily know when the parameter of the vector size is set to f, the expected execution time in a round is $\tau_f = \frac{ft_{ID}}{96}$. And to get the maximum efficiency, f must be set to $|\mathbb{X}|$. Therefore, we have $\tau_f = \frac{|\mathbb{X}|t_{ID}}{96}$. From Eqn 7, we know $\zeta = \lceil \frac{\ln(\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|})}{\ln(1-e^{-1})} \rceil$. To re-write the execution time of the first phase, we get:

$$T_1 = \lceil \frac{\ln(\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|})}{\ln(1-e^{-1})} \rceil \frac{|\mathbb{X}|t_{ID}}{96}. \tag{8}$$

For the second phase, the reader broadcasts a request with parameter $\langle p, r_p \rangle$ to split the tags into p groups. Then the execution time of the partition phase is $T_2 = pt_{ID}/96$.

For the third phase, the reader computes the indicating vector and broadcasts the parameter and the vector to search the wanted set. Due to the complexity of the analysis of the linear system, we make an approximation to the total execution time of the search phase through the simulation results, which are shown in Section VI. Then the execution time of the third phase is approximately $T_3 = (1+\alpha)|\mathbb{X}|t_l$.

Therefore, the total execution time of the three phases is:

$$\begin{aligned} T_{total} &= T_1 + T_2 + T_3 \\ &= \lceil \frac{\ln(1-\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|})}{\ln(1-e^{-1})} \rceil \frac{|\mathbb{X}|t_{ID}}{96} + \frac{pt_{ID}}{96} + (1+\alpha)|\mathbb{X}|t_l \\ &\approx \frac{\ln(1-\frac{\alpha|\mathbb{X}|}{|\mathbb{N}-\mathbb{X}|})}{\ln(1-e^{-1})} \frac{|\mathbb{X}|t_{ID}}{96} + (1+\alpha)|\mathbb{X}|t_l \end{aligned} \tag{9}$$

In Eqn 9, we know T_{total} is a function of α. Fig 3(b) shows the value of T_{total} with respect to α, when $|\mathbb{X}| = 2,000$ and $|\mathbb{N}| = 50,000$. To compute the minimum value of T_{total}, we let the first derivative of T_{total} be zero $\frac{\partial T_{total}}{\partial \alpha} = 0$. Solving the equation, numerically, we can find the optimal α minimizes T_{total}. For example, the optimal α in Fig 3(b) is 0.136.

V. Implementation

A. Setup

In this section, we propose a heuristic symbol clustering (HSC) algorithm to detect the number of tags replying in a slot. To explore the efficiency of HSC in actual environment, we setup a testbed with USRP software-defined platform and programmable WISP tags. Our test environment is shown in Fig 4(a). USRP1 in the prototype has two complete RFX900 daughterboards which are designed for operation in the 900 MHz band. The RFID tag is implemented with the WISP programmable device based on the DL-WISP4.1 firmware. The WISP tag generally comprises two parts: the first part is the MSP430F2132 microcontroller which can work in ultra-low power, the second part is an antenna circuitry which can

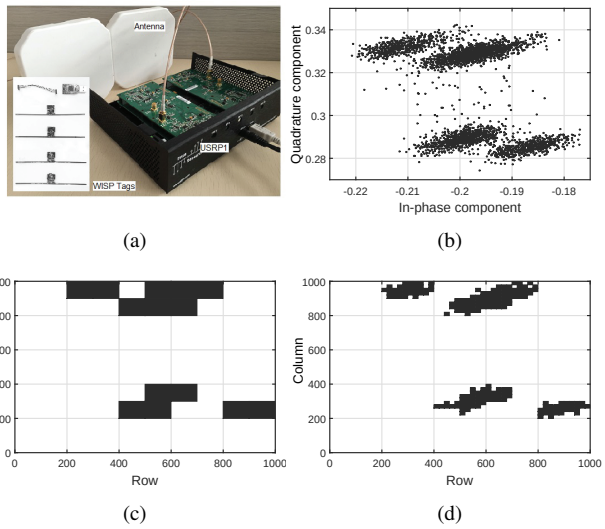


Fig. 4. An example of HSC algorithm: (a) Experiment environment with an USRP1 platform and WISP tags. (b) Gathered symbols from 2 tags in a slot. (c) Clusters in the first round of the filtration phase. (d) Clusters in the second round of the filtration phase.

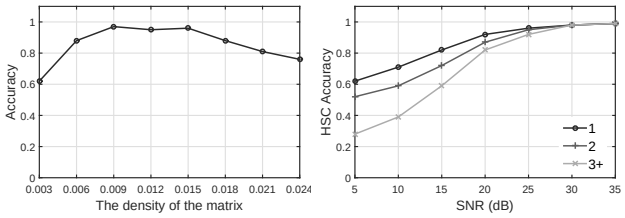
gather and backscatter signals. In the firmware of both USRP and WISP, most of operations (e.g., QUERY, ACK) specified in the EPCglobal Gen-2 standard have been implemented. Therefore, we can easily modify the source code of the firmware to support our protocol.

B. heuristic symbol clustering Algorithm

HSC aims at identifying the number of efficient clusters, based on the idea of defining cluster as connected dense components. Since areas of low-point density can be arbitrarily shaped in the data space, we first roughly filter the noise of the physical layer symbols. After the filtration, we cluster the signals to detect each slot state. HSC needs to cluster the physical layer signals in at most a slot with 10 bits.

Our algorithm HSC consists of three phases, which are the pre-processing phase, the filtration phase and the calculation phase respectfully. In the pre-processing phase, the reader captures the physical layer signals that tags concurrently transmit in a slot. Since the symbols are represented in decimal format, we make discretization for the physical layer symbols to speed up computation. We first measure the range of both in-phase and quadrature components over all the data points in a slot. Thus the server can concentrate on the signal data without irrelevant parts. We then map the coordinate of each points to $1,000 \times 1,000$ matrix. If the physical layer symbols locate in the corresponding position of the matrix, we set the element to '1'. On the contrary, if no physical layer symbols is in the position, we set the element to '0'. After mapping all the signal data, we can easily obtain the symbol matrix M.

In the filtration phase, we filter the noise of the physical layer symbols in two rounds. Since the size of the symbol matrix M is $1,000 \times 1,000$, we divide it into 100 sub-matrices M_i ($i \in [1, 100]$) with 100×100 size in the first round. Then by counting the number of '1's in each sub-matrix, we



(a) (b)

Fig. 5. Efficiency investigation of HSC. (a) The impact of the threshold on the accuracy. (b) The HSC accuracy with different channel errors.

can calculate the density of each sub-matrix as $\rho = \frac{k}{10{,}000}$. With a threshold r, some sub-matrices with low density can be filtered. The sub-matrices with high density can be used to facilitate the computation of the number of clusters. We denote the set of the sub-matrices with high density as Υ. Therefore, we fast zoom into the interested parts among sub-matrices. Fig 4(c) shows the first round of filtration. The most of the sub-matrices have been filtered. In the second round, we divide each sub-matrices from the set Υ into 25 sub-matrices with 20×20 size. By counting the number of '1's in each sub-matrix, we can also calculate the density of the sub-matrix as $\rho = \frac{k}{400}$. With the same threshold r, the sub-matrices with high density can be selected, which denoted as Φ (Fig 4(d)).

In the calculation phase, we extract the number of clusters from the set Φ. If the vertexes of two sub-matrices in Φ are connected, we put the sub-matrices into the same clusters. On checking for all the sub-matrices in Φ, we can get multiple clusters. In Fig 4(d), the server obtains five clusters. But note that a cluster contains only one 20×20 sub-matrix. We can set the threshold to filter this kind of clusters. We can count the number of sub-matrices in each clusters, which is denoted as u. The number of the cluster is denoted as v. Then we only count the number of sub-matrices above $0.1\frac{u}{v}$. As discussed above, we know if n tags collide in a slot, 2^n clusters will formed. When we get the number of clusters C, the number of tags from a slot can be calculated as $\lceil \log_2 C \rceil$.

C. HSC Evaluation

We then study the impact of the threshold in the filtration process. Note the problem of finding suitable density thresholds is also a challenging issue. Due to the focus of the paper is on designing the fast searching protocol, as the same in [14], we only give empirical thresholds based on the gathered data. We test the efficiency of our protocol based on 200 slots, then average the results which are shown in Fig 5(a). As the change of the threshold, the accuracy of the identification may vary. The density between 0.009 and 0.015 is moderate for the case which can achieve the accuracy above 0.92. When clustering the physical layer signals to exploit the number of replying tags, the noise tends to disrupt the algorithm and makes it difficult to detect the cluster structure. Due to the two rounds filtration in the second phase, HSC can dive into smaller scale to investigate the density of each clusters without scarifying efficiency and accuracy.

TABLE I
PERFORMANCE COMPARISON OF TAG SEARCH PROTOCOLS.
$|\mathbb{Y}| = 50{,}000$, $P_{REQ} = 10^{-3}$ AND $R_{INTS} = 0.2$

$\lvert\mathbb{X}\rvert$	PLAT			E-STEP	ITSP	CATS
	R_S=1.1	1.5	1.9			
2,000	2.8	2.9	2.8	8.4	14.1	21.9
6,000	8.0	8.2	8.1	22.2	36.4	60.4
10,000	12.9	13.3	13.3	34.3	57.1	95.9
14,000	17.7	18.3	18.3	45.4	75.3	129.5
18,000	22.6	22.9	23.2	55.8	92.2	162.2
22,000	27.3	27.9	28.1	66.1	110.5	195.2
26,000	31.8	32.5	33.3	76.0	125.3	226.5
30,000	36.0	37.0	38.0	84.6	141.1	258.5

D. Channel Error

In an actual environment, wireless communications is error-prone. Channel error may corrupt the data exchanged between the reader and tags. For example, if the preamble from a tag disturbed by the channel noise, the accuracy of HSC algorithm will decrease. We evaluate the impact of the channel error on the accuracy of the HSC algorithm. Fig 5(b) plots the average accuracy of the HSC algorithm through the different signal to noise ratio (SNR). As shown in the figure, the higher SNR, the more accurate identification result. When the SNR is between 20 and 30, HSC algorithm can keep in a high accuracy (80% above) for 1, 2 and 3+ cases.

VI. PERFORMANCE EVALUATION

A. Simulation Settings

We implement a simulator with Python to evaluate the performance of our protocol in large-scale systems. Our protocol is compared with the state-of-the-art protocols, including CATS [11], ITSP [12] and E-STEP [13]. Based on the specification of the Philips I-Code system [17], we set $t_s = 0.4ms$, $t_l = 0.8ms$ and $t_{ID} = 2.4ms$. Since the protocols CATS, ITSP and E-STEP are designed to search the tags with a guaranteed false rate, we define the required false positives ratio as $P_{REQ} = \frac{|W^* - \mathbb{X} \cap \mathbb{Y}|}{|\mathbb{X} - \mathbb{X} \cap \mathbb{Y}|}$, where W^* is the set of tags in the search result. Then we set $P_{REQ} = 10^{-3}$ as the same in [13]. Each simulation experiment was conducted for 100 times and then we report the averaged results of the independent trials.

B. Performance Comparison

We evaluate the performance of PLAT, comparing with the state-of-the-art searching protocols E-STEP, ITSP and CATS. In Table I, we set $|\mathbb{Y}| = 50{,}000$, $R_{INT} = 0.2$ ($R_{INT} = \frac{|\mathbb{X} \cap \mathbb{Y}|}{|\mathbb{X}|}$), and vary $|\mathbb{X}|$ from $2{,}000$ to $30{,}000$. In Table II, we set $|\mathbb{X}| = 10{,}000$, $R_{INT} = 0.2$, and vary $|\mathbb{Y}|$ from $40{,}000$ to $100{,}000$. Since the execution time of our protocol is related to $|\mathbb{N}|$, we define the ratio $R_s = \frac{|\mathbb{N}|}{|\mathbb{Y}|}$ and let $R_s = 1.1, 1.5, 1.9$.

Generally, both tables show that PLAT performs much better than E-STEP, ITSP and CATS. In Table I, when $|\mathbb{X}|$ becomes large, the execution time of PLAT, E-STEP, ITSP and CATS increase. For example, when $|\mathbb{X}| = 6{,}000$ and $R_s = 1.5$, PLAT finishes searching in $8.2s$. In the meantime,

TABLE II
PERFORMANCE COMPARISON OF TAG SEARCH PROTOCOLS.
$|\mathbb{X}| = 10,000$, $P_{REQ} = 10^{-3}$ AND $R_{INTS} = 0.2$

$\|\mathbb{Y}\|$	PLAT			E-STEP	ITSP	CATS
	R_S=1.1	1.5	1.9			
$40,000$	12.9	13.0	13.2	40.2	66.9	93.6
$50,000$	13.0	13.2	13.2	41.0	68.4	95.4
$60,000$	13.1	13.4	13.5	41.9	70.8	96.2
$70,000$	13.3	13.4	13.5	43.1	71.3	97.9
$80,000$	13.2	13.5	13.7	43.4	72.3	99.5
$90,000$	13.4	13.5	13.6	43.8	73.7	99.7
$100,000$	13.4	13.6	13.7	44.9	74.6	101.1

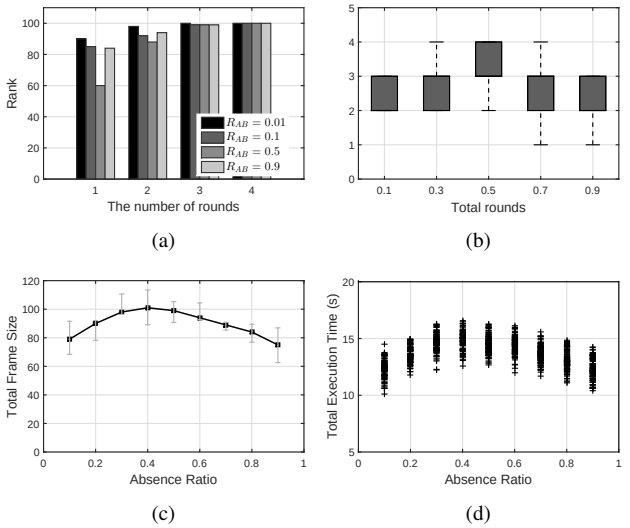


Fig. 6. The efficiency of solving linear system. (a) The average rank of the linear system in the first round. (b) The total execution rounds. (c) The total frame size. (d) The total execution time.

E-STEP has the execution time with $22.2s$. As $|\mathbb{X}|$ increases to $18,000$, PLAT and E-STEP finish searching in $27.9s$ and $66.1s$ respectively. In Table II, when $|\mathbb{Y}|$ becomes large, the execution time of E-STEP, ITSP and CATS also increase, but PLAT is not affected. This is because the execution time of PLAT is mainly related to $|\mathbb{N}|$. We observe when $R_s = \frac{|\mathbb{N}|}{|\mathbb{Y}|}$ grows, PLAT has an increase in execution time. Since the joint optimization ensures the high efficiency of searching process, the impact of the change of R_s is small. From Table I and II, we know PLAT always outperforms E-STEP, ITSP and CATS. In addition, PLAT can identify completely the tags from $\mathbb{X} \cap \mathbb{Y}$. But E-STEP, ITSP and CATS has the false positives in the search results.

C. Performance of Solving Linear System

1) Efficiency Investigation: We first evaluate the efficiency of search phase with fixed tag number γ ($|\mathbb{X}| \leq \gamma$). Since in the partition phase, the reader divides γ tags into p groups, the system constructs p linear systems to search Γ. For the simplicity of description, we only focus on the i-th group S_i where 100 tags are in the set S_i (i.e., $|S_i| = 100$). We define the absence ratio (R_{AB}) to represent the percentage of the absent tag number in S_i. For example, if $R_{AB} = 0.1$ and $|S_i| = 100$, it means 10 tags in S_i are absent.

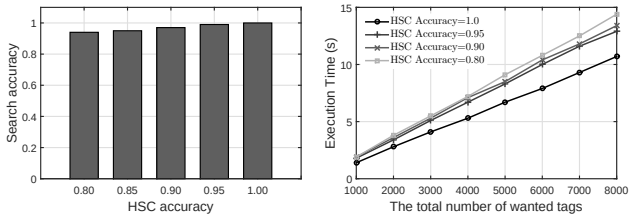


Fig. 7. Evaluating the impact of HSC accuracy. (a) The impact of HSC accuracy on the total execution time. (b) The impact of HSC accuracy on the total search accuracy.

Fig 6(a) plots the average rank of the linear system in the first four rounds, when we fix $|S_i| = 100$ and vary $R_{AB} = 0.01, 0.1, 0.5, 0.9$. We observe in all the cases, PLAT can achieve the rank with 100 in only four rounds. When a little number of tags are absent, PLAT has a reduction in the number of execution round. For example, if $R_{AB} = 0.01$, in only 3 rounds, the rank of the linear system nearly approaches 100. But if $R_{AB} = 0.1$, PLAT must execute about 4 rounds to obtain 100 rank.

Fig 6(b) plots the total execution rounds which the reader needs to finish searching in S_i, when we fix $|S_i| = 100$ and vary $R_{AB} = 0.1, 0.3, 0.5, 0.7, 0.9$. From the figure, we observe in the most cases, PLAT finishes in searching in no more than 3 rounds. When the more number of tags are absent or present in the set S_i, the less rounds PLAT needs to finish searching. For example, if $R_{AB} = 0.9$, on average, the reader can identify the set S_i in 3 rounds.

Fig 6(c) plots the total number of slots, in which the reader needs to complete the searching of S_i, when we fix $|S_i| = 100$ and vary $R_{AB} = 0.1, 0.3, 0.5, 0.7, 0.9$. Fig 6(d) plots the total execution time, in which we fix $|\mathbb{N}| = 75,000$ and $|\mathbb{X}| = 10,000$ to evaluate and vary $R_{AB} = 0.1, 0.3, 0.5, 0.7, 0.9$. Since in the acknowledgement step of the search phase, the reader sends acknowledgement (ACK or NAK) to eliminate the identified tags. When R_{AB} approaches 0.5, the more time the reader needs to finish searching.

2) Impact of HSC Accuracy: We then evaluate the impact of HSC accuracy on the total search process. If HSC gets wrong slot type, the system of linear equations possibly has no solutions. Thus, the reader simply infers the presence of tags through two kinds of slots (Section III). The reader obtains the number of tag replying in one of three kinds of slots (1, 2, 3+). Fig 7(a) plots the accuracy of tag searching when HSC identification error exists, where we fix $|\mathbb{X}| = 5,000$, $|\mathbb{Y}| = 40,000$, $|\mathbb{N}| = 50,000$, and varying the HSC accuracy among $0, 0.05, 0.1, 0.2$. If the HSC identification error exists, the search results suffer from biases.

Fig 7(b) plots the impact of HSC accuracy on the total execution time, with fixing $|\mathbb{N}| = 50,000$, and varying the HSC accuracy among $0, 0.05, 0.1, 0.2$, $|\mathbb{X}|$ from $1,000$ to $8,000$. In practice, the HSC accuracy can vary due to various

factors, such as the channel error, the tag position or the transmission power of the reader. In the ideal situation, there would be no HSC identification error, which means a perfect searching result. But if the HSC identification error exists, the reader will take more time to finish searching. As shown in Fig 7(b), PLAT with HSC accuracy 1.0 and $|\mathbb{X}| = 5,000$ takes $6.7s$. In the meantime, PLAT with HSC accuracy 0.90 takes $10.4s$.

VII. Related Work

Existing tag identification protocols fall into two categories: Tree-based protocols [19]–[22] and Aloha-based protocols [23]–[26]. In Tree-based protocols, the reader sends an initialization request and transmits one bit of ID at a time. If there exists collisions, the reader divides the another two groups. Tags with matching bits will reply. Only until at least one of groups contains one tag, the reader can identify the tag. In Aloha-based identification protocols, the reader broadcasts an initialization command the tags use to reply in a random slot. When receiving the query request, each tag transmits its ID in the corresponding slot.

There are also some work on the tag searching problem. In CATS [11], it first addresses the tag searching problem, which proposes an efficient compact approximation-based tag searching protocol, by employing Bloom filters to compact the information. In ITSP [12], it introduces a filtering vector technique which is a compact one-dimension bit array constructed from tag IDs. The filtering vector improves the accuracy and reduces the execution time. In E-STEP [13], it designs a testing slot technique which tests the presence of wanted tags without tag ID transmissions. Therefore, the reader can finish tag searching with high time efficiency and accuracy.

VIII. Conclusion

This paper studies the tag searching problem in large RFID systems. For many RFID-enabled applications, fast searching a particular set of RFID tags is practically important. To address the problem, we propose PLAT, a physical layer tag searching protocol, which can make a fundamental improvement on the searching performance. We also propose a heuristic symbol clustering (HSC) algorithm to fast identify the number of replying tags in a slot. We implement a prototype system and validate the feasibility of HSC algorithm based on USRP and WISP platform. Extensive simulations show PLAT has a great improvement compared with the existing promising works.

Acknowledgment

This research is financially supported by the National Natural Science Foundation of China (No.61272418, 61373181), the National Science and Technology Support Program of China (No.2012BAK26B02), the Future Network Prospective Research Program of Jiangsu Province (No.BY2013095-5-02), the Lianyungang City Science and Technology Project (Industry Development) (No.CG1420), the Fundamental Research Funds for the Central Universities and HK PolyU G-SB40.

References

[1] L. Xie, H. Han, Q. Li, J. Wu, and S. Lu, "Efficient protocols for collecting histograms in large-scale rfid systems," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 2015.

[2] J. Han, H. Ding, C. Qian, D. Ma, W. Xi, Z. Wang, Z. Jiang, and L. Shangguan, "Cbid: A customer behavior identification system using passive tags," in *Proc. of IEEE ICNP*, 2014, pp. 47–58.

[3] J. Han, C. Qian, X. Wang, D. Ma, J. Zhao, P. Zhang, W. Xi, and Z. Jiang, "Twins: Device-free object tracking using passive tags," in *Proc. of IEEE INFOCOM*, 2014, pp. 469–476.

[4] J. Wang, D. Vasisht, and D. Katabi, "Rf-idraw: virtual touch screen in the air using rf signals," in *Proc. of ACM SIGCOMM*, 2014.

[5] K. Bu, B. Xiao, Q. Xiao, and S. Chen, "Efficient misplaced-tag pinpointing in large rfid systems," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 23, no. 11, pp. 2094–2106, 2012.

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

[7] F. Zhu, J. Liu, and L. Chen, "Fast physical-layer unknown tag identification in large-scale RFIDsystems," in *Proc. of IEEE GLOBECOM*, 2014, pp. 511–516.

[8] J. Liu and L. Chen, "Placement of multiple rfid reader antennas to alleviate the negative effect of tag orientation," in *Proc. of IEEE ICPADS*, Dec 2012, pp. 432–439.

[9] S. Zhang, X. Liu, J. Wang, J. Cao, and G. Min, "Energy-efficient active tag searching in large scale rfid systems," *Information Sciences*, vol. 317, pp. 143 – 156, 2015.

[10] X. Liu, S. Zhang, B. Xiao, and K. Bu, "Flexible and time-efficient tag scanning with handheld readers," *IEEE Transactions on Mobile Computing (TMC)*, vol. 15, no. 4, pp. 840–852, 2016.

[11] Y. Zheng and M. Li, "Fast tag searching protocol for large-scale rfid systems," *IEEE Transactions on Networking (TON)*, 2013.

[12] M. Chen, W. Luo, Z. Mo, S. Chen, and Y. Fang, "An efficient tag search protocol in large-scale rfid systems," in *Proc. of IEEE INFOCOM*, 2013.

[13] X. Liu, B. Xiao, S. Zhang, K. Bu, and A. Chan, "Step: A time-efficient tag searching protocol in large rfid systems," *IEEE Transaction on Computers (TOC)*, 2015.

[14] Y. Hou, J. Ou, Y. Zheng, and M. Li, "Place: Physical layer cardinality estimation for large-scale rfid systems," in *Proc. of IEEE INFOCOM*, 2015.

[15] J. Liu, B. Xiao, S. Chen, F. Zhu, and L. Chen, "Fast rfid grouping protocols," in *Proc. of IEEE INFOCOM*, 2015.

[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 (TPDS)*, 2014.

[17] P. Semiconductors, "I-code smart label rfid tags," http://www.nxp.com/acrobat_download/other/identification/SL092030.pdf, 2004.

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

[19] 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 Systems (TPDS)*, 2007.

[20] M. Shahzad and A. X. Liu, "Probabilistic optimal tree hopping for rfid identification," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, no. 1. ACM, 2013, pp. 293–304.

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

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

[23] L. Xie, B. Sheng, C. Tan, H. Han, Q. Li, and D. Chen, "Efficient tag identification in mobile rfid systems," in *Proc. of IEEE INFOCOM*, March 2010, pp. 1–9.

[24] V. Namboodiri and L. Gao, "Energy-aware tag anticollision protocols for rfid systems," *IEEE Transactions on Mobile Computing (TMC)*, 2010.

[25] B. Zhen, M. Kobayashi, and M. Shimizu, "Framed aloha for multiple rfid objects identification," *IEICE Transactions on Communications*, 2005.

[26] S.-R. Lee, S.-D. Joo, and C.-W. Lee, "An enhanced dynamic framed slotted aloha algorithm for rfid tag identification," in *Proc. IEEE MobiQuitous*, July 2005, pp. 166–172.